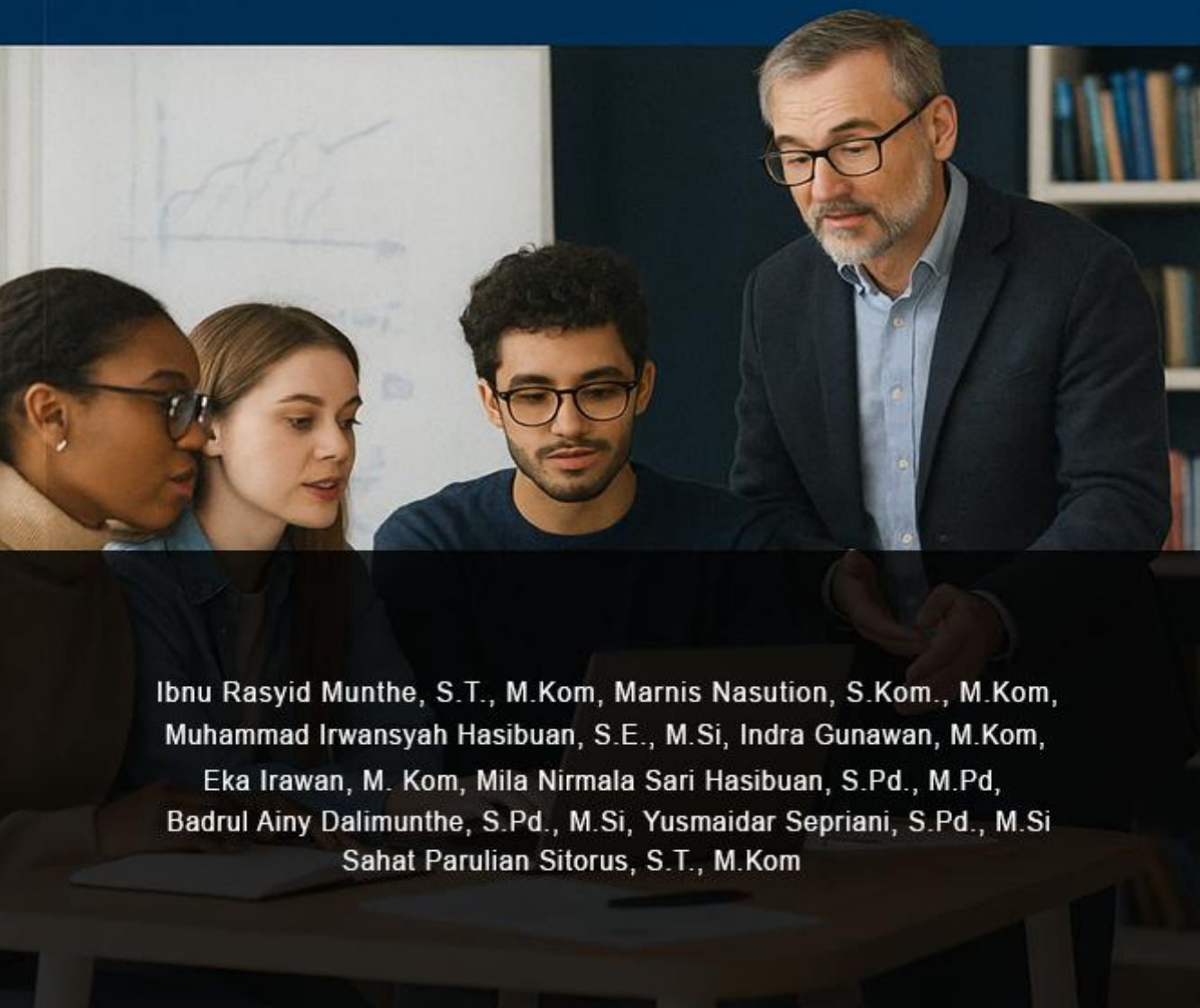


BUKU AJAR METODOLOGI PENELITIAN

Perancangan Sistem Informasi
dan Data Sains



Ibnu Rasyid Munthe, S.T., M.Kom, Marnis Nasution, S.Kom., M.Kom,
Muhammad Irwansyah Hasibuan, S.E., M.Si, Indra Gunawan, M.Kom,
Eka Irawan, M. Kom, Mila Nirmala Sari Hasibuan, S.Pd., M.Pd,
Badrul Ainy Dalimunthe, S.Pd., M.Si, Yusmaidar Sepriani, S.Pd., M.Si
Sahat Parulian Sitorus, S.T., M.Kom

BUKU AJAR
METODOLOGI PENELITIAN PERANCANGAN SISTEM
INFORMASI DAN DATA SAINS

Ibnu Rasyid Munthe, S.T., M.Kom,
Marnis Nasution, S.Kom., M.Kom,
Muhammad Irwansyah Hasibuan, S.E., M.Si,
Indra Gunawan, M.Kom,
Eka Irawan, M. Kom,
Mila Nirmala Sari Hasibuan, S.Pd., M.Pd,
Badrul Ainy Dalimunthe, S.Pd., M.Si,
Yusmaidar Sepriani, S.Pd., M.Si,
Sahat Parulian Sitorus, S,T., M.Kom



BUKU AJAR
METODOLOGI PENELITIAN PERANCANGAN SISTEM INFORMASI DAN DATA SAINS

**Ibnu Rasyid Munthe, S.T., M.Kom, Marnis Nasution, S.Kom., M.Kom, Muhammad
Irwansyah Hasibuan, S.E., M.Si, Indra Gunawan, M.Kom, Eka Irawan, M. Kom,
Mila Nirmala Sari Hasibuan, S.Pd., M.Pd, Badrul Ainy Dalimunthe, S.Pd., M.Si,
Yusmaidar Sepriani, S.Pd., M.Si, Sahat Parulian Sitorus, S.T., M.Kom**

Editor :
Sahat Parulian Sitorus

Desain Cover :
Nurhasanah

Sumber :
<https://isbn.jndi.my.id/products/buku-ajar-metodologi-penelitian-perancangan-sistem-informasi-dan-data-sains>

Tata Letak :
Ida Royani Sitorus

Ukuran :
Jml hal judul 410, Jml hal isi naskah 409, Uk: 17x25 cm

ISBN :
978-634-04-5439-0

Cetakan Pertama :
11 November 2025

Hak Cipta 2025, Pada Penulis
Isi diluar tanggung jawab percetakan

Copyright © 2025 by PT.JNDI
All Right Reserved

Hak cipta dilindungi undang-undang
Dilarang keras menerjemahkan, memfotokopi, atau
memperbanyak sebagian atau seluruh isi buku ini
tanpa izin tertulis dari Penerbit.

PENERBIT PT. JASA NIAGA DIGITAL INDONESIA
Jl. H. Maulana Link. VI. Labuhan Ruku, Talawi, Batu Bara, Sumatera Utara 21254
Telp/Wa : 082361274081
<https://isbn.jndi.my.id>
E-mail:office@jndi.my.id

"Data is the new gold, and the right methodology is the key to transforming it into groundbreaking innovation."

PRAKATA

Puji syukur ke hadirat Tuhan Yang Maha Esa, berkat rahmat dan karunia-Nya buku ajar ini dapat hadir di tangan pembaca dengan judul **“Buku Ajar Metodologi Penelitian Perancangan Sistem Informasi dan Data Sains.”**

Buku ini dirancang bukan sekadar menjadi bacaan akademik, melainkan juga panduan praktis yang menjawab kebutuhan nyata di era digital. Pembaca akan diajak memahami metodologi penelitian secara sistematis, menguasai perancangan sistem informasi, serta menggali potensi data sains untuk menghasilkan solusi inovatif. Setiap bab disusun dengan bahasa yang lugas, dilengkapi studi kasus, latihan soal, hingga tugas kelompok sehingga pembelajaran lebih aplikatif dan bermakna.

Kami yakin buku ini layak menjadi rujukan utama bagi mahasiswa, dosen, peneliti, dan praktisi yang ingin memperdalam metodologi penelitian sekaligus menguasai keterampilan teknologi informasi modern. Semoga kehadirannya dapat memperkaya khasanah literatur dan menjadi kontribusi nyata dalam mencetak generasi unggul di bidang sistem informasi dan data sains.

KATA PENGANTAR

Alhamdulillah, puji syukur kami panjatkan ke hadirat Tuhan Yang Maha Esa atas rahmat dan karunia-Nya sehingga buku ajar ini dapat disusun dengan judul **“Buku Ajar Metodologi Penelitian Perancangan Sistem Informasi dan Data Sains.”**

Buku ini hadir untuk memberikan pemahaman mendalam mengenai metodologi penelitian yang berfokus pada integrasi antara perancangan sistem informasi dan data sains. Materi yang disajikan tidak hanya bersifat teoretis, tetapi juga dilengkapi contoh kasus, latihan soal, dan tugas kelompok yang relevan dengan kebutuhan akademik maupun dunia kerja. Dengan penyajian yang sistematis, buku ini diharapkan mampu membantu mahasiswa, dosen, dan peneliti dalam mengembangkan keterampilan analitis, berpikir kritis, serta menciptakan inovasi berbasis data.

Kami menyadari bahwa penyusunan buku ini masih jauh dari sempurna, sehingga kritik dan saran sangat kami harapkan demi penyempurnaan edisi berikutnya. Semoga buku ini memberi manfaat luas serta menjadi rujukan yang inspiratif bagi generasi pembelajar di bidang sistem informasi dan data sains.

Rantauprapat, Oktober 2025

Tim Penulis

DAFTAR ISI

PRAKATA	5
KATA PENGANTAR.....	6
DAFTAR ISI.....	7
Pengantar Perancangan Sistem Informasi.....	221
dan Data Science	221
Konsep dan Elemen Utama Perancangan Sistem Informasi.....	221
Peran Perancangan Sistem Informasi dalam Organisasi.....	22
Pengertian dan Lingkup Data Sains	23
Pemahaman Awal sebagai Pondasi	24
Siklus Data Sains (Data Science Lifecycle)	25
Relevansi di Era Digital.....	26
Soal Uraian.....	31
Tugas Kelompok.....	32
Metodologi Perancangan Sistem Informasi	35
Pengertian Metodologi Perancangan Sistem Informasi	35
Tujuan	36
Peran dalam SDLC.....	37
Jenis-Jenis Metodologi Perancangan Sistem Informasi.....	37
A. Structured System Analysis and Design (SSAD)	37
B. Object-Oriented Analysis and Design (OOAD).....	40
C. Prototyping	41
D. Metode Agile	43
E. Rapid Application Development (RAD).....	45

F. DevOps.....	46
G. Design Thinking.....	47
Tahapan dalam Metodologi Perancangan Sistem Informasi	49
A. Analisis Kebutuhan Pengguna dan Bisnis	49
B. Perancangan Sistem.....	49
C. Implementasi	49
D. Pengujian	50
E. Evaluasi dan Pemeliharaan	50
Teknik Evaluasi dalam Metodologi Perancangan Sistem Informasi.....	50
A. Evaluasi Kinerja	50
B. Evaluasi Kualitas.....	51
C. Evaluasi Keberlanjutan.....	51
Metodologi yang Relevan untuk Masa Depan	52
A. Agile dan DevOps.....	52
B. Integrasi Design Thinking.....	53
C. Metodologi Adaptif.....	53
Soal Uraian.....	54
Tugas Kelompok.....	55
Identifikasi Kebutuhan Perancangan Sistem Informasi.....	57
Pentingnya Identifikasi Kebutuhan	58
Metode Identifikasi Kebutuhan	59
A. Wawancara	59
B. Observasi	59
C. Dokumentasi.....	60
D. Kuesioner	60
Pencatatan Masalah pada Sistem Lama	60
Penerjemahan Kebutuhan Menjadi Fitur dan Fungsi.....	61

Hubungan dengan Desain Konseptual.....	61
Validasi dengan Pendekatan Berbasis Data	62
Tantangan dalam Identifikasi Kebutuhan	62
Relevansi Identifikasi Kebutuhan dalam Dunia Nyata	63
Soal Uraian.....	67
Tugas Kelompok.....	68
Studi Kelayakan Perancangan Sistem Informasi dan Data Sains	71
Tujuan Studi Kelayakan	72
Studi Kelayakan dalam Perancangan Sistem Informasi	72
Studi Kelayakan dalam Data Sains	73
Empat Aspek Utama Studi Kelayakan	73
A. Laporan Formal Studi Kelayakan	75
B. Pengambilan Keputusan Organisasi	75
C. Evaluasi Berbasis Data	76
Soal Uraian.....	81
Tugas Kelompok.....	82
Pemodelan Sistem dengan DFD dan ERD	85
Konsep Dasar Data Flow Diagram (DFD)	86
A. Komponen DFD	86
B. Tingkatan DFD.....	87
C. Spesifikasi Proses dan Kamus Data.....	90
Konsep Dasar Entity Relationship Diagram (ERD)	93
A. Pengertian ERD	93
B. Komponen ERD	93
C. Kardinalitas dan Opsionalitas	94
D. Penyusunan Struktur dan Bentuk Normal.....	95
Hubungan dan Kesesuaian DFD–ERD	95

Simbol dan Aturan Penamaan	96
Pemodelan sebagai Dasar Rancangan Basis Data	96
Contoh Penerapan DFD–ERD.....	97
A. Sistem Akademik Perguruan Tinggi	97
B. Sistem Perbankan	98
C. Sistem E-Commerce.....	100
Tantangan Umum dan Cara Menanganinya	101
Soal Uraian.....	102
Tugas Kelompok.....	103
Pemodelan dengan UML.....	109
Pentingnya UML dalam Pemodelan	110
Komponen Utama UML.....	110
Use Case Diagram	111
A. Komponen Utama.....	111
B. Kegunaan Use Case Diagram	112
C. Contoh Penerapan	112
Class Diagram	113
A. Komponen Utama.....	113
B. Kegunaan Class Diagram.....	113
C. Contoh Penerapan	114
Activity Diagram	114
A. Komponen Utama.....	115
B. Kegunaan Activity Diagram.....	115
C. Contoh Penerapan	115
Hubungan Antara Use Case, Class, dan Activity Diagram	117
Tantangan dalam Pemodelan UML.....	117
Soal Uraian.....	119

Tugas Kelompok.....	120
Desain Sistem Informasi	125
Desain Konseptual	126
Komponen Utama.....	126
Peran Diagram Konseptual	127
Validasi.....	127
Desain Logis	127
A. Detail Struktur Data	128
B. Proses Bisnis	128
Desain Fisik	128
A. Komponen Desain Fisik.....	129
B. Keamanan dan Pemeliharaan	129
C. Hasil Desain Fisik.....	129
<i>Prototyping</i> Sistem Informasi	130
A. Tujuan <i>Prototyping</i>	130
B. Jenis Prototype	130
C. Proses <i>Prototyping</i>	131
D. Keunggulan <i>Prototyping</i>	131
Integrasi Antar Tahap Desain.....	131
Tantangan dalam Desain Sistem Informasi.....	132
Soal Uraian.....	139
Tugas Kelompok.....	140
Model dan Tahapan Pengembangan	144
dan Pengujian Sistem Informasi	144
System Development Life Cycle (SDLC)	145
Model Waterfall.....	148
Model Agile.....	151

Model Pengembangan Sistem Lain	154
Model Spiral.....	154
V-Model	158
Prototype	161
Tahapan Pengembangan dan Pasangan Pengujian	167
A. Spesifikasi Kebutuhan (Requirements Specification)	167
B. Desain Sistem (<i>System Design</i>)	167
C. Desain Detail (<i>Detailed Design</i>)	167
D. Implementasi (<i>Implementation</i>)	168
Pengujian Sistem	168
Jenis Pengujian	168
Pengujian Unit Testing	168
A. Pentingnya Pengujian Unit.....	168
B. Studi Kasus Login ke Email	169
C. Analisa Pengujian Unit.....	170
Pengujian Integration Testing	170
A. Pentingnya Integration Testing	171
B. Contoh Studi Kasus Login ke Email	171
C. Analisa Integration Testing.....	172
Pengujian System Testing	173
A. Pentingnya System Testing.....	173
B. Studi Kasus Nyata Login ke Email	173
C. Analisa System Testing.....	174
Pengujian Acceptance Testing (UAT.....	175
A. Pentingnya UAT	175
B. Studi Kasus Login ke Email	176
C. Analisis UAT.....	177

Pengujian Regression Testing	178
A. Pentingnya Regression Testing.....	178
B. Studi Kasus Login ke Email	178
C. Analisis Regression Testing	179
Metode Pengujian Khusus.....	1800
Black Box Testing	1800
A. Prinsip Dasar	1800
B. Studi Kasus Aplikasi Login dengan Email	181
White Box Testing	182
A. Prinsip Dasar White Box Testing	183
B. Studi Kasus Aplikasi Login dengan Email	184
Grey Box Testing	186
A. Prinsip Dasar Grey Box Testing	187
B. Studi Kasus Aplikasi Login dengan Email	188
Soal Uraian.....	189
Tugas Kelompok.....	1911
Ujian Tengah Semester (UTS)	194
Contoh Soal Uraian	194
Contoh Soal Studi Kasus	195
Metodologi Data Sains.....	196
Konsep Dasar Metodologi Penelitian Data Sains.....	196
Karakteristik Utama	197
Tahapan Umum	199
A. Identifikasi masalah	199
B. Pengumpulan data.....	199
C. Preprocessing data.	200
D. Pemodelan.....	200

E. Evaluasi	200
F. Deployment dan pemantauan.....	201
Framework Penelitian Artificial Intelligence (AI)	20201
Tahapan Framework AI	202
A. Problem Formulation.....	202
B. Knowledge Representation.....	202
C. Algorithm Selection	202
D. Model Training.....	203
E. Evaluation	203
F. Deployment	203
Kelebihan dan Kekurangan	204
Framework Penelitian Machine Learning (ML)	205
Tahapan Framework ML	205
A. Data Collection.....	205
B. Data Cleaning & Preprocessing.....	206
C. Feature Engineering.....	206
D. Model Selection.....	207
E. Training & Validation.....	207
F. Testing.....	207
G. Deployment	207
H. Monitoring.....	208
Kelebihan dan Kekurangan	208
Framework Penelitian Deep Learning (DL).....	209
Tahapan Framework DL	209
A. Data Preparation.....	209
B. Model Architecture Design.....	210
C. Training with GPUs/TPUs.....	210

D. Regularization & Optimization.....	211
E. Evaluation	211
F. Deployment	211
Kelebihan dan Kekurangan	211
Perbandingan AI, ML, dan DL	213
A. Kompleksitas Data	213
B. Kebutuhan Data	213
C. Interpretabilitas	213
D. Komputasi	214
E. Aplikasi.....	214
Studi Kasus Penerapan	214
A. Bisnis	214
B. Kesehatan	223
C. Sosial	231
D. Teknologi.....	238
Soal Uraian.....	248
Tugas Kelompok.....	248
Machine Learning.....	250
Definisi dan Konsep Dasar	250
Tahap Pengumpulan Data.....	253
A. Pemilihan Algoritma	255
B. Proses Pelatihan Model	257
C. Evaluasi Model.....	259
D. Deployment Model.....	263
E. Pemeliharaan dan Pembaruan Model.....	263
Tantangan dalam <i>Machine Learning</i>	265
A. Kualitas Data	265

B. Kompleksitas Algoritma	265
C. Etika dan Privasi	265
D. Transparansi Model (<i>Black Box</i>).....	266
Soal Uraian.....	266
Tugas Kelompok.....	267
Supervised Learning	271
Data Terstruktur dan Tidak Terstruktur	272
A. Definisi Data Terstruktur	272
B. Definisi Data Tidak Terstruktur	273
C. Signifikansi Perbedaan Data	274
Pengertian Model Supervised Learning	274
Karakteristik Utama	275
A. Data Berlabel dengan Pasangan <i>Input-Output</i>	275
B. Tujuan Prediksi Nilai Kontinu maupun Kategori	276
C. Evaluasi Kinerja dengan Membandingkan Prediksi dan Keluaran Aktual.....	276
Jenis-Jenis Algoritma <i>Supervised Learning</i>	276
A. Regresi	276
B. Klasifikasi	278
Tahapan dalam Model <i>Supervised Learning</i>	284
A. Pengumpulan Data	284
B. Praproses Data.....	284
C. Pelatihan Model.....	285
D. Evaluasi Model.....	285
E. Deployment	285
Teknik Evaluasi di <i>Supervised Learning</i>	286
A. Regresi	286

B. Klasifikasi	288
C. Pentingnya Evaluasi	290
Algoritma <i>Supervised Learning</i> untuk AI Masa Depan	290
A. Deep Learning.....	290
B. Ensemble Learning	290
C. Hybrid Models	2911
D. Tren Mendatang	2911
Soal Uraian.....	2911
Tugas Kelompok.....	292
Unsupervised Learning	295
Pengertian Model Unsupervised Learning	296
Perbedaan dengan Supervised Learning	297
Contoh Aplikasi Nyata.....	298
Jenis-Jenis Algoritma untuk Model <i>Unsupervised Learning</i>	299
A. Clustering	299
B. Dimensionality Reduction.....	304
C. Association Rule Learning.....	308
Tahapan di Model Unsupervised Learning	311
A. Pengumpulan data.....	311
B. Preprocessing.....	312
C. Pemilihan algoritma.....	312
D. Pelatihan model.....	312
E. Evaluasi hasil	312
Teknik Evaluasi di Unsupervised Learning	313
A. Internal Evaluation Metrics.....	313
B. External Evaluation Metrics	317
C. Visualisasi Data	320

Jenis Algoritma <i>Unsupervised Learning</i> untuk AI di Masa Depan	322
A. Deep Autoencoder.....	322
B. Generative Adversarial Networks (GANs).....	322
C. Self-Organizing Maps (SOM).....	322
D. Graph-Based Clustering	322
Soal Uraian.....	323
Tugas Kelompok.....	324
Reinforcement Learning.....	328
Pengertian Model Reinforcement Learning	329
Perbedaan dengan Pendekatan Lain	330
A. Supervised learning	330
B. Unsupervised learning.....	331
C. Reinforcement learning	331
Contoh Aplikasi	332
Jenis-Jenis Algoritma pada <i>Reinforcement Learning</i>	337
A. Value-Based Methods.....	337
B. Policy-Based Methods	341
C. Actor-Critic Methods	344
Tahapan dalam Reinforcement Learning.....	349
A. Definisi Lingkungan (<i>Environment</i>)	349
B. Interaksi Agen dengan Lingkungan	349
C. Feedback dari Lingkungan	350
D. Update Policy	350
E. Evaluasi Agen	350
Contoh Tahapan RL.....	351
A. Pengamatan Awal (<i>State Observation</i>).....	351
B. Pemilihan Tindakan (<i>Action Selection</i>)	351

C. Respon Lingkungan (<i>Reward and New State</i>).....	351
D. Pembaruan Kebijakan (<i>Policy Update</i>)	352
E. Proses Iteratif Menuju Optimal	352
Teknik Evaluasi dalam <i>Reinforcement Learning</i>	353
A. Cumulative reward	353
B. Average reward.....	354
Convergence rate	355
Algoritma <i>Reinforcement Learning</i> untuk AI Masa Depan	358
A. Deep Reinforcement Learning (DRL)	358
B. Proximal Policy Optimization (PPO)	358
C. Deep Deterministic Policy Gradient (DDPG).....	358
D. AlphaZero-like Algorithms	359
Soal Uraian.....	359
Tugas Kelompok.....	360
Imbalance Data	363
Pengertian Imbalance Data	364
Contoh Kasus	365
A. Kesehatan	365
B. Keuangan	366
C. Keamanan Siber	366
Karakteristik	366
A. Distribusi Tidak Merata Antar Kelas	367
B. Bias Model ke Kelas Mayoritas	367
C. Keterbatasan Metrik Akurasi	367
Jenis-Jenis Masalah	368
A. Binary Class Imbalance	368
B. Multi-class Imbalance	369

C. Within-class Imbalance.....	371
D. Imbalance Dinamis	371
Dampak terhadap Model	371
A. Penurunan Akurasi Kelas Minoritas.....	371
B. Peningkatan <i>False Negative</i> dalam Medis.....	372
C. Menurunnya Kemampuan Generalisasi	372
D. Risiko <i>Overfitting</i> pada Mayoritas	372
Teknik Penanganan	374
A. Resampling Methods	375
B. Algorithmic Methods.....	378
C. Hybrid Methods.....	382
D. Advanced Approaches	384
Contoh Penerapan	392
A. Kesehatan	392
B. Keamanan Siber	393
C. Keuangan	394
Penerapan Imbalance Di Masa Depan	395
Soal Uraian.....	397
Tugas Kelompok.....	398
Ujian Akhir Semester	400
Contoh Soal Uraian	400
Contoh Soal Studi Kasus	401
DAFTAR PUSTAKA	402

Pengantar Perancangan Sistem Informasi dan Data Science

A. CPL

Mahasiswa memahami dasar perancangan sistem informasi dan Data Sains.

B. CPMK

Mahasiswa mampu menjelaskan definisi, tujuan, dan ruang lingkup.

C. Sub-CPMK

Menjelaskan peran perancangan sistem informasi dalam organisasi dan Data Sains di era digital.

Pemahaman tentang **perancangan sistem informasi** dan **data sains** merupakan fondasi penting bagi mahasiswa yang sedang mempelajari dunia teknologi dan manajemen informasi. Keduanya saling melengkapi dan menghadirkan kerangka berpikir yang lebih kuat untuk menghadapi tantangan era digital. Dalam bagian ini, akan dibahas secara terstruktur mengenai pengertian, elemen, peran, serta integrasi perancangan sistem informasi dan data sains, sekaligus memberikan gambaran mengenai siklus *data science* yang menjadi acuan penting dalam praktik profesional.

Konsep dan Elemen Utama Perancangan Sistem Informasi

Perancangan sistem informasi dapat dipahami sebagai upaya menyusun dan mengintegrasikan berbagai elemen untuk

menghasilkan informasi yang bermanfaat. Elemen tersebut terdiri dari **perangkat keras, perangkat lunak, data, jaringan, dan manusia**.

- a. **Perangkat keras** mencakup komputer, server, perangkat penyimpanan, serta infrastruktur lain yang menopang pemrosesan data.
- b. **Perangkat lunak** adalah instruksi yang memungkinkan perangkat keras bekerja, baik dalam bentuk aplikasi maupun sistem operasi.
- c. **Data** merupakan bahan mentah yang harus dikelola dengan benar agar menghasilkan informasi yang relevan.
- d. **Jaringan** menghubungkan berbagai komponen sehingga pertukaran informasi berjalan lancar.
- e. **Manusia** berperan sebagai pengendali dan pengambil keputusan dari hasil pengolahan informasi.

Dalam proses perancangan, semua elemen ini dipetakan dan dihubungkan agar membentuk suatu ekosistem yang terintegrasi. Mahasiswa perlu memahami bahwa setiap keputusan desain akan berdampak pada efektivitas sistem yang dibangun.

Peran Perancangan Sistem Informasi dalam Organisasi

Peran **perancangan sistem informasi** dalam organisasi tidak bisa dipandang sebelah mata. Pada sektor bisnis, perancangan sistem informasi mendukung kegiatan akuntansi, pemasaran, dan pengelolaan sumber daya manusia. Pada sektor pendidikan, rancangan sistem informasi digunakan untuk mengelola data

akademik, administrasi, dan layanan mahasiswa. Sementara itu, dalam sektor kesehatan, rancangan sistem informasi membantu pencatatan rekam medis pasien dan mempermudah proses diagnosis.

Yang perlu ditekankan adalah bagaimana perancangan sistem informasi tidak hanya menghasilkan alat bantu teknis, tetapi juga menghadirkan instrumen strategis untuk meningkatkan efisiensi dan efektivitas organisasi. Dengan rancangan yang baik dan terintegrasi, manajemen dapat mengambil **keputusan yang lebih cepat dan tepat** karena informasi yang diperoleh bersifat akurat, real-time, dan relevan dengan kebutuhan.

Pengertian dan Lingkup Data Sains

Istilah *data science* atau data sains merujuk pada suatu bidang yang berfokus pada pengumpulan, pengolahan, dan interpretasi data dalam berbagai bentuk. Data yang dikelola bisa berupa **data terstruktur** maupun **data tidak terstruktur**.

- a. **Data terstruktur** biasanya tersimpan dalam basis data dengan format tabel yang jelas.
- b. **Data tidak terstruktur** mencakup teks bebas, gambar, audio, video, dan media sosial.

Data sains memanfaatkan algoritma, komputasi modern, serta teknik statistik untuk menemukan pola tersembunyi dalam data. Hasilnya dapat digunakan untuk berbagai keperluan, mulai dari prediksi tren pasar hingga pengembangan layanan kesehatan berbasis kecerdasan buatan. Pemahaman tentang data sains

memungkinkan mahasiswa tidak hanya menjadi pengguna informasi, tetapi juga pengembang solusi berbasis data.

Pentingnya Integrasi Perancangan Sistem Informasi dan Data Sains

Integrasi antara **perancangan sistem informasi** dan **data sains** menghasilkan nilai tambah yang besar. Perancangan sistem informasi menyediakan struktur, alur, dan wadah untuk mengelola data, sedangkan data sains memberikan alat untuk menggali pengetahuan dari data tersebut. Contoh nyata integrasi dapat dilihat dalam layanan perbankan digital. Perancangan sistem informasi mencakup pencatatan transaksi keuangan secara real-time, sementara data sains membantu mendeteksi transaksi mencurigakan melalui analisis pola. Integrasi ini membuat layanan perbankan menjadi lebih aman dan andal.

Integrasi juga terlihat pada bidang pendidikan. Perancangan sistem informasi akademik memastikan pencatatan nilai mahasiswa dilakukan secara rapi, sementara data sains digunakan untuk memprediksi tingkat kelulusan atau risiko mahasiswa tidak menyelesaikan studi tepat waktu. Dengan demikian, institusi dapat melakukan intervensi lebih dini untuk meningkatkan kualitas pembelajaran.

Pemahaman Awal sebagai Pondasi

Pemahaman awal tentang **perancangan sistem informasi** dan **data sains** merupakan pondasi yang tidak bisa diabaikan. Mahasiswa

perlu menguasai konsep dasar agar mampu mengembangkan rancangan dan strategi pengelolaan informasi di masa depan. Dengan dasar yang kuat, mahasiswa dapat lebih mudah memahami topik lanjutan seperti perancangan basis data, pemodelan sistem, serta penerapan algoritma *machine learning*.

Pondasi ini juga memberikan kemampuan berpikir sistematis. Mahasiswa dilatih untuk melihat hubungan antar elemen dalam rancangan sistem dan bagaimana data dapat diolah menjadi informasi yang bernilai. Dengan kemampuan tersebut, mereka tidak hanya siap menghadapi tantangan akademik, tetapi juga siap bersaing dalam dunia kerja yang semakin kompetitif.

Siklus Data Sains (Data Science Lifecycle)

Siklus *data science* menjelaskan tahapan yang harus dilalui dalam mengolah data hingga menghasilkan informasi yang bermakna. Tahapan tersebut meliputi:

1. **Pengumpulan data**

Data diperoleh dari berbagai sumber, seperti basis data internal, sensor, aplikasi daring, maupun media sosial.

2. **Pembersihan data**

Pada tahap ini, data diperiksa untuk menghilangkan duplikasi, memperbaiki kesalahan, serta menangani ketidakkonsistenan.

3. **Eksplorasi data**

Tahap eksplorasi dilakukan untuk menemukan pola awal melalui visualisasi sederhana.

4. **Pemodelan data**

Algoritma digunakan untuk membangun model prediktif atau klasifikasi.

5. **Evaluasi model**

Hasil model diuji untuk mengukur tingkat akurasi dan keandalannya.

6. **Pelaporan hasil**

Informasi yang diperoleh disajikan dalam bentuk laporan yang mudah dipahami oleh pengambil keputusan.

Pemahaman siklus ini membuat mahasiswa mengerti bagaimana data diubah menjadi pengetahuan yang bermanfaat. Tahapan ini bersifat berulang sehingga model dapat terus diperbaiki sesuai kebutuhan.

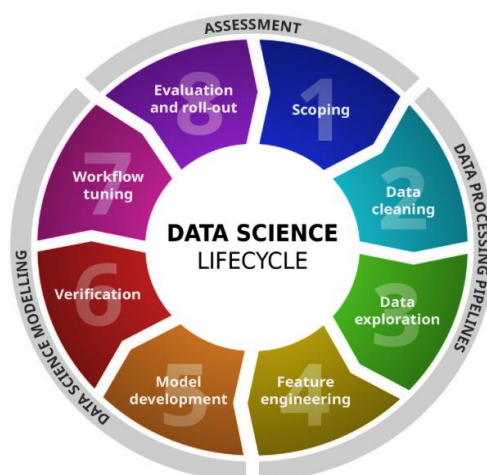
Relevansi di Era Digital

Era digital ditandai dengan volume data yang sangat besar. Tanpa **perancangan sistem informasi** yang terstruktur, data akan sulit dikelola. Tanpa **data sains**, informasi yang dihasilkan dari sistem yang dirancang tidak akan maksimal. Integrasi keduanya memungkinkan organisasi untuk memanfaatkan potensi data secara penuh.

Contoh paling nyata adalah pada perusahaan teknologi besar yang memanfaatkan data pelanggan untuk memberikan layanan personalisasi. Perancangan sistem informasi mereka memastikan pencatatan aktivitas pengguna, sedangkan data sains memproses data tersebut untuk menghasilkan rekomendasi yang relevan. Hal ini

membuktikan bahwa kombinasi dua bidang ini menjadi kunci dalam era digital.

Materi ini menekankan pentingnya pemahaman dasar mengenai **perancangan sistem informasi** dan **data sains**. Mahasiswa diajak untuk memahami elemen dasar rancangan sistem, mengenal lingkup data sains, melihat bagaimana keduanya terintegrasi, serta mempelajari siklus *data science*. Semua ini menjadi fondasi penting yang akan mendukung proses belajar di tahap berikutnya. Dengan penguasaan awal ini, mahasiswa tidak hanya siap menghadapi tantangan akademik, tetapi juga siap berkontribusi dalam pengembangan solusi inovatif di dunia nyata.



Gambar 1. Siklus Data Sains Terintegrasi

Gambar ini menampilkan **siklus data sains** yang terdiri dari delapan tahap utama yang saling berhubungan. Setiap tahap menggambarkan proses penting yang harus dilalui dalam mengelola data hingga menjadi informasi yang bermanfaat untuk pengambilan keputusan. Siklus ini relevan dengan pembahasan pada bagian

Materi karena memperlihatkan alur yang terstruktur mulai dari pengumpulan data hingga tahap evaluasi dan penerapan.

Tahap pertama adalah *scoping*, yaitu mendefinisikan ruang lingkup dan tujuan dari sebuah proyek berbasis data. Setelah itu, dilakukan *data cleaning* untuk memastikan data yang digunakan bebas dari kesalahan atau duplikasi. Tahap berikutnya adalah *data exploration*, di mana data dianalisis secara awal untuk menemukan pola, tren, atau anomali. Kemudian dilakukan *feature engineering* yang berfokus pada pemilihan serta pembuatan atribut yang relevan agar model yang dibangun lebih akurat.

Selanjutnya, masuk ke tahap *model development* yang mencakup pembangunan model prediktif atau klasifikasi dengan algoritma tertentu. Model yang dihasilkan perlu melewati tahap *verification* untuk memastikan performanya sesuai standar. Setelah itu, dilakukan *workflow tuning* untuk menyempurnakan kinerja keseluruhan proses, sehingga hasil yang diperoleh semakin optimal. Siklus ini ditutup dengan *evaluation and roll-out*, di mana hasil akhir dievaluasi, dilaporkan, dan diimplementasikan ke dalam sistem nyata.



Gambar 2. Siklus Perancangan Sistem Informasi

Gambar ini menampilkan **siklus perancangan sistem informasi** yang terdiri dari lima tahap utama, yaitu perencanaan, analisis, desain, implementasi, dan pemeliharaan. Setiap tahap memiliki peran yang penting untuk memastikan bahwa sistem informasi yang dirancang mampu mendukung kebutuhan organisasi secara efektif. Siklus ini sangat relevan dengan pembahasan pada bagian **Materi**, karena menunjukkan keterhubungan langsung antara rancangan sistem informasi dengan integrasi data sains dalam menghasilkan informasi yang berkualitas.

Tahap pertama adalah **perencanaan**, yang berfokus pada identifikasi kebutuhan organisasi serta menentukan tujuan utama dari sistem yang akan dikembangkan. Tahap ini melibatkan pengumpulan informasi awal untuk memastikan rancangan sistem sesuai dengan visi organisasi. Tahap berikutnya adalah **analisis**, di mana kebutuhan pengguna dipetakan secara detail, potensi masalah diidentifikasi, dan persyaratan sistem didefinisikan. Analisis ini menjadi dasar penting bagi tahap berikutnya, yaitu **desain**. Pada tahap desain, rancangan logis maupun fisik dari sistem dibuat, mencakup struktur basis data, alur proses, serta antarmuka pengguna.

Setelah desain selesai, tahap **implementasi** dilakukan untuk membangun sistem sesuai rancangan yang telah dibuat. Implementasi mencakup pemrograman, pengujian, serta integrasi dengan infrastruktur yang ada. Tahap terakhir adalah **pemeliharaan**, yang memastikan sistem tetap berjalan optimal melalui perbaikan bug, pembaruan, serta penyesuaian terhadap kebutuhan baru.

Kedua gambar yang ditampilkan, yaitu **Siklus Data Sains Terintegrasi** dan **Siklus Perancangan Sistem Informasi**, saling

melengkapi dalam membangun pemahaman komprehensif mengenai pengelolaan dan pemanfaatan data dalam organisasi modern. Hubungan keduanya erat karena masing-masing memberikan perspektif berbeda yang pada akhirnya mengarah pada tujuan yang sama, yakni menghasilkan sistem dan informasi yang bermanfaat serta dapat mendukung pengambilan keputusan.

Siklus Perancangan Sistem Informasi menitikberatkan pada tahapan pengembangan sistem, dimulai dari perencanaan, analisis, desain, implementasi, hingga pemeliharaan. Siklus ini memastikan bahwa sistem yang dibangun dapat memenuhi kebutuhan organisasi, memiliki rancangan yang jelas, dan tetap relevan melalui pemeliharaan berkelanjutan. Dalam konteks ini, sistem yang dirancang menjadi wadah yang terstruktur untuk mengelola data secara konsisten dan teratur.

Sementara itu, **Siklus Data Sains Terintegrasi** berfokus pada pemrosesan data yang tersimpan di dalam sistem. Siklus ini mencakup tahapan *scoping*, *data cleaning*, *data exploration*, *feature engineering*, pengembangan model, verifikasi, penyempurnaan alur kerja, hingga evaluasi dan penerapan. Tujuannya adalah mengubah data mentah menjadi wawasan yang memiliki nilai tambah bagi organisasi. Dengan demikian, data sains bertindak sebagai alat analitik yang melengkapi fungsi sistem informasi yang telah dirancang. Keterhubungan kedua siklus terlihat jelas ketika perancangan sistem informasi menyediakan struktur dan alur pengelolaan data, sedangkan siklus data sains mengolah data tersebut untuk menghasilkan pengetahuan baru

Soal Uraian

1. Jelaskan bagaimana perancangan sistem informasi dapat meningkatkan efisiensi operasional dalam sebuah organisasi.
2. Uraikan lima peran penting manusia dalam perancangan sistem informasi.
3. Apa perbedaan utama antara *data science lifecycle* dengan siklus perancangan sistem informasi?
4. Berikan contoh nyata bagaimana data terstruktur digunakan dalam pengambilan keputusan di sektor kesehatan.
5. Bagaimana data tidak terstruktur seperti teks media sosial dapat dimanfaatkan melalui pendekatan data sains?
6. Jelaskan hubungan antara tahap analisis pada perancangan sistem informasi dengan tahap eksplorasi dalam siklus data sains.
7. Sebutkan manfaat pemeliharaan sistem informasi bagi keberlangsungan organisasi dalam era digital.
8. Berikan contoh kasus sederhana penerapan *machine learning* yang mendukung perancangan sistem informasi akademik.
9. Uraikan perbedaan mendasar antara hasil keluaran perancangan sistem informasi dengan keluaran dari analisis data sains.
10. Mengapa integrasi antara perancangan sistem informasi dan data sains dianggap sebagai pondasi inovasi digital modern? Jelaskan secara argumentatif.

Tugas Kelompok

Perancangan Sistem Informasi Akademik di Perguruan Tinggi

Deskripsi Tugas

Mahasiswa diminta untuk bekerja dalam kelompok yang terdiri dari 4-5 orang dan mendiskusikan contoh nyata mengenai integrasi **perancangan sistem informasi** dalam pengelolaan akademik di perguruan tinggi. Fokus utama dari studi kasus ini adalah bagaimana rancangan sistem informasi dapat membantu menyelesaikan permasalahan administrasi akademik, meningkatkan efisiensi kerja, dan mendukung pengambilan keputusan oleh pihak manajemen.

Petunjuk Tugas

1. **Identifikasi Masalah**

Tentukan permasalahan utama dalam administrasi akademik, misalnya kesalahan pengisian KRS, keterlambatan penginputan nilai, atau kesulitan mengakses jadwal kuliah.

2. **Rancangan Sistem**

Susun rancangan sistem informasi akademik yang dapat mengatasi masalah tersebut. Jelaskan komponen utamanya, seperti basis data mahasiswa, antarmuka pengguna, serta fitur integrasi antarbagian akademik.

3. **Simulasi Alur Kerja**

Buat alur kerja sederhana yang memperlihatkan proses mulai dari input data hingga keluaran informasi.

4. **Kesimpulan Diskusi**

Sampaikan manfaat nyata dari rancangan sistem ini terhadap peningkatan efisiensi akademik.

Output yang Diharapkan

- a. Laporan kelompok sepanjang 5-7 halaman.
- b. Diagram alur untuk menjelaskan rancangan sistem.
- c. Paparan hasil diskusi di kelas.

Studi Kasus 2: Integrasi Data Sains

Pemanfaatan Data Sains untuk Deteksi Dini Penyakit

Deskripsi Tugas

Mahasiswa diminta untuk bekerja dalam kelompok yang terdiri dari 4-5 orang dan mendiskusikan contoh nyata mengenai integrasi **data sains** dalam layanan kesehatan digital. Fokus utama dari studi kasus ini adalah bagaimana data sains dapat membantu meningkatkan akurasi diagnosis, mendukung deteksi dini penyakit, serta mempercepat proses pengambilan keputusan oleh tenaga medis.

Petunjuk Tugas

1. Identifikasi Masalah

Tentukan permasalahan utama dalam layanan kesehatan, misalnya keterlambatan diagnosis, kesalahan pencatatan data pasien, atau keterbatasan tenaga medis.

2. Penerapan Data Sains

Jelaskan bagaimana data sains dapat digunakan, misalnya melalui pemodelan prediktif, analisis rekam medis, atau pemanfaatan data tidak terstruktur dari citra medis.

3. Simulasi Alur Kerja

Buat alur kerja sederhana yang memperlihatkan hubungan antara data pasien, proses analisis data sains, dan hasil keluaran berupa rekomendasi medis.

4. **Kesimpulan Diskusi**

Sampaikan manfaat nyata dari integrasi data sains dalam meningkatkan layanan kesehatan.

Output yang Diharapkan

- a. Laporan kelompok sepanjang 5-7 halaman.
- b. Diagram alur atau ilustrasi sederhana untuk menjelaskan integrasi data sains.
- c. Paparan hasil diskusi di kelas.

Metodologi Perancangan Sistem Informasi

A. CPL

Memahami metodologi perancangan sistem informasi dengan integritas.

B. CPMK

Membandingkan metodologi perancangan sistem informasi

C. Sub-CPMK

Menentukan metodologi perancangan sistem informasi yang tepat.

Metodologi perancangan sistem informasi merupakan kerangka sistematis yang digunakan untuk memastikan proses pengembangan sistem berjalan dengan terstruktur, terarah, dan memiliki landasan akademik serta praktik yang kuat. Dalam dunia modern yang penuh dinamika, metodologi menjadi pedoman penting agar pengembangan sistem tidak hanya efisien, tetapi juga relevan dengan kebutuhan pengguna dan perkembangan teknologi terkini. Penjelasan dalam bab ini akan menguraikan secara komprehensif pengertian metodologi, jenis-jenis metodologi yang banyak digunakan, tahapan dalam metodologi, teknik evaluasi, hingga relevansi metodologi kontemporer bagi masa depan.

Pengertian Metodologi Perancangan Sistem Informasi

Metodologi perancangan sistem informasi adalah *framework* yang memberikan panduan terstruktur dalam setiap fase

pengembangan sistem. Melalui metodologi, tim pengembang dapat mengurangi ketidakpastian, meningkatkan kualitas desain, serta menyelaraskan sistem dengan kebutuhan pengguna.

Tujuan

Tujuan metodologi dalam perancangan sistem informasi merupakan landasan penting yang membimbing keseluruhan proses pengembangan. Tujuan ini tidak hanya bersifat teknis, tetapi juga menyangkut aspek manajerial dan strategis untuk memastikan bahwa sistem yang dihasilkan sesuai dengan kebutuhan organisasi.

1. Arah yang Jelas

Metodologi memberikan arah yang jelas dalam setiap tahapan pengembangan. Dengan adanya pedoman terstruktur, tim dapat menghindari kebingungan, mempercepat proses, dan meningkatkan efisiensi kerja.

2. Konsistensi Antar Tahapan

Salah satu peran utama metodologi adalah menjamin konsistensi hasil di setiap tahap. Hal ini penting untuk menjaga keterpaduan sistem sehingga setiap komponen saling mendukung.

3. Pengelolaan Risiko

Metodologi juga membantu dalam mengurangi risiko kegagalan proyek. Dengan prosedur yang sistematis, potensi kesalahan dapat dideteksi lebih awal dan segera ditangani.

4. Dokumentasi dan Evaluasi

Selain itu, metodologi menyediakan dokumentasi lengkap sebagai dasar untuk evaluasi. Dokumentasi ini penting untuk pembelajaran, pengendalian kualitas, serta pengembangan sistem di masa depan.

Peran dalam SDLC

Metodologi memiliki hubungan yang sangat erat dengan *System Development Life Cycle (SDLC)*. *SDLC* berperan sebagai **siklus hidup** yang menguraikan tahapan pengembangan sistem dari awal hingga akhir, sedangkan metodologi berfungsi sebagai **panduan teknis dan praktis** yang menjabarkan cara penerapan pada setiap tahap. Dengan demikian, keduanya saling melengkapi untuk mencapai hasil yang optimal. Tahapan dalam *SDLC* meliputi **analisis kebutuhan, desain sistem, implementasi, pengujian, dan pemeliharaan**. Pada setiap tahap, metodologi membantu:

1. Menyediakan prosedur **sistematis** dan **terarah**.
2. Menjamin **konsistensi** antar hasil.
3. Mengurangi risiko **kesalahan teknis** atau miskomunikasi.
4. Mendukung penyusunan **dokumentasi** yang akurat sebagai referensi.

Jenis-Jenis Metodologi Perancangan Sistem Informasi

A. Structured System Analysis and Design (SSAD)

Structured System Analysis and Design (SSAD) merupakan salah satu metodologi klasik dalam perancangan sistem informasi yang hingga kini masih sering dijadikan acuan dalam dunia akademik maupun praktik profesional. Metodologi ini menekankan pada **alur**

proses serta pemodelan menggunakan *Data Flow Diagram (DFD)* untuk menggambarkan bagaimana data mengalir dan berinteraksi di dalam sistem.

Kelebihan SSAD

Keunggulan utama dari SSAD adalah sifatnya yang **jelas, terstruktur**, dan relatif **mudah dipahami** oleh pengembang maupun pengguna. Dengan adanya model yang sistematis, komunikasi antar anggota tim dapat lebih terarah, sehingga meminimalisasi kesalahpahaman. Selain itu, representasi grafis melalui *DFD* membuat hubungan antar proses dan data lebih transparan.

Keterbatasan SSAD

Namun, SSAD juga memiliki kelemahan, terutama dalam hal **fleksibilitas**. Metodologi ini cenderung kurang adaptif ketika menghadapi perubahan kebutuhan bisnis yang dinamis. Hal ini karena rancangan yang sudah ditetapkan sejak awal akan sulit untuk diubah tanpa menimbulkan dampak pada tahapan lainnya.

Contoh Penerapan

1. Sistem Informasi Akademik

a) Context diagram

Entitas luar **Mahasiswa, Dosen, BAAK** berinteraksi dengan proses **Sistem Akademik**.

b) Level-0

Proses utama **Registrasi, Kelola Kurikulum, Penjadwalan**; *data store* **Mahasiswa, matakuliah, Jadwal**.

c) **Aliran data**

Formulir registrasi → **Registrasi**; hasil penjadwalan → **Mahasiswa/Dosen**; pembaruan kurikulum → **matakuliah**.

d) **Manfaat DFD**

Memperjelas batas sistem, tanggung jawab proses, dan titik kontrol validasi data.

2. **Sistem Perbankan Tradisional**

a) **Context diagram**

Entitas **Nasabah, Teller/ATM, Otoritas** berinteraksi dengan **Sistem Perbankan**.

b) **Level-0**

Proses **Setoran, Penarikan, Transfer, Rekonsiliasi**; *data store* **Rekening, Transaksi, Nasabah**.

c) **Aliran data**

Instruksi transaksi dari **Nasabah** → proses; output berupa **bukti transaksi** dan **saldo terkini**.

d) **Manfaat DFD**

Memastikan **konsistensi** arus informasi, membantu audit trail, dan mendukung perancangan kontrol **keamanan**.

Dengan mengaitkan contoh penerapan langsung ke artefak *DFD* (mulai dari *context diagram* hingga *level-0*), pembaca memperoleh gambaran **terstruktur** tentang entitas, proses, *data store*, dan **aliran data** sehingga dokumentasi lebih **komprehensif** dan mudah dievaluasi.

B. Object-Oriented Analysis and Design (OOAD)

Object-Oriented Analysis and Design (OOAD) merupakan metodologi perancangan sistem informasi yang memanfaatkan konsep **objek** dan **class diagram** untuk memodelkan sistem. Metode ini selaras dengan paradigma pemrograman modern karena berorientasi pada objek yang merepresentasikan entitas nyata di dunia nyata. Keunggulan utama dari OOAD terletak pada kemampuannya mendukung **reuse kode** sehingga pengembangan sistem dapat lebih efisien dan berkelanjutan. Namun demikian, kelemahan mendasar dari pendekatan ini adalah **kompleksitas awal yang relatif tinggi**, terutama bagi pengembang yang belum terbiasa dengan konsep pemodelan berorientasi objek. Kompleksitas ini sering kali muncul pada tahap perancangan, di mana konsistensi antar objek dan relasi harus didefinisikan secara jelas agar sistem tidak kehilangan arah.

Pengembangan

OOAD memberikan kerangka kerja yang kuat dalam membangun sistem yang fleksibel, skalabel, dan mudah dipelihara. Dengan adanya **class diagram**, hubungan antar komponen sistem dapat tergambar lebih realistis sesuai dengan kebutuhan bisnis. Selain itu, *use case diagram* turut memperjelas bagaimana aktor berinteraksi dengan sistem sehingga mendukung pemahaman lintas tim, baik teknis maupun non-teknis. Dalam praktiknya, OOAD sangat relevan digunakan pada proyek-proyek berskala besar, seperti pengembangan perangkat lunak enterprise atau sistem informasi yang melibatkan banyak modul.

Penerapan OOAD juga mendukung prinsip **encapsulation**, **inheritance**, dan **polymorphism**, yang memperkaya desain sekaligus mempermudah pengembangan lanjutan. Meski membutuhkan waktu lebih lama dalam tahap perancangan awal, hasil akhir dari OOAD cenderung lebih terstruktur dan siap menghadapi perubahan kebutuhan di masa depan. Dengan demikian, OOAD menjadi salah satu metodologi yang penting untuk dipahami dan diaplikasikan dalam dunia modern pengembangan sistem informasi.

Contoh Penerapan

Sebagai contoh nyata, penerapan OOAD dapat dilihat pada pengembangan **Sistem Manajemen Perpustakaan Digital**. Dalam sistem ini, setiap entitas seperti **Buku**, **Anggota**, dan **Petugas** direpresentasikan sebagai objek dengan atribut dan metode tertentu. Misalnya, objek *Buku* memiliki atribut judul, pengarang, dan status ketersediaan, serta metode untuk *peminjaman* atau *pengembalian*. Melalui *class diagram*, hubungan antar objek dapat dipetakan, seperti relasi antara *Anggota* dengan *Buku* melalui proses peminjaman. Dengan desain berorientasi objek, sistem ini lebih mudah dikembangkan ulang ketika fitur baru ditambahkan, misalnya integrasi dengan aplikasi mobile atau fitur pencarian cerdas.

C. Prototyping

Prototyping merupakan metodologi pengembangan sistem yang berfokus pada pembuatan **purwarupa (prototype)** sebagai representasi awal dari sistem yang akan dibangun. Tujuan utama dari

pendekatan ini adalah untuk memberikan gambaran nyata kepada pengguna sehingga mereka dapat mencoba, menilai, dan memberikan umpan balik sebelum sistem final dikembangkan sepenuhnya. Dengan demikian, *prototyping* memungkinkan adanya komunikasi yang lebih efektif antara pengembang dan pengguna. Kelebihan dari metode ini adalah kemampuannya menghasilkan **umpan balik cepat** melalui pendekatan iteratif, sehingga kesalahan desain dapat diminimalisasi sejak dini. Namun, kelemahan yang sering muncul adalah kebutuhan akan **sumber daya tambahan**, baik dari sisi waktu maupun biaya, terutama karena revisi yang harus dilakukan secara berulang.

Pengembangan

Metodologi *prototyping* sangat bermanfaat dalam proyek yang memiliki tingkat ketidakpastian tinggi atau ketika kebutuhan pengguna belum terdefinisi dengan jelas. Dengan adanya **iterasi**, setiap perubahan yang diusulkan dapat langsung diuji pada versi purwarupa berikutnya. Hal ini membuat pengembang lebih mudah menyesuaikan rancangan sesuai dengan kebutuhan nyata pengguna. Selain itu, *prototyping* dapat meningkatkan **kepuasan pengguna**, karena mereka merasa dilibatkan secara aktif dalam proses pengembangan.

Dalam konteks praktis, *prototyping* sering digunakan pada pengembangan **aplikasi perangkat lunak interaktif**, seperti sistem e-commerce, aplikasi mobile, maupun sistem manajemen berbasis web. Sebagai contoh, dalam pengembangan aplikasi belanja online, pengguna dapat mencoba versi awal antarmuka untuk melakukan

simulasi pembelian, memberikan masukan terkait navigasi, maupun desain tampilan. Melalui mekanisme ini, sistem akhir yang dihasilkan akan lebih **user-friendly**, relevan dengan kebutuhan pasar, dan memiliki kualitas yang lebih baik.

Contoh Penerapan

Contoh penerapan nyata *prototyping* dapat dilihat pada pengembangan **aplikasi layanan perbankan digital**. Pada tahap awal, tim pengembang membuat purwarupa sederhana berupa antarmuka login, menu utama, dan fitur dasar seperti cek saldo atau transfer dana. Purwarupa ini kemudian diuji langsung oleh calon pengguna untuk memberikan masukan terkait kemudahan penggunaan, keamanan tampilan, dan kecepatan akses. Berdasarkan umpan balik tersebut, pengembang melakukan revisi berulang hingga menghasilkan sistem yang aman, intuitif, dan sesuai dengan kebutuhan nasabah. Dengan cara ini, *prototyping* terbukti mempercepat tercapainya desain sistem yang lebih matang dan berkualitas.

D. Metode Agile

Agile Methodology adalah pendekatan pengembangan sistem yang menekankan **fleksibilitas, kolaborasi tim**, serta penggunaan **iterasi singkat** dalam setiap tahap pengerjaan. Metodologi ini muncul sebagai respons terhadap keterbatasan metode tradisional yang kaku dalam menghadapi perubahan kebutuhan. Keunggulan utama *Agile* adalah kemampuannya untuk **responsif terhadap**

perubahan, sehingga sangat sesuai diterapkan pada proyek yang dinamis dan memiliki tingkat ketidakpastian tinggi. Namun, penerapan *Agile* juga memiliki tantangan, seperti tuntutan akan **keterlibatan pengguna yang tinggi** dan kebutuhan akan **disiplin komunikasi** yang konsisten dalam tim.

Pengembangan

Dalam praktiknya, *Agile* memanfaatkan kerangka kerja seperti *Scrum* atau *Kanban* untuk mengatur alur kerja tim. Melalui iterasi singkat yang disebut *sprint*, tim dapat menghasilkan **incremental product** yang siap diuji dan dievaluasi oleh pengguna. Hal ini memungkinkan umpan balik diperoleh lebih cepat, sehingga perbaikan dapat segera dilakukan tanpa menunggu siklus pengembangan panjang.

Contoh Penerapan

Sebagai contoh penerapan nyata, *Agile* banyak digunakan dalam pengembangan **platform e-learning**. Fitur-fitur seperti sistem kuis interaktif, forum diskusi, dan modul pembelajaran dapat dirilis secara bertahap melalui *sprint*. Setiap kali fitur baru diuji, pengguna memberikan masukan terkait kemudahan akses maupun desain antarmuka. Dengan demikian, pengembang dapat segera menyesuaikan produk sesuai kebutuhan aktual, menghasilkan platform yang lebih **user-friendly**, relevan, dan terus berkembang sejalan dengan kebutuhan pengguna.

E. Rapid Application Development (RAD)

Rapid Application Development (RAD) adalah metodologi pengembangan perangkat lunak yang menitikberatkan pada **kecepatan pengembangan** melalui pemanfaatan perangkat bantu otomatisasi, prototyping cepat, dan keterlibatan intensif pengguna. Metode ini dirancang untuk mempercepat siklus hidup pengembangan sehingga produk dapat dihasilkan dalam waktu yang lebih singkat dibandingkan dengan pendekatan tradisional. Kelebihan utama RAD adalah **percepatan waktu produksi**, yang memungkinkan organisasi merespons kebutuhan bisnis dengan cepat. Namun, kelemahan yang sering muncul adalah risiko **dokumentasi yang kurang lengkap**, karena fokus lebih besar diberikan pada kecepatan implementasi daripada detail administratif.

Pengembangan

Pendekatan RAD sangat sesuai digunakan pada proyek yang memiliki batas waktu ketat namun membutuhkan hasil yang dapat segera digunakan. Dengan adanya **tools otomatisasi**, tim pengembang dapat mempercepat pembuatan antarmuka pengguna, integrasi modul, maupun pengujian. Keterlibatan pengguna secara aktif dalam setiap iterasi membantu memastikan bahwa produk akhir sesuai dengan kebutuhan dan mudah diadopsi.

Contoh Penerapan

Sebagai contoh penerapan nyata, RAD sering digunakan dalam pengembangan **sistem informasi rumah sakit**. Melalui RAD, tim pengembang dapat dengan cepat menghasilkan purwarupa

untuk modul registrasi pasien, pencatatan rekam medis, atau sistem penjadwalan dokter. Purwarupa ini diuji langsung oleh tenaga medis dan staf administrasi, yang kemudian memberikan umpan balik untuk perbaikan segera. Dengan pendekatan ini, rumah sakit dapat memperoleh sistem yang fungsional dalam waktu singkat, meskipun dokumentasi formal mungkin lebih minimal dibandingkan metodologi tradisional.

F. DevOps

DevOps adalah pendekatan yang mengintegrasikan proses **pengembangan (development)** dan **operasional (operations)** dalam satu alur kerja terpadu. Prinsip utama yang diusung adalah *continuous integration* (CI) dan *continuous delivery* (CD), yang memastikan kode baru dapat diuji, diintegrasikan, dan dirilis dengan cepat serta aman. Keunggulan dari *DevOps* adalah kemampuannya menjaga **keberlanjutan sistem** melalui otomatisasi, kolaborasi lintas tim, dan monitoring intensif. Metode ini sangat relevan diterapkan pada sistem berskala besar yang membutuhkan stabilitas sekaligus kecepatan dalam merespons perubahan. Namun, tantangan yang muncul adalah kebutuhan akan **budaya kerja kolaboratif** yang kuat serta kesiapan infrastruktur teknologi yang memadai.

Pengembangan

Dengan *DevOps*, siklus pengembangan perangkat lunak menjadi lebih singkat, transparan, dan terukur. Praktik seperti *automated testing*, *infrastructure as code*, serta penggunaan pipeline CI/CD

membantu organisasi menghasilkan produk dengan kualitas tinggi. Selain itu, monitoring berkelanjutan memungkinkan tim untuk segera mendeteksi masalah dan melakukan perbaikan proaktif.

Contoh Penerapan

Contoh nyata penerapan *DevOps* dapat ditemukan pada pengembangan **platform layanan streaming**. Setiap kali fitur baru, seperti sistem rekomendasi film atau optimasi performa video, selesai dikembangkan, fitur tersebut langsung melewati pipeline CI/CD untuk diuji dan dirilis ke server produksi. Tim operasi secara simultan memantau performa sistem melalui dashboard monitoring. Dengan cara ini, platform dapat menghadirkan pembaruan secara cepat, menjaga kualitas layanan tetap tinggi, sekaligus memastikan pengalaman pengguna berjalan lancar.

G. Design Thinking

Design Thinking adalah pendekatan inovatif dalam pengembangan sistem yang menempatkan **pengguna sebagai pusat desain**. Metodologi ini menekankan pentingnya **empati, eksplorasi masalah, ideasi, prototyping**, dan **pengujian** sebagai tahapan utama. Dengan berfokus pada pengalaman pengguna (*user experience*), *Design Thinking* mendorong solusi yang tidak hanya fungsional tetapi juga relevan dengan kebutuhan nyata. Keunggulannya terletak pada kemampuannya memperkuat **inovasi** sekaligus meningkatkan kepuasan pengguna. Namun, penerapan

metode ini menuntut waktu, keterlibatan lintas disiplin, serta keterbukaan terhadap proses iteratif yang panjang.

Pengembangan

Pendekatan *Design Thinking* sangat bermanfaat dalam proyek yang mengutamakan pengalaman pengguna, terutama ketika kebutuhan belum jelas sejak awal. Dengan tahapan yang sistematis, pengembang dapat menggali akar masalah, menghasilkan ide kreatif, dan menguji solusi secara berulang. Hal ini memungkinkan terciptanya produk yang lebih adaptif, relevan, dan memiliki nilai tambah bagi pengguna.

Contoh Penerapan

Sebagai contoh nyata, *Design Thinking* banyak digunakan dalam pengembangan **aplikasi transportasi online**. Tim pengembang memulai dengan mengidentifikasi kesulitan pengguna dalam mencari kendaraan, menghitung tarif, dan memantau perjalanan. Dari hasil empati dan wawancara, dibuatlah purwarupa fitur *real-time tracking* dan sistem estimasi biaya yang transparan. Setelah melalui pengujian berulang, fitur ini terbukti meningkatkan rasa aman sekaligus kenyamanan pengguna. Dengan cara ini, *Design Thinking* berhasil menghadirkan inovasi yang berorientasi pada kebutuhan pengguna.

Tahapan dalam Metodologi Perancangan Sistem Informasi

A. Analisis Kebutuhan Pengguna dan Bisnis

Tahap **analisis kebutuhan** merupakan fondasi dalam perancangan sistem informasi. Proses ini bertujuan untuk mengidentifikasi kebutuhan pengguna maupun organisasi melalui berbagai teknik, seperti wawancara, observasi, dan analisis dokumen. Hasil dari tahap ini berupa **spesifikasi fungsional** yang jelas, sehingga sistem yang dikembangkan dapat selaras dengan tujuan bisnis dan mendukung efisiensi operasional.

B. Perancangan Sistem

Tahap **perancangan sistem** berfokus pada penyusunan blueprint sistem secara menyeluruh. Aspek yang dirancang meliputi **arsitektur sistem, desain basis data, pemodelan proses bisnis, dan antarmuka pengguna**. Untuk memvisualisasikan rancangan, digunakan alat bantu seperti *Unified Modeling Language (UML)*, *Entity Relationship Diagram (ERD)*, dan *Data Flow Diagram (DFD)*. Perancangan yang baik memastikan sistem memiliki struktur yang terukur, fleksibel, dan mudah dikembangkan lebih lanjut.

C. Implementasi

Tahap **implementasi** mencakup proses penerjemahan rancangan ke dalam kode program (*coding*), integrasi antar modul, serta penerapan sistem di lingkungan operasional. Keberhasilan tahap ini bergantung pada kualitas kode, penggunaan standar pemrograman, serta dokumentasi yang lengkap.

D. Pengujian

Tahap **pengujian** dilakukan secara bertahap, mulai dari *unit testing*, *integration testing*, *system testing*, hingga *usability testing*. Tujuannya adalah untuk memastikan bahwa sistem berfungsi sesuai kebutuhan, memiliki performa optimal, serta bebas dari kesalahan yang dapat mengganggu operasional.

E. Evaluasi dan Pemeliharaan

Tahap terakhir adalah **evaluasi dan pemeliharaan**. Evaluasi dilakukan untuk mengukur efektivitas sistem dalam mendukung tujuan bisnis. Sementara itu, pemeliharaan mencakup perbaikan bug, penyesuaian terhadap kebutuhan baru, serta peningkatan performa sistem secara berkelanjutan. Tahapan ini penting untuk menjamin keberlanjutan dan relevansi sistem dalam jangka panjang.

Teknik Evaluasi dalam Metodologi Perancangan Sistem Informasi

A. Evaluasi Kinerja

Tahap **evaluasi kinerja** berfokus pada pengukuran kemampuan sistem dalam mendukung proses bisnis secara efektif. Teknik yang umum digunakan antara lain *performance testing*, yang menguji kecepatan sistem dalam menangani transaksi, serta *stress testing*, yang mengukur batas kapasitas sistem ketika dihadapkan pada beban kerja tinggi. Melalui pengujian ini, organisasi dapat mengetahui apakah sistem mampu menjaga stabilitas, responsivitas, dan reliabilitas meskipun dalam kondisi ekstrem.

Sebagai contoh pada sistem perbankan digital dilakukan *stress testing* dengan mensimulasikan ribuan transaksi simultan untuk

memastikan layanan tetap berjalan lancar saat terjadi lonjakan aktivitas nasabah, seperti pada periode pembayaran gaji bulanan atau saat promosi besar-besaran berlangsung.

B. Evaluasi Kualitas

Evaluasi kualitas menekankan pada pengalaman pengguna dalam berinteraksi dengan sistem. Salah satu teknik penting adalah *usability testing*, yaitu pengujian untuk menilai kemudahan penggunaan, navigasi, serta desain antarmuka. Selain itu, dilakukan *User Acceptance Test (UAT)* untuk mengukur tingkat kepuasan pengguna dan menilai apakah sistem sudah memenuhi kebutuhan yang dirumuskan pada tahap analisis. Kualitas sistem yang tinggi tidak hanya meningkatkan produktivitas, tetapi juga mendorong penerimaan teknologi oleh pengguna.

Sebagai contoh pada aplikasi e-commerce dilakukan *usability testing* dengan melibatkan pengguna untuk menilai kemudahan dalam mencari produk, melakukan checkout, dan menggunakan metode pembayaran digital, sehingga hasilnya dapat digunakan untuk memperbaiki desain antarmuka dan meningkatkan kenyamanan pengguna.

C. Evaluasi Keberlanjutan

Tahap **evaluasi keberlanjutan** melibatkan analisis menyeluruh terhadap faktor biaya dan manfaat melalui *cost-benefit analysis*. Selain itu, evaluasi mencakup pengukuran tingkat kepuasan pengguna dan efektivitas operasional dalam jangka panjang. Proses

ini bertujuan untuk menentukan kelayakan sistem agar tetap relevan, efisien, dan adaptif terhadap perubahan kebutuhan bisnis maupun perkembangan teknologi.

Sebagai contoh pada implementasi sistem manajemen rumah sakit dilakukan analisis biaya-manfaat untuk menilai apakah investasi pada perangkat keras, perangkat lunak, dan pelatihan staf memberikan manfaat lebih besar dibandingkan biaya, serta memastikan sistem mampu digunakan secara berkelanjutan untuk mendukung layanan kesehatan jangka panjang.

Metodologi yang Relevan untuk Masa Depan

A. Agile dan DevOps

Kombinasi **Agile** dan **DevOps** semakin populer karena mampu mendukung pengembangan berkelanjutan melalui iterasi cepat, pengiriman berkesinambungan, serta monitoring otomatis. *Agile* menekankan pada fleksibilitas dalam merespons perubahan kebutuhan, sementara *DevOps* menekankan pada kolaborasi antara tim pengembang dan operasional. Integrasi keduanya memungkinkan organisasi untuk menghasilkan sistem yang **responsif, andal, dan efisien**.

Alasan logisnya, pendekatan ini membantu mengurangi risiko kegagalan proyek karena setiap perubahan dapat segera diuji dan diintegrasikan tanpa menunggu siklus panjang. Selain itu, otomatisasi dalam *DevOps* mengurangi kesalahan manual, mempercepat waktu rilis, serta menekan biaya operasional. Sebagai contoh, perusahaan rintisan teknologi sering menggunakan pendekatan ini untuk

mempercepat perilisan fitur baru sekaligus menjaga kualitas layanan, sehingga mampu bersaing di pasar yang bergerak cepat.

B. Integrasi Design Thinking

Design Thinking berperan penting dalam merancang sistem informasi yang berfokus pada kebutuhan manusia (*human-centered approach*). Pendekatan ini menekankan pada kreativitas, empati terhadap pengguna, serta iterasi solusi yang inovatif. Dengan mengintegrasikan *Design Thinking* ke dalam metodologi pengembangan, organisasi dapat menghasilkan sistem yang tidak hanya berfungsi dengan baik, tetapi juga memberikan **pengalaman pengguna yang optimal**.

Alasan logisnya, pendekatan ini mampu menurunkan risiko kegagalan sistem karena kebutuhan pengguna sudah teridentifikasi sejak awal, sekaligus meningkatkan nilai bisnis melalui inovasi yang sesuai dengan perilaku konsumen. Misalnya, perusahaan layanan transportasi daring memanfaatkan *Design Thinking* untuk menyederhanakan proses pemesanan, sehingga pengguna merasa nyaman, loyal, dan perusahaan dapat menjaga daya saing di pasar digital yang dinamis.

C. Metodologi Adaptif

Masa depan perancangan sistem akan semakin dipengaruhi oleh kemajuan teknologi seperti *Artificial Intelligence (AI)*, *Big Data Analytics*, *Cloud Computing*, dan *Internet of Things (IoT)*. Oleh karena itu, **metodologi adaptif** diperlukan agar sistem dapat menyesuaikan diri dengan dinamika global yang terus berubah. Pendekatan ini

memungkinkan pengembang merespons kebutuhan baru secara cepat dan efektif.

Alasan logisnya, sistem yang adaptif mampu memperpanjang umur penggunaan aplikasi karena tidak harus dibangun ulang setiap kali ada perkembangan teknologi, serta mengurangi biaya transformasi digital dengan cara menyesuaikan modul yang sudah ada. Sebagai contoh, penggunaan *AI* dalam sistem kesehatan memungkinkan prediksi penyakit lebih dini, sementara *Cloud Computing* mendukung fleksibilitas akses data lintas lokasi, sehingga efisiensi operasional dan kualitas layanan dapat meningkat secara signifikan.

Soal Uraian

1. Jelaskan pengertian metodologi perancangan sistem informasi dan sebutkan perannya dalam *System Development Life Cycle (SDLC)*.
2. Bandingkan metodologi Structured System Analysis and Design (SSAD) dengan Object-Oriented Analysis and Design (OOAD) dari segi pendekatan, kelebihan, dan kekurangan.
3. Apa yang dimaksud dengan Prototyping dalam perancangan sistem informasi? Berikan contoh kasus penggunaannya pada sistem layanan kesehatan.
4. Mengapa Agile Methodology dianggap lebih adaptif dibandingkan metodologi tradisional? Jelaskan dengan contoh penerapan di industri e-commerce.

5. Uraikan tahapan perancangan sistem informasi mulai dari analisis kebutuhan hingga pemeliharaan, serta kaitannya dengan keberhasilan implementasi sistem.
6. Jelaskan perbedaan antara performance testing dan usability testing dalam evaluasi sistem informasi dengan memberikan contoh skenario.
7. Mengapa user acceptance test (UAT) penting dilakukan sebelum sistem informasi diterapkan? Berikan minimal dua alasan dengan ilustrasi nyata.
8. Apa saja kelebihan metodologi Rapid Application Development (RAD) dan kapan metodologi ini tepat digunakan? Sertakan contoh dalam pengembangan aplikasi mobile.
9. Diskusikan bagaimana DevOps dan Design Thinking dapat saling melengkapi dalam pengembangan sistem informasi masa depan, khususnya pada sistem layanan publik.
10. Menurut Anda, metodologi apa yang paling relevan untuk mendukung pengembangan sistem berbasis Artificial Intelligence (AI), Big Data, dan Internet of Things (IoT)? Jelaskan alasan Anda disertai contoh penerapan nyata.

Tugas Kelompok

Judul

Analisis dan Penerapan Perancangan Sistem Informasi pada Studi Kasus

Instruksi:

1. Bentuk kelompok terdiri dari **4-5 mahasiswa**.

2. Pilih satu studi kasus sistem informasi (misalnya: **sistem informasi akademik, sistem kasir retail, aplikasi layanan kesehatan, atau sistem manajemen proyek**).
3. Analisis kebutuhan pengguna dan organisasi untuk studi kasus yang dipilih.
4. Tentukan **satu metodologi perancangan sistem informasi** yang paling sesuai (misalnya: SSAD, OOAD, Prototyping, Agile, RAD, DevOps, atau Design Thinking).
5. Buat laporan yang berisi:
 - a. Latar belakang permasalahan
 - b. Alasan pemilihan metodologi
 - c. Tahapan penerapan metodologi pada studi kasus
 - d. Teknik evaluasi yang akan digunakan
 - e. Potensi pengembangan di masa depan (misalnya integrasi AI, Big Data, Cloud, IoT)
 - f. Sajikan hasil analisis dalam bentuk:
 - a. **Dokumen laporan** (5–7 halaman)
 - b. **Presentasi kelompok** (10–15 menit) dengan visualisasi diagram/tabel/flowchart.

Identifikasi Kebutuhan Perancangan Sistem Informasi

A. CPL

Mampu memahami kebutuhan pengguna.

B. CPMK

Menentukan kebutuhan dasar sistem.

C. Sub-CPMK

Mengidentifikasi masalah yang sering muncul pada sistem berjalan.

Tahap **identifikasi kebutuhan** merupakan fondasi utama dalam proses perancangan sistem informasi. Tanpa pemahaman yang tepat mengenai kebutuhan pengguna, rancangan sistem berpotensi tidak efektif, sulit dioperasikan, dan tidak mampu menyelesaikan permasalahan yang dihadapi organisasi. Oleh karena itu, pemahaman komprehensif tentang bagaimana mengidentifikasi kebutuhan harus dimiliki oleh mahasiswa agar dapat merancang sistem yang relevan dan bermanfaat. Materi ini akan menguraikan secara menyeluruh berbagai aspek yang terkait dengan identifikasi kebutuhan, mulai dari metode yang digunakan, pencatatan masalah pada sistem lama, penerjemahan kebutuhan ke dalam fitur dan fungsi, hingga validasi dengan pendekatan berbasis data.

Pembahasan mengenai **perancangan sistem informasi** menegaskan bahwa identifikasi kebutuhan merupakan tahapan inti yang memastikan rancangan konseptual memiliki arah jelas serta sesuai dengan tujuan organisasi. Proses ini mencakup pemetaan alur

kerja, struktur data, serta kebutuhan pengguna yang harus diterjemahkan ke dalam desain logis maupun fisik. Rancangan yang matang akan memudahkan tahap implementasi dan pemeliharaan karena sejak awal sudah mempertimbangkan faktor teknis dan operasional.

Di sisi lain, perspektif **data sains** menekankan bahwa setiap kebutuhan yang diidentifikasi sebaiknya divalidasi dengan pendekatan berbasis data. Analisis historis, eksplorasi pola, serta visualisasi data memberikan bukti objektif untuk memastikan kebutuhan yang dicatat sesuai dengan kondisi nyata. Contoh nyata adalah rancangan sistem informasi akademik yang dilengkapi dengan analisis data sains untuk mengukur kecepatan akses data mahasiswa dan mendeteksi pola penggunaan. Dengan integrasi ini, mahasiswa memahami bahwa identifikasi kebutuhan bukan hanya kegiatan administratif, tetapi sinergi antara desain sistem yang sistematis dan pemanfaatan data sains yang analitis, sehingga hasilnya lebih akurat, aplikatif, dan relevan.

Pentingnya Identifikasi Kebutuhan

Tahap **identifikasi kebutuhan** memiliki posisi yang sangat penting karena menjadi **awal dari seluruh rangkaian pengembangan sistem**. Setiap rancangan sistem informasi yang baik selalu diawali dengan pemetaan kebutuhan secara jelas. Jika kebutuhan pengguna tidak terpetakan dengan benar, hasil akhirnya dapat berupa sistem yang kompleks namun tidak bermanfaat, atau bahkan tidak digunakan sama sekali oleh penggunanya. Dengan kata

lain, kegagalan pada tahap ini akan memengaruhi seluruh tahapan selanjutnya. Oleh sebab itu, mahasiswa harus memahami bahwa identifikasi kebutuhan tidak sekadar proses teknis, melainkan sebuah upaya strategis untuk memastikan kesesuaian antara rancangan sistem dengan realitas operasional.

Metode Identifikasi Kebutuhan

Ada beberapa metode yang dapat digunakan dalam mengidentifikasi kebutuhan sistem, antara lain:

A. Wawancara

Wawancara dilakukan secara langsung dengan pengguna sistem, baik individu maupun kelompok. Melalui wawancara, pengembang dapat menggali informasi terkait masalah yang sering muncul pada sistem lama, fitur yang dibutuhkan, serta harapan terhadap sistem baru. Kelebihan metode ini adalah adanya komunikasi dua arah yang memungkinkan klarifikasi secara langsung.

B. Observasi

Observasi dilakukan dengan cara mengamati penggunaan sistem lama dalam aktivitas nyata sehari-hari. Dengan observasi, pengembang dapat menemukan masalah yang mungkin tidak disebutkan oleh pengguna dalam wawancara. Misalnya, alur kerja yang terlalu panjang atau prosedur yang

membbingungkan dapat terlihat jelas saat diamati secara langsung.

C. Dokumentasi

Dokumentasi berupa arsip, laporan kerja, atau catatan manual sistem lama dapat menjadi sumber informasi yang berharga. Melalui dokumentasi, pengembang dapat meninjau bagaimana alur sistem sebelumnya dijalankan, aturan yang digunakan, serta struktur data yang dikelola. Dokumentasi membantu memperkuat hasil wawancara dan observasi karena berfungsi sebagai bukti tertulis.

D. Kuesioner

Selain tiga metode utama di atas, penggunaan kuesioner juga dapat dipertimbangkan. Kuesioner memungkinkan pengembang untuk menjangkau banyak pengguna dalam waktu singkat. Pertanyaan dalam kuesioner dapat dirancang untuk mengungkapkan pengalaman, harapan, dan masalah pengguna terkait sistem yang berjalan.

Pencatatan Masalah pada Sistem Lama

Identifikasi kebutuhan tidak hanya berhenti pada pengumpulan informasi, tetapi juga melibatkan **pencatatan masalah** secara sistematis. Masalah yang ada pada sistem lama harus diinventarisasi dengan jelas, lengkap dengan dampaknya terhadap kinerja organisasi. Misalnya, masalah keterlambatan dalam pemrosesan data harus dijelaskan bersama dampaknya, seperti

penundaan layanan kepada pengguna atau lambatnya proses pengambilan keputusan. Dengan pencatatan masalah yang rapi, pengembang dapat menyusun prioritas kebutuhan, sehingga sistem baru mampu mengatasi permasalahan paling krusial terlebih dahulu.

Penerjemahan Kebutuhan Menjadi Fitur dan Fungsi

Hasil identifikasi kebutuhan kemudian harus **diterjemahkan ke dalam fitur dan fungsi sistem**. Tahap ini sangat penting karena memastikan bahwa kebutuhan yang bersifat konseptual dapat diwujudkan dalam bentuk konkret pada sistem. Sebagai contoh, kebutuhan akan kemudahan akses data mahasiswa dapat diterjemahkan menjadi fitur pencarian cepat dengan filter tertentu. Sementara itu, kebutuhan akan keamanan data dapat diterjemahkan menjadi fungsi otorisasi dan autentikasi pengguna. Proses penerjemahan kebutuhan ini harus dilakukan dengan teliti agar fitur yang dirancang benar-benar sesuai dengan ekspektasi pengguna.

Hubungan dengan Desain Konseptual

Identifikasi kebutuhan menjadi **jembatan menuju desain konseptual**. Desain konseptual merupakan representasi awal dari sistem yang akan dikembangkan, biasanya dalam bentuk diagram atau model logis. Jika identifikasi kebutuhan dilakukan secara detail, maka desain konseptual dapat dibuat dengan lebih jelas dan akurat. Sebaliknya, jika tahap identifikasi dilakukan secara terburu-buru, desain konseptual akan kehilangan arah dan berpotensi menghasilkan sistem yang tidak sesuai dengan kebutuhan nyata.

Validasi dengan Pendekatan Berbasis Data

Pendekatan berbasis data atau *data-driven approach* dapat digunakan untuk memvalidasi kebutuhan yang telah diidentifikasi. Data historis dari sistem lama dapat dianalisis untuk memastikan kebenaran kebutuhan yang disebutkan pengguna. Misalnya, jika pengguna mengeluhkan lambatnya akses data, maka pengembang dapat memeriksa waktu rata-rata yang dibutuhkan sistem lama untuk menampilkan data. Validasi dengan data memberikan bukti objektif yang memperkuat hasil identifikasi, sehingga kebutuhan yang akan diwujudkan tidak hanya berdasarkan persepsi subjektif, tetapi juga pada fakta yang dapat diukur.

Tantangan dalam Identifikasi Kebutuhan

Proses identifikasi kebutuhan tidak selalu berjalan mulus. Ada beberapa tantangan yang sering muncul, antara lain:

1. **Perbedaan persepsi** antara pengguna dan pengembang. Pengguna cenderung memandang dari sisi kenyamanan, sedangkan pengembang memandang dari sisi teknis.
2. **Keterbatasan waktu dan sumber daya** dalam melakukan wawancara, observasi, atau analisis dokumentasi.
3. **Resistensi pengguna** yang enggan menyampaikan masalah pada sistem lama karena sudah terbiasa dengan cara kerja tersebut.

Mahasiswa perlu memahami tantangan ini agar mampu mencari solusi yang tepat, seperti membangun komunikasi yang baik, menggunakan metode identifikasi yang bervariasi, serta memanfaatkan data untuk memperkuat argumen.

Relevansi Identifikasi Kebutuhan dalam Dunia Nyata

Dalam praktik nyata, tahap identifikasi kebutuhan memiliki dampak besar terhadap keberhasilan proyek pengembangan sistem informasi. Organisasi yang melakukan identifikasi kebutuhan secara detail biasanya menghasilkan sistem yang lebih sesuai, lebih efisien, dan lebih mudah diterima oleh pengguna. Sebaliknya, organisasi yang melewati tahap ini seringkali menghadapi kegagalan implementasi karena sistem yang dibangun tidak menjawab kebutuhan sebenarnya.

Sebagai contoh, pada sistem informasi rumah sakit, identifikasi kebutuhan dapat mencakup pencatatan rekam medis pasien, integrasi dengan laboratorium, hingga sistem antrean digital. Tanpa identifikasi yang jelas, rumah sakit mungkin membangun sistem yang mahal namun tidak mampu mengurangi antrian pasien. Contoh ini menegaskan betapa pentingnya tahap identifikasi kebutuhan untuk memastikan keberhasilan sistem.

Tahap **identifikasi kebutuhan** adalah inti dari perancangan sistem informasi. Proses ini mencakup wawancara, observasi, dokumentasi, serta validasi berbasis data. Masalah pada sistem lama harus dicatat secara sistematis, kemudian diterjemahkan menjadi fitur dan fungsi pada sistem baru. Identifikasi kebutuhan juga berfungsi

sebagai fondasi menuju desain konseptual yang lebih akurat. Dengan pendekatan berbasis data, validasi kebutuhan menjadi lebih kuat dan dapat dipertanggungjawabkan.



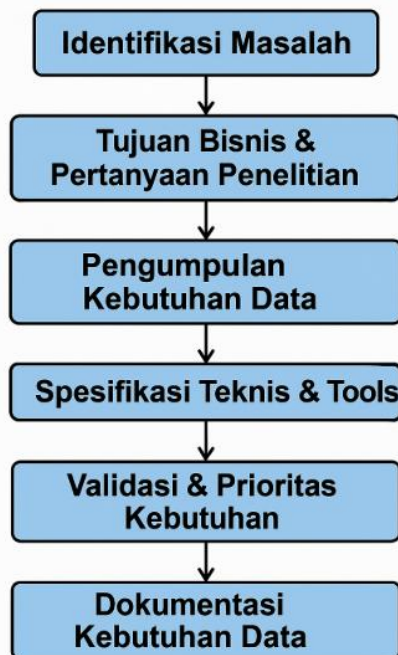
Gambar 3. Tahapan Identifikasi Kebutuhan Sistem

Proses identifikasi kebutuhan merupakan salah satu bagian terpenting dalam pengembangan sistem informasi. Tahap ini memastikan bahwa sistem yang dibangun benar-benar menjawab masalah organisasi dan memenuhi harapan pengguna. Setiap langkah dalam proses ini memiliki peran penting untuk menciptakan sistem yang efektif, efisien, dan dapat diandalkan.

Langkah awal adalah **pengumpulan informasi**, yang dilakukan melalui observasi, wawancara, studi dokumen, maupun survei. Informasi yang diperoleh digunakan untuk menggambarkan kebutuhan nyata dari pengguna dan organisasi. Setelah itu, masuk ke tahap **analisis proses bisnis saat ini**. Analisis ini membantu menemukan alur kerja yang tidak efisien, hambatan, atau peluang perbaikan yang dapat ditingkatkan melalui sistem baru.

Berikutnya adalah **identifikasi kebutuhan non-fungsional**, seperti keandalan, keamanan, performa, dan kemudahan penggunaan. Hal ini penting agar sistem tidak hanya berjalan sesuai fungsi, tetapi juga berkualitas tinggi dalam operasional sehari-hari. Selanjutnya, dilakukan **validasi dan prioritas kebutuhan** bersama pemangku kepentingan. Validasi memastikan kebutuhan yang diidentifikasi benar-benar relevan, sedangkan penentuan prioritas membantu mengatur urutan implementasi.

Tahap terakhir adalah **dokumentasi spesifikasi kebutuhan**, yang merangkum semua hasil analisis dalam dokumen formal. Dokumentasi ini menjadi dasar perancangan dan pengembangan, sekaligus referensi untuk menghindari kesalahpahaman antara pengguna dan pengembang.



Gambar 4. Tahapan Identifikasi Kebutuhan Data Sains

Tahapan identifikasi kebutuhan dalam Data Sains merupakan langkah awal yang sangat penting untuk memastikan bahwa proyek analisis data berjalan sesuai tujuan. Proses ini diawali dengan **identifikasi masalah**, yaitu memahami isu atau tantangan utama yang dihadapi organisasi. Tanpa pemahaman yang jelas, analisis data berisiko tidak relevan. Selanjutnya, dilakukan **penetapan tujuan bisnis dan pertanyaan penelitian** yang berfungsi sebagai panduan arah dalam pengumpulan data maupun pemilihan metode analisis.

Setelah tujuan jelas, tahapan berikutnya adalah **pengumpulan kebutuhan data**, di mana tim menentukan data apa saja yang diperlukan, baik data internal seperti transaksi penjualan maupun data eksternal seperti tren pasar. Proses ini harus dilanjutkan dengan **analisis sumber data** untuk memastikan ketersediaan, kualitas, serta format data yang dapat digunakan. Kemudian, tahap **spesifikasi teknis dan tools** dilakukan untuk menetapkan infrastruktur, perangkat lunak, dan metode yang sesuai.

Sebelum implementasi, dilakukan **validasi dan prioritas kebutuhan** untuk memastikan bahwa data yang dikumpulkan benar-benar mendukung tujuan analisis serta sesuai dengan keterbatasan sumber daya yang dimiliki. Akhirnya, semua hasil identifikasi didokumentasikan dalam **dokumen kebutuhan data** sehingga menjadi acuan resmi bagi tim teknis maupun pemangku kepentingan.

Soal Uraian

1. Mengapa tahap identifikasi kebutuhan dianggap sebagai fondasi awal dalam pengembangan sistem informasi?
2. Sebutkan dan jelaskan dua kelebihan serta dua kelemahan dari teknik wawancara sebagai metode identifikasi kebutuhan.
3. Bagaimana catatan manual organisasi dapat membantu dalam memperjelas kebutuhan sistem baru?
4. Uraikan keluaran yang diharapkan setelah proses identifikasi kebutuhan selesai dilakukan.
5. Jelaskan bagaimana keterlibatan pengguna sejak awal dapat meminimalkan risiko kegagalan sistem baru.
6. Bagaimana penggunaan data historis dapat meningkatkan validitas hasil identifikasi kebutuhan?
7. Berikan satu contoh masalah pada sistem pelayanan publik dan jelaskan dampaknya bagi masyarakat.
8. Jelaskan keterkaitan antara identifikasi kebutuhan dengan pembuatan desain konseptual sistem.
9. Bagaimana metode observasi dapat mengungkap masalah tersembunyi yang tidak disampaikan oleh pengguna?
10. Jelaskan perbedaan mendasar antara fitur inti yang wajib ada dan fitur tambahan yang hanya bersifat pendukung dalam sistem informasi.

Tugas Kelompok

Identifikasi Kebutuhan pada Perancangan Sistem Informasi

Rancangan Sistem Informasi Perpustakaan Digital

Deskripsi Tugas

Mahasiswa diminta untuk bekerja dalam kelompok yang terdiri dari 4-5 orang dan melakukan identifikasi kebutuhan pada **perancangan sistem informasi** untuk perpustakaan digital di perguruan tinggi. Fokus studi kasus ini adalah menemukan kebutuhan pengguna (mahasiswa, dosen, pustakawan) dalam mengakses, meminjam, serta mengelola koleksi buku secara daring.

Petunjuk Tugas

1. **Identifikasi Masalah**

Kelompok harus mengamati masalah yang sering muncul pada sistem manual, seperti keterlambatan pengembalian buku, pencarian koleksi yang sulit, atau keterbatasan akses bagi pengguna.

2. **Teknik Identifikasi**

Gunakan wawancara, observasi, dan dokumentasi untuk mengumpulkan informasi dari pihak perpustakaan.

3. **Daftar Kebutuhan**

Susun kebutuhan fungsional (pencarian buku, sistem peminjaman otomatis) dan kebutuhan non-fungsional (keamanan, kemudahan akses).

4. **Hasil Diskusi**

Kelompok membuat ringkasan kebutuhan yang dapat menjadi dasar desain sistem informasi perpustakaan.

Output yang Diharapkan

1. Laporan kelompok sepanjang 5-7 halaman.
2. Tabel daftar kebutuhan sistem.
3. Paparan hasil diskusi di kelas.

Identifikasi Kebutuhan pada Data Sains

Penerapan Data Sains untuk Prediksi Kebutuhan Energi Kota

Deskripsi Tugas

Mahasiswa diminta untuk bekerja dalam kelompok dan melakukan identifikasi kebutuhan dalam proyek **data sains** yang berfokus pada prediksi kebutuhan energi di sebuah kota besar. Fokus studi kasus ini adalah menentukan data yang diperlukan, metode analisis yang relevan, dan potensi manfaat hasil prediksi bagi pemerintah kota.

Petunjuk Tugas

1. **Identifikasi Masalah**

Tentukan permasalahan energi, misalnya konsumsi listrik yang tinggi pada jam tertentu atau distribusi energi yang tidak merata.

2. **Sumber Data:**

Identifikasi data yang dibutuhkan, seperti catatan penggunaan listrik, data cuaca, dan pola aktivitas masyarakat.

3. **Pendekatan Analitis**

Tentukan pendekatan data sains, misalnya regresi, *time series analysis*, atau *machine learning* sederhana.

4. **Hasil Diskusi**

Kelompok menyusun laporan tentang kebutuhan data dan metode analisis yang akan digunakan untuk memprediksi kebutuhan energi.

Output yang Diharapkan

1. Laporan kelompok sepanjang 5-7 halaman.
2. Diagram alur kebutuhan data dan analisis.
3. Paparan hasil diskusi di kelas.

Studi Kelayakan Perancangan Sistem Informasi dan Data Sains

A. CPL

Mampu mengevaluasi kelayakan sistem baru dan data sains

B. CPMK

Menilai aspek teknis, ekonomi, operasional, hukum.

C. Sub-CPMK

Memahami manfaat kelayakan bagi organisasi.

Tahap **studi kelayakan** merupakan salah satu fase yang paling penting dalam proses pengembangan sistem, baik pada ranah **perancangan sistem informasi** maupun **data sains**. Studi kelayakan berfungsi untuk menilai sejauh mana sebuah rancangan dapat diwujudkan dan bermanfaat bagi organisasi. Berdasarkan kajian dalam literatur *Analisa Perancangan Sistem Informasi* dan *Principles of Data Science*, studi kelayakan tidak hanya dilihat dari sisi teknis, tetapi juga dari bagaimana rancangan mampu memberikan manfaat nyata dalam pengelolaan informasi dan pemanfaatan data untuk pengambilan keputusan. Materi ini akan memaparkan secara komprehensif berbagai aspek yang terlibat dalam studi kelayakan, mulai dari tujuan, pendekatan, hingga manfaat yang diperoleh. Dengan demikian, mahasiswa akan memperoleh gambaran menyeluruh mengenai bagaimana studi kelayakan menjadi landasan utama dalam pengambilan keputusan strategis.

Tujuan Studi Kelayakan

Tujuan utama dari **studi kelayakan** adalah memastikan apakah rancangan yang disusun layak untuk diimplementasikan. Kelayakan mencakup aspek teknis, ekonomi, operasional, dan hukum. Dengan adanya evaluasi kelayakan, organisasi dapat menghindari risiko kerugian akibat sistem yang tidak sesuai kebutuhan atau sulit dilaksanakan. Studi kelayakan juga memberikan justifikasi kuat terhadap penggunaan sumber daya, sehingga setiap investasi memiliki dasar analitis yang jelas. Dalam konteks **perancangan sistem informasi**, tujuan ini berkaitan dengan penciptaan sistem yang fungsional, efisien, dan sesuai kebutuhan pengguna sebagaimana dijelaskan dalam *Analisa Perancangan Sistem Informasi*. Sedangkan dalam **data sains**, seperti yang dijelaskan pada *Principles of Data Science*, tujuannya adalah memastikan bahwa analisis data yang dilakukan mampu memberikan nilai tambah nyata bagi pengambilan keputusan.

Studi Kelayakan dalam Perancangan Sistem Informasi

Dalam **perancangan sistem informasi**, studi kelayakan menilai apakah sistem baru yang dirancang benar-benar dapat meningkatkan efisiensi organisasi. *Analisa Perancangan Sistem Informasi* menekankan bahwa perancangan yang baik dimulai dengan identifikasi kebutuhan, dilanjutkan dengan evaluasi infrastruktur, serta analisis kesiapan sumber daya manusia. Misalnya, universitas yang ingin mengembangkan sistem akademik digital harus menilai kesiapan jaringan, kapasitas server, serta pelatihan staf.

Tanpa evaluasi menyeluruh, proyek berisiko menghasilkan sistem yang mahal namun tidak efektif. Studi kelayakan berperan sebagai **alat seleksi** untuk memastikan hanya rancangan dengan prospek keberhasilan tinggi yang diteruskan ke tahap implementasi.

Studi Kelayakan dalam Data Sains

Pada **data sains**, studi kelayakan berfokus pada ketersediaan dan kualitas data, metode analisis yang akan digunakan, serta manfaat dari hasil analisis. *Principles of Data Science* menekankan pentingnya data yang memiliki volume, variasi, dan validitas yang baik. Sebagai contoh, proyek prediksi kebutuhan energi hanya dapat berhasil jika data historis, data cuaca, dan pola aktivitas masyarakat tersedia dengan kualitas memadai. Selain itu, studi kelayakan juga mencakup evaluasi kesiapan perangkat lunak, algoritma, serta keterampilan sumber daya manusia dalam menggunakan teknik seperti *time series analysis* atau *machine learning*. Hal ini memastikan proyek yang dijalankan realistis secara praktis, tidak hanya menarik secara teoretis.

Empat Aspek Utama Studi Kelayakan

Studi kelayakan didasarkan pada evaluasi empat aspek utama: teknis, ekonomi, operasional, dan hukum. Keempat aspek ini, menurut *Analisa Perancangan Sistem Informasi* dan *Principles of Data Science*, saling melengkapi dalam memberikan gambaran komprehensif.

1. Aspek Teknis

Menilai kemampuan infrastruktur dan teknologi yang dimiliki organisasi. *Analisa Perancangan Sistem Informasi* menyoroti kebutuhan perangkat keras dan perangkat lunak yang sesuai dalam rancangan sistem informasi, sedangkan *Principles of Data Science* menekankan kesesuaian algoritma dan *tools* analisis data.

2. Aspek Ekonomi

Menilai keseimbangan antara biaya dan manfaat. Perancangan sistem informasi memerlukan analisis biaya implementasi dan pemeliharaan, sementara data sains menekankan nilai ekonomis dari prediksi atau pola yang ditemukan.

3. Aspek Operasional

Menganalisis kesiapan pengguna, budaya organisasi, dan prosedur kerja. *Analisa Perancangan Sistem Informasi* menunjukkan bahwa resistensi pengguna sering menjadi kendala dalam sistem baru, sedangkan *Principles of Data Science* menggarisbawahi perlunya pelatihan dalam penggunaan algoritma analisis.

4. Aspek Hukum

Memastikan kepatuhan terhadap regulasi. *Analisa Perancangan Sistem Informasi* mengaitkan aspek ini dengan perlindungan data organisasi, sedangkan *Principles of Data Science* mengaitkannya dengan isu privasi dan regulasi penggunaan data pribadi dalam analisis.

A. Laporan Formal Studi Kelayakan

Hasil studi kelayakan dituangkan dalam **laporan formal** yang mencakup analisis keempat aspek utama. Laporan ini menjadi acuan bagi manajemen untuk mengambil keputusan strategis. *Analisa Perancangan Sistem Informasi* menekankan bahwa laporan dalam perancangan sistem informasi berfungsi sebagai dokumen teknis dan komunikasi manajerial, sedangkan *Principles of Data Science* menekankan bahwa laporan dalam proyek data sains harus menampilkan analisis data sebagai bukti objektif. Keduanya menunjukkan bahwa laporan formal bukan hanya dokumen administratif, tetapi juga dasar legitimasi bagi keputusan organisasi.

B. Pengambilan Keputusan Organisasi

Keputusan organisasi sangat dipengaruhi oleh hasil studi kelayakan. Jika laporan menyatakan layak, maka sistem informasi atau proyek data sains dapat dilanjutkan dengan keyakinan besar. Jika dinyatakan tidak layak, organisasi dapat menunda atau mengubah arah proyek. Dengan mengacu pada pembahasan *Analisa Perancangan Sistem Informasi* dan *Principles of Data Science*, studi kelayakan berfungsi sebagai **alat mitigasi risiko** yang menyeimbangkan perspektif teknis dan data analitis dalam pengambilan keputusan.

C. Evaluasi Berbasis Data

Evaluasi berbasis data atau *data-driven evaluation* memperkuat argumen studi kelayakan. *Principles of Data Science* menekankan bahwa data historis, eksplorasi pola, dan visualisasi memberikan dasar objektif yang kuat. Di sisi lain, *Analisa Perancangan Sistem Informasi* menunjukkan bagaimana data teknis sistem lama, seperti jumlah transaksi harian atau kapasitas pemrosesan, dapat digunakan untuk memperkuat analisis kelayakan sistem informasi. Dengan menggabungkan keduanya, laporan kelayakan menjadi lebih kredibel.

Relevansi Studi Kelayakan bagi Organisasi

Manfaat studi kelayakan sangat besar, baik dalam perancangan sistem informasi maupun data sains. *Analisa Perancangan Sistem Informasi* menegaskan bahwa studi kelayakan mencegah organisasi membangun sistem yang tidak efisien, sedangkan *Principles of Data Science* menekankan bahwa studi kelayakan membantu organisasi memaksimalkan pemanfaatan data. Dengan demikian, studi kelayakan menjadi instrumen penting dalam strategi manajemen modern yang mengintegrasikan teknologi dan analisis data.

Tahap **studi kelayakan** merupakan komponen penting dalam pengembangan **perancangan sistem informasi** dan **data sains**. Studi ini menilai kelayakan dari segi teknis, ekonomi, operasional, dan hukum, serta dituangkan dalam laporan formal yang menjadi dasar pengambilan keputusan organisasi. *Analisa Perancangan Sistem Informasi* menekankan aspek desain sistem, sedangkan *Principles of*

Data Science menekankan analisis data. Keduanya saling melengkapi, sehingga evaluasi kelayakan tidak hanya berdasarkan intuisi, tetapi juga analisis yang terukur. Dengan pemahaman ini, mahasiswa dapat melihat bahwa studi kelayakan adalah sinergi antara rancangan sistem yang matang dan pemanfaatan data sains yang analitis, sehingga menjadi proses strategis yang menentukan keberhasilan organisasi dalam menghadapi tantangan era digital

Tabel 1. Dimensi Studi Kelayakan Perancangan Sistem Informasi

Dimensi	Aspek yang Dinilai	Pertanyaan Kunci	Output/ Indikator
Kelayakan Teknis	Ketersediaan teknologi (hardware, software, jaringan) Kapasitas SDM TI Integrasi dengan sistem lama	Apakah teknologi yang dibutuhkan tersedia dan terjangkau? Apakah tim mampu mengoperasikan?	Spesifikasi teknis, kebutuhan perangkat, kesenjangan teknologi.
Kelayakan Ekonomi	Biaya investasi (hardware, software, SDM) Biaya operasional & pemeliharaan Analisis cost-benefit	Apakah manfaat sistem lebih besar dari biayanya? Kapan ROI tercapai?	Estimasi biaya, perbandingan biaya vs manfaat, periode balik modal.

Dimensi	Aspek yang Dinilai	Pertanyaan Kunci	Output/ Indikator
Kelayakan Operasional	Dukungan manajemen Kesiapan pengguna Perubahan budaya kerja SOP & proses bisnis	Apakah sistem bisa dijalankan dalam operasi sehari-hari? Apakah pengguna mau menerima perubahan?	Tingkat penerimaan pengguna, kesiapan prosedur, komitmen organisasi.
Kelayakan Hukum & Etika	Kepatuhan terhadap regulasi Hak cipta software Keamanan data & privasi	Apakah sistem sesuai dengan peraturan pemerintah? Apakah data pengguna aman secara hukum?	Checklist kepatuhan regulasi, analisis risiko hukum.
Kelayakan Jadwal	Estimasi waktu pengembangan Ketersediaan SDM & resources Deadline Proyek	Apakah sistem dapat diselesaikan sesuai jadwal? Apakah ada risiko keterlambatan?	Timeline proyek, milestone, risiko keterlambatan.
Kelayakan Sosial	Dampak pada stakeholder Perubahan pola kerja Tanggung jawab sosial organisasi	Apakah sistem membawa dampak positif pada masyarakat/pengguna? Apakah ada resistensi?	Analisis dampak sosial, tingkat resistensi, penerimaan stakeholder eksternal.

Tabel 2. Dimensi Studi Kelayakan Proyek Data Sains

Dimensi	Aspek yang Dinilai	Pertanyaan Kunci	Output/ Indikator
Kelayakan Data	Kualitas data (akurasi, kelengkapan, konsistensi) Ketersediaan data historis Aksesibilitas data	Apakah data cukup untuk melatih model? Apakah data mudah diakses dan sesuai regulasi?	Profil kualitas data, jumlah sampel, kesiapan data pipeline.
Kelayakan Teknis	Infrastruktur komputasi (server, cloud, GPU) Tools & framework (Python, R, TensorFlow, dll.) SDM (data scientist, data engineer)	Apakah infrastruktur mampu mengolah data besar? Apakah tim menguasai teknologi yang dipakai?	Spesifikasi infrastruktur, gap kompetensi teknis, kebutuhan software/hardware.
Kelayakan Ekonomi	Biaya penyimpanan & komputasi Biaya lisensi software Biaya SDM spesialis Estimasi ROI (Return on Investment)	Apakah hasil analitik/AI memberikan keuntungan lebih besar dari biaya? Kapan ROI tercapai?	Estimasi cost-benefit, perhitungan ROI, proyeksi efisiensi/pendapatan tambahan.

Dimensi	Aspek yang Dinilai	Pertanyaan Kunci	Output/ Indikator
Kelayakan Operasional	Dukungan manajemen Integrasi hasil analisis dengan proses bisnis Kesiapan pengguna memahami output model	Apakah hasil model dapat digunakan dalam keputusan bisnis sehari-hari? Apakah manajemen siap?	Tingkat penerimaan pengguna, kesiapan adopsi hasil analitik, integrasi ke workflow.
Kelayakan Hukum & Etika	Kepatuhan terhadap regulasi data (misal: GDPR, UU ITE, HIPAA) Privasi & keamanan data Bias & fairness algoritma	Apakah penggunaan data sesuai hukum? Apakah model bebas dari diskriminasi/etika?	Checklist regulasi, analisis risiko privasi, fairness audit model.
Kelayakan Jadwal	Estimasi waktu pengumpulan & pembersihan data Durasi pelatihan model Deadline bisnis/proyek	Apakah proyek dapat selesai sesuai target bisnis? Apakah ada bottleneck data cleaning?	Timeline proyek, milestone, risiko keterlambatan, estimasi training & deployment time.

Dimensi	Aspek yang Dinilai	Pertanyaan Kunci	Output/ Indikator
Kelayakan Sosial	Dampak hasil model bagi pengguna & masyarakat Persepsi publik terhadap penggunaan AI/analitik Transparansi hasil	Apakah proyek meningkatkan kepercayaan publik? Apakah ada potensi resistensi dari user?	Analisis dampak sosial, penerimaan stakeholder, strategi komunikasi hasil model.

Soal Uraian

1. Jelaskan mengapa studi kelayakan menjadi tahap penting sebelum implementasi sistem baru.
2. Sebutkan dan uraikan empat aspek utama dalam studi kelayakan dengan contoh singkat.
3. Bagaimana aspek ekonomi dapat memengaruhi keputusan organisasi dalam mengadopsi sistem baru?
4. Mengapa kepatuhan hukum menjadi faktor yang menentukan dalam studi kelayakan suatu proyek?
5. Apa bentuk keluaran yang dihasilkan setelah studi kelayakan selesai dilakukan?
6. Bagaimana cara menilai kelayakan teknis pada proyek pengembangan aplikasi layanan publik?
7. Uraikan dampak positif yang dapat diperoleh organisasi setelah melakukan studi kelayakan secara menyeluruh.
8. Berikan contoh aspek operasional yang perlu diperhatikan dalam sistem manajemen rumah sakit.

9. Jelaskan bagaimana data historis dapat digunakan untuk memperkuat argumen dalam laporan kelayakan.
10. Siapa saja pihak yang terlibat dalam penyusunan laporan studi kelayakan dan apa peran mereka?

Tugas Kelompok

Studi Kasus 1: Rancangan Sistem Informasi Perpustakaan Digital

Laporan Singkat Studi Kelayakan Rancangan Sistem Informasi Perpustakaan Digital

Deskripsi Tugas

Mahasiswa diminta bekerja dalam kelompok 4-5 orang untuk menyusun laporan singkat studi kelayakan dengan tema **Rancangan Sistem Informasi Perpustakaan Digital**. Fokus tugas ini adalah menilai apakah rancangan sistem dapat dilaksanakan dengan mempertimbangkan aspek teknis, ekonomi, operasional, dan hukum.

Petunjuk Tugas

1. Aspek Teknis

Evaluasi infrastruktur yang tersedia seperti jaringan, server, dan perangkat lunak pendukung.

2. Aspek Ekonomi

Hitung estimasi biaya implementasi dan manfaat yang diperoleh, misalnya efisiensi waktu pencarian buku.

3. Aspek Operasional

Analisis kesiapan pustakawan, mahasiswa, dan staf administrasi dalam menggunakan sistem baru.

4. **Aspek Hukum**

Pastikan sistem mendukung perlindungan hak cipta koleksi digital dan keamanan data pengguna.

5. **Kesimpulan**

Kelompok menyimpulkan kelayakan rancangan sistem dan memberikan rekomendasi.

Output yang Diharapkan

1. Laporan kelompok sepanjang 5 halaman.
2. Ringkasan tabel aspek kelayakan.
3. Presentasi singkat di kelas.

Studi Kasus 2: Penerapan Data Sains untuk Prediksi Kebutuhan Energi Kota

Laporan Singkat Studi Kelayakan Penerapan Data Sains untuk Prediksi Kebutuhan Energi Kota

Deskripsi Tugas

Mahasiswa diminta bekerja dalam kelompok untuk menyusun laporan singkat studi kelayakan dengan tema **Penerapan Data Sains untuk Prediksi Kebutuhan Energi Kota**. Fokus tugas ini adalah menilai apakah proyek data sains dapat dilaksanakan secara efektif dengan memperhatikan empat aspek utama.

Petunjuk Tugas

1. **Aspek Teknis**

Analisis ketersediaan data historis energi, perangkat analisis, serta keandalan server.

2. **Aspek Ekonomi**

Perkirakan biaya pemrosesan data dan manfaat berupa efisiensi distribusi energi.

3. **Aspek Operasional**

Identifikasi kesiapan tim analisis data dan dukungan pemerintah kota.

4. **Aspek Hukum**

Pastikan penggunaan data mematuhi regulasi privasi dan perlindungan data warga.

5. **Kesimpulan**

Kelompok menyimpulkan kelayakan proyek dan memberikan rekomendasi.

Output yang Diharapkan

1. Laporan kelompok sepanjang 5 halaman.
2. Diagram alur kebutuhan data dan analisis.
3. Presentasi singkat di kelas.

Pemodelan Sistem dengan DFD dan ERD

A. CPL

Mahasiswa mampu membuat pemodelan proses dan data.

B. CPMK

Menggunakan DFD dan ERD dalam pemodelan.

C. Sub-CPMK

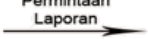
Menyusun alur data dan hubungan antar entitas.

Pemodelan sistem merupakan tahapan **penting** dalam pengembangan **perancangan sistem informasi**. Tahap ini membantu menyusun gambaran utuh mengenai bagaimana data bergerak, diolah, dan disimpan sebelum sistem diwujudkan dalam bentuk fisik. Dua teknik utama yang menjadi landasan dalam kajian *Analisa Perancangan Sistem Informasi* adalah **Data Flow Diagram (DFD)** dan **Entity Relationship Diagram (ERD)**. DFD memusatkan perhatian pada **alur data** dan transformasinya melalui proses-proses terdefinisi, sedangkan ERD berfokus pada **struktur data** beserta **hubungan antar entitas** di dalam rancangan basis data. Dengan menguasai keduanya, mahasiswa membangun kemampuan untuk menata proses dan data secara **terstruktur, logis, dan konsisten**, sejalan dengan prinsip yang dibahas dalam *Analisa Perancangan Sistem Informasi*.

Konsep Dasar Data Flow Diagram (DFD)

Data Flow Diagram (DFD) adalah representasi grafis yang menggambarkan arus data dalam suatu sistem. DFD menjelaskan dari mana data berasal, bagaimana data diolah, dan ke mana data dikirimkan. Keunggulan DFD terletak pada penyajian informasi secara ringkas dan mudah dipahami oleh berbagai pemangku kepentingan.

A. Komponen DFD

Simbol	Makna	Contoh
	Entity	
	Alir Data	
	Proses	
	Data Storage	

Gambar 5. Simbol DFD dan Fungsinya

Data Flow Diagram (DFD) adalah alat pemodelan sistem yang digunakan untuk menggambarkan aliran data dalam sebuah proses. Simbol utama DFD terdiri dari empat bagian: *Entity* (kotak), yang merepresentasikan sumber atau tujuan data; *Alir Data* (panah), yang menunjukkan pergerakan data antar komponen; *Proses* (lingkaran), yang menggambarkan transformasi data; serta *Data Storage* (dua garis paralel), yang melambangkan penyimpanan data. Dengan menggunakan simbol-simbol ini, analis sistem dapat dengan mudah

memvisualisasikan bagaimana data diproses, disimpan, dan ditransmisikan antar entitas dalam suatu sistem informasi.

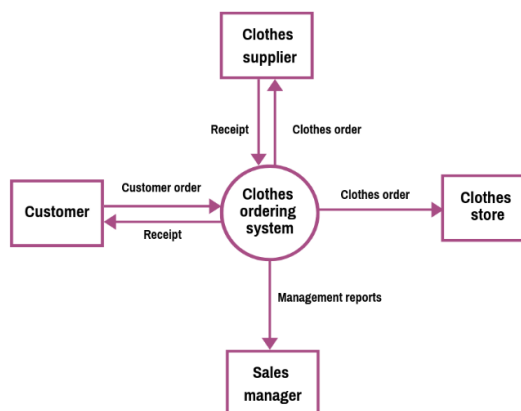
DFD menggunakan simbol standar agar komunikasi rancangan **seragam**:

1. **Entitas Eksternal** (persegi panjang): sumber atau tujuan data di luar sistem.
2. **Proses** (lingkaran/oval): pengolahan yang mengubah masukan menjadi keluaran.
3. **Aliran Data** (panah): perpindahan data antar elemen.
4. **Penyimpanan Data** (garis ganda horizontal): tempat penyimpanan sementara atau permanen.

B. Tingkatan DFD

DFD disusun bertingkat untuk menata rincian proses secara bertahap:

1. **Level 0** atau *context diagram*: menampilkan sistem sebagai satu proses utama dengan entitas eksternal dan aliran data yang terkait; menggambarkan sistem **secara utuh**.

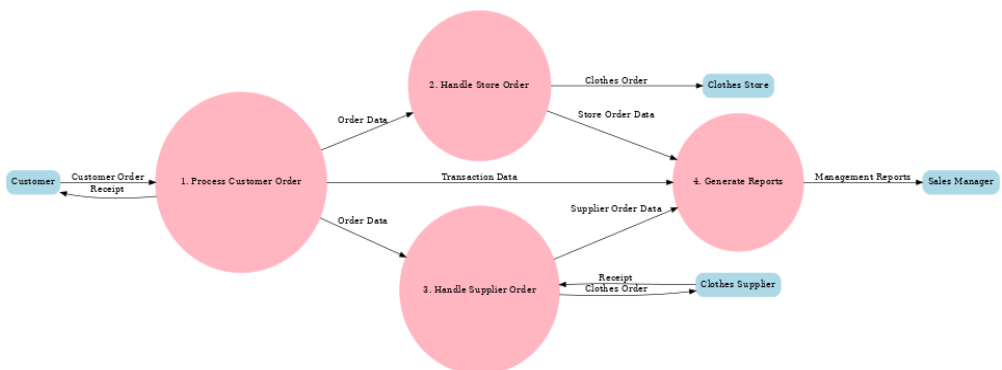


Gambar 6. Contoh *Context Diagram* atau *Level 0*

Diagram di atas menggambarkan *Context Diagram* atau *Level 0 Data Flow Diagram (DFD)* dari sebuah sistem pemesanan pakaian. Sistem digambarkan sebagai satu proses utama dengan entitas eksternal yang berinteraksi langsung. Entitas eksternal tersebut meliputi *Customer* (pelanggan), *Clothes Supplier* (pemasok pakaian), *Clothes Store* (toko pakaian), dan *Sales Manager* (manajer penjualan). Aliran data yang ditunjukkan mencerminkan komunikasi antara sistem dan para entitas.

Contohnya, pelanggan mengirimkan *Customer Order* (pesanan pelanggan) dan menerima *Receipt* (kwitansi) sebagai bukti transaksi. Sistem juga mengirimkan *Clothes Order* (pesanan pakaian) ke pemasok maupun toko pakaian, serta menerima konfirmasi berupa *Receipt*. Selain itu, sistem menghasilkan *Management Reports* (laporan manajerial) yang diberikan kepada manajer penjualan untuk kebutuhan analisis bisnis.

2. **Level 1**: memecah proses utama menjadi beberapa subproses.



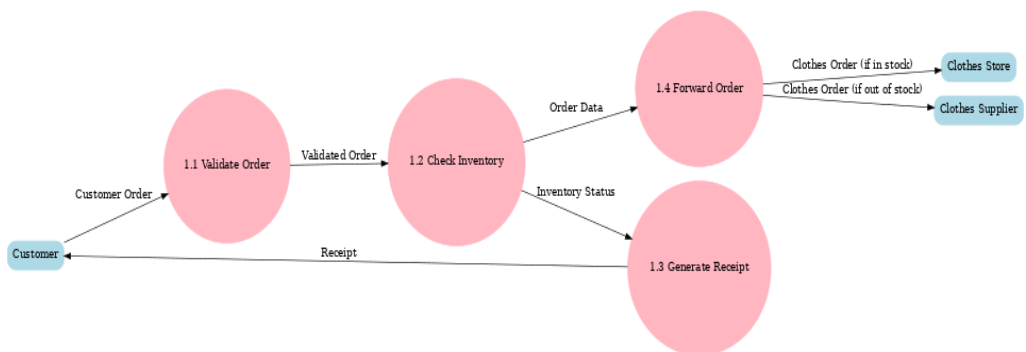
Gambar 7. Data Flow Diagram (DFD) Level 1

Data Flow Diagram (DFD) Level 1 menggambarkan aliran data utama dalam sistem pemesanan pakaian. Proses dimulai ketika

customer mengirimkan *customer order* ke sistem. Pesanan ini kemudian diproses dan menghasilkan *receipt* yang dikirim kembali kepada customer. Selanjutnya, sistem mengelola pesanan ke dua pihak utama: **clothes store** dan **clothes supplier**.

Jika barang tersedia, sistem membuat *clothes order* ke store; jika tidak tersedia, maka sistem meneruskan order ke supplier. Supplier memberikan *receipt* sebagai konfirmasi, yang kembali masuk ke dalam sistem. Semua transaksi yang terjadi, baik dari customer, store, maupun supplier, diolah lebih lanjut untuk menghasilkan *management reports* yang disampaikan kepada **sales manager**. Laporan ini berguna untuk analisis penjualan, evaluasi stok, dan pengambilan keputusan bisnis.

3. **Level 2** dan seterusnya: digunakan bila diperlukan penjabaran lebih rinci.



Gambar 8. Data Flow Diagram (DFD) Level 2

Gambar ini menampilkan Data Flow Diagram (DFD) Level 2 yang menjelaskan lebih detail proses *Process Customer Order*. Proses ini diuraikan menjadi empat subproses. Pertama, **Validate Order**, yaitu sistem memeriksa pesanan dari customer untuk memastikan data valid. Setelah validasi, pesanan diteruskan ke tahap **Check**

Inventory, di mana sistem mengecek ketersediaan barang di dalam stok. Hasil dari tahap ini menghasilkan *Inventory Status* yang akan menentukan langkah selanjutnya. Jika stok tersedia, pesanan diteruskan ke **Forward Order** untuk dikirim ke *Clothes Store*. Jika stok tidak tersedia, order dialihkan ke *Clothes Supplier*. Sementara itu, berdasarkan hasil pengecekan, sistem juga membuat **Generate Receipt** yang berfungsi sebagai bukti transaksi dan dikirimkan kembali ke customer.

Prinsip *decomposition* (perincian bertahap) dan *balancing* (keseimbangan aliran antar level) menjadi acuan agar input-output pada satu level **selaras** dengan level di bawahnya. Hal ini selaras dengan penekanan dalam *Analisa Perancangan Sistem Informasi* tentang disiplin pemecahan proses dan penjagaan konsistensi antar level.

C. Spesifikasi Proses dan Kamus Data

Tabel 3. Kamus Data Sistem Pemesanan Pakaian

Data Flow	Deskripsi	Elemen Data
Customer Order	Data pesanan dari customer	Customer_ID, Nama, Alamat, Item_Code, Jumlah, Tanggal_Order
Validated Order	Pesanan yang telah diverifikasi	Customer_ID, Item_Code, Jumlah, Status_Validasi
Inventory Status	Status ketersediaan stok barang	Item_Code, Stok_Tersedia, Stok_Dibutuhkan
Receipt	Bukti transaksi pemesanan untuk customer	Receipt_No, Customer_ID, Item_Code, Jumlah, Harga, Total, Status_Order

Data Flow	Deskripsi	Elemen Data
Clothes Order	Pesanan ke store atau supplier	Order_ID, Item_Code, Jumlah, Tujuan (Store/Supplier)
Store Order Data	Data order yang dikirim ke Clothes Store	Order_ID, Item_Code, Jumlah, Status_Pengiriman
Supplier Order Data	Data order yang dikirim ke Clothes Supplier	Order_ID, Item_Code, Jumlah, Status_Konfirmasi
Supplier Receipt	Bukti konfirmasi order dari supplier	Receipt_No, Supplier_ID, Order_ID, Status_Pengiriman
Management Reports	Laporan manajemen hasil analisis penjualan	Periode, Total_Order, Total_Pendapatan, Barang_Terlaris, Status_Stok

Tabel ini berfungsi untuk menjelaskan aliran data yang digunakan dalam sistem pemesanan pakaian. Setiap arus data memiliki deskripsi dan elemen penting yang mendukung pemrosesan informasi. Dengan adanya kamus data, sistem menjadi lebih mudah dipahami secara teknis maupun fungsional. Data yang dikirimkan customer, hasil validasi, status stok, hingga laporan manajemen, semuanya didefinisikan secara jelas agar tidak terjadi ambiguitas. Kamus data juga menjadi referensi utama bagi pengembang maupun analis dalam merancang, mengelola, dan memastikan konsistensi sistem.

Tabel 4. Data Dictionary Sistem Pemesanan Pakaian

Data Flow	Deskripsi	Elemen Data
Customer Order	Data pesanan yang dikirim oleh customer	Customer_ID, Nama, Alamat, Item_Code, Jumlah, Tanggal_Order

Data Flow	Deskripsi	Elemen Data
Validated Order	Data order yang telah diverifikasi sistem	Customer_ID, Item_Code, Jumlah, Status_Validasi
Inventory Status	Status ketersediaan stok barang	Item_Code, Stok_Tersedia, Stok_Dibutuhkan
Receipt	Bukti transaksi untuk customer	Receipt_No, Customer_ID, Item_Code, Jumlah, Harga, Total, Status_Order
Clothes Order	Pesanan diteruskan ke store/supplier	Order_ID, Item_Code, Jumlah, Tujuan (Store/Supplier)
Store Order Data	Data order yang dikirim ke Clothes Store	Order_ID, Item_Code, Jumlah, Status_Pengiriman
Supplier Order Data	Data order yang dikirim ke Clothes Supplier	Order_ID, Item_Code, Jumlah, Status_Konfirmasi
Supplier Receipt	Bukti konfirmasi order dari supplier	Receipt_No, Supplier_ID, Order_ID, Status_Pengiriman
Management Reports	Laporan hasil analisis penjualan	Periode, Total_Order, Total_Pendapatan, Barang_Terlaris, Status_Stok

Tabel Data Dictionary ini menjelaskan arus data yang digunakan dalam sistem pemesanan pakaian. Setiap arus data dijabarkan melalui nama, deskripsi, dan elemen datanya. Dengan adanya data dictionary, setiap proses dalam sistem menjadi lebih jelas dan terdefinisi, sehingga memudahkan analisis, perancangan, dan implementasi. Customer Order, Validated Order, hingga Management Reports memiliki struktur yang baku untuk menghindari ambiguitas. Elemen-elemen data seperti Customer_ID, Item_Code,

hingga Status_Stok, memastikan bahwa informasi yang mengalir dalam sistem dapat dipahami konsisten oleh semua pihak yang terlibat.

Agar DFD dapat dioperasikan, setiap proses memerlukan **spesifikasi proses** (*process specification*) dan pendefinisian arus melalui **kamus data** (*data dictionary*). Spesifikasi proses berisi uraian langkah pengolahan, sumber data, serta keluaran yang dihasilkan. Kamus data menjelaskan struktur, tipe, dan batasan elemen data. Keduanya membantu menjaga **keterbacaan** dan **ketepatan** rancangan sebagaimana dianjurkan dalam *Analisa Perancangan Sistem Informasi*.

Konsep Dasar Entity Relationship Diagram (ERD)

A. Pengertian ERD

Entity Relationship Diagram (ERD) adalah model grafis yang menggambarkan **entitas**, **atribut**, dan **relasi** (*relationship*) di dalam rancangan basis data. ERD memandu penyusunan struktur data yang **terorganisasi** sehingga kebutuhan informasi dapat dipenuhi tanpa kelebihan data.

B. Komponen ERD

SIMBOL	NAMA
	Entitas (<i>Entity</i>)
	Atribut (<i>Attribute</i>)
	Relasi/Hubungan (<i>Relationship</i>)

Gambar 9. Simbol ERD

ambar ini menunjukkan simbol-simbol dasar yang digunakan dalam diagram Entity Relationship (ERD). Simbol pertama berbentuk persegi panjang melambangkan **Entitas (Entity)**, yaitu objek nyata atau konsep yang datanya disimpan, seperti mahasiswa, dosen, atau mata kuliah. Simbol kedua berbentuk oval mewakili **Atribut (Attribute)**, yaitu sifat atau karakteristik dari entitas, misalnya nama, alamat, atau tanggal lahir. Simbol ketiga berbentuk belah ketupat digunakan untuk menunjukkan **Relasi/Hubungan (Relationship)**, yakni penghubung antar entitas, contohnya mahasiswa mengambil mata kuliah atau dosen mengajar kelas tertentu.

Selain itu, terdapat **garis penghubung** yang mengikat entitas, relasi, dan atribut sehingga hubungan data menjadi jelas. Dalam praktik perancangan, konsep kunci utama (*primary key*) dan kunci asing (*foreign key*) diterapkan untuk menjamin keunikan serta keterkaitan data. Penentuan kunci ini merupakan langkah pokok dalam Analisa Perancangan Sistem Informasi agar basis data yang dihasilkan konsisten, terstruktur, dan mampu mendukung kebutuhan informasi secara optimal.

C. Kardinalitas dan Opsionalitas

Kardinalitas menyatakan jumlah keterlibatan entitas dalam relasi (*one-to-one*, *one-to-many*, *many-to-many*). **Opsionalitas** menunjukkan apakah keterlibatan itu wajib atau opsional. Penandaan kardinalitas dan opsionalitas membantu menyelaraskan ERD dengan DFD, karena pola arus data harus selaras dengan hubungan data yang ditetapkan.

D. Penyusunan Struktur dan Bentuk Normal

Agar struktur data efisien, dilakukan penataan melalui *normalization* menuju bentuk normal yang sesuai kebutuhan (misalnya 1NF, 2NF, 3NF). Tujuannya menghindari **redundansi** dan menjaga **integritas** data. *Analisa Perancangan Sistem Informasi* menegaskan pentingnya penataan ini sebelum rancangan fisik dibuat.

Hubungan dan Kesesuaian DFD-ERD

DFD dan ERD menyajikan dua sisi dari rancangan yang **saling melengkapi**:

- a) **DFD** menata **alur kerja data** dari entitas eksternal menuju proses, penyimpanan, dan keluaran.
- b) **ERD** menata **relasi data** yang menyokong alur tersebut.

Kesesuaian dilakukan melalui langkah berikut (selaras dengan *Analisa Perancangan Sistem Informasi*):

1. **Menelusuri aliran utama** di DFD Level 0 dan Level 1; memastikan setiap aliran memiliki asal dan tujuan yang jelas.
2. **Memetakan aliran ke struktur data** di ERD; setiap penyimpanan pada DFD berpadanan dengan tabel/entitas yang relevan.
3. **Memastikan keseimbangan**: input-output pada proses yang memanfaatkan penyimpanan harus selaras dengan entitas dan relasi pada ERD.
4. **Menguji skenario penggunaan** untuk melihat konsistensi perubahan data dari aliran pada DFD terhadap keterhubungan pada ERD.

Dengan pola ini, rancangan **terhindar** dari kesenjangan antara proses dan data, serta memudahkan transisi ke rancangan fisik.

Simbol dan Aturan Penamaan

Penggunaan **simbol standar** dan **aturan penamaan** yang konsisten memperlancar komunikasi rancangan. *Analisa Perancangan Sistem Informasi* menganjurkan:

- a) Penamaan proses dengan kata kerja yang **jelas**, misalnya **Kelola Data Mahasiswa**.
- b) Penamaan penyimpanan data sebagai **kata benda jamak**, misalnya **Data Mahasiswa**.
- c) Penghindaran singkatan yang tidak baku; bila harus menggunakan singkatan, sertakan entri di *data dictionary*.

Disiplin penamaan ini memudahkan pemangku kepentingan non-teknis memahami rancangan tanpa ambiguitas.

Pemodelan sebagai Dasar Rancangan Basis Data

Pemodelan DFD-ERD menjadi dasar bagi rancangan **basis data** dan rancangan antarmuka. Rangkaian langkah yang dianjurkan dalam *Analisa Perancangan Sistem Informasi* adalah:

1. **Membangun DFD Level 0** untuk menampilkan gambaran utuh sistem.
2. **Merinci ke Level 1/2** guna menata proses dan aliran utama.
3. **Menyusun ERD awal** dari istilah data, penyimpanan pada DFD, dan kebutuhan pelaporan.

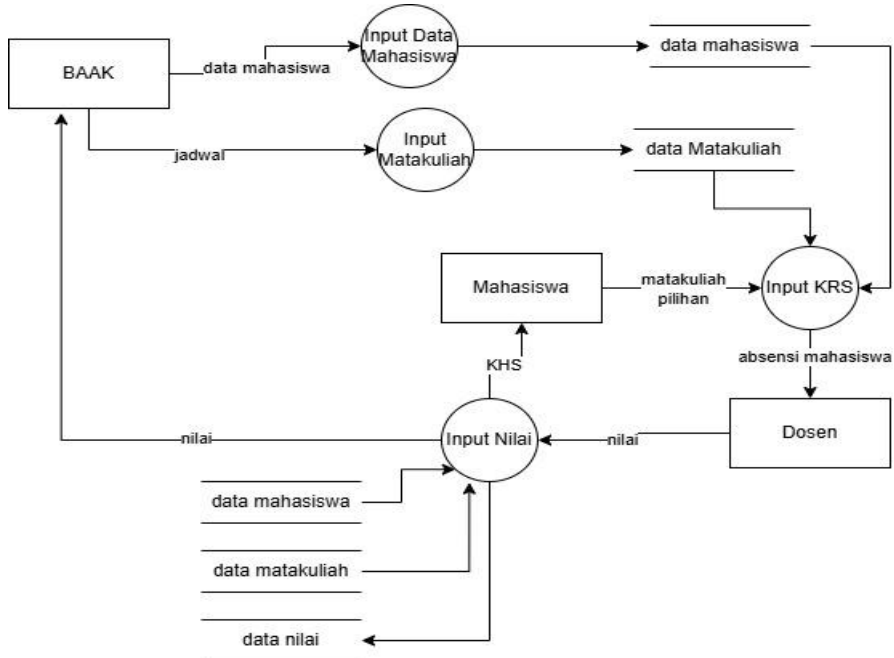
4. **Menata kunci dan kardinalitas**, lalu melakukan *normalization* seperlunya.
5. **Melengkapi kamus data**, termasuk definisi, tipe, dan batasan setiap elemen.
6. **Menyelaraskan kembali** DFD dan ERD hingga tidak ada aliran/relasi yang bertentangan.

Hasil akhirnya ialah rancangan yang **siap** diturunkan menjadi skema basis data dan rancangan antarmuka yang menyalurkan data sesuai alur pada DFD.

Contoh Penerapan DFD-ERD

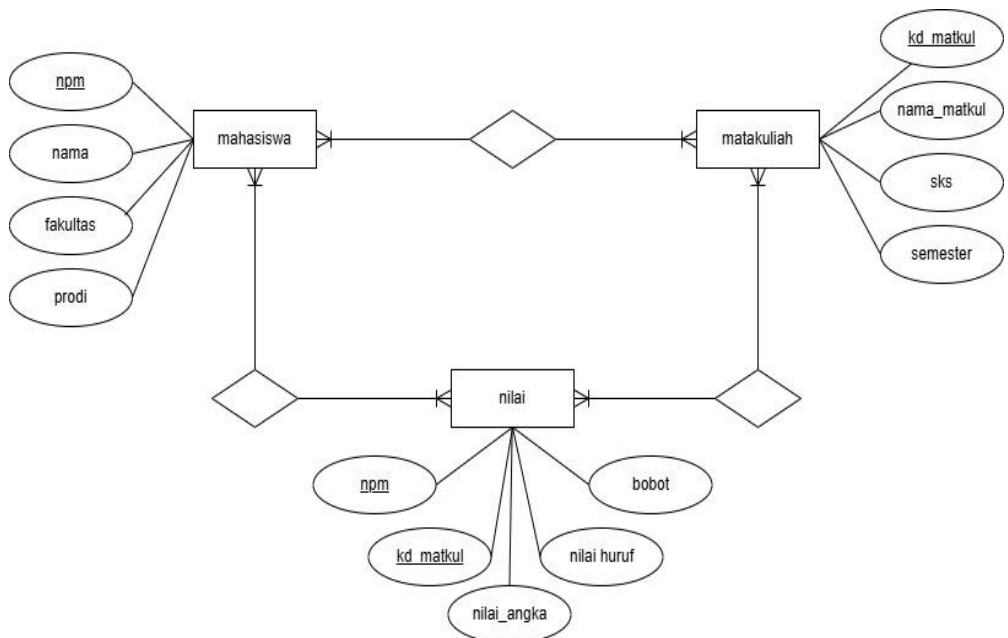
A. Sistem Akademik Perguruan Tinggi

- a) **DFD**: pendaftaran, pengisian KRS, pengolahan nilai, penerbitan transkrip.



Gambar 10. DFD Sistem Akademik Perguruan Tinggi

- b) **ERD**: entitas Mahasiswa, Dosen, MataKuliah, Kelas; relasi pengambilan, pengampu, dan penjadwalan.



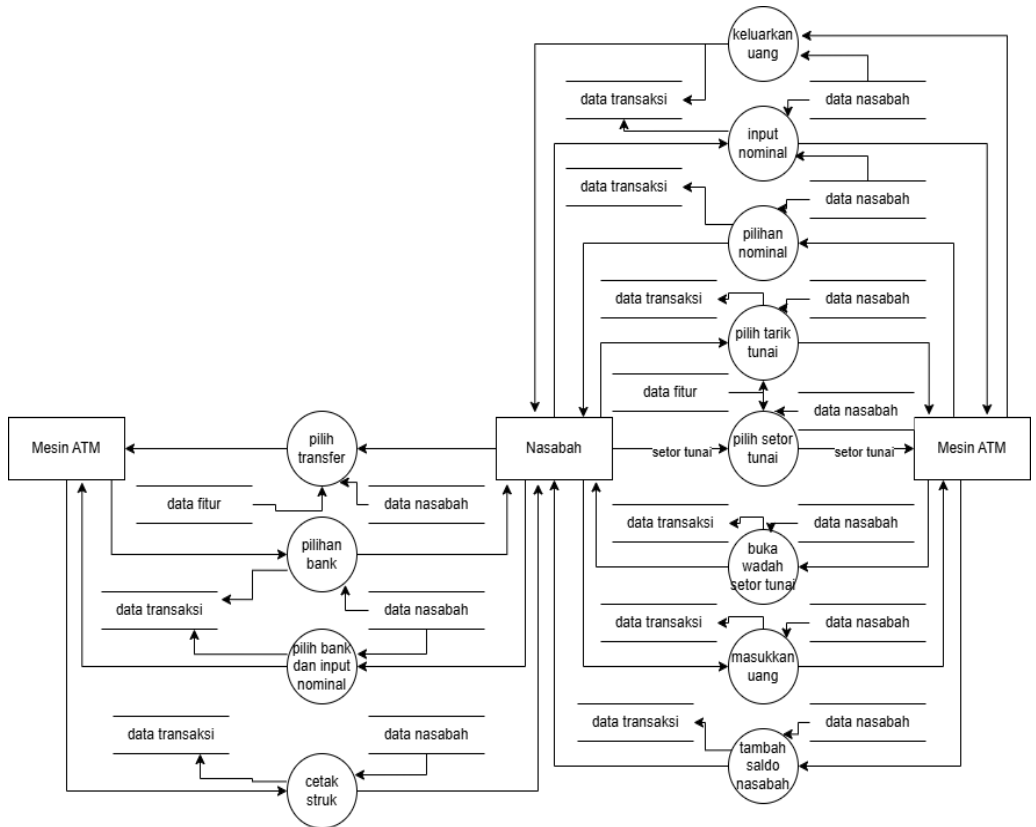
Gambar 11. ERD Sistem Akademik

c) **Kesesuaian**

Penyimpanan **Data Nilai** pada DFD berpadanan dengan tabel **Nilai** pada ERD; aliran **Input Nilai** dari proses dosen harus selaras dengan relasi Mahasiswa-matakuliah.

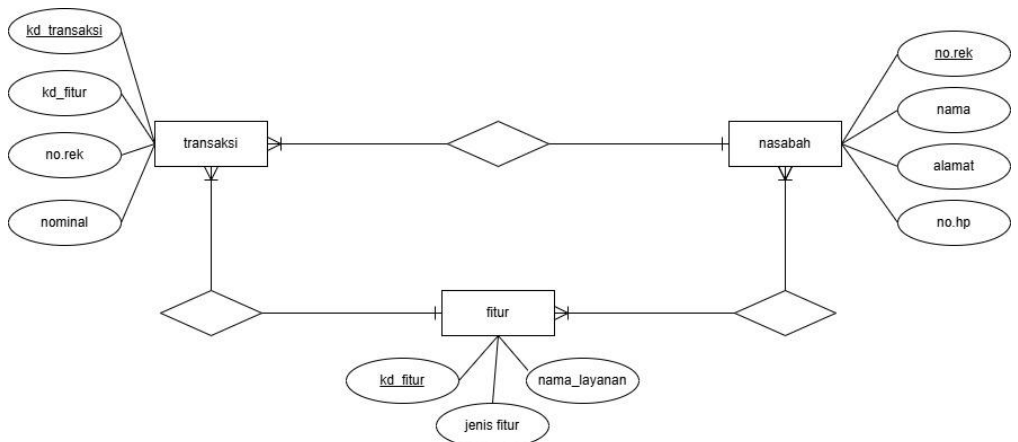
B. Sistem Perbankan

- a) **DFD**: setoran, penarikan, transfer, dan pembuatan laporan.



Gambar 12. DFD LVL 0 Sistem Perbankan

b) **ERD**: entitas Nasabah, Rekening, Transaksi; relasi kepemilikan dan histori transaksi.

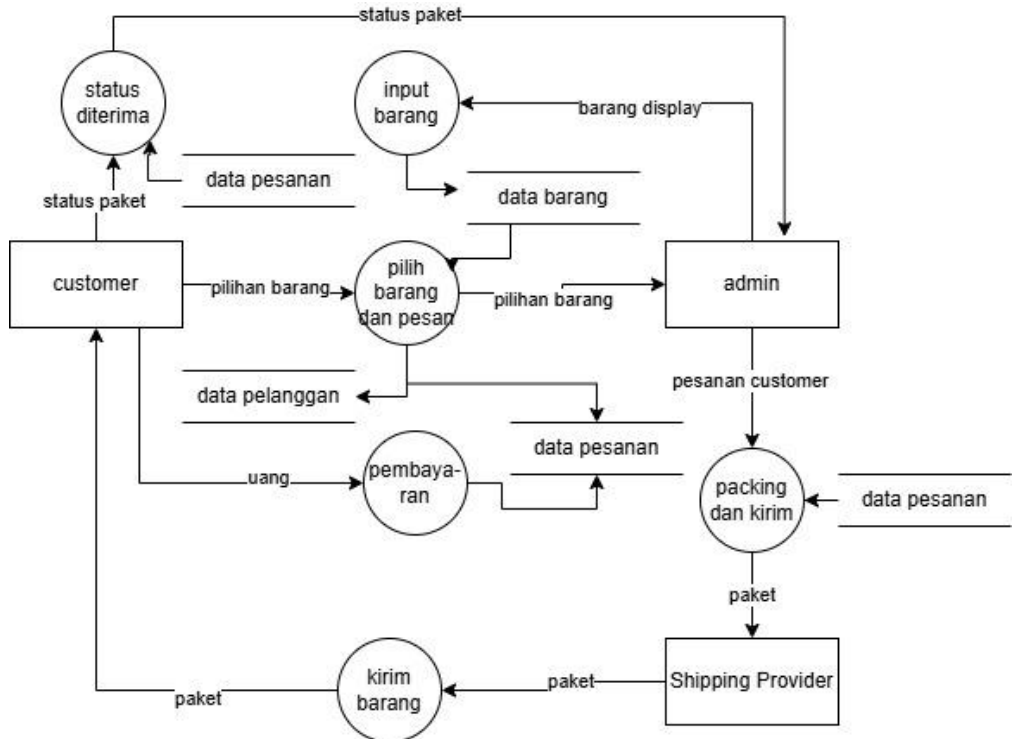


Gambar 13. ERD Sistem Perbankan

- c) **Kesesuaian**: aliran **Debit/Kredit** pada DFD harus bersesuaian dengan keterkaitan Rekening-Transaksi pada ERD agar saldo **terjaga** konsistensinya.

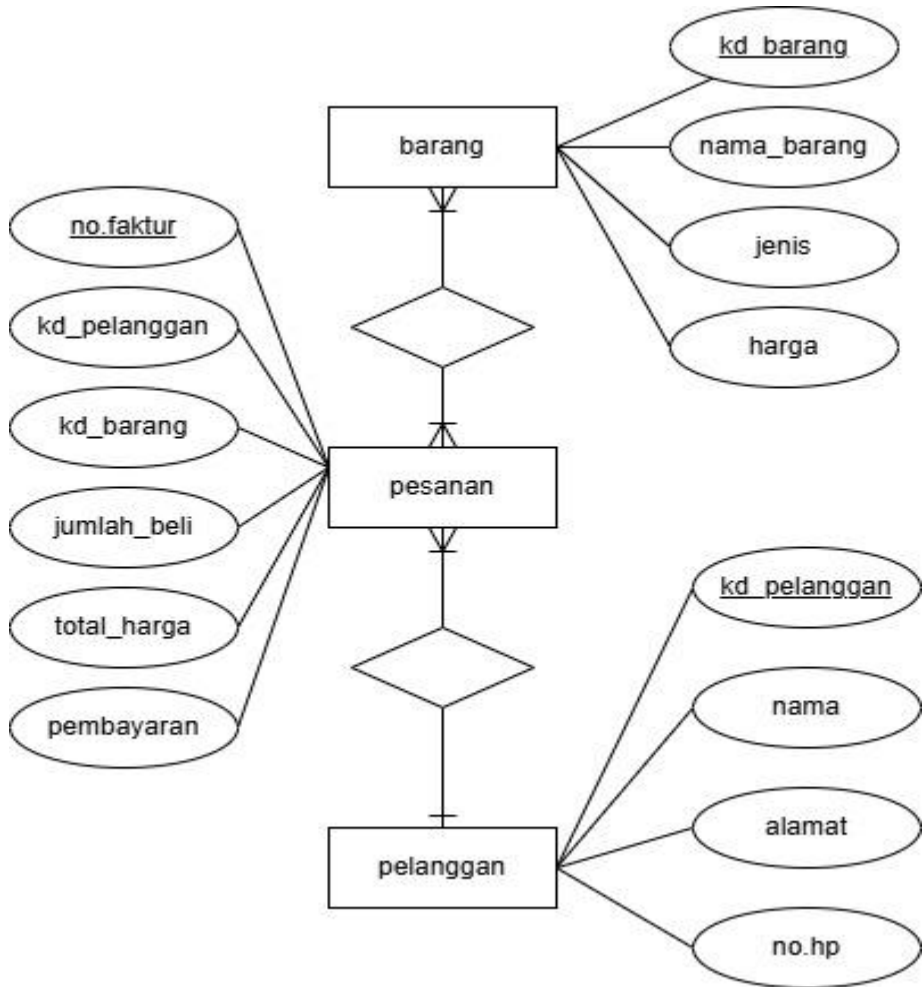
C. Sistem E-Commerce

- a) **DFD**: pemilihan produk, pemesanan, pembayaran, pengiriman.



Gambar 14.DFD Lvl 0 Sistem E-Commerce

- b) **ERD**: entitas Pengguna, Produk, Pesanan, Pembayaran; relasi pesanan berisi banyak detail barang.



Gambar 15.ERD Sistem *E-Commerce*

- c) **Kesesuaian:** penyimpanan **Data Pesanan** pada DFD dipetakan ke tabel **Pesanan** dan **DetailPesanan** pada ERD, menandai pola *one-to-many* yang **jelas**.

Tantangan Umum dan Cara Menanganinya

1. **Kompleksitas proses:** gunakan *decomposition* bertahap dan pastikan *balancing* saat berpindah level.

2. **Perbedaan persepsi:** rujuk **kamus data** dan lakukan tinjauan bersama pemangku kepentingan.
3. **Redundansi data:** tata ulang ERD dengan *normalization* agar duplikasi **berkurang**.
4. **Ketidaksesuaian DFD-ERD:** lakukan penelusuran *traceability* dari aliran ke entitas hingga ditemukan titik tidak selaras, lalu perbaiki.
5. **Dokumentasi tersebar:** konsolidasikan artefak dalam repositori tunggal dengan versi yang **terkendali**.

Prinsip-prinsip ini ditekankan dalam *Analisa Perancangan Sistem Informasi* untuk menjaga mutu rancangan dari awal hingga siap diwujudkan.

Soal Uraian

1. Jelaskan pengertian Data Flow Diagram (DFD) dan berikan satu contoh penggunaannya dalam sistem manajemen rumah sakit.
2. Apa peran utama Entity Relationship Diagram (ERD) dalam merancang basis data, serta berikan contoh penerapannya pada sistem perpustakaan digital.
3. Bagaimana keterkaitan antara DFD dan ERD dalam memastikan keselarasan antara proses bisnis dan struktur data?
4. Sebutkan dan jelaskan empat simbol utama dalam DFD disertai fungsi masing-masing.
5. Mengapa ERD dianggap penting dalam perancangan basis data relasional, dan apa dampaknya jika tidak digunakan?

6. Apa perbedaan antara entitas dan atribut dalam ERD, sertakan contoh untuk masing-masing.
7. Bagaimana langkah-langkah menyusun DFD level 0 pada sistem layanan transportasi daring?
8. Berikan dua contoh entitas yang mungkin ditemukan dalam sistem informasi rumah sakit, beserta penjelasan singkat.
9. Apa tujuan utama pemodelan sistem dalam tahap analisis dan desain perangkat lunak?
10. Bagaimana pemodelan dengan DFD dan ERD dapat meningkatkan efektivitas komunikasi antara analis sistem, pengembang, dan pemangku kepentingan non-teknis?

Tugas Kelompok

Mahasiswa mampu menyusun **DFD level 0, level 1, level 2**, serta **ERD** untuk sistem perpustakaan digital dengan penamaan yang konsisten, alur data yang jelas, dan struktur data yang tertata.

Ketentuan Umum

1. Bekerja dalam kelompok **4-5 orang**.
2. Gunakan notasi standar untuk *Data Flow Diagram (DFD)* dan *Entity Relationship Diagram (ERD)*.
3. Istilah asing ditulis *italic*; kata penting ditulis **tebal**.
4. Jaga konsistensi: nama proses (kata kerja), nama *data store* (kata benda jamak), nama alur data (kata benda), dan nama entitas (kata benda tunggal).

5. Sertakan **kamus data** ringkas untuk setiap alur data dan *data store*.

Ruang Lingkup Sistem Perpustakaan Digital

Aktivitas utama meliputi: **registrasi anggota, pencarian koleksi, peminjaman, pengembalian, reservasi, denda, pengadaan, dan pelaporan**. Pihak yang terlibat: **Anggota, Pustakawan, Admin Sistem, Pemasok**. Sumber data: data anggota, koleksi buku, transaksi peminjaman/pengembalian, pembayaran denda, data pengadaan.

Bagian A

DFD Level 0 (Context Diagram)

Tugas: Gambar satu proses utama, daftar entitas eksternal, serta alur data masuk/keluar.

- a) Proses utama: **Sistem Perpustakaan Digital**.
- b) Entitas eksternal minimal: **Anggota, Pustakawan, Pemasok** (opsional **Pembayaran Denda/Payment Gateway** bila digunakan).
- c) Contoh alur data masuk: **Form Registrasi, Permintaan Pencarian, Permintaan Peminjaman, Pengembalian**.
- d) Contoh alur data keluar: **Status Peminjaman, Hasil Pencarian, Slip Pengembalian, Ringkasan Denda**.
- e) **Keluaran yang diharapkan:** satu diagram rapi + daftar alur data (bullet) + kamus data ringkas (maks. 1-2 kalimat per item).

Bagian B

DFD Level 1

Tugas: Uraikan proses utama menjadi 4-6 subproses.

- a) Contoh subproses: **Kelola Anggota, Kelola Koleksi, Kelola Peminjaman, Kelola Pengembalian, Kelola Denda, Kelola Pelaporan.**
- b) Tambahkan *data store* utama: **Data Anggota, Data Koleksi, Data Peminjaman, Data Pengembalian, Data Denda.**
- c) Jaga *balancing*: seluruh alur pada level 0 harus muncul/logis pada level 1.
- d) **Keluaran yang diharapkan**: satu diagram + tabel *data store* (nama, isi pokok) + revisi kamus data bila diperlukan.

Bagian C

DFD Level 2

Tugas: Pilih **dua** subproses dari Level 1 dan uraikan detailnya.

- a) Rekomendasi: **Kelola Peminjaman** dan **Kelola Pengembalian.**
- b) Contoh rincian **Kelola Peminjaman**: verifikasi status anggota → cek ketersediaan → catat transaksi → terbitkan bukti → perbarui stok.
- c) Contoh rincian **Kelola Pengembalian**: terima buku → cek keterlambatan → hitung denda → catat pembayaran (bila ada) → perbarui stok.
- d) Tambahkan alur kontrol yang relevan (mis. **Notifikasi Jatuh Tempo** bila ingin ditampilkan sebagai alur data terstruktur).
- e) **Keluaran yang diharapkan**: dua diagram Level 2 + daftar alur baru yang muncul + penyelarasan kamus data.

Bagian D

ERD (Entity Relationship Diagram)

Tugas: Susun ERD yang selaras dengan DFD.

- a) **Entitas inti (minimal):** Anggota, Buku, Eksemplar, Peminjaman, Pengembalian, Denda, Pustakawan, Pemasok.
- b) **Atribut contoh:**
 - a) Anggota(id_anggota*, nama, alamat, email, status)
 - b) Buku(id_buku*, judul, pengarang, penerbit, tahun, isbn, kategori)
 - c) Eksemplar(id_eksemplar*, id_buku FK, lokasi_rak, status_eksemplar)
 - d) Peminjaman(id_pinjam*, id_anggota FK, id_eksemplar FK, tgl_pinjam, tgl_jatuh_tempo, status)
 - e) Pengembalian(id_kembali*, id_pinjam FK, tgl_kembali, kondisi_buku)
 - f) Denda(id_denda*, id_pinjam FK, nominal, status_bayar, tgl_bayar)
 - g) Pustakawan(id_pustakawan*, nama, jabatan)
 - h) Pemasok(id_pemasok*, nama, kontak)
- c) **Relasi pokok:**
 - a) Buku **1..n** Eksemplar
 - b) Anggota **1..n** Peminjaman
 - c) Eksemplar **1..n** Peminjaman
 - d) Peminjaman **1..1** Pengembalian
 - e) Peminjaman **0..1** Denda
 - f) Pemasok **1..n** Buku (untuk pengadaan)
- d) Terapkan *primary key* dan *foreign key* dengan penamaan konsisten.

- e) **Keluaran yang diharapkan:** satu diagram ERD + daftar atribut tiap entitas + daftar *primary/foreign key*.

Kamus Data

Cantumkan tabel berisi: **Nama Item, Jenis** (alur/store/entitas), **Deskripsi, Struktur Ringkas** (mis. format kode, panjang, tipe), **Sumber/Tujuan**.

Aturan Penamaan & Gaya

1. Proses: **kata kerja + objek** (mis. **Verifikasi Keanggotaan**).
2. *Data store*: **kata benda jamak** (mis. **Data Peminjaman**).
3. Alur data: **kata benda** (mis. **Permintaan Peminjaman**).
4. Entitas: **kata benda tunggal** (mis. **Anggota**).
5. Gunakan bahasa Indonesia baku sesuai KBBI dan PUEBI.

Asumsi yang Diperbolehkan

1. Denda dihitung **per hari** keterlambatan.
2. Satu **eksemplar** hanya dapat dipinjam oleh **satu anggota** pada waktu yang sama.
3. Reservasi hangus bila buku tidak diambil dalam **2 hari kerja**.
4. Opsi pembayaran denda dapat dicatat manual atau melalui *payment gateway*.

Format Pengumpulan

1. Satu berkas **PDF** berisi: halaman judul, DFD Level 0-2, ERD, kamus data ringkas, daftar asumsi.

2. Sertakan **halaman orisinalitas** berisi pernyataan bahwa rancangan dibuat oleh kelompok.
3. Penamaan berkas:
DFD_ERD_Perpustakaan_NamaKelompok.pdf.

Pemodelan dengan UML

A. CPL

Mampu menggunakan pemodelan berorientasi objek.

B. CPMK

Membuat diagram UML.

C. Sub-CPMK

Menjelaskan kegunaan class diagram, use case, activity.

Pemodelan sistem perangkat lunak merupakan bagian yang tidak terpisahkan dalam proses **perancangan sistem informasi**. Untuk menjaga agar rancangan tetap konsisten, terstruktur, dan dapat dipahami oleh semua pemangku kepentingan, digunakanlah **Unified Modeling Language (UML)**. UML adalah bahasa pemodelan standar yang menyediakan seperangkat notasi grafis untuk memvisualisasikan, merancang, serta mendokumentasikan sistem perangkat lunak. Dalam literatur *Analisa Perancangan Sistem Informasi*, UML ditekankan sebagai sarana untuk menyelaraskan kebutuhan bisnis dengan rancangan teknis.

Keunggulan UML terletak pada sifatnya yang fleksibel, sehingga dapat digunakan untuk berbagai jenis sistem, mulai dari aplikasi sederhana hingga sistem skala besar. Pemahaman mendalam mengenai UML membantu mahasiswa maupun profesional dalam menata rancangan dengan cara yang lebih efektif dan efisien.

Pentingnya UML dalam Pemodelan

UML bukan sekadar alat bantu visual, melainkan kerangka konseptual yang menyatukan berbagai pandangan mengenai sistem. Dengan UML, rancangan tidak lagi bergantung pada interpretasi individu, tetapi ditampilkan dalam bentuk diagram yang memiliki standar internasional. Keberadaan standar ini menjadikan UML sebagai bahasa universal dalam dunia pengembangan perangkat lunak.

Dalam *Analisa Perancangan Sistem Informasi*, UML dipandang sebagai instrumen untuk memperjelas komunikasi lintas fungsi. Oleh karena itu, **UML menjadi sarana komunikasi utama** antara analis, desainer, pengembang, serta pemangku kepentingan non-teknis.

Manfaat penggunaan UML antara lain:

1. **Meningkatkan komunikasi** antar anggota tim pengembang.
2. **Mengurangi ambiguitas** dalam interpretasi kebutuhan.
3. **Menyediakan dokumentasi formal** yang dapat dipertanggungjawabkan.
4. **Membantu validasi dan verifikasi** kebutuhan pengguna.
5. **Mempercepat pengembangan** karena rancangan yang jelas meminimalisasi revisi di tahap implementasi.

Komponen Utama UML

Dalam kerangka UML, terdapat berbagai jenis diagram yang dapat digunakan sesuai tujuan. Namun, dalam materi ini difokuskan pada tiga jenis utama, yaitu **use case diagram**, **class diagram**, dan **activity diagram**. Ketiga diagram ini dipilih karena mampu

memberikan gambaran yang menyeluruh, mulai dari interaksi pengguna dengan sistem, struktur internal objek, hingga alur aktivitas yang terjadi. *Analisa Perancangan Sistem Informasi* menekankan ketiganya sebagai fondasi pemodelan awal.

Use Case Diagram

Use case diagram adalah diagram yang digunakan untuk menggambarkan interaksi antara aktor dengan sistem. Diagram ini menjawab pertanyaan: siapa yang menggunakan sistem dan apa yang dapat dilakukan oleh pengguna tersebut. Dengan demikian, use case diagram menjadi titik awal yang penting dalam memahami kebutuhan sistem.

A. Komponen Utama

a) Aktor

Pihak yang berinteraksi dengan sistem, bisa berupa manusia, sistem lain, atau perangkat eksternal.

b) Use Case

Layanan atau fungsi yang diberikan oleh sistem untuk aktor.

c) Hubungan

Garis yang menghubungkan aktor dengan use case.

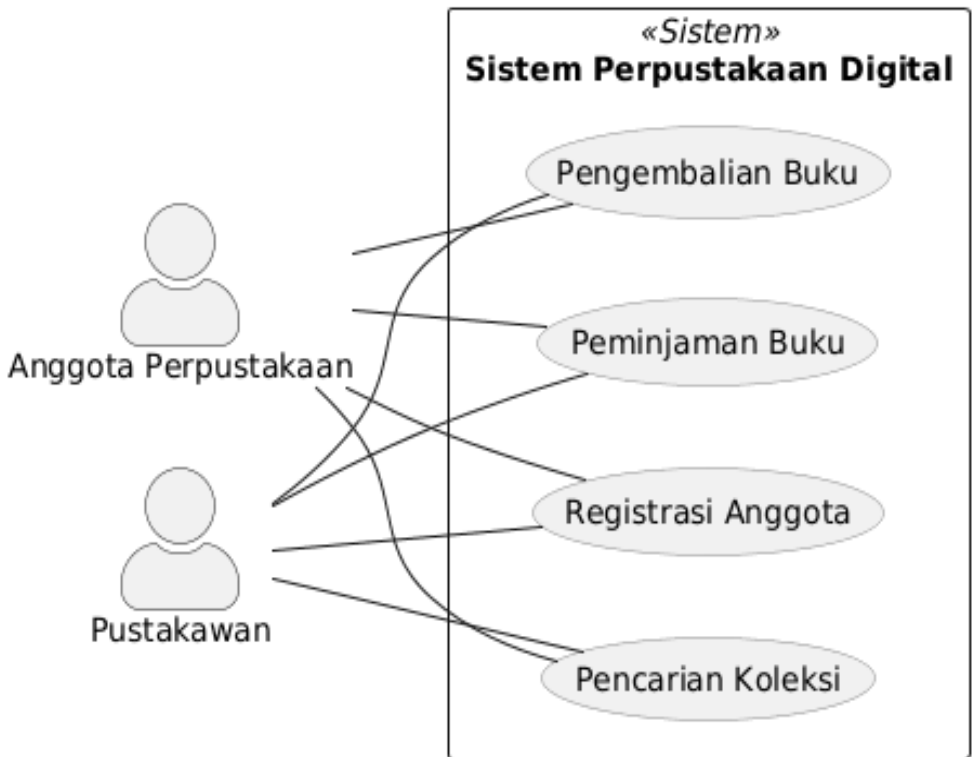
d) Boundary System

Kotak yang menggambarkan batas sistem, memisahkan antara bagian internal dan eksternal.

B. Kegunaan Use Case Diagram

- Mengidentifikasi kebutuhan sistem dari perspektif pengguna.
- Mempermudah komunikasi antara pengembang dan pengguna.
- Menjadi dasar bagi perancangan diagram lainnya.

C. Contoh Penerapan



Gambar 16. Contoh Use Case Diagram

Dalam sistem perpustakaan digital, aktor utama adalah **anggota perpustakaan** dan **pustakawan**. Use case yang relevan meliputi *registrasi anggota*, *peminjaman buku*, *pengembalian buku*, dan *pencarian koleksi*. Dalam *Analisa Perancangan Sistem Informasi*, use case semacam ini dikaitkan langsung dengan kebutuhan fungsional yang akan dijabarkan lebih rinci dalam diagram lain.

Class Diagram

Class diagram menggambarkan struktur objek dalam sistem, termasuk atribut, metode, serta hubungan antar kelas. Diagram ini menjawab pertanyaan: apa saja objek yang ada dalam sistem dan bagaimana mereka saling terhubung.

A. Komponen Utama

a) **Kelas**

Digambarkan sebagai persegi panjang yang terbagi menjadi tiga bagian (nama kelas, atribut, metode).

b) **Atribut**

Variabel atau data yang dimiliki kelas.

c) **Metode**

Fungsi atau perilaku yang dimiliki kelas.

d) **Relasi**

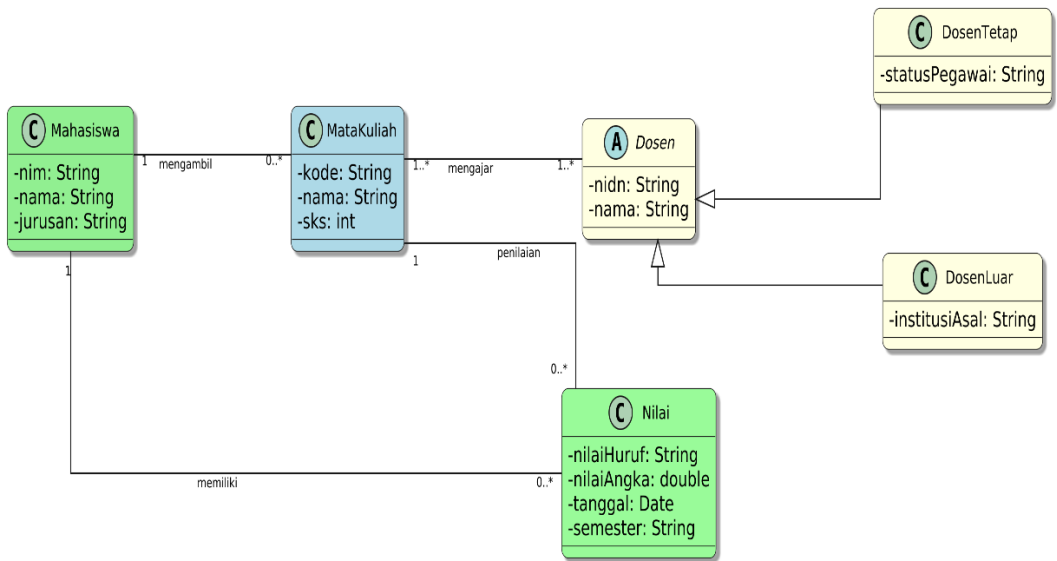
Garis penghubung yang menunjukkan keterkaitan antar kelas, meliputi asosiasi, pewarisan (*inheritance*), komposisi, dan agregasi.

B. Kegunaan Class Diagram

- a) Merancang struktur logis perangkat lunak.
- b) Menentukan tanggung jawab setiap objek dalam sistem.
- c) Menjadi dasar implementasi kode program.

C. Contoh Penerapan

Class Diagram Sistem Akademik



Gambar 17. Contoh Class Diagram

Pada sistem akademik, kelas yang terlibat antara lain Mahasiswa, Dosen, MataKuliah, dan Nilai. Relasi pewarisan dapat digunakan untuk menggambarkan bahwa *DosenTetap* dan *DosenLuar* merupakan turunan dari kelas *Dosen*. Dengan class diagram, struktur data dan perilaku sistem dapat digambarkan secara jelas dan sistematis. *Analisa Perancangan Sistem Informasi* menekankan bahwa class diagram harus menjaga konsistensi dengan kebutuhan yang ditentukan pada use case.

Activity Diagram

Activity diagram digunakan untuk menggambarkan alur aktivitas dalam sistem. Diagram ini mirip dengan diagram alir

(*flowchart*), tetapi lebih fokus pada aktivitas yang dilakukan oleh sistem maupun pengguna.

A. Komponen Utama

a) **Initial Node**

Lingkaran hitam sebagai titik awal.

b) **Activity**

Persegi panjang dengan sudut tumpul untuk menggambarkan aktivitas.

c) **Decision Node**

Belah ketupat untuk menunjukkan percabangan.

d) **Flow/Transition**

Panah yang menunjukkan alur perpindahan aktivitas.

e) **Final Node**

Lingkaran hitam dengan lingkaran luar sebagai titik akhir.

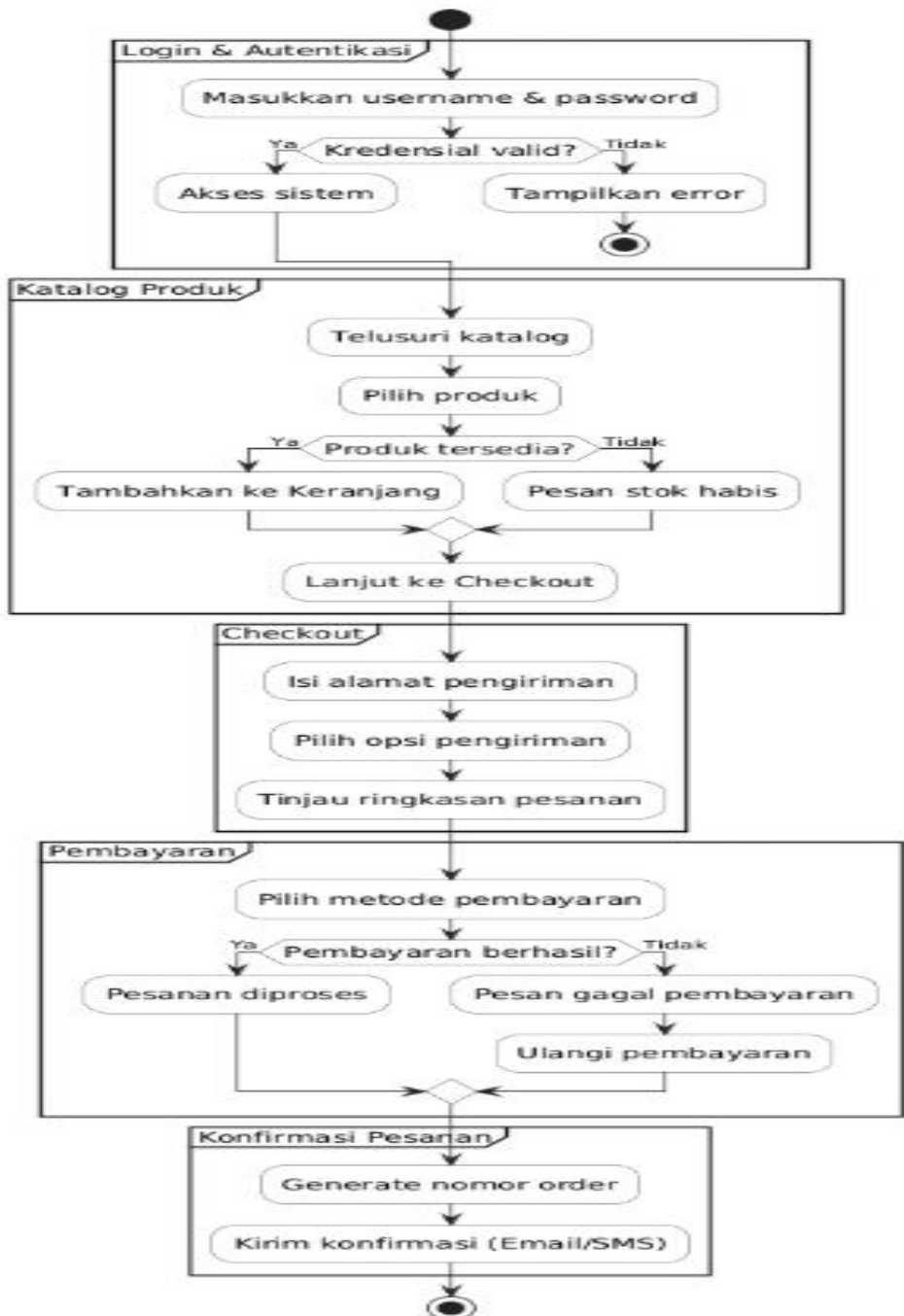
B. Kegunaan Activity Diagram

- a) Memvisualisasikan alur kerja sistem secara detail.
- b) Membantu memahami proses bisnis yang kompleks.
- c) Mengidentifikasi aktivitas paralel maupun bercabang.

C. Contoh Penerapan

Dalam sistem e-commerce, alur aktivitas meliputi *login pengguna, memilih produk, menambahkan ke keranjang, melakukan pembayaran, dan menerima konfirmasi pesanan*. Dengan activity diagram, alur tersebut dapat divisualisasikan secara jelas. Dalam *Analisa Perancangan Sistem Informasi*,

diagram aktivitas dipandang penting untuk menggambarkan dinamika proses bisnis dan menilai efisiensi alur kerja.



Gambar 18. Contoh Activity Diagram

Hubungan Antara Use Case, Class, dan Activity Diagram

Ketiga diagram UML tersebut tidak berdiri sendiri, melainkan saling mendukung. **Use case diagram** memberikan gambaran kebutuhan pengguna. **Class diagram** menunjukkan bagaimana kebutuhan tersebut diimplementasikan dalam bentuk struktur objek. **Activity diagram** menampilkan alur aktivitas yang terjadi ketika kebutuhan dijalankan. Hubungan ini memastikan bahwa rancangan sistem mencerminkan kebutuhan pengguna sekaligus dapat diimplementasikan secara teknis. *Analisa Perancangan Sistem Informasi* menekankan pentingnya keterpaduan ini agar sistem tidak hanya valid di atas kertas, tetapi juga siap diterapkan.

Sebagai contoh, pada sistem akademik:

- a) *Use case diagram* memperlihatkan bahwa mahasiswa dapat melakukan registrasi.
- b) *Class diagram* menunjukkan kelas Mahasiswa dengan atribut id, nama, dan metode registrasi().
- c) *Activity diagram* menggambarkan langkah-langkah registrasi mulai dari input data hingga verifikasi.

Tantangan dalam Pemodelan UML

Meskipun UML menyediakan kerangka yang jelas, terdapat tantangan dalam penerapannya:

1. Kompleksitas Sistem

Semakin besar sistem, semakin banyak diagram yang diperlukan.

2. Keterbatasan Pemahaman

Tidak semua pemangku kepentingan familiar dengan notasi UML.

3. **Konsistensi Antar Diagram**

Diagram yang berbeda harus tetap konsisten agar tidak menimbulkan konflik interpretasi.

4. **Waktu dan Biaya**

Penyusunan diagram yang lengkap membutuhkan waktu dan sumber daya tambahan.

Unified Modeling Language (UML) merupakan standar internasional dalam pemodelan sistem berbasis objek. Dengan memanfaatkan **use case diagram**, **class diagram**, dan **activity diagram**, pengembang dapat menyusun rancangan yang menyeluruh, mulai dari kebutuhan pengguna hingga alur aktivitas sistem. UML membantu menjaga konsistensi, mengurangi ambiguitas, serta meningkatkan komunikasi antar pihak yang terlibat. *Analisa Perancangan Sistem Informasi* menegaskan bahwa pemodelan dengan UML memberikan dasar konseptual yang kuat dalam perancangan sistem informasi. Bagi mahasiswa, pemahaman tentang UML akan menjadi fondasi penting untuk mengembangkan keterampilan analitis dan praktis, sehingga mampu menghadapi tantangan di dunia kerja maupun penelitian lebih lanjut.

Soal Uraian

1. Apa yang dimaksud dengan *Unified Modeling Language (UML)* dan sebutkan satu contoh penggunaannya dalam pengembangan perangkat lunak manajemen rumah sakit.
2. Jelaskan fungsi utama dari use case diagram serta berikan ilustrasi penggunaannya pada sistem pemesanan tiket daring.
3. Sebutkan dan jelaskan tiga komponen utama yang ada dalam class diagram dengan contoh kelas Mahasiswa.
4. Apa perbedaan mendasar antara activity diagram dan sequence diagram, sertakan contoh sederhana untuk masing-masing.
5. Mengapa UML dianggap penting dalam perancangan sistem informasi skala besar seperti sistem e-commerce?
6. Bagaimana penggunaan UML dapat mendukung komunikasi efektif antar anggota tim dalam proyek pengembangan aplikasi mobile?
7. Apa yang ditunjukkan oleh use case diagram dalam sistem pembayaran digital, dan siapa aktor yang terlibat?
8. Berikan contoh class diagram sederhana untuk menggambarkan sistem absensi sekolah.
9. Apa keuntungan yang diperoleh organisasi dengan menggunakan UML dalam dokumentasi proyek perangkat lunak?
10. Bagaimana cara membaca activity diagram pada sistem registrasi mahasiswa baru, mulai dari titik awal hingga titik akhir?

Tugas Kelompok

Mahasiswa mampu merancang **use case diagram**, **class diagram**, dan **activity diagram** untuk studi kasus **Sistem Pembayaran Kuliah**, dengan penamaan yang konsisten, keterpaduan antar-diagram, serta dokumentasi yang rapi sesuai kaidah *perancangan sistem informasi*.

Deskripsi Singkat Studi Kasus

Perguruan tinggi memerlukan sistem untuk mengelola **tagihan UKT/SPP**, **beasiswa/potongan**, **cara bayar** (loket, *virtual account*, *e-wallet*), **verifikasi pembayaran**, **penerbitan bukti bayar**, dan **pelaporan keuangan**. Sistem juga harus berinteraksi dengan **Sistem Akademik** (untuk status registrasi) dan **Gateway Pembayaran** (untuk notifikasi transaksi). Fokus tugas: memodelkan kebutuhan dan alur utama pembayaran kuliah secara jelas.

Lingkup Fungsional

1. **Manajemen Tagihan**: pembuatan tagihan per mahasiswa per semester, skema cicilan/termin, potongan beasiswa.
2. **Pembayaran**: pilihan metode, validasi nominal, pengunggahan bukti (bila manual), atau *callback* dari *payment gateway*.
3. **Verifikasi & Rekonsiliasi**: otomatis/manual, pencatatan status berhasil/gagal, penanganan pembayaran ganda.
4. **Bukti & Pelaporan**: penerbitan **kuitansi digital**, ringkasan per mahasiswa, rekap fakultas/semester, ekspor ke sistem keuangan.

5. **Keterkaitan Akademik:** status **registrasi** aktif setelah lunas/minimal cicilan terpenuhi; blokir KRS bila tunggakan ada.

Bagian A

Use Case Diagram

Tugas: Gambar *use case diagram* lengkap dengan aktor, *use case*, *system boundary*, dan relasi (*include/extend* bila relevan).

a) **Aktor minimum**

Mahasiswa, Bendahara/Keuangan, Admin Sistem, Gateway Pembayaran, Sistem Akademik.

b) **Use case**

Lihat Tagihan, Pilih Metode Pembayaran, Bayar UKT, Unggah Bukti Bayar, Verifikasi Pembayaran, Terbitkan Kuitansi, Rekonsiliasi Transaksi, Sinkron Status Registrasi.

c) **Keluaran yang diharapkan**

1 diagram rapi + daftar definisi singkat setiap aktor dan *use case* (1-2 kalimat) untuk menjaga keseragaman makna.

Bagian B

Class Diagram

Tugas

Rancang **class diagram** yang menyokong *use case* di atas, lengkap dengan atribut, operasi, relasi, dan *multiplicity*.

a) **Kelas Diagram**

Mahasiswa, Tagihan, RincianTagihan, Pembayaran, MetodePembayaran, Kuitansi, Beasiswa, Pengguna, Peran, TransaksiGateway, LogRekonsiliasi.

b) **Atribut**

- 1) Tagihan(id_tagihan*, semester, total_tagihan, status)
- 2) RincianTagihan(id_rincian*, id_tagihan FK, deskripsi, nominal)
- 3) Pembayaran(id_bayar*, id_tagihan FK, tanggal, nominal_bayar, kanal, status)
- 4) Beasiswa(id_beasiswa*, nama, persentase, periode)
- 5) Kuitansi(id_kuitansi*, id_bayar FK, nomor_kuitansi, url_dokumen)

c) **Relasi kunci:**

- a. Mahasiswa **1..n** Tagihan
- b. Tagihan **1..n** RincianTagihan
- c. Tagihan **0..n** Pembayaran
- d. Pembayaran **0..1** Kuitansi
- e. Mahasiswa **0..n** Beasiswa (melalui tabel asosiasi bila diperlukan)

Catatan desain

Gunakan *enumeration* untuk status (mis. Draft, terbit, lunas, tunggak), terapkan *primary key/foreign key* yang konsisten, dan pisahkan tanggung jawab kelas agar *low coupling* dan *high cohesion* terjaga.

Bagian C

Activity Diagram

Tugas

Buat **tiga** *activity diagram* terpisah untuk alur berikut:

1. **Terbitkan Tagihan Semester** (oleh Bendahara/otomatis):
mulai → ambil data mahasiswa → hitung komponen biaya →

terapkan beasiswa/potongan → terbitkan tagihan → notifikasi mahasiswa → selesai.

2. **Proses Pembayaran oleh Mahasiswa:** mulai → lihat tagihan → pilih metode → lakukan bayar (unggah bukti / redirect ke *gateway*) → terima notifikasi/unggah bukti → tampilkan status → jika berhasil, unduh kuitansi → selesai.
3. **Verifikasi & Rekonsiliasi:** mulai → cek notifikasi *gateway* / validasi bukti → cocokkan nominal dan ID transaksi → tandai status → jika ganda/salah nominal, buat tiket koreksi → sinkron ke **Sistem Akademik** → selesai.

Gunakan **initial/final node, decision/merge, swimlane** (Mahasiswa/Keuangan/Gateway) untuk memperjelas peran.

Kamus Istilah & Aturan Penamaan

- a) Istilah asing ditulis *italic*: *use case, actor, activity, payment gateway, callback*.
- b) **Nama proses/aktivitas:** kata kerja + objek (mis. **Validasi Pembayaran**).
- c) **Nama kelas/entitas:** kata benda tunggal (mis. **Tagihan, Pembayaran**).
- d) **Atribut:** *snake_case* atau *camelCase* konsisten di seluruh diagram.

Asumsi yang Diperbolehkan

1. Pembayaran dapat **parsial/cicilan** sesuai kebijakan.
2. Notifikasi dari *gateway* menggunakan *callback* yang berisi **id_transaksi, status, nominal**.

3. Registrasi akademik **aktif** setelah status tagihan **LUNAS** atau **MINIMAL_BAYAR** terpenuhi.
4. Koreksi transaksi dilakukan melalui *workflow* tiket internal.

Output yang Harus Dikumpulkan

1. 1 **Use Case Diagram** + tabel definisi aktor/*use case*.
2. 1 **Class Diagram** + daftar atribut/operasi + daftar relasi dan *multiplicity*.
3. 3 **Activity Diagram** (sesuai tiga alur).
4. **Kamus Data/Glosarium** istilah dan status.
5. **Ringkasan Konsistensi** (maks. 1 halaman) yang menjelaskan keterpaduan *use case* → *class* → *activity*.

Format berkas: **PDF** (A4), orientasi potret, font 11-12 pt, margin 2.5 cm. Nama berkas: UML_PembayaranKuliah_NamaKelompok.pdf.

Desain Sistem Informasi

A. CPL

Mahasiswa memahami rancangan awal hingga pembuatan purwarupa sistem informasi.

B. CPMK

Menyusun desain konseptual, logis, fisik, serta membuat prototype sistem informasi.

C. Sub-CPMK

1. Menentukan komponen utama sistem dari hasil identifikasi.
2. Menentukan spesifikasi teknis.
3. Menguji desain dengan model sederhana (prototype).

Perancangan sistem informasi adalah fondasi utama dalam pengembangan perangkat lunak yang terarah dan efektif. Tahapan ini bukan hanya sekadar kegiatan teknis, melainkan proses sistematis yang menghubungkan kebutuhan pengguna dengan solusi teknologi yang akan dibangun. **Desain sistem informasi** terdiri atas beberapa lapisan penting, mulai dari **desain konseptual**, **desain logis**, **desain fisik**, hingga tahap pembuatan *prototype*. Setiap lapisan memiliki fungsi yang berbeda, tetapi saling terkait untuk menciptakan sistem yang utuh dan konsisten. Dalam pembahasan ini, akan dipaparkan secara mendalam bagaimana tahapan desain disusun, apa saja komponen yang terlibat, dan bagaimana pengujian awal dilakukan melalui model sederhana. Penjelasan ini diperkuat dengan acuan dari *Analisa Perancangan Sistem Informasi*, yang

menekankan pentingnya rancangan terstruktur untuk memastikan kesesuaian antara kebutuhan pengguna dan solusi teknologi.

Desain Konseptual

Desain konseptual merupakan langkah awal yang menyajikan **gambaran umum solusi sistem informasi**. Pada tahap ini, perancang belum masuk pada detail teknis, melainkan berfokus pada identifikasi komponen utama yang diperlukan. Tujuan utamanya adalah menyatukan pemahaman antara pengembang dan pemangku kepentingan mengenai bentuk sistem yang akan dibangun. Literatur *Analisa Perancangan Sistem Informasi* menekankan bahwa desain konseptual berfungsi sebagai jembatan komunikasi yang mengurangi perbedaan persepsi.

Komponen Utama

Beberapa komponen penting yang dirumuskan dalam desain konseptual antara lain:

a) **Basis data**

Menggambarkan kebutuhan data yang akan disimpan dan dikelola.

b) **Antarmuka**

Menampilkan rancangan awal interaksi pengguna dengan sistem.

c) **Proses utama**

Mendefinisikan aktivitas inti yang harus dijalankan oleh sistem.

Peran Diagram Konseptual

Diagram konseptual sering digunakan pada tahap ini untuk mempermudah komunikasi. Diagram dapat berupa diagram alir, diagram konteks, atau model entitas sederhana. Fungsinya adalah menyajikan informasi dalam bentuk visual sehingga lebih mudah dipahami oleh berbagai pihak, termasuk mereka yang tidak berlatar belakang teknis. Dalam *Analisa Perancangan Sistem Informasi*, penggunaan diagram ini dipandang sebagai praktik yang wajib untuk menyamakan persepsi awal.

Validasi

Validasi desain konseptual dilakukan bersama pemangku kepentingan. Proses ini melibatkan diskusi, klarifikasi kebutuhan, dan penyesuaian agar rancangan tetap sesuai dengan tujuan organisasi. Validasi sejak awal penting untuk mencegah terjadinya kesalahan rancangan yang dapat berdampak besar di tahap implementasi. Keterlibatan aktif pemangku kepentingan sebagaimana diuraikan dalam *Analisa Perancangan Sistem Informasi* menjadi faktor kunci keberhasilan tahap ini.

Desain Logis

Desain logis merupakan tahap lanjutan setelah rancangan konseptual disepakati. Fokus utamanya adalah menjabarkan **struktur data** dan **proses bisnis** secara lebih detail. Dalam desain logis, perancang mulai menyusun skema basis data, diagram relasi entitas, serta alur proses yang lebih rinci.

A. Detail Struktur Data

Salah satu aspek penting dalam desain logis adalah normalisasi data. Proses ini bertujuan untuk:

1. Menghindari **redundansi data**.
2. Menjamin **konsistensi informasi**.
3. Meningkatkan **efisiensi pengelolaan basis data**.

Contohnya, pada sistem akademik, entitas Mahasiswa, MataKuliah, dan Dosen diatur sedemikian rupa agar hubungan antar data tercatat secara jelas. Normalisasi memastikan bahwa setiap data hanya disimpan sekali, tetapi dapat diakses melalui relasi yang sesuai.

B. Proses Bisnis

Desain logis juga menguraikan bagaimana proses bisnis berjalan di dalam sistem. Misalnya, bagaimana alur pendaftaran mahasiswa baru, pengisian KRS, hingga pencetakan transkrip nilai. Semua proses tersebut dituangkan dalam diagram yang menekankan logika kerja tanpa memperhitungkan aspek perangkat keras atau perangkat lunak tertentu.

Desain Fisik

Desain fisik adalah tahap di mana detail teknis mulai dimasukkan ke dalam rancangan. Tahap ini menjawab pertanyaan: *Bagaimana sistem akan diimplementasikan secara nyata?* Jika desain

logis menggambarkan *apa* yang harus dilakukan sistem, maka desain fisik menggambarkan *bagaimana* sistem tersebut diwujudkan.

A. Komponen Desain Fisik

Komponen yang dirancang pada tahap fisik mencakup:

a) Perangkat keras

Spesifikasi server, komputer klien, perangkat jaringan, serta kapasitas penyimpanan.

b) Perangkat lunak

Sistem operasi, perangkat basis data, bahasa pemrograman, serta kerangka kerja (*framework*) yang akan digunakan.

c) Jaringan

Arsitektur jaringan, kebutuhan bandwidth, serta konfigurasi keamanan.

B. Keamanan dan Pemeliharaan

Selain spesifikasi teknis, desain fisik juga menekankan aspek keamanan dan pemeliharaan. Fitur keamanan meliputi mekanisme autentikasi, otorisasi, enkripsi, dan *backup* berkala. Aspek pemeliharaan mencakup rencana pembaruan perangkat lunak serta pemantauan kinerja sistem secara rutin.

C. Hasil Desain Fisik

Output dari desain fisik berupa dokumen spesifikasi lengkap yang akan digunakan sebagai pedoman implementasi.

Dokumen ini memuat semua kebutuhan teknis sehingga tim pengembang dapat membangun sistem sesuai dengan rancangan.

Prototyping Sistem Informasi

Prototyping adalah teknik pembuatan purwarupa sistem yang berfungsi sebagai alat evaluasi. Dengan *prototype*, pengguna dapat melihat bentuk awal sistem dan memberikan masukan sebelum sistem dikembangkan lebih lanjut.

A. Tujuan Prototyping

- a) Memastikan bahwa kebutuhan pengguna telah terakomodasi.
- b) Menyediakan sarana komunikasi visual yang mudah dipahami.
- c) Mengurangi risiko kesalahan implementasi.

B. Jenis Prototype

1. Low Fidelity Prototype

Berupa sketsa sederhana atau *wireframe* yang menggambarkan alur navigasi.

2. High Fidelity Prototype

Model interaktif yang mendekati produk akhir, lengkap dengan desain grafis dan fungsi dasar.

C. Proses Prototyping

1. Membuat rancangan awal antarmuka.
2. Menyajikan rancangan kepada pengguna untuk memperoleh umpan balik.
3. Melakukan perbaikan berdasarkan masukan.
4. Mengulang proses hingga rancangan sesuai dengan harapan pengguna.

D. Keunggulan Prototyping

- a) **Mempercepat siklus pengembangan** karena masalah dapat ditemukan lebih awal.
- b) **Meningkatkan keterlibatan pengguna** sehingga sistem lebih sesuai dengan kebutuhan.
- c) **Mengurangi biaya perbaikan** di tahap implementasi karena kesalahan sudah dikoreksi sebelumnya.

Integrasi Antar Tahap Desain

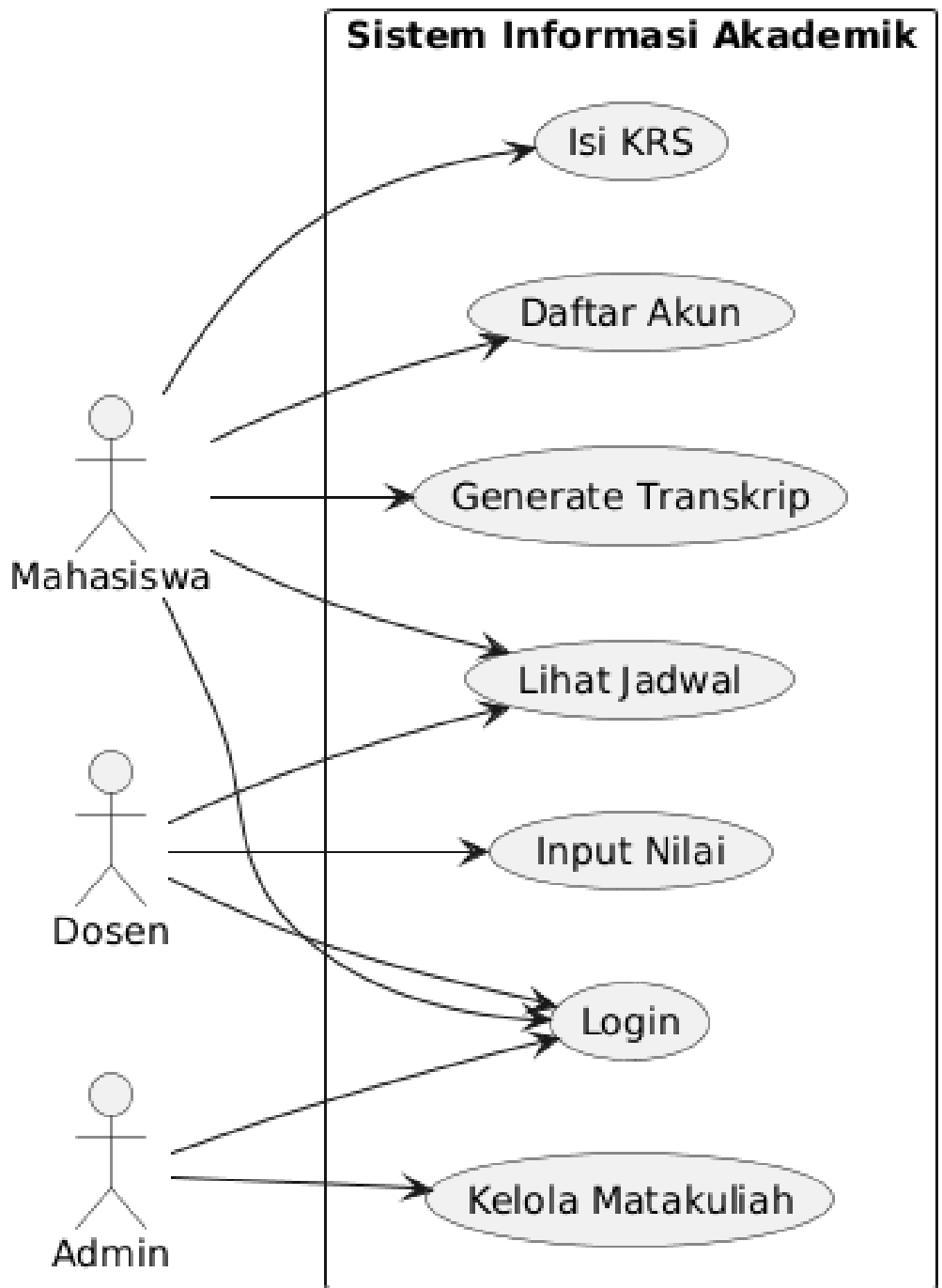
Ketiga tahap desain, yaitu konseptual, logis, dan fisik, bersama dengan pembuatan *prototype*, membentuk suatu alur yang berkesinambungan. **Desain konseptual** berperan sebagai kerangka awal, **desain logis** menguraikan detail struktur dan proses, sedangkan **desain fisik** menerjemahkan rancangan ke dalam spesifikasi teknis. Sementara itu, *prototyping* memberikan sarana evaluasi agar rancangan dapat diuji sejak dini. Integrasi ini memastikan bahwa sistem yang dibangun tidak hanya memenuhi kebutuhan fungsional, tetapi juga siap untuk diimplementasikan dengan teknologi yang ada.

Tantangan dalam Desain Sistem Informasi

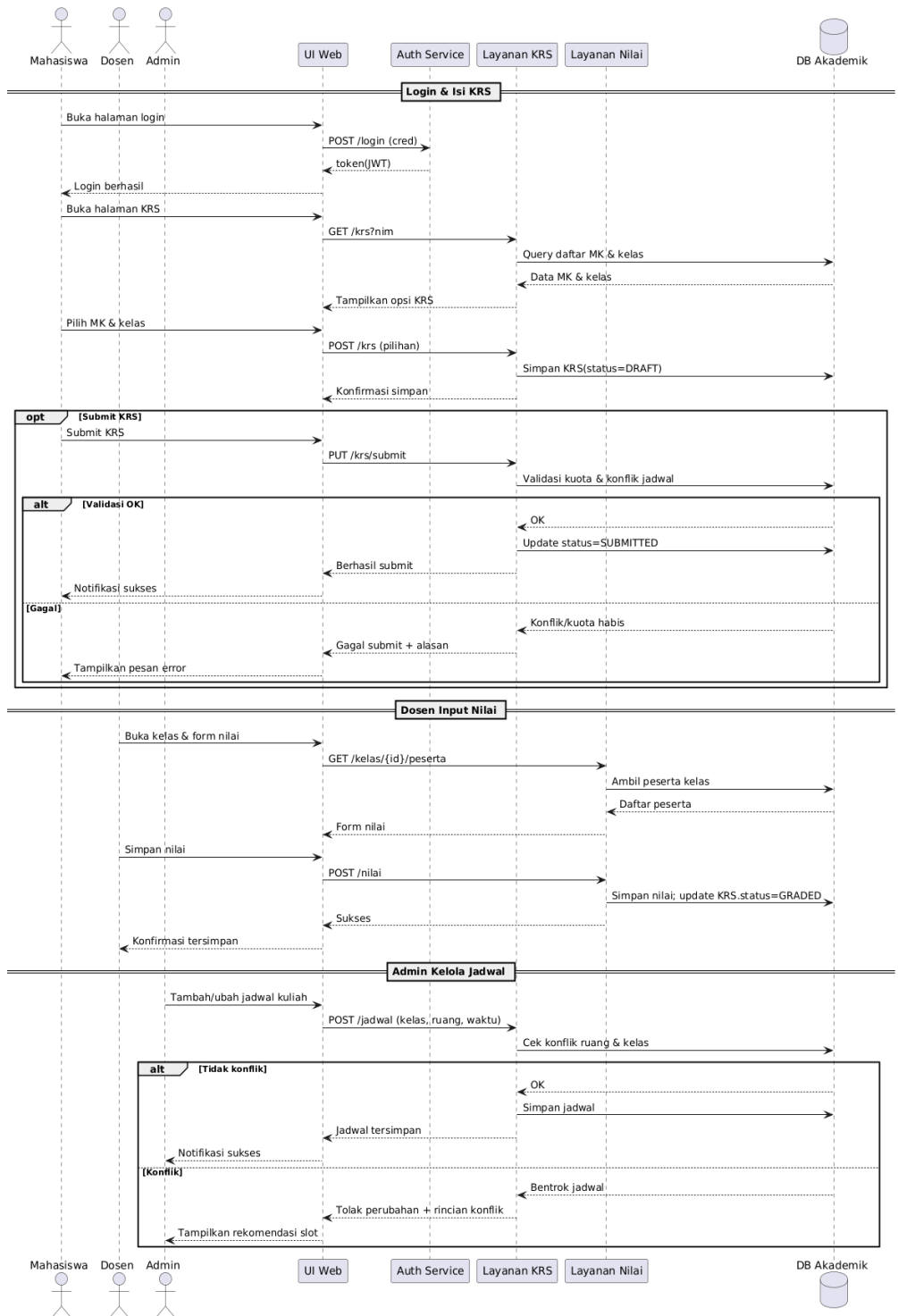
Proses desain sistem informasi menghadapi berbagai tantangan, di antaranya:

1. **Kebutuhan pengguna yang berubah-ubah** sehingga desain harus adaptif.
2. **Keterbatasan sumber daya** seperti anggaran, waktu, dan tenaga ahli.
3. **Kompleksitas sistem** yang semakin tinggi seiring bertambahnya fitur.
4. **Kesenjangan komunikasi** antara pengguna non-teknis dan pengembang.

Desain sistem informasi adalah proses penting yang menentukan keberhasilan implementasi perangkat lunak. Dengan memahami perbedaan dan keterkaitan antara desain konseptual, logis, fisik, serta pentingnya *prototyping*, mahasiswa dapat menyusun rancangan yang terstruktur, jelas, dan sesuai kebutuhan pengguna. Naskah ini menegaskan bahwa desain bukan hanya soal teknis, melainkan juga sarana komunikasi, validasi, dan evaluasi. Oleh karena itu, pemahaman mendalam tentang desain sistem informasi menjadi landasan yang tidak bisa diabaikan dalam setiap proyek pengembangan. Contoh Diagram desain konseptual sistem informasi akademik.

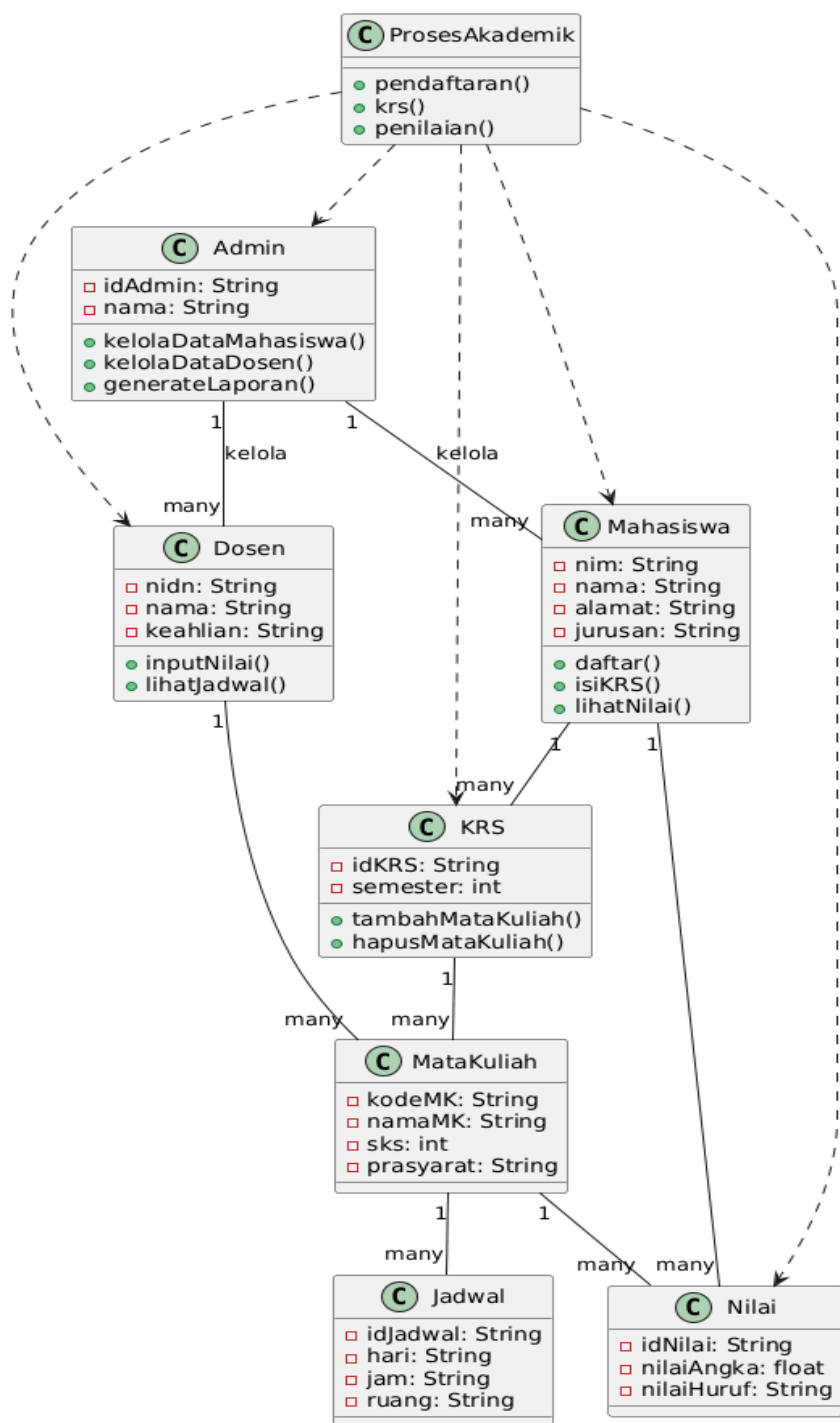


Gambar 19. Use Case Diagram Sistem Informasi Akademik

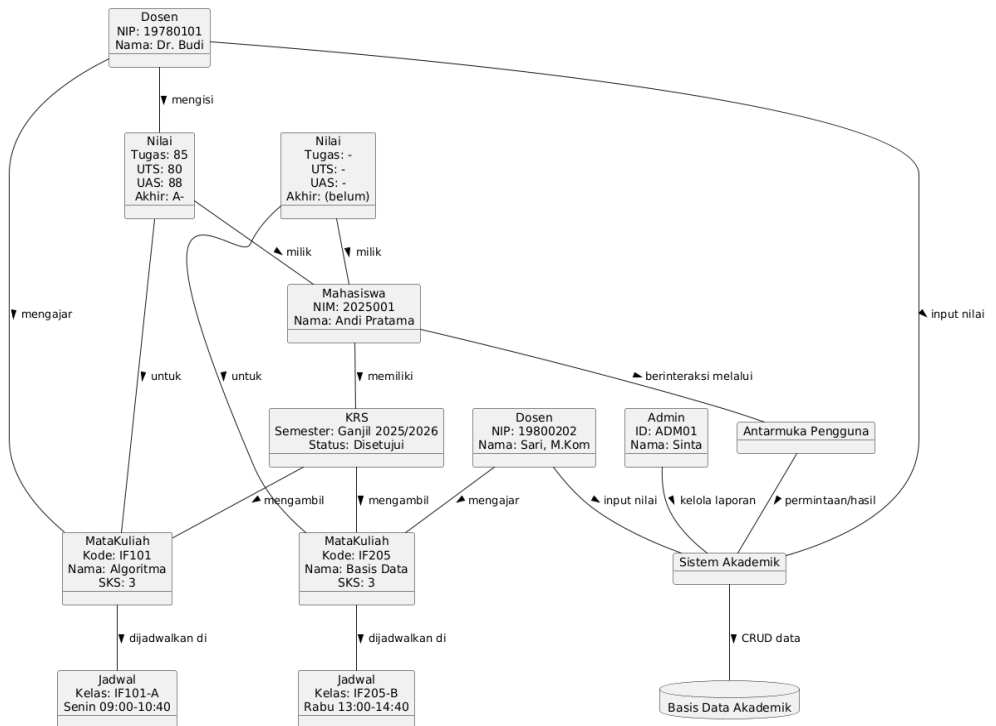


Gambar 20. Sequence Diagram Sistem Informasi Akademik

Class Diagram Sistem Informasi Akademik

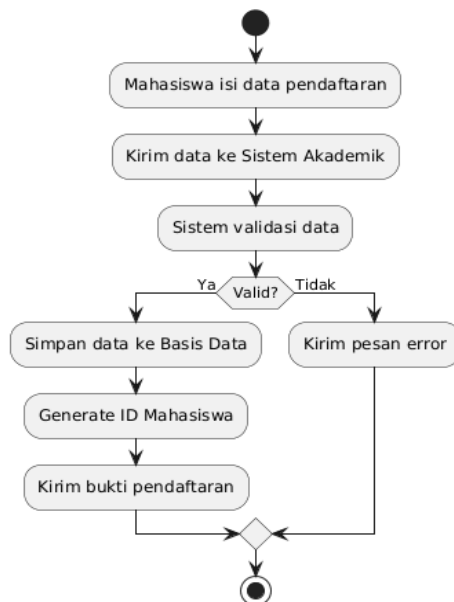


Gambar 21. Class Diagram Sistem Informasi Akademik



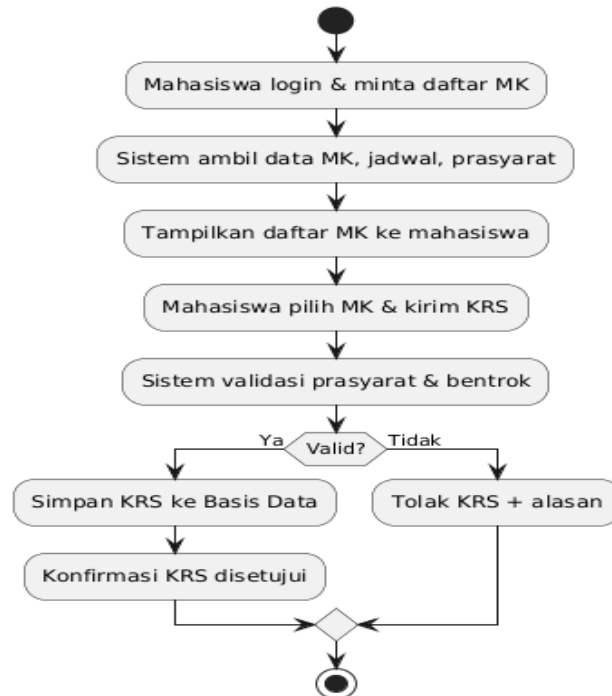
Gambar 22. Object Diagram Sistem Informasi Akademik

Activity Diagram - Pendaftaran Mahasiswa



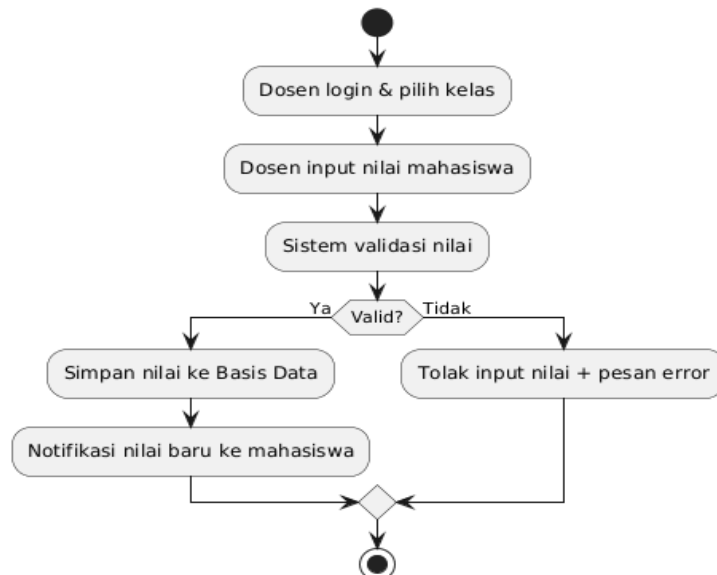
Gambar 23. Activity Diagram Pendaftaran Mahasiswa

Activity Diagram - Proses KRS



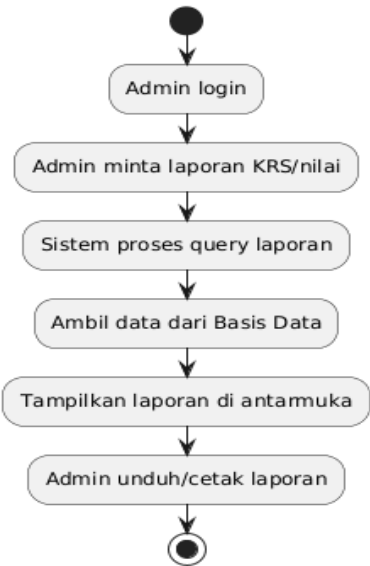
Gambar 24. Activity Diagram Proses KRS

Activity Diagram - Proses Penilaian



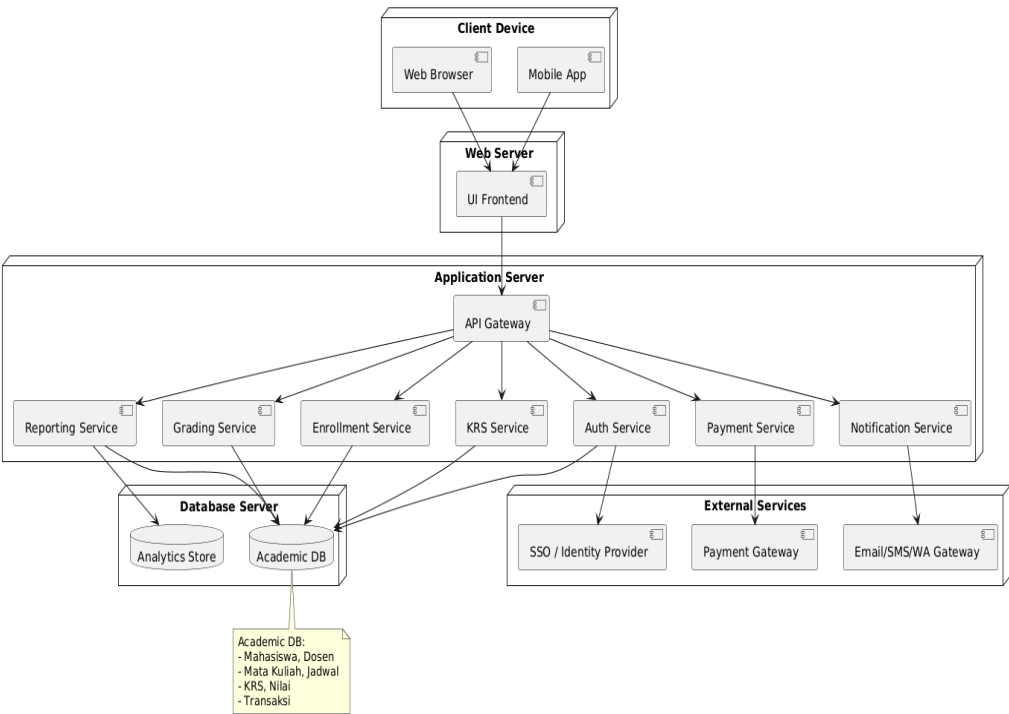
Gambar 25. Activity Diagram Proses Penilaian

Activity Diagram - Laporan Admin



Gambar 26. Activity Diagram Laporan Admin

Deployment Diagram - Sistem Informasi Akademik



Gambar 27. Deployment Diagram Sistem Infromasi Akademik

Soal Uraian

1. Jelaskan yang dimaksud dengan desain konseptual dalam pembangunan sistem informasi dan berikan contoh sederhana pada sistem manajemen rumah sakit.
2. Apa tujuan utama dari desain konseptual dalam proyek pengembangan sistem, dan bagaimana tahap ini memengaruhi keberhasilan implementasi?
3. Sebutkan dan uraikan perbedaan mendasar antara desain konseptual, desain logis, dan desain fisik, sertakan contoh singkat untuk masing-masing.
4. Bagaimana proses validasi desain konseptual dapat dilakukan bersama pemangku kepentingan dalam proyek sistem e-learning?
5. Mengapa normalisasi data sangat penting dalam tahap desain logis, dan apa dampaknya bila diabaikan pada sistem akademik?
6. Sebutkan contoh spesifikasi perangkat keras yang biasanya dirancang dalam desain fisik, misalnya pada pengembangan sistem e-commerce berskala besar.
7. Apa yang dimaksud dengan prototyping dalam perancangan sistem informasi, serta sebutkan satu tujuannya bagi pengembang maupun pengguna.
8. Bagaimana cara memperoleh umpan balik pengguna dari sebuah *prototype* dalam sistem aplikasi mobile?
9. Apa perbedaan utama antara *prototype* dengan produk akhir dalam siklus pengembangan perangkat lunak?

10. Bagaimana sebuah prototype dapat membantu dalam proses validasi kebutuhan sistem, khususnya pada sistem layanan perbankan digital?

Tugas Kelompok

Mahasiswa mampu menyusun **desain konseptual**, **desain logis**, dan **purwarupa antarmuka** untuk sistem perpustakaan digital yang rapi, konsisten, berbasis teori perancangan sistem informasi, dan siap diwujudkan pada tahap berikutnya.

Deskripsi Singkat Studi Kasus

Sebuah perguruan tinggi akan membangun **perpustakaan digital** untuk mengelola keanggotaan, koleksi, peminjaman, pengembalian, reservasi, denda, serta pelaporan. Sistem harus mudah digunakan, aman, dan dapat terhubung dengan layanan lain (mis. *payment gateway*). Tugas ini berfokus pada **rancangan** sebelum implementasi, dengan mengacu pada prinsip dalam literatur *Analisa Perancangan Sistem Informasi*.

Bagian A. Desain Konseptual

Tugas: susun satu berkas berisi gambaran umum solusi.

Cakupan minimal:

1. **Tujuan layanan** (poin singkat) dan batasan sistem (sertakan juga hal yang tidak termasuk lingkup sistem).
2. **Pemetaan pemangku kepentingan** (Anggota, Pustakawan, Admin, Pemasok, Keuangan).

3. **Diagram konseptual** (boleh *context diagram*/gambar blok):
 - a) Entitas luar ↔ sistem ↔ keluaran/masukan utama.
 - b) Alur informasi tingkat tinggi (tanpa detail teknis).
4. **Daftar komponen utama: basis data, antarmuka, proses pokok** (peminjaman, pengembalian, reservasi, pengelolaan koleksi, denda, pelaporan).
5. **Verifikasi awal** dengan skenario singkat: contoh alur "anggota meminjam buku" dari awal hingga akhir.

Keluaran: 1-2 halaman penjelasan + 1 diagram konseptual.

Bagian B

Desain Logis

Tugas: turunkan rancangan konseptual ke struktur yang lebih rinci tanpa menyebut perangkat tertentu.

Cakupan minimal:

1. **Struktur data** (tabel/entitas dan atribut utama) beserta hubungan antar entitas (gunakan notasi standar seperti ERD, tuliskan *primary key* dan *foreign key*).
2. **Aturan bisnis** pokok:
 - Status keanggotaan;
 - Syarat peminjaman;
 - Perhitungan denda;
 - Masa berlaku reservasi.
3. **Alur proses rinci** per layanan utama (poin berurutan).
4. **Penataan data** untuk menghindari pengulangan dan menjamin **konsistensi**.

5. **Kamus data ringkas** (nama elemen, deskripsi, format/jenis, sumber/tujuan).

Keluaran: 2-3 halaman uraian + 1 skema hubungan entitas berbasis ERD + kamus data.

Bagian C

Prototype Antarmuka

Tugas: buat *prototype* layar kunci untuk alur prioritas.

Layar minimal:

1. **Beranda** (ringkas; akses cepat ke pencarian, peminjaman, pengembalian).
2. **Pencarian Koleksi** (hasil pencarian + filter sederhana).
3. **Detail Koleksi** (info eksemplar, ketersediaan, tombol pinjam/reservasi).
4. **Ruang Anggota** (riwayat pinjam, status jatuh tempo, denda).
5. **Panel Pustakawan** (validasi peminjaman/pengembalian, pengolahan denda).

Ketentuan *prototype*:

- a) Boleh **low-fidelity** (*wireframe*) atau **high-fidelity** interaktif.
- b) Setiap layar memuat **judul**, **tujuan singkat**, dan **alur tindakan** (1-2 kalimat).
- c) Pastikan istilah konsisten dengan Bagian A-B.
- d) Sertakan kriteria evaluasi **usability** seperti konsistensi navigasi dan aksesibilitas.

Keluaran: berkas gambar atau pranala *prototype* + 1 halaman panduan alur.

Pedoman Kualitas & Konsistensi

- a) Gunakan bahasa Indonesia baku (KBBI/PUEBI), susunan SPOK jelas.
- b) Istilah asing ditulis *italic*; unsur penting ditulis **tebal**.
- c) Penamaan seragam: proses = kata kerja (mis. **Kelola Peminjaman**), data = kata benda jamak (mis. **Data Peminjaman**), entitas = kata benda tunggal (mis. **Anggota**).
- d) Keterpaduan antardokumen: komponen di Bagian A harus tampak pada Bagian B; istilah pada Bagian B muncul konsisten pada *prototype* di Bagian C.

Asumsi yang Diperbolehkan

- 1. Satu eksemplar hanya dapat dipinjam oleh satu anggota pada saat yang sama.
- 2. Denda dihitung per hari keterlambatan.
- 3. Reservasi otomatis dibatalkan bila buku tidak diambil dalam **2 hari kerja**.
- 4. Pembayaran denda bisa melalui kasir atau *payment gateway*.
- 5. Sistem menerapkan aspek **keamanan data** seperti autentikasi pengguna dan hak akses berbeda untuk pustakawan vs anggota.

Format Pengumpulan

- a) Satu berkas **PDF** berisi: halaman judul, Bagian A, Bagian B, Bagian C, kamus data, dan daftar rujukan istilah.
- b) Lampirkan berkas gambar diagram/tautan *prototype* pada lampiran.
- c) Penamaan berkas:
Desain_Prototype_Perpustakaan_NamaKelompok.pdf.

Model dan Tahapan Pengembangan dan Pengujian Sistem Informasi

A. CPL

Menganalisis serta memahami tahapan, model pengembangan sistem, dan metode pengujian yang relevan.

B. CPMK

- a) Menjelaskan secara komprehensif konsep SDLC beserta tahapan utamanya.
- b) Menganalisis dan membandingkan karakteristik, kelebihan, serta kekurangan model Waterfall dan Agile.
- c) Menguraikan berbagai model pengembangan sistem yang sering digunakan, serta menjelaskan metode pengujian yang relevan seperti Black Box, White Box, dan Grey Box Testing.

C. Sub-CPMK

- a. Menyebutkan serta menjelaskan tahapan utama SDLC secara runtut.
- b. Menentukan kondisi yang tepat untuk penerapan model Waterfall maupun Agile berdasarkan karakteristik proyek.
- c. Mengidentifikasi dan mendeskripsikan model pengembangan sistem yang umum digunakan dalam praktik.
- d. Menjelaskan jenis-jenis pengujian sistem, termasuk contoh penerapannya.

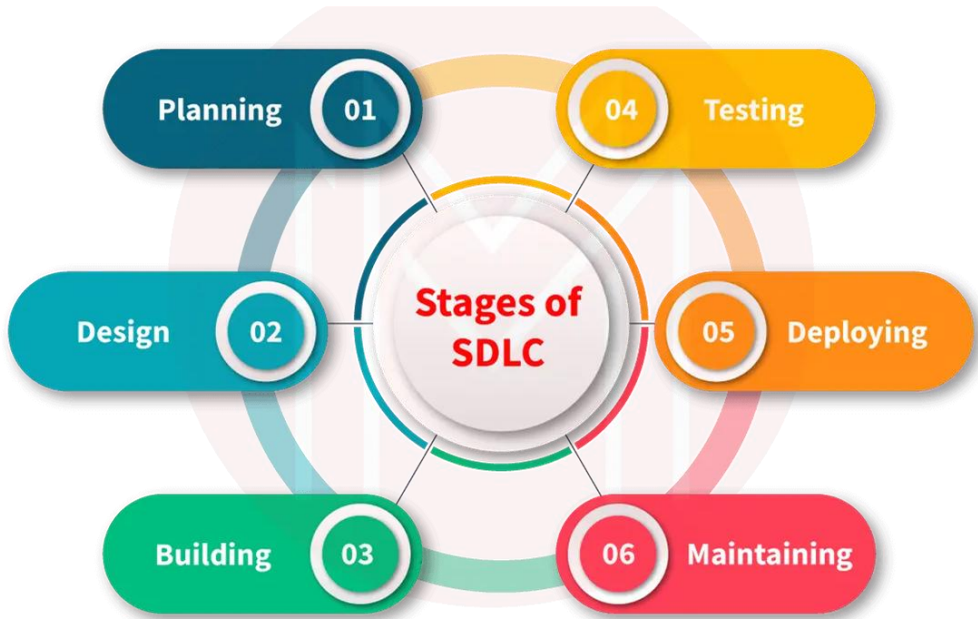
Pengembangan sistem informasi memerlukan **perencanaan, perancangan, dan pengujian** yang menyeluruh agar sistem benar-

benar sesuai dengan kebutuhan pengguna. Proses ini bersifat teknis sekaligus konseptual karena melibatkan **kerangka kerja pengembangan** seperti *System Development Life Cycle (SDLC)* serta model turunannya, misalnya **Waterfall** dan **Agile**. Pemahaman yang utuh atas model-model ini membantu mahasiswa maupun praktisi memilih pendekatan yang relevan dengan karakteristik proyek.

Keberhasilan pengembangan sistem juga bergantung pada penerapan **metode pengujian** yang tepat untuk menjamin kualitas perangkat lunak. Dalam literatur *Analisa Perancangan Sistem Informasi*, dijelaskan bahwa pemahaman menyeluruh atas siklus hidup pengembangan dan model yang dipilih akan sangat memengaruhi kualitas rancangan sistem serta keefektifan hasil akhir. Hubungan ini menegaskan bahwa teori tidak dapat dipisahkan dari praktik, sehingga setiap model pengembangan harus dikaitkan dengan analisis perancangan yang terstruktur.

System Development Life Cycle (SDLC)

System Development Life Cycle (SDLC) adalah **kerangka klasik** yang menuntun pengembangan sistem melalui tahapan analisis, desain, implementasi, pengujian, hingga pemeliharaan. Tujuannya menyediakan **alur kerja sistematis** agar setiap langkah terukur dan dapat dipertanggungjawabkan.



Gambar 28. System Development Life Cycle (SDLC)

Gambar 28 System Development Life Cycle (SDLC) adalah pendekatan terstruktur dalam pengembangan perangkat lunak yang memastikan **integritas, konsistensi, dan kualitas sistem**. Dalam konteks *Sistem Informasi Akademik (SIA)*, setiap tahap memainkan peran penting untuk menghasilkan solusi yang efektif, andal, dan mudah dipelihara. Tahap **Planning** berfungsi untuk menganalisis kebutuhan pengguna seperti mahasiswa, dosen, dan staf administrasi, misalnya identifikasi kebutuhan pendaftaran online, manajemen nilai, serta jadwal kuliah.

Selanjutnya pada tahap **Design**, perancangan dilakukan menggunakan *Unified Modeling Language (UML)* dan *Entity Relationship Diagram (ERD)* dengan fokus menciptakan arsitektur sistem yang **jelas, fleksibel, dan konsisten**. Tahap **Building** adalah proses pengembangan sistem dengan bahasa pemrograman seperti *Java* atau *PHP*, serta penggunaan *MySQL* sebagai basis data

akademik, dengan penekanan pada **keteraturan dalam penulisan kode** agar mudah dikelola.

Kemudian tahap **Testing** dilakukan untuk memastikan setiap modul berjalan tanpa kesalahan, meliputi *unit testing* hingga *user acceptance testing (UAT)* yang melibatkan pengguna langsung. Setelah itu, tahap **Deploying** mengimplementasikan sistem di server kampus agar dapat diakses secara luas dengan perhatian utama pada **keamanan data**. Terakhir, tahap **Maintaining** berfokus pada pembaruan fitur seperti integrasi e-learning atau mobile apps, serta mencakup perbaikan bug dan peningkatan performa. Dengan menerapkan SDLC secara disiplin, *Sistem Informasi Akademik* mampu memberikan layanan yang **berkelanjutan, terpercaya, dan sesuai kebutuhan pengguna**.

Tahapan Utama SDLC

1. **Analisis**

Mengidentifikasi kebutuhan, meninjau sistem lama, dan mendefinisikan ruang lingkup sistem baru.

2. **Desain**

Menyusun rancangan konseptual, logis, dan fisik.

3. **Implementasi**

Membangun sistem melalui kode, konfigurasi perangkat keras, serta integrasi modul.

4. **Pengujian**

Memverifikasi kesesuaian sistem dengan kebutuhan.

5. **Pemeliharaan**

Melakukan pembaruan, perbaikan, dan peningkatan performa.

Kelebihan dan Kekurangan

Kelebihan SDLC adalah **struktur yang jelas** sehingga setiap anggota tim memahami tahapan yang harus ditempuh. Kekurangannya, SDLC **kurang fleksibel** terhadap perubahan mendadak karena setiap tahap bergantung pada tahap sebelumnya. Meski demikian, SDLC menjadi landasan penting bagi berbagai model pengembangan.

Model Waterfall

Model *Waterfall* merupakan salah satu model klasik dalam *Software Development Life Cycle* (SDLC) yang bersifat linear dan berurutan. Setiap tahapan harus diselesaikan sebelum melangkah ke tahap berikutnya, sehingga tidak ada proses yang berjalan secara paralel. Adapun karakteristik utama Model *Waterfall* adalah:

1. **Proses linear dan sistematis**

Setiap tahap dilakukan secara berurutan, mulai dari analisis kebutuhan, desain sistem, implementasi, pengujian, hingga pemeliharaan.

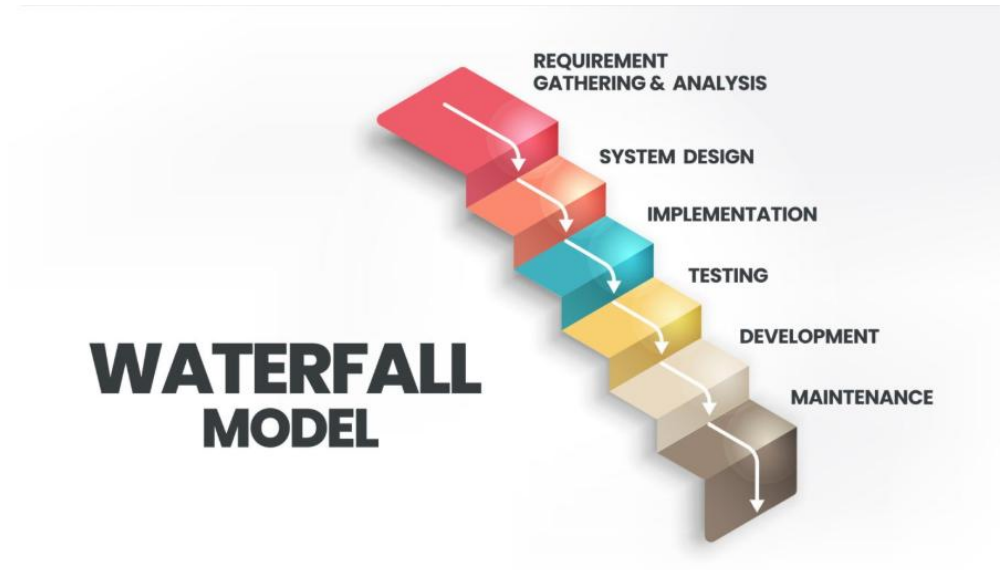
2. **Setiap tahap menghasilkan keluaran yang jelas**

Produk sementara berupa dokumen atau hasil kerja menjadi acuan bagi tahap berikutnya, sehingga memudahkan kontrol serta dokumentasi.

3. **Kurang fleksibel terhadap perubahan**

Perubahan sulit dilakukan setelah satu tahap berjalan, karena alur pengembangannya kaku dan terikat urutan.

Dengan karakteristik tersebut, Model *Waterfall* **cocok digunakan pada proyek dengan kebutuhan yang stabil dan jelas sejak awal**, tetapi **kurang ideal untuk proyek yang dinamis atau sering mengalami revisi**.



Gambar 29. Model *Waterfall*

Model *Waterfall* merupakan pendekatan klasik dalam *Software Development Life Cycle* (SDLC) yang bersifat linear dan berurutan. Setiap tahapan harus diselesaikan sebelum melanjutkan ke tahap berikutnya. Hal ini menjadikan *Waterfall* mudah dipahami dan diimplementasikan, terutama pada proyek dengan kebutuhan yang stabil. Contoh penerapannya dapat dilihat pada pengembangan **Sistem Informasi Akademik (SIA)** di perguruan tinggi.

Tahapan *Waterfall* meliputi: (1) *Requirement Gathering & Analysis*, yaitu mengidentifikasi kebutuhan pengguna seperti pendaftaran mahasiswa, jadwal kuliah, atau pengolahan nilai. (2) *System Design*, yang mendefinisikan arsitektur SIA, termasuk rancangan basis data dan antarmuka. (3) *Implementation*, di mana

kode program mulai ditulis sesuai desain. (4) *Testing*, yaitu menguji fungsionalitas agar sistem berjalan sesuai kebutuhan. (5) *Development* (deployment), yakni penerapan sistem di lingkungan nyata. (6) *Maintenance*, yaitu pemeliharaan agar sistem tetap optimal.

Insight pentingnya adalah bahwa setiap tahap menghasilkan keluaran yang jelas, seperti dokumen kebutuhan atau modul sistem, yang menjadi acuan tahap berikutnya. Namun, **tantangan utama Model Waterfall adalah kesulitannya dalam menghadapi perubahan**. Misalnya, bila setelah tahap implementasi terdapat perubahan aturan akademik, maka akan sulit melakukan revisi tanpa mengulang dari tahap awal.

Dengan demikian, Model *Waterfall* **sangat sesuai untuk Sistem Informasi Akademik dengan kebutuhan yang sudah pasti dan stabil**, tetapi **kurang ideal untuk proyek yang berpotensi mengalami perubahan signifikan**.

Kelebihan dan Kekurangan

a) Kelebihan

Dokumentasi **lengkap, kontrol proyek** lebih mudah, cocok untuk kebutuhan stabil.

b) Kekurangan

Tidak fleksibel, pengguna baru melihat hasil akhir setelah selesai, berisiko tidak sesuai kebutuhan.

Model Agile

Model **Agile** merupakan pendekatan manajemen proyek modern yang menekankan pada **kelincahan (agility)**, **fleksibilitas**, serta **kecepatan** dalam pengembangan produk. Pendekatan ini membagi proyek besar menjadi iterasi kecil yang disebut *sprint*, di mana hasil kerja dapat dievaluasi dan ditingkatkan secara berkelanjutan. Dengan cara ini, produk tidak hanya selesai lebih cepat, tetapi juga dapat menyesuaikan dengan kebutuhan pengguna dan perubahan pasar yang dinamis.

Prinsip Dasar Agile

Agile berlandaskan pada *Manifesto Agile* yang menekankan empat nilai utama:

1. **Individu dan interaksi** lebih diutamakan daripada proses dan alat.
2. **Perangkat lunak yang berfungsi** lebih penting daripada dokumentasi yang komprehensif.
3. **Kolaborasi dengan pelanggan** lebih utama daripada kontrak formal.
4. **Merespons perubahan** lebih baik daripada mengikuti rencana yang kaku.

Karakteristik Model Agile

a) **Iteratif dan Inkremental**

Proyek dipecah menjadi siklus pendek (*sprint*).

b) **Fleksibel dan Adaptif**

Cepat menyesuaikan kebutuhan yang berubah.

c) **Berorientasi pada Pengguna**

Umpan balik pelanggan menjadi prioritas.

d) **Kolaboratif**

Komunikasi terbuka serta keterlibatan aktif seluruh tim.

Manfaat Model Agile

1. **Pengiriman lebih cepat** melalui tahapan rilis bertahap.
2. **Peningkatan kualitas** berkat evaluasi dan *feedback* yang berkelanjutan.
3. **Responsif terhadap perubahan**, sehingga risiko berkurang.
4. **Kepuasan tim meningkat** melalui kolaborasi yang transparan.

Contoh Framework Agile

a) **Scrum**

Menggunakan *sprint* dan peran seperti *Scrum Master* serta *Product Owner*.

b) **Kanban**

Visualisasi alur kerja dan pembatasan pekerjaan yang sedang berlangsung (*Work-In-Progress*).

c) **Extreme Programming (XP)**

Menekankan praktik teknis seperti *Test-Driven Development (TDD)* dan integrasi berkelanjutan.



Gambar 30. Model Agile

Gambar di atas memperlihatkan siklus kerja **Agile** yang terdiri dari enam tahap penting: *Requirements*, *Design*, *Development*, *Testing*, *Deployment*, dan *Review*. Model ini menggambarkan bagaimana setiap proses dilakukan secara berulang dalam bentuk iterasi untuk menghasilkan produk yang selalu berkembang. Pendekatan ini menekankan bahwa setiap tahap tidak berdiri sendiri, melainkan saling berhubungan dalam siklus berkesinambungan.

Tahap pertama adalah **Requirements**, yaitu proses mengidentifikasi kebutuhan utama dari pengguna maupun bisnis. Selanjutnya pada tahap **Design**, tim merancang solusi yang sesuai dengan kebutuhan tersebut. Proses berlanjut ke **Development**, di mana rancangan diubah menjadi produk nyata melalui kode dan fungsi. Kemudian dilakukan **Testing** untuk memastikan kualitas dan mengurangi kesalahan. Hasil yang sudah stabil akan masuk ke tahap **Deployment** atau penerapan, sebelum akhirnya masuk pada tahap

Review guna mengevaluasi efektivitas dan mendapatkan *feedback* untuk iterasi berikutnya.

Metode ini sangat **sesuai untuk aplikasi mobile atau sistem berbasis web yang dinamis dan cepat berubah**. Misalnya, dalam pengembangan aplikasi e-commerce, kebutuhan konsumen dapat berganti dengan cepat seiring tren belanja daring. Dengan model Agile, tim dapat segera merespons perubahan tersebut melalui siklus iterasi singkat. Hal ini memberi keuntungan berupa **pengiriman lebih cepat, kualitas terjaga, dan adaptasi terhadap kebutuhan pasar**. Oleh karena itu, Agile dianggap sebagai pendekatan yang relevan dan efektif dalam era digital yang penuh ketidakpastian.

Model Pengembangan Sistem Lain

Model Spiral

Model spiral adalah metodologi pengembangan perangkat lunak berbasis iterasi yang menggabungkan prinsip **pengembangan bertahap** (*incremental* atau pengembangan bertahap) dengan **manajemen risiko**. Setiap siklus atau *spiral* berfungsi menghasilkan penyempurnaan produk sekaligus mengevaluasi risiko yang muncul. Model ini relevan untuk proyek yang **kompleks, berskala besar**, dan memiliki **ketidakpastian tinggi**. Pendekatan ini memungkinkan penyesuaian kebutuhan secara dinamis, mendorong **eksperimen teknologi baru**, serta memastikan **risiko dikendalikan sejak awal**.

Fase Utama dalam Model Spiral

Setiap putaran spiral mencakup empat fase utama yang saling berkaitan:

1. **Perencanaan (Planning)**

Menentukan tujuan, alternatif solusi, serta batasan proyek. Pada tahap ini kebutuhan sistem untuk iterasi dirumuskan secara jelas.

2. **Analisis Risiko (Risk Analysis)**

Mengidentifikasi potensi risiko dari sisi teknologi, biaya, jadwal, maupun penerimaan pengguna. Strategi mitigasi, seperti pembuatan prototipe atau studi kelayakan, digunakan untuk mengurangi ketidakpastian.

3. **Rekayasa (Engineering)**

Meliputi aktivitas pengembangan, mulai dari desain, pembuatan kode, hingga pengujian modul atau fungsi. Prototipe sering dipakai untuk menguji kelayakan teknis dan mengurangi risiko implementasi.

4. **Evaluasi (Evaluation)**

Meninjau hasil iterasi bersama pemangku kepentingan dan pengguna. Dari sini diputuskan apakah proyek akan dilanjutkan, direvisi, atau dihentikan.

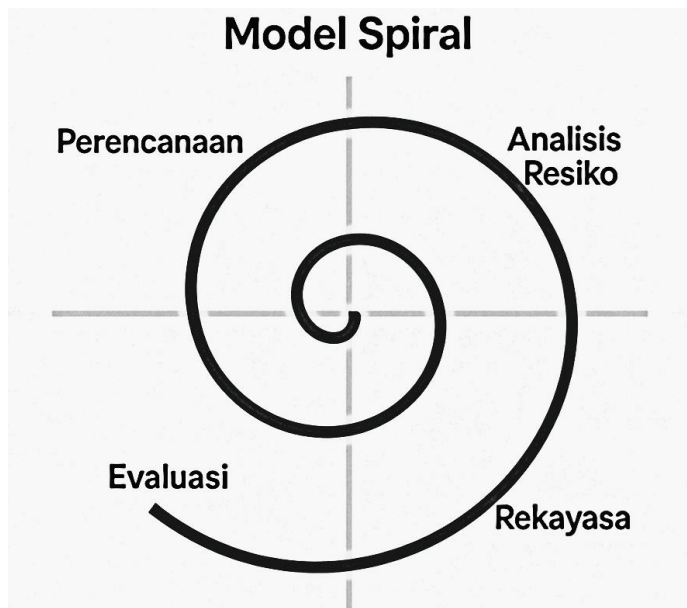
Kelebihan Model Spiral

- a) Mengintegrasikan **manajemen risiko** secara sistematis, sehingga potensi masalah dapat diantisipasi lebih awal.
- b) **Fleksibel** terhadap perubahan kebutuhan, mendukung penyesuaian sesuai dinamika pasar.

- c) Cocok untuk proyek dengan **kompleksitas tinggi**, karena setiap iterasi menghasilkan evaluasi mendalam.
- d) Mendukung peninjauan hasil secara **bertahap**, memungkinkan keterlibatan pengguna sejak awal.

Kekurangan Model Spiral

- a) Membutuhkan **biaya tinggi**, karena banyak iterasi, analisis, dan prototyping yang memerlukan sumber daya tambahan.
- b) Memerlukan **keahlian manajemen risiko** yang kuat agar proses pengendalian benar-benar efektif.
- c) Kurang sesuai untuk proyek **berskala kecil**, karena iterasi yang berulang dapat memperlambat dan membebani tim dengan sumber daya terbatas.



Gambar 31. Model *Spiral*

Contoh penerapan model spiral dapat dilihat dalam pengembangan aplikasi media sosial baru yang memadukan

Augmented Reality (AR) dan *Artificial Intelligence (AI)*. Proyek semacam ini umumnya **kompleks** dan penuh **ketidakpastian**, sehingga membutuhkan metodologi yang mampu menyesuaikan kebutuhan sekaligus mengendalikan risiko. Model spiral sangat sesuai karena menghadirkan siklus **perencanaan, analisis risiko, rekayasa**, dan **evaluasi** secara berulang.

Pada **siklus pertama**, tim berfokus pada perencanaan visi awal serta pembuatan prototipe sederhana untuk menguji fitur AR. Berdasarkan hasil evaluasi, tim dapat menilai **kelayakan teknologi AR** dan **penerimaan awal pengguna** terhadap integrasi AI. Identifikasi risiko sejak awal memberikan gambaran apakah teknologi yang digunakan layak dikembangkan lebih lanjut.

Memasuki **siklus kedua**, pengembangan diarahkan pada **integrasi AR dengan AI** dalam versi *beta*. Proses ini melibatkan pengujian performa aplikasi, efektivitas integrasi sistem, serta keamanan data. Dengan evaluasi berulang, tim memastikan **stabilitas sistem** tetap terjaga dan fitur inti berfungsi dengan baik.

Pada **siklus berikutnya**, fokus diarahkan pada penyempurnaan berkelanjutan. Langkah-langkah meliputi penambahan **personalisasi berbasis AI** untuk meningkatkan interaksi sosial, optimalisasi performa, serta perumusan strategi monetisasi, misalnya **fitur premium** atau **iklan berbasis AI**. Pendekatan ini memastikan aplikasi tidak hanya **meningkatkan pengalaman pengguna**, tetapi juga memberikan **keunggulan kompetitif** di pasar.

V-Model

V-Model merupakan pengembangan dari model **Waterfall** yang menekankan keterkaitan langsung antara setiap fase pengembangan dengan fase pengujian yang relevan. Dengan pendekatan ini, pengujian dilakukan sejak tahap awal sehingga kesalahan dapat terdeteksi lebih dini. Sebagai contoh, dalam pengembangan sistem aplikasi e-commerce sederhana, fase awal berupa spesifikasi kebutuhan akan dipasangkan dengan *acceptance testing* untuk memastikan fitur yang dirancang sesuai dengan keinginan pengguna.

Manfaat V-Model

Penerapan V-Model memberikan dua manfaat utama:

1. **Deteksi dini kesalahan**, karena pengujian dilakukan sejak awal sehingga meminimalisasi risiko perbaikan mahal di akhir siklus.
2. **Kualitas terjamin**, sebab setiap fase pengembangan selalu berpasangan dengan fase pengujian yang relevan.

Kelebihan V-Model

- a) **Struktur terorganisasi**
Setiap fase memiliki keluaran yang jelas dan langsung diuji.
- b) **Meningkatkan keandalan**
Pengujian paralel memastikan kesalahan teridentifikasi sejak awal.
- c) **Dokumentasi lengkap**

Cocok untuk proyek yang membutuhkan jejak dokumentasi kuat.

d) **Kualitas lebih baik**

Proses yang disiplin menjamin hasil perangkat lunak sesuai spesifikasi.

Kekurangan V-Model

a) **Kurang fleksibel**

Sulit menyesuaikan perubahan kebutuhan di tengah pengembangan.

b) **Waktu relatif panjang**

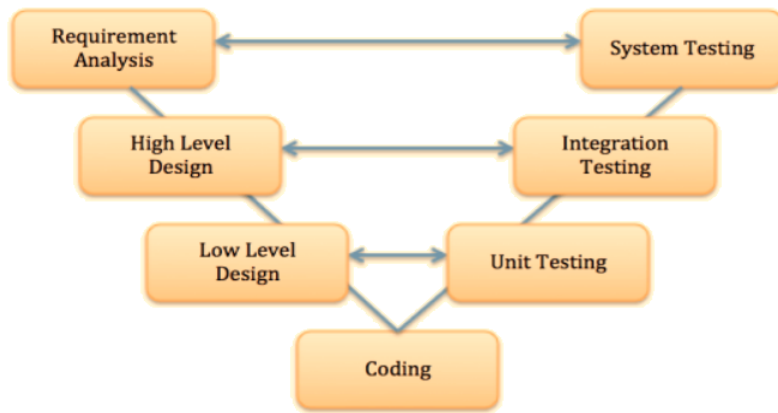
Karena setiap tahap harus diselesaikan sebelum melanjutkan ke tahap berikutnya.

c) **Biaya lebih tinggi**

Dokumentasi dan pengujian intensif membutuhkan sumber daya lebih.

d) **Tidak cocok untuk proyek kecil**

Lebih relevan untuk proyek berskala besar dengan kebutuhan stabil.



Gambar 32. V-Model

Penerapannya dapat dijelaskan melalui studi kasus pengembangan **aplikasi e-commerce sederhana**. Pada tahap **Requirement Analysis**, tim mengidentifikasi kebutuhan utama seperti fitur login, pencarian produk, keranjang belanja, hingga pembayaran. Fase ini dipasangkan dengan **System Testing**, yaitu pengujian menyeluruh yang memastikan aplikasi berjalan sesuai spesifikasi dan mampu memenuhi kebutuhan pengguna dalam skenario nyata, misalnya alur pembelian lengkap dari pendaftaran akun hingga proses pembayaran.

Selanjutnya, tahap **High Level Design** berfokus pada perancangan arsitektur sistem, termasuk rancangan antarmuka pengguna dan database. Pengujian pasangannya adalah **Integration Testing**, yang menitikberatkan pada interaksi antar modul. Sebagai contoh, data dari modul pencarian produk harus terintegrasi mulus dengan modul keranjang belanja dan diteruskan ke modul pembayaran tanpa adanya kesalahan komunikasi.

Pada tahap **Low Level Design**, perincian modul lebih detail seperti algoritma diskon dirancang. Fase ini dipasangkan

dengan **Unit Testing**, yaitu pengujian bagian kode terkecil secara individual untuk memastikan setiap fungsi bekerja sesuai harapan. Misalnya, memverifikasi validasi format email atau perhitungan diskon saat checkout. Setelah itu, tahap **Coding** menerjemahkan desain ke dalam kode program sekaligus mempersiapkan pengujian sesuai lingkup unit, integrasi, maupun sistem.

Dengan struktur ini, **V-Model** memungkinkan deteksi dini kesalahan serta menjamin kualitas perangkat lunak melalui **pengujian paralel**, meskipun kelemahannya adalah kurang fleksibel terhadap perubahan kebutuhan yang muncul di tengah proses pengembangan.

Prototype

Metode Prototype merupakan salah satu model dalam pengembangan perangkat lunak yang menekankan pada pembuatan **purwarupa (prototype)** sebelum sistem dikembangkan sepenuhnya. Prototype ini berfungsi sebagai **gambaran awal** dari sistem sehingga klien dapat memahami wujud nyata aplikasi lebih awal. Melalui pendekatan ini, terjadi interaksi intensif antara pengembang dan pengguna sejak tahap awal. Interaksi tersebut membantu mengurangi kesalahpahaman dalam memahami kebutuhan sistem yang diinginkan.

Kelebihan Metode Prototype

1. **Interaksi Intensif dengan Klien**

Klien dapat memberikan masukan secara langsung terhadap purwarupa. Hal ini membuat pengembang lebih mudah memahami kebutuhan pengguna.

2. **Deteksi Kesalahan Lebih Cepat**

Kesalahan dapat ditemukan sejak awal karena sistem diuji dalam bentuk prototype sebelum implementasi penuh.

3. **Efisiensi Waktu dan Biaya**

Revisi dilakukan di tahap awal sehingga lebih hemat dibandingkan memperbaiki sistem setelah selesai dibuat.

4. **Kejelasan Kebutuhan Sistem**

Adanya visualisasi nyata membuat kebutuhan sistem lebih jelas, terukur, dan tidak ambigu.

Kekurangan Metode Prototype

1. **Kompleksitas Bertambah**

Permintaan fitur tambahan dari klien dapat meningkatkan kompleksitas proyek melebihi rencana awal.

2. **Risiko Kurang Analisi**

Fokus berlebihan pada prototype bisa menyebabkan analisis kebutuhan tidak dilakukan secara menyeluruh.

3. **Ketergantungan pada Komunikas**

Jika komunikasi antara pengembang dan klien tidak efektif, maka proses pengembangan dapat terhambat.

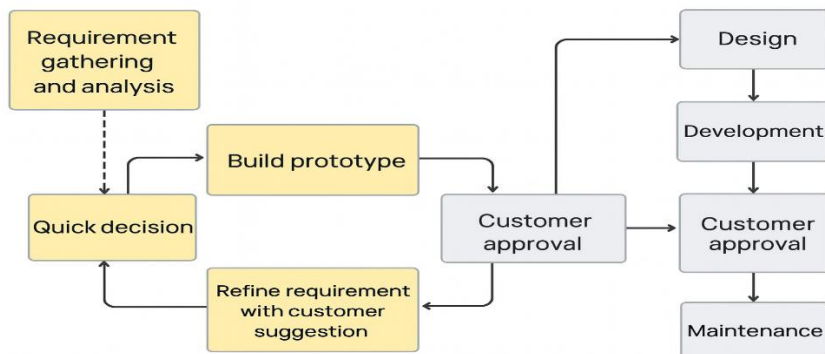
4. **Perubahan Terus-Menerus**

Klien berpotensi terus menambahkan permintaan baru sehingga mengganggu timeline dan konsistensi sistem.

Tahapan dalam Metode Prototype

Enam tahapan utama dalam model prototype:

1. **Requirements Gathering & Analysis** (*Analisis Kebutuhan*)
mendefinisikan kebutuhan sistem secara rinci melalui diskusi antara klien dan tim pengembang.
2. **Quick Design** (*Desain Cepat*) membuat desain sederhana untuk memberi gambaran awal sistem.
3. **Build Prototype** (*Membangun Purwarupa*)
Mengembangkan desain cepat menjadi purwarupa sistem.
4. **User Evaluation** (*Evaluasi Pengguna*) prototype diuji dan dievaluasi oleh klien untuk memberikan masukan.
5. **Refining Prototype** (*Perbaikan Purwarupa*) prototype diperbaiki sesuai umpan balik klien hingga mencapai kesepakatan.
6. **Implement Product & Maintain** (*Implementasi dan Pemeliharaan*)
Sistem akhir dikembangkan, diuji, diterapkan, dan dipelihara agar berjalan optimal.



Gambar 33. Metode Prototype

Sebuah toko kerajinan tangan lokal yang berstatus Usaha Mikro, Kecil, dan Menengah (UMKM) ingin memperluas jangkauan pasarnya dengan membuat platform *e-commerce* sendiri. Pemilik UMKM tersebut tidak memiliki pengalaman teknis dan tidak bisa mendefinisikan secara pasti semua fitur yang diinginkan. Oleh karena itu, tim pengembang menggunakan **model prototype** untuk memastikan produk akhir sesuai dengan kebutuhan bisnis dan pemahaman pemilik UMKM. Proses penerapan dimulai dengan antara lain :

- a. **Pengumpulan Kebutuhan Awal** melalui wawancara dengan pemilik UMKM, sehingga teridentifikasi beberapa fitur dasar seperti halaman katalog produk, fitur keranjang belanja, proses checkout sederhana, halaman detail pesanan, serta sistem manajemen produk. Setelah itu, tim melakukan.
- b. **Pembuatan Prototype** interaktif menggunakan Figma, yang mencakup alur pemilihan produk, keranjang, hingga pembayaran, termasuk contoh desain berupa halaman beranda dengan produk unggulan, tombol *Tambah ke Keranjang*, keranjang belanja, dan halaman checkout. Prototype ini kemudian memasuki tahap.
- c. **Evaluasi Prototype** langsung oleh pemilik dan staf UMKM. Mereka mencoba alur belanja dan memberikan masukan berupa kebutuhan tambahan, seperti kolom kode diskon, notifikasi email untuk pesanan baru, serta fitur ulasan pembeli. Masukan tersebut ditindaklanjuti dengan

- d. **Iterasi dan Perbaikan** prototype, sehingga ditambahkan kolom kode promo, simulasi notifikasi email, dan elemen ulasan produk. Proses iterasi dilakukan berulang hingga semua pihak puas dan menyetujui desain final. Setelah prototype disepakati, tim pengembang memulai
- e. **Implementasi Produk Akhir** dengan proses pengkodean sesuai alur dan desain yang telah dikembangkan.

Model prototype ini memberikan beberapa manfaat nyata, antara lain **mengurangi risiko salah paham** antara pengembang dan klien, menghasilkan produk yang **sesuai kebutuhan pasar** karena melibatkan pemilik dan staf secara aktif, menyediakan **fleksibilitas dalam perubahan** di tahap awal, serta mendukung **efisiensi waktu dan biaya** karena revisi dilakukan sebelum pengembangan penuh.

Tabel 5. Metode Pengembangan Aplikasi E-Commerce UMKM

Metode	Karakteristik	Kelebihan	Kekurangan	Penerapan
Waterfall	Linear, kebutuhan pasti.	Mudah dipahami, cocok untuk fitur sederhana.	Tidak fleksibel terhadap perubahan.	Tidak cocok, karena kebutuhan UMKM sering berubah.
Agile	Iteratif, fleksibel.	Cepat menyesuaikan kebutuhan, hasil bisa terlihat tiap sprint.	Butuh keterlibatan aktif pemilik UMKM.	Sangat sesuai, pemilik bisa memberi masukan rutin.

Metode	Karakteristik	Kelebihan	Kekurangan	Penerapan
Spiral	Iteratif, fokus risiko.	Bagus mengantisipasi risiko besar seperti keamanan.	Kompleks, mahal untuk UMKM kecil.	Terlalu berlebihan untuk skala UMKM awal.
V-Model	Linear dengan fokus pengujian.	Kualitas tinggi karena pengujian di tiap tahap.	Sama kaku dengan Waterfall.	Cocok jika UMKM butuh jaminan kualitas stabil.
Prototype	Iterasi cepat, umpan balik langsung.	Mudah dipahami pemilik non-teknis, interaktif.	Perlu waktu untuk pembuatan prototipe berulang.	Sangat ideal, pemilik dapat coba langsung sebelum final.

Metode pengembangan perangkat lunak memiliki keunggulan dan kelemahan masing-masing, terutama ketika diterapkan pada konteks UMKM yang cenderung memiliki kebutuhan dinamis. Waterfall dan V-Model lebih cocok jika kebutuhan sudah pasti sejak awal, tetapi kurang fleksibel. Spiral sangat efektif dalam mitigasi risiko, namun terlalu rumit untuk skala kecil. Agile menjadi pilihan ideal karena mampu menyesuaikan perubahan dengan cepat, walau menuntut keterlibatan tinggi dari pemilik. Prototype justru sangat sesuai dengan studi kasus ini karena memungkinkan pemilik UMKM yang awam teknologi untuk melihat wujud aplikasi sejak awal dan memberikan masukan langsung. Kombinasi Prototype dan Agile adalah pendekatan paling relevan untuk pengembangan e-commerce UMKM agar produk akhir sesuai kebutuhan, berkualitas, dan mudah digunakan.

Tahapan Pengembangan dan Pasangan Pengujian

A. Spesifikasi Kebutuhan (Requirements Specification)

Pada tahap ini, semua **fitur dan persyaratan pengguna** dikumpulkan, misalnya fitur login, pencarian produk, keranjang belanja, dan pembayaran. Pengujian pasangannya adalah **Acceptance Testing** yang berfokus memastikan sistem memenuhi persyaratan fungsional maupun non-fungsional. Contoh penerapan adalah skenario pengguna mendaftar, mencari produk, hingga menyelesaikan pesanan.

B. Desain Sistem (System Design)

Tahap ini merancang **arsitektur sistem** mencakup database, antarmuka pengguna, serta interaksi antar komponen. Pasangan ujiannya adalah **Integration Testing** yang memverifikasi interaksi antar modul berjalan lancar. Contoh: alur pencarian produk hingga checkout tanpa hambatan komunikasi.

C. Desain Detail (Detailed Design)

Tahap ini merinci setiap **komponen perangkat lunak**, seperti desain database dan algoritma. Pengujian pasangannya adalah **Module Testing** yang menguji modul secara terpisah, misalnya fungsi perhitungan diskon.

D. Implementasi (*Implementation*)

Desain yang telah dibuat diterjemahkan ke dalam **kode program**. Pasangan pengujian adalah **Unit Testing**, yaitu pengujian pada bagian kode terkecil, misalnya fungsi validasi email.

Pengujian Sistem

Pengujian memastikan sistem bekerja sesuai kebutuhan dan mengurangi risiko kesalahan fatal. Tanpa pengujian, kualitas sistem tidak dapat dijamin.

Jenis Pengujian

Pengujian Unit Testing

Pengujian unit (*unit testing*) adalah metode pengujian perangkat lunak yang berfokus pada **unit atau komponen terkecil** dari kode yang dapat diuji secara terisolasi. Unit ini biasanya berupa **fungsi, metode, atau kelas** dalam sebuah program. Tujuan utama pengujian unit adalah memastikan bahwa setiap unit berfungsi sesuai dengan **spesifikasi** dan menghasilkan output yang benar ketika diberikan input tertentu.

A. Pentingnya Pengujian Unit

Pengujian unit sangat penting dalam proses pengembangan perangkat lunak karena dapat:

- a) **Menjamin keandalan**

Setiap unit yang diuji dengan baik akan mengurangi kemungkinan kesalahan di tahap integrasi.

b) **Menghemat waktu dan biaya**

Kesalahan yang ditemukan lebih awal lebih mudah dan murah untuk diperbaiki.

c) **Meningkatkan kualitas kode**

Dengan pengujian unit, pengembang dapat menulis kode yang lebih rapi, terstruktur, dan mudah dipelihara.

B. Studi Kasus Login ke Email

Bayangkan sebuah sistem **login ke email**. Proses login biasanya melibatkan:

1. Pengguna memasukkan **alamat email** dan **kata sandi**.
2. Sistem memvalidasi input tersebut.
3. Jika sesuai, pengguna diarahkan ke **inbox**; jika tidak, sistem menampilkan pesan **kesalahan**.

Dalam pengujian unit, kita hanya fokus pada **fungsi validasi login**. Misalnya, ada sebuah metode `validateLogin(email, password)`. Pengujian unit akan menguji berbagai skenario seperti:

- a) Input email dan kata sandi benar → hasil **sukses**.
- b) Input email benar tetapi kata sandi salah → hasil **gagal**.
- c) Input email salah tetapi kata sandi benar → hasil **gagal**.
- d) Input kosong atau tidak sesuai format → hasil **error**.

Dengan pengujian unit, kita dapat memastikan bahwa **fungsi login** berjalan sesuai harapan sebelum diintegrasikan dengan komponen lain, seperti database atau antarmuka pengguna. Hal ini

membantu menjaga **kualitas sistem secara keseluruhan** serta meminimalkan risiko kesalahan fatal di tahap produksi.

C. Analisa Pengujian Unit

Pengujian unit pada studi kasus login ke email berfokus pada validasi fungsi inti yaitu `validateLogin(email, password)` untuk memastikan mekanisme autentikasi berjalan sesuai spesifikasi. Analisis ini penting karena login merupakan pintu utama akses pengguna ke sistem, sehingga kesalahan kecil dapat berdampak besar terhadap keamanan maupun pengalaman pengguna. Melalui pengujian unit, berbagai skenario dapat diuji secara terisolasi, seperti kondisi ketika email dan kata sandi benar, kombinasi salah satu input yang tidak valid, maupun input kosong atau tidak sesuai format. Dengan pendekatan ini, pengembang dapat mendeteksi error sejak dini sebelum proses integrasi dengan database atau antarmuka pengguna, sehingga meminimalisasi risiko kegagalan saat sistem digunakan secara nyata. Selain itu, pengujian unit pada kasus login juga memberikan jaminan bahwa kode lebih stabil, mudah dipelihara, serta efisien dalam waktu dan biaya pengembangan. Hal ini membuktikan pentingnya unit testing sebagai pondasi kualitas perangkat lunak.

Pengujian Integration Testing

Integration Testing adalah metode pengujian perangkat lunak yang fokus pada pemeriksaan **interaksi antar modul**. Tujuannya adalah untuk memastikan bahwa setiap modul yang telah

diuji secara terpisah (melalui *unit testing*) dapat bekerja dengan baik ketika digabungkan. Pengujian ini sangat penting karena dalam sistem nyata, sebuah modul jarang berdiri sendiri, melainkan saling berhubungan dengan modul lain.

A. Pentingnya Integration Testing

Tanpa **integration testing**, meskipun setiap modul berfungsi dengan baik secara individual, masih ada risiko terjadinya kesalahan ketika modul tersebut saling berinteraksi. Kesalahan umum meliputi kesalahan komunikasi data, ketidakcocokan format, atau alur logika yang tidak terhubung dengan benar. Oleh karena itu, tahap ini membantu mengurangi risiko kegagalan sistem saat digunakan secara menyeluruh.

B. Contoh Studi Kasus Login ke Email

Bayangkan sebuah sistem **login ke email**. Proses ini melibatkan beberapa modul:

1. **Modul Input:** menerima masukan berupa *username* dan *password* dari pengguna.
2. **Modul Validasi:** memeriksa apakah input sesuai format yang benar (misalnya panjang minimal, karakter valid).
3. **Modul Autentikasi:** mencocokkan *username* dan *password* dengan data pada basis data.
4. **Modul Dashboard:** menampilkan halaman utama setelah login berhasil.

Jika dilakukan **integration testing**, penguji akan memeriksa apakah:

- a) Data dari **modul input** berhasil diteruskan dengan benar ke **modul validasi**.
- b) Hasil validasi diproses oleh **modul autentikasi** sesuai kebutuhan.
- c) Apabila autentikasi berhasil, sistem dapat berpindah ke **modul dashboard** tanpa kesalahan.

Sebagai contoh sederhana, jika *username* benar namun *password* salah, maka integrasi antara **modul validasi** dan **modul autentikasi** harus memastikan pesan error ditampilkan. Sebaliknya, jika keduanya benar, sistem harus berhasil menampilkan **dashboard**. Hal ini memperlihatkan bahwa **integration testing** memastikan kelancaran aliran data dan logika antar modul, sehingga sistem dapat berfungsi sesuai kebutuhan pengguna.

C. Analisa Integration Testing

Integration Testing pada sistem login email berperan penting untuk memastikan interaksi antar modul berjalan konsisten dan sesuai alur logika yang diharapkan. Misalnya, ketika pengguna memasukkan *username* dan *password*, data dari *modul input* harus diteruskan ke *modul validasi* tanpa ada kehilangan atau perubahan format. Setelah lolos validasi, *modul autentikasi* wajib mampu mencocokkan data dengan benar di basis data, termasuk menangani kondisi error jika *password* salah atau akun tidak terdaftar. Jika autentikasi berhasil, sistem harus menampilkan *modul dashboard* secara mulus. Tanpa pengujian integrasi, risiko terjadinya bug seperti salah kirim data, gagal menangani error, atau transisi modul yang tidak sinkron akan

lebih besar. Dengan demikian, Integration Testing memastikan aliran data dan logika antar modul berjalan lancar sehingga pengalaman pengguna tetap konsisten, aman, dan sesuai kebutuhan bisnis.

Pengujian System Testing

System Testing merupakan tahap pengujian perangkat lunak yang dilakukan untuk mengevaluasi sistem secara **keseluruhan**. Tujuannya adalah memastikan bahwa semua komponen aplikasi dapat bekerja bersama sesuai dengan spesifikasi dan kebutuhan pengguna. Pada tahap ini, fokus bukan hanya pada modul tertentu, tetapi pada integrasi fungsionalitas sehingga sistem benar-benar siap digunakan.

A. Pentingnya System Testing

System Testing sangat penting karena dapat mengidentifikasi **ketidaksesuaian** antara sistem dengan kebutuhan awal. Selain itu, pengujian ini memberikan jaminan bahwa aplikasi telah melalui proses validasi menyeluruh sebelum diterapkan kepada pengguna akhir. Dengan demikian, System Testing membantu menjaga kualitas, keamanan, dan keandalan perangkat lunak.

B. Studi Kasus Nyata Login ke Email

Sebagai contoh sederhana, mari kita gunakan studi kasus **login ke email**. Login merupakan salah satu fitur paling penting dalam sistem berbasis akun. Berikut gambaran proses pengujian:

1. Skenario Positif

Pengguna memasukkan alamat email dan kata sandi yang benar. Sistem harus berhasil mengarahkan pengguna ke halaman utama (*inbox*).

2. **Skenario Negatif**

Pengguna memasukkan kata sandi yang salah. Sistem harus menampilkan pesan error seperti "*Password salah*" tanpa memberikan akses ke akun.

3. **Uji Keamanan**

Sistem harus memblokir percobaan login berulang yang mencurigakan untuk mencegah serangan *brute force*.

4. **Uji Kinerja**

Sistem diuji dengan ribuan pengguna yang login secara bersamaan untuk memastikan server tetap stabil.

Dari contoh login ke email, dapat dipahami bahwa **System Testing** mencakup pengujian dari berbagai aspek, mulai dari fungsi utama, keamanan, hingga kinerja. Dengan pengujian menyeluruh, sistem dapat dipastikan **handal, aman, dan sesuai kebutuhan pengguna**. Pendekatan ini membuat System Testing menjadi bagian krusial dalam siklus hidup pengembangan perangkat lunak.

C. Analisa System Testing

System Testing pada studi kasus login ke email merupakan tahap penting untuk memastikan bahwa seluruh komponen aplikasi bekerja sesuai kebutuhan pengguna dan persyaratan sistem. Pengujian ini dilakukan secara menyeluruh dengan melibatkan aspek fungsionalitas, keamanan, serta kinerja. Misalnya, ketika pengguna

memasukkan email dan kata sandi yang valid, sistem harus dapat mengarahkan pengguna ke halaman inbox tanpa kendala. Sebaliknya, ketika kata sandi yang salah dimasukkan, sistem harus menampilkan pesan error yang jelas tanpa memberikan akses. Selain itu, aspek keamanan juga diuji, seperti perlindungan terhadap serangan brute force dengan membatasi percobaan login berulang yang mencurigakan. Tidak hanya itu, uji kinerja dilakukan untuk memastikan sistem tetap stabil saat menghadapi ribuan pengguna yang login secara bersamaan. Dengan pendekatan ini, System Testing memastikan bahwa fitur login tidak hanya berfungsi secara benar, tetapi juga aman, handal, dan sesuai dengan ekspektasi pengguna akhir.

Pengujian Acceptance Testing (UAT)

Acceptance Testing (UAT) adalah tahap akhir dalam proses pengujian perangkat lunak yang dilakukan oleh **pengguna akhir** (*end user*). Tujuan utamanya adalah untuk **memastikan kesesuaian sistem** dengan kebutuhan bisnis dan memvalidasi bahwa aplikasi dapat digunakan dalam skenario nyata sebelum resmi diluncurkan.

A. Pentingnya UAT

User Acceptance Testing (UAT) memiliki peranan yang sangat penting dalam proses pengembangan perangkat lunak. Tahap ini memastikan bahwa sistem tidak hanya berfungsi dari sisi teknis, tetapi juga sesuai dengan *business requirements* yang telah ditentukan. Melalui *UAT*, pengguna akhir dapat memverifikasi bahwa setiap fitur

aplikasi berjalan sebagaimana mestinya dalam skenario nyata. Tanpa adanya tahap ini, risiko terjadinya kesalahan implementasi atau ketidaksesuaian dengan kebutuhan pengguna akan lebih besar. Selain itu, *UAT* juga membantu membangun kepercayaan antara tim pengembang dan pihak bisnis, karena memberikan kesempatan bagi mereka untuk mengevaluasi sistem sebelum *go-live*. Proses ini sering kali menggunakan *test scenarios* yang merepresentasikan aktivitas sehari-hari pengguna. Dengan demikian, *UAT* bukan hanya langkah teknis, tetapi juga strategi untuk memastikan kualitas, kepuasan pengguna, serta kesuksesan implementasi aplikasi secara menyeluruh.

B. Studi Kasus Login ke Email

Sebagai ilustrasi sederhana, kita gunakan contoh **login ke email**. Fitur ini sangat penting karena berkaitan dengan **akses awal** pengguna terhadap sistem. Melalui *UAT*, pengguna memastikan bahwa proses login berfungsi dengan benar, baik untuk input valid maupun salah.

1. Menentukan Skenario Uji

- a) Skenario login dengan **username dan password benar**.
- b) Skenario login dengan **password salah**.
- c) Skenario login dengan **username tidak terdaftar**.

2. Pelaksanaan Uji

- a) Pengguna membuka halaman login email.
- b) Memasukkan **email** dan **password** sesuai skenario.

- c) Sistem menampilkan **halaman utama email** jika data valid.
- d) Jika data tidak valid, sistem menampilkan **pesan error** yang jelas.

3. Hasil yang Diharapkan

- a) Input benar → akses diberikan.
- b) Input salah → akses ditolak dengan notifikasi tepat.
- c) Proses login berjalan **cepat, stabil**, dan tanpa error teknis.

C. Analisis UAT

Pengujian *User Acceptance Testing (UAT)* pada fitur *login* bertujuan memastikan bahwa sistem dapat digunakan dengan baik oleh pengguna akhir sesuai kebutuhan bisnis. Dalam studi kasus *Login ke Email*, UAT difokuskan pada validasi proses autentikasi. Pengguna menguji skenario *login* dengan kombinasi data yang benar, salah, maupun tidak terdaftar. Hasil uji menunjukkan bahwa sistem berhasil mengizinkan akses ketika *username* dan *password* valid, serta menolak dengan pesan error yang jelas jika data salah. Hal ini membuktikan bahwa sistem tidak hanya berfungsi secara teknis, tetapi juga mendukung pengalaman pengguna dengan notifikasi yang informatif. Selain itu, performa sistem diuji agar proses *login* berlangsung cepat dan stabil. Melalui UAT, dapat dipastikan bahwa fitur login sesuai dengan kebutuhan bisnis, mengurangi risiko kesalahan saat *go-live*, serta meningkatkan kepercayaan pengguna terhadap kualitas aplikasi.

Pengujian Regression Testing

Regression Testing merupakan jenis pengujian perangkat lunak yang dilakukan untuk **memastikan perubahan atau pembaruan kode tidak merusak fungsi lain** yang sudah ada sebelumnya. Tujuan utamanya adalah menjaga agar fitur lama tetap berjalan dengan baik setelah adanya **modifikasi, perbaikan bug, atau penambahan fitur baru**. Uji ini sangat penting untuk menjamin **stabilitas, keandalan, dan kualitas sistem**.

A. Pentingnya Regression Testing

Regression Testing sangat penting karena berfungsi sebagai penjaga kualitas sistem setelah adanya perubahan kode. Setiap kali dilakukan *bug fixing*, update, atau penambahan fitur baru, regression testing memastikan bahwa fungsi lama tetap berjalan tanpa masalah. Tanpa tahap ini, ada risiko besar terjadinya kerusakan (*side effect*) yang tidak terdeteksi dan dapat mengganggu pengalaman pengguna. Dengan regression testing, tim pengembang dapat menjaga **stabilitas, konsistensi, dan keandalan** aplikasi. Selain itu, proses ini membantu meningkatkan kepercayaan pengguna dan mempercepat *time to market* dengan mengurangi potensi error pada tahap produksi.

B. Studi Kasus Login ke Email

Sebagai contoh nyata, kita gunakan kasus **login ke email**. Misalnya, tim pengembang menambahkan fitur **otentikasi dua faktor (two-factor authentication / 2FA)** guna meningkatkan

keamanan. Setelah penambahan ini, Regression Testing diperlukan untuk memastikan bahwa proses login lama tidak mengalami gangguan.

Langkah-Langkah Regression Testing

1. Identifikasi Fitur Lama yang Terdampak

- a) Fungsi login dengan **username** dan **password**.
- b) Fungsi logout.
- c) Tampilan halaman utama setelah login.

2. Pelaksanaan Uji

- a) Pengguna mencoba login hanya dengan **username dan password** tanpa aktivasi **2FA**.
- b) Sistem menampilkan halaman utama dengan benar.
- c) Logout diuji untuk memastikan pengguna bisa keluar tanpa error.

3. Hasil yang Diharapkan

- a) Fitur login lama tetap berfungsi seperti sebelumnya.
- b) Penambahan **2FA** tidak mengganggu akses dasar.
- c) Tidak muncul error baru setelah perubahan kode.

C. Analisis Regression Testing

Pengujian ini memastikan bahwa meskipun ada **fitur tambahan** seperti **2FA**, fungsi inti aplikasi yaitu login dan logout tetap berjalan normal. Dengan demikian, Regression Testing berperan penting dalam menjamin **konsistensi sistem, kenyamanan**

pengguna, serta keberlangsungan operasional. Jika hasil uji konsisten dengan harapan, aplikasi dapat dianggap **siap digunakan.**

Metode Pengujian Khusus

Black Box Testing

Black Box Testing merupakan salah satu metode pengujian perangkat lunak yang menitikberatkan pada hubungan **input-output** tanpa memperhatikan struktur internal atau baris kode program. Dengan kata lain, penguji tidak perlu mengetahui bagaimana algoritma diimplementasikan, melainkan cukup mengevaluasi apakah sistem telah memberikan **output sesuai harapan** berdasarkan **input tertentu**. Dalam istilah *software engineering*, penguji ibarat melihat sebuah "kotak hitam" tanpa mengetahui apa yang terjadi di dalamnya.

A. Prinsip Dasar

Prinsip utama dalam **Black Box Testing** adalah:

1. Menentukan **input** yang akan diberikan ke sistem.
2. Menentukan **output** yang diharapkan.
3. Melakukan perbandingan antara hasil aktual dengan hasil yang diharapkan.
4. Tidak memerlukan pemahaman mendalam terhadap kode atau logika program.

Dengan fokus tersebut, metode ini membantu memvalidasi **fungsionalitas sistem** dari sudut pandang pengguna sehingga berorientasi pada pengalaman penggunaan (*user experience*).

B. Studi Kasus Aplikasi Login dengan Email

Sebagai ilustrasi, bayangkan sebuah **aplikasi login** yang mengharuskan pengguna memasukkan **email** dan **password** untuk dapat mengakses sistem.

Skenario Pengujian Black Box

1. Input valid:

- a) Email: *user@example.com*
- b) Password: *benar123*
- c) **Output yang diharapkan:** muncul pesan **"Login berhasil"** dan sistem mengarahkan ke halaman utama.

2. Input salah pada password:

- a) Email: *user@example.com*
- b) Password: *salah123*
- c) **Output yang diharapkan:** muncul pesan **"Password salah"**.

3. Input salah pada format email:

- a) Email: *userexample.com* (tanpa @)
- b) Password: *benar123*
- c) **Output yang diharapkan:** muncul pesan **"Format email tidak valid"**.

4. Input kosong:

- a) Email: *(kosong)*
- b) Password: *(kosong)*
- c) **Output yang diharapkan:** muncul pesan **"Email dan password wajib diisi"**.

Tabel 6. Skenario Pengujian Black Box

No	Skenario	Input (Email)	Input (Password)	Langkah Uji	Output yang Diharapkan
1	Login valid	<i>user@example.com</i>	<i>benar123</i>	Isi kolom lalu klik Login	Muncul "Login berhasil" dan diarahkan ke beranda
2	Password salah	<i>user@example.com</i>	<i>salah123</i>	Isi kolom lalu klik Login	Muncul "Password salah" , tidak login
3	Format email tidak valid	<i>userexample.com</i>	<i>benar123</i>	Masukkan email tanpa @, klik Login	"Format email tidak valid" , permintaan auth tidak dikirim
4	Input kosong	<i>(kosong)</i>	<i>(kosong)</i>	Biarkan kedua kolom kosong, klik Login	"Email dan password wajib diisi" , field ditandai

Dari studi kasus login di atas, jelas terlihat bahwa **Black Box Testing** berfokus pada pengujian respons sistem terhadap berbagai **input** pengguna tanpa harus memahami kode program. Metode ini efektif untuk menilai **keandalan fungsi** aplikasi serta memastikan pengalaman pengguna berjalan sesuai harapan. Dengan pendekatan ini, integritas fungsional sistem dapat diuji secara objektif dan komprehensif.

White Box Testing

White Box Testing memiliki peran penting dalam fase pengembangan perangkat lunak karena dapat membantu menemukan bug tersembunyi pada jalur logika internal yang sering

kali tidak terdeteksi oleh metode pengujian lain. Dengan menelusuri kondisi dan struktur kode, pengembang dapat lebih cepat mengidentifikasi kesalahan logika, potensi celah keamanan, serta meningkatkan keandalan sistem sebelum produk dirilis.

White Box Testing adalah metode pengujian perangkat lunak yang berfokus pada struktur internal kode program. Pendekatan ini memungkinkan penguji untuk memahami bagaimana logika, kondisi, serta alur kontrol dijalankan di dalam sistem. Dalam istilah *software testing*, metode ini sering disebut juga *glass box testing* atau *clear box testing*.

Berbeda dengan **Black Box Testing** yang hanya melihat input dan output, White Box Testing menekankan validasi terhadap **jalur logika internal**.

A. Prinsip Dasar White Box Testing

1. Mengetahui Struktur Internal Kode

Penguji harus memahami alur logika, struktur program, serta kondisi yang digunakan di dalam kode. Tanpa pengetahuan ini, pengujian tidak dapat dilakukan secara efektif.

2. Pengujian Berdasarkan Jalur Logika (Control Flow)

Fokus utama adalah memastikan semua jalur logika, *decision point*, dan kondisi percabangan sudah diuji agar tidak ada bagian kode yang terlewat.

3. Memeriksa Kondisi dan Pernyataan (Condition & Statement Coverage)

4. Prinsip ini memastikan bahwa setiap pernyataan (*statement*) dan kondisi (*if, else, loop*) sudah dieksekusi minimal sekali dalam pengujian.

5. **Mengidentifikasi Kode Mati (*Dead Code*)**

White Box Testing dapat membantu menemukan bagian kode yang tidak pernah dijalankan, sehingga bisa dioptimalkan atau dihapus.

6. **Validasi Keamanan Internal**

Dengan menguji jalur logika internal, potensi celah keamanan (misalnya akses tanpa validasi) bisa terdeteksi sejak dini.

7. **Kombinasi Input dan Cabang Eksekusi**

Setiap kemungkinan kombinasi input harus diuji terhadap jalur kode yang sesuai untuk memastikan hasil program konsisten.

8. **Diterapkan Secara Iteratif**

Prinsipnya dilakukan berulang selama siklus pengembangan perangkat lunak (SDLC) agar bug dapat ditemukan lebih cepat sebelum tahap akhir pengujian.

B. Studi Kasus Aplikasi Login dengan Email

Selain logika dasar validasi email dan password, pengujian juga dapat diperluas dengan menambahkan jalur alternatif, misalnya percabangan kondisi keamanan tambahan seperti *captcha* atau verifikasi dua faktor. Hal ini penting karena pengujian White Box dapat memetakan bagaimana kode menangani mekanisme perlindungan

ekstra tersebut, termasuk jalur ketika *captcha* gagal diverifikasi atau ketika kode verifikasi tambahan tidak dimasukkan dengan benar. Untuk memahami secara sederhana, mari gunakan studi kasus **aplikasi login** dengan input berupa **email** dan **password**. Di dalam kode program, terdapat logika seperti:

1. Mengecek apakah input *email* terisi.
2. Mengecek format *email* apakah valid (misalnya mengandung simbol "@").
3. Mengecek apakah *password* tidak kosong.
4. Memverifikasi kombinasi *email* dan *password* dengan data pada basis data.
5. Jika validasi berhasil, pengguna diarahkan ke *dashboard*.

Tabel 7. Skenario Pengujian White Box

No	Jalur Logika	Input yang Diuji	Kondisi Awal / Catatan Eksekusi	Output yang Diharapkan
1	Validasi input email kosong	Email: "" Password: "abc123"	Koneksi DB aktif; captcha nonaktif; akun tidak diblokir	Pesan: "Email harus diisi"
2	Validasi format email tidak sesuai	Email: "userexample.com" Password: "abc123"	Validator format aktif (regex); captcha nonaktif	Pesan: "Format email salah"
3	Validasi password kosong	Email: "user@example.com" Password: ""	Email dianggap valid; rate limit aman	Pesan: "Password harus diisi"

No	Jalur Logika	Input yang Diuji	Kondisi Awal / Catatan Eksekusi	Output yang Diharapkan
4	Kombinasi email & password salah	Email: "user@exampl e.com" Password: "wrong123"	Email terdaftar aktif; hash password tidak cocok; rate limit +1	Pesan: "Email atau password salah"
5	Login berhasil	Email: "user@exampl e.com" Password: "abc123"	Email terdaftar & aktif; captcha lulus; 2FA nonaktif	Arahkan ke dashboard
6	Captcha gagal diverifikasi	Email: "user@exampl e.com" Password: "abc123" Captcha: salah	Captcha diaktifkan; threshold percobaan tersisa ≥ 1	Pesan: "Verifikasi captcha gagal"; login ditolak
7	2FA/OTP salah	Email: "user@exampl e.com" Password: "abc123" OTP: "000000"	2FA diaktifkan; kredensial dasar valid; captcha lulus	Pesan: "Kode verifikasi salah"; tetap di layar 2FA

Grey Box Testing

Grey Box Testing adalah metode pengujian perangkat lunak yang menggabungkan keunggulan **Black Box Testing** dan **White Box Testing**. Dalam pendekatan ini, **tester mengetahui sebagian struktur internal** aplikasi, namun tetap berfokus pada pengujian dari sudut pandang pengguna. Dengan demikian, pengujian ini menekankan pada **fungsi aplikasi** sekaligus **alur logika internal** yang relevan.

A. Prinsip Dasar Grey Box Testing

1. Pengetahuan Parsial tentang Sistem

Tester memiliki akses terbatas terhadap desain, arsitektur, atau kode sumber, cukup untuk memahami alur utama tanpa detail penuh.

2. Fokus pada Fungsionalitas dan Logika Internal

Pengujian tidak hanya melihat hasil input-output, tetapi juga memvalidasi jalur proses internal yang penting.

3. Pengujian Berbasis Skenario

Test case biasanya dibuat berdasarkan *use case* pengguna, tetapi diperkuat dengan pemahaman struktur internal agar lebih efektif.

4. Kombinasi Pendekatan

Menggunakan teknik *Black Box* (fokus pada fungsi dan kebutuhan bisnis) serta *White Box* (fokus pada jalur logika tertentu yang kritis).

5. Deteksi Masalah Integrasi dan Keamanan

Grey Box Testing efektif untuk menemukan masalah seperti *data flow*, *session management*, serta *security loopholes* yang mungkin tidak terlihat hanya dengan Black Box Testing.

6. Pendekatan Realistis

Metode ini sering dianggap lebih praktis karena mencerminkan situasi nyata di mana penguji memiliki informasi terbatas, mirip dengan kondisi dunia nyata dalam eksplorasi bug.

B. Studi Kasus Aplikasi Login dengan Email

Untuk memahami penerapan Grey Box Testing, mari kita gunakan contoh nyata yaitu **fitur login menggunakan email** pada sebuah aplikasi.

1. Informasi yang Diketahui Tester

Tester memiliki akses pada sebagian dokumentasi, misalnya:

- a) Struktur tabel *user* di basis data yang menyimpan **email** dan **password**.
- b) Informasi bahwa sistem menggunakan metode *hashing* untuk menyimpan kata sandi.
- c) Alur proses autentikasi, mulai dari validasi input hingga verifikasi di server.

2. Langkah Pengujian

1. Uji validasi input

Memasukkan format email salah (contoh: user@abc) untuk memastikan sistem menolak input tidak valid.

2. Uji data tidak terdaftar

Memasukkan email yang tidak ada di database, sistem harus menampilkan pesan **"akun tidak ditemukan"**.

3. Uji password salah

Memasukkan email terdaftar dengan password salah, sistem menampilkan pesan **"password salah"**.

4. Uji login sukses

Memasukkan email dan password benar, sistem harus memberikan akses.

5. Uji keamanan hashing

Dengan pengetahuan bahwa password di-hash, tester memverifikasi bahwa database tidak menyimpan password dalam bentuk teks biasa (*plain text*).

3. Keunggulan Grey Box Testing

- a) Mendeteksi **kerentanan keamanan** dengan memanfaatkan pengetahuan internal.
- b) Memastikan **fungsionalitas eksternal** tetap berjalan sesuai harapan.
- c) Menyeimbangkan perspektif **pengguna akhir** dan **pengembang**.

Dengan pendekatan **Grey Box Testing**, pengujian aplikasi login menjadi lebih **komprehensif**. Tester tidak hanya menguji dari sisi antarmuka, tetapi juga memahami bagaimana sistem memproses data secara internal. Hal ini memastikan aplikasi lebih **andalan, aman**, dan sesuai kebutuhan pengguna.

Soal Uraian

1. Apa yang dimaksud dengan *System Development Life Cycle* (SDLC) dan sebutkan lima tahapan utamanya secara runtut beserta penjelasan singkat.
2. Jelaskan kelebihan dan kelemahan SDLC sebagai kerangka klasik dalam pengembangan sistem informasi, sertakan contoh penerapan sederhana.

3. Apa perbedaan mendasar antara model Waterfall dan Agile dalam hal alur kerja serta cara menghadapi perubahan kebutuhan.
4. Kapan model Waterfall lebih cocok digunakan? Berikan contoh proyek yang sesuai untuk model ini, misalnya sistem administrasi keuangan.
5. Mengapa Agile dianggap lebih fleksibel dalam menghadapi perubahan kebutuhan? Jelaskan dengan contoh pengembangan aplikasi layanan daring.
6. Sebutkan model pengembangan sistem selain Waterfall dan Agile, lalu jelaskan satu kelebihanannya, misalnya pada Spiral atau V-Model.
7. Jelaskan peran Acceptance Testing dan Regression Testing dalam proses pengujian sistem, sertakan situasi praktis di mana kedua pengujian tersebut penting.
8. Apa yang dimaksud dengan Black Box Testing dan White Box Testing? Sertakan contoh sederhana masing-masing, seperti form login atau fungsi kalkulasi.
9. Siapa saja pihak yang terlibat dalam proses SDLC (misalnya analis sistem, pengembang, pengguna, manajer proyek) dan bagaimana kontribusi mereka dalam keberhasilan proyek.
10. Mengapa pemilihan model pengembangan dan metode pengujian penting untuk keberhasilan proyek? Jelaskan dengan menautkan manfaatnya terhadap kualitas, efisiensi, dan kepuasan pengguna.

Tugas Kelompok

Mahasiswa mampu menganalisis kelemahan model SDLC klasik, mensimulasikan jadwal proyek berbasis Agile, membandingkan berbagai model pengembangan sistem, serta merancang skenario pengujian terintegrasi untuk sistem perpustakaan digital modern. Seluruh aktivitas harus dikaitkan dengan prinsip yang dijelaskan dalam literatur *Analisa Perancangan Sistem Informasi* agar hasil diskusi berbasis teori yang kokoh sekaligus aplikatif.

Instruksi Tugas

1. **Diskusikan kelemahan penerapan SDLC pada studi kasus sistem perpustakaan digital modern.**
 1. Identifikasi tahapan SDLC yang sulit diterapkan pada sistem perpustakaan digital yang bersifat dinamis.
 2. Kaitkan kelemahan tersebut dengan penjelasan teori dalam *Analisa Perancangan Sistem Informasi*.
 3. Berikan contoh bagaimana keterbatasan fleksibilitas SDLC berdampak pada kebutuhan pengguna yang berubah.
 4. Tuliskan argumentasi kelompok dalam 1-2 halaman.
2. **Buat simulasi jadwal proyek sistem perpustakaan digital dengan model Agile.**
 1. Bagi proyek menjadi *sprint* (durasi 2 minggu).
 2. Tentukan output tiap *sprint* (misalnya desain antarmuka, modul pencarian, modul peminjaman).

3. Sajikan dalam bentuk tabel jadwal proyek lengkap dengan deliverables.
4. Lampirkan deskripsi singkat tentang bagaimana kolaborasi tim dan pengguna dilakukan di setiap *sprint*.
5. Hubungkan penerapan ini dengan prinsip fleksibilitas dan iterasi yang ditekankan dalam literatur perancangan.

3. **Analisis kelebihan dan kelemahan masing-masing model pengembangan sistem (Waterfall, Agile, Spiral, V-Model, Prototype) dalam konteks perpustakaan digital.**

1. Buat tabel perbandingan yang memuat kelebihan, kelemahan, serta kondisi penerapan paling sesuai.
2. Berikan ulasan naratif 2-3 halaman yang menjelaskan alasan kelompok dalam menilai relevansi tiap model.
3. Sertakan rujukan teori dari *Analisa Perancangan Sistem Informasi* untuk memperkuat argumen.

4. **Buat contoh skenario pengujian sistem (Unit, Integration, Black Box, White Box) untuk fitur pencarian buku pada sistem perpustakaan digital.**

1. Rancang 2-3 kasus uji untuk setiap metode pengujian.
2. Sertakan deskripsi input, langkah pengujian, serta hasil yang diharapkan.
3. Jelaskan bagaimana skenario ini memastikan kualitas fitur pencarian.
4. Kaitkan desain pengujian dengan konsep validasi kebutuhan yang dijelaskan dalam literatur perancangan.

5. **Simulasikan penerapan Black Box dan White Box Testing secara terintegrasi pada studi kasus sistem perpustakaan digital.**

1. Tentukan satu fitur penting (misalnya proses login anggota).
2. Buat rencana uji dengan pendekatan Black Box (fokus input-output) dan White Box (fokus logika kode).
3. Tunjukkan bagaimana kombinasi keduanya memberikan validasi menyeluruh.
4. Tulis hasil simulasi dalam laporan ringkas 1-2 halaman.
5. Kaitkan integrasi pengujian dengan manfaat yang dijelaskan dalam *Analisa Perancangan Sistem Informasi*.

Format Pengumpulan

1. Laporan kelompok dalam bentuk PDF dengan struktur: pendahuluan, pembahasan setiap poin, tabel atau diagram pendukung, kesimpulan.
2. Lampirkan jadwal proyek, tabel perbandingan, dan skenario pengujian dalam lampiran.
3. Penamaan berkas:
Tugas_Kelompok_PerpustakaanDigital_NamaKelompok.pdf

Ujian Tengah Semester (UTS)

Mata Kuliah: Metode Penelitian

Bentuk Soal: Uraian (10 Soal)

Cakupan Materi: Pertemuan 1-8

Contoh Soal Uraian

A. Soal Uraian

1. Jelaskan integrasi antara perancangan sistem informasi dan data sains dalam mendukung pengambilan keputusan di organisasi modern.
2. Bandingkan metodologi Structured System Analysis and Design (SSAD) dengan Object-Oriented Analysis and Design (OOAD) dari segi pendekatan, kelebihan, dan keterbatasannya.
3. Uraikan tahapan dalam siklus data sains (Data Science Lifecycle) serta jelaskan manfaat praktisnya bagi organisasi.
4. Mengapa tahap identifikasi kebutuhan dianggap sebagai fondasi utama dalam pengembangan sistem informasi?
5. Jelaskan perbedaan mendasar antara performance testing dan usability testing dalam evaluasi sistem informasi.
6. Apa saja peran manusia dalam siklus perancangan sistem informasi? Jelaskan secara rinci.
7. Diskusikan kelebihan dan kelemahan metodologi Prototyping.

8. Mengapa studi kelayakan harus mencakup aspek teknis, ekonomi, operasional, dan hukum?
9. Jelaskan simbol-simbol utama dalam Data Flow Diagram (DFD) beserta fungsinya.
10. Apa perbedaan keluaran perancangan sistem informasi dengan keluaran analisis data sains?

Contoh Soal Studi Kasus

B. Soal Studi Kasus

Kasus:

Sebuah perguruan tinggi menghadapi masalah dalam pengelolaan KRS (Kartu Rencana Studi) mahasiswa. Sering terjadi keterlambatan penginputan nilai, kesalahan jadwal kuliah, dan kesulitan mahasiswa mengakses informasi akademik secara real-time.

Instruksi:

1. Identifikasi minimal tiga masalah utama dari sistem akademik manual tersebut.
2. Susun kebutuhan fungsional dan non-fungsional sistem informasi akademik yang baru.
3. Gambarkan Context Diagram (DFD Level 0) untuk sistem informasi akademik tersebut.
4. Buat ERD sederhana yang memuat entitas utama (misalnya Mahasiswa, Dosen, Mata Kuliah, Nilai, Jadwal).
5. Uraikan manfaat yang diharapkan setelah sistem akademik digital diimplementasikan.

Metodologi Data Sains

A. CPL

Memahami metodologi data sains dengan integritas akademik.

B. CPMK

Mampu membandingkan metodologi data sains pada Artificial Intelligence (AI), Machine Learning (ML), dan Deep Learning (DL).

C. Sub-CPMK

Mampu menentukan tahapan-tahapan yang tepat dalam framework penelitian pada AI, ML, dan DL sesuai dengan permasalahan yang diteliti.

Metodologi data sains adalah fondasi penting untuk memastikan setiap langkah dalam pengolahan data, *machine learning*, dan *artificial intelligence* dilakukan dengan sistematis dan dapat dipertanggungjawabkan. Dengan kerangka metodologi yang jelas, praktisi dapat menavigasi kompleksitas data modern, menghubungkan teori dengan implementasi nyata, serta menghasilkan solusi yang **relevan dan bermanfaat**. Bab ini membahas secara komprehensif konsep dasar metodologi data sains, framework penelitian pada AI, ML, dan DL, perbandingan ketiganya, hingga studi kasus pada berbagai domain.

Konsep Dasar Metodologi Penelitian Data Sains

Metodologi data sains merupakan kerangka kerja **sistematis** yang berfungsi untuk merancang, melaksanakan, serta mengevaluasi

proyek berbasis data dengan tujuan menghasilkan keluaran yang relevan dan bermanfaat. Konsep ini tidak hanya menekankan pada *technical process* seperti algoritma atau pemrograman, melainkan juga menegaskan pentingnya **nilai praktis** dan **** penerapan****. Dalam praktiknya, metodologi ini mencakup tahap **pengumpulan data** dari berbagai sumber yang valid, diikuti dengan **pembersihan data** guna meminimalkan bias dan kesalahan. Selanjutnya, dilakukan **eksplorasi data** melalui visualisasi maupun analisis statistik untuk memahami pola awal, kemudian berlanjut ke tahap **pemodelan** menggunakan metode statistik maupun *machine learning* yang menghasilkan prediksi atau klasifikasi. Tahap akhir berupa **interpretasi hasil**, agar temuan yang diperoleh dapat dipahami secara luas dan dimanfaatkan dalam pengambilan keputusan

Karakteristik Utama

1. Interdisipliner

Data sains bersifat **interdisipliner** karena menggabungkan berbagai bidang pengetahuan seperti **statistika**, **matematika**, **ilmu komputer**, serta pengetahuan **domain spesifik**. Kombinasi ini memungkinkan analisis data dilakukan secara lebih **komprehensif**, di mana aspek teknis dan teoritis berpadu dengan konteks nyata dari permasalahan yang diteliti. Pendekatan lintas disiplin ini memberikan kekuatan dalam mengembangkan model yang tidak hanya akurat secara numerik, tetapi juga relevan dengan kebutuhan praktis.

2. **Berbasis data**

Karakteristik berikutnya adalah bersifat **berbasis data**, artinya setiap hasil analisis, prediksi, maupun rekomendasi didukung oleh **data yang valid** dan dapat diverifikasi. Prinsip ini sejalan dengan konsep *evidence-based decision making*, yang menekankan pentingnya bukti kuat dalam pengambilan keputusan. Dengan demikian, data sains tidak bergantung pada asumsi semata, melainkan menegakkan **integritas akademik** serta akuntabilitas ilmiah.

3. **Iteratif**

Proses data sains bersifat **iteratif**, yakni dilakukan secara **berulang** untuk menemukan pola terbaik dan memperbaiki model yang ada. Tahapan iterasi ini mencakup pembersihan ulang, pengujian metode berbeda, serta evaluasi hasil yang diperoleh. Pendekatan iteratif mencerminkan dinamika ilmu pengetahuan modern yang selalu terbuka terhadap **perbaikan berkelanjutan** (*continuous improvement*), sehingga model yang dikembangkan semakin handal dan adaptif terhadap perubahan data.

4. **Aplikatif**

Terakhir, data sains memiliki sifat **aplikatif**, karena fokus utamanya adalah menyelesaikan **masalah nyata** yang dihadapi oleh organisasi maupun masyarakat. Hasil analisis tidak berhenti pada aspek teoritis, tetapi diterjemahkan menjadi **solusi praktis** yang dapat digunakan untuk mendukung pengambilan

keputusan strategis, menciptakan inovasi, serta meningkatkan efisiensi. Dengan sifat aplikatif ini, data sains membuktikan relevansinya dalam dunia modern yang penuh kompleksitas..

Tahapan Umum

A. Identifikasi masalah

Tahap awal dalam **proyek data sains** adalah **identifikasi masalah**. Pada fase ini, peneliti atau praktisi harus memahami secara mendalam konteks, tujuan, serta permasalahan yang ingin diselesaikan. Identifikasi yang baik mencakup analisis kebutuhan bisnis maupun penelitian, sehingga arah proyek lebih jelas. Misalnya, perusahaan ritel tidak hanya ingin meningkatkan prediksi penjualan musiman, tetapi juga memahami faktor eksternal seperti tren pasar atau pola perilaku konsumen. Pemahaman yang komprehensif ini akan membuat analisis lebih fokus dan bernilai strategis.

B. Pengumpulan data

Setelah masalah terdefinisi, dilakukan **pengumpulan data** dari berbagai sumber, baik internal maupun eksternal. Sumber data harus dipilih berdasarkan relevansi dan kualitas. Data dapat berasal dari basis data perusahaan, survei pelanggan, *application programming interface (API)*, maupun data publik dari lembaga resmi. Sebagai contoh, data transaksi penjualan, data cuaca, serta indeks harga konsumen dapat digabungkan untuk memprediksi permintaan produk secara lebih akurat.

C. Preprocessing data.

Tahap **preprocessing data** mencakup pembersihan, transformasi, normalisasi, serta integrasi data agar siap dianalisis. Proses ini penting untuk menghindari bias dan kesalahan analisis. Data yang tidak lengkap, duplikat, atau mengandung *outlier* perlu ditangani dengan metode yang sesuai. Sebagai ilustrasi, jika ditemukan nilai penjualan negatif akibat kesalahan input, maka data tersebut harus diperbaiki, diimputasi, atau dihapus sesuai standar kualitas data.

D. Pemodelan

Dalam tahap **pemodelan**, digunakan teknik statistik maupun *machine learning* untuk membangun model prediksi atau klasifikasi. Pemilihan algoritma harus mempertimbangkan kompleksitas data, interpretabilitas, serta tujuan analisis. Misalnya, algoritma *random forest* dapat dipakai untuk memprediksi kategori pelanggan, sementara *logistic regression* digunakan untuk menganalisis faktor penyebab churn.

E. Evaluasi

Setelah model dibuat, dilakukan **evaluasi** menggunakan metrik tertentu seperti *accuracy*, *precision*, *recall*, maupun *F1-score*. Evaluasi harus menggunakan data uji yang terpisah agar hasilnya objektif. Sebagai contoh, model prediksi churn

pelanggan dapat dibandingkan dengan data aktual untuk menilai tingkat keandalannya.

F. Deployment dan pemantauan

Tahap terakhir adalah **deployment** model ke dalam sistem operasional serta **pemantauan** kinerjanya secara berkelanjutan. Pemantauan meliputi pengecekan akurasi, adaptasi terhadap data baru, serta perbaikan model bila performa menurun. Misalnya, model rekomendasi produk pada e-commerce perlu diperbarui secara berkala agar tetap sesuai dengan perubahan preferensi pelanggan.

Framework Penelitian Artificial Intelligence (AI)

Kecerdasan Buatan atau *Artificial Intelligence (AI)* merupakan cabang ilmu komputer yang berfokus pada pengembangan sistem yang mampu **meniru kemampuan manusia** seperti pengambilan keputusan, pengenalan pola, hingga pemrosesan bahasa alami. AI tidak hanya menekankan pada aspek teknis komputasi, tetapi juga menonjolkan kemampuan sistem untuk **beradaptasi, menggeneralisasi, dan belajar dari pengalaman**. Dalam konteks penelitian, AI menjadi fondasi penting untuk menghadirkan solusi inovatif yang relevan dengan tantangan global sekaligus mempertimbangkan aspek etika.

Tahapan Framework AI

A. Problem Formulation

Tahap awal dalam **framework AI** adalah **problem formulation**, yaitu mendefinisikan masalah serta tujuan yang ingin dicapai secara jelas dan terukur. Kejelasan rumusan masalah akan memengaruhi pemilihan data, metode, serta indikator evaluasi. Sebagai contoh, sebuah perusahaan perbankan tidak hanya ingin mendeteksi potensi penipuan transaksi secara real-time, tetapi juga mengurangi tingkat *false positive* agar sistem tetap efisien.

B. Knowledge Representation

Tahap berikutnya adalah **knowledge representation**, yakni menentukan cara menyimpan, mengorganisasi, dan merepresentasikan informasi agar dapat diproses secara efektif. Representasi yang baik memungkinkan sistem mengenali pola dan hubungan antar-entitas. Contohnya, penggunaan *graph database* untuk merepresentasikan koneksi antar pengguna dalam jaringan sosial atau relasi transaksi keuangan.

C. Algorithm Selection

Setelah informasi terstruktur, dilakukan **algorithm selection** untuk memilih algoritma pencarian, inferensi, atau optimisasi yang paling relevan. Pemilihan algoritma mempertimbangkan ketersediaan data, kompleksitas perhitungan, serta

interpretabilitas hasil. Misalnya, *decision tree* dipilih untuk analisis yang memerlukan transparansi, sedangkan *deep learning* digunakan untuk pengenalan citra berskala besar.

D. Model Training

Tahap **model training** adalah proses melatih sistem dengan data agar dapat mengenali pola dan membuat prediksi. Kualitas data sangat menentukan hasil pelatihan. Sebagai ilustrasi, *convolutional neural networks* (CNN) dapat digunakan untuk menganalisis citra medis dalam mendeteksi tumor.

E. Evaluation

Setelah model dilatih, dilakukan **evaluation** dengan menggunakan metrik seperti *accuracy*, *precision*, *recall*, dan *F1-score*. Evaluasi harus dilakukan pada data uji yang berbeda dari data pelatihan untuk menghindari *overfitting*. Hal ini memastikan model mampu bekerja pada situasi nyata dengan tingkat keandalan tinggi.

F. Deployment

Tahap akhir adalah **deployment**, yaitu menerapkan model ke lingkungan nyata dan memastikan integrasi dengan sistem yang ada. Contohnya, sistem rekomendasi produk pada e-commerce yang dioperasikan secara otomatis sekaligus dipantau performanya. Pemantauan berkelanjutan penting

agar model tetap relevan seiring dengan perubahan data dan kebutuhan pengguna.

Sebagai contoh nyata, sebuah rumah sakit besar mengembangkan sistem **AI berbasis CNN** untuk mendeteksi kanker paru-paru dari citra CT-Scan. Proses dimulai dengan **problem formulation** untuk mempercepat diagnosis medis. Data ribuan citra pasien dikumpulkan dan disiapkan melalui **knowledge representation** yang tepat. Algoritma *deep learning* dipilih pada tahap **algorithm selection**. Setelah melalui **model training**, sistem diuji dengan metrik *accuracy* dan *recall* untuk memastikan sensitivitas terhadap kasus kanker. Model yang lulus evaluasi kemudian di-**deployment** ke rumah sakit, sehingga dokter terbantu dalam membuat keputusan klinis lebih cepat dan akurat.

Kelebihan dan Kekurangan

a. Kelebihan

Salah satu **kelebihan utama** dari *Artificial Intelligence (AI)* adalah sifatnya yang **fleksibel** dan mampu diaplikasikan di berbagai bidang, mulai dari kesehatan, pendidikan, hingga industri. AI dapat membantu **otomatisasi proses**, meningkatkan **efisiensi kerja**, serta menghasilkan **prediksi yang lebih akurat** melalui analisis data yang kompleks. Contohnya, dalam bidang kesehatan, AI digunakan untuk mendukung diagnosis medis dengan memanfaatkan citra radiologi.

b. Kekurangan

Meskipun menawarkan banyak manfaat, AI juga memiliki **kekurangan** yang signifikan. Pengembangan dan implementasi AI membutuhkan **sumber daya komputasi besar**, termasuk perangkat keras dengan performa tinggi dan energi yang tidak sedikit. Selain itu, terdapat risiko **bias data** serta keterbatasan dalam hal **interpretabilitas model**, sehingga hasil yang diberikan kadang sulit dipahami pengguna non-teknis. Oleh karena itu, penelitian dan penggunaan AI harus memperhatikan aspek **efisiensi**, **etika**, dan **keberlanjutan** agar dampaknya benar-benar bermanfaat bagi masyarakat.

Framework Penelitian Machine Learning (ML)

Machine Learning (ML) adalah cabang dari *Artificial Intelligence (AI)* yang berfokus pada pengembangan algoritma yang mampu **belajar dari data** serta meningkatkan kinerja seiring waktu tanpa instruksi eksplisit. ML menjadi fondasi penting dalam era digital karena diaplikasikan dalam sistem rekomendasi, pengenalan wajah, kendaraan otonom, hingga analisis prediktif di bidang kesehatan dan keuangan. Definisi ini menegaskan bahwa ML bukan hanya tentang otomatisasi, melainkan tentang **kemampuan sistem beradaptasi** dengan pola baru yang muncul dalam data.

Tahapan Framework ML

A. Data Collection

Tahap awal dalam **framework Machine Learning (ML)** adalah **data collection**, yakni proses mengumpulkan dataset dari

sumber yang relevan dan terpercaya. Contoh, perusahaan transportasi mengumpulkan data dari **sensor IoT** pada kendaraan untuk memantau penggunaan bahan bakar. Kualitas serta keragaman data menjadi pondasi penting agar model dapat menghasilkan prediksi yang akurat dan dapat diandalkan.

B. Data Cleaning & Preprocessing

Tahap berikutnya adalah **data cleaning & preprocessing** untuk menjamin kualitas data. Proses ini mencakup penanganan data hilang, deteksi dan penghapusan *outliers*, serta transformasi dan normalisasi. Misalnya, data kecepatan yang tidak logis dari sensor diperbaiki agar tidak mengganggu analisis. Tahap ini memastikan model tidak bias akibat kualitas data yang buruk.

C. Feature Engineering

Pada tahap **feature engineering**, peneliti memilih atau membangun fitur yang relevan untuk memperkaya informasi yang dimiliki model. Contohnya, dari data perjalanan kendaraan dibuat variabel baru seperti rata-rata konsumsi bahan bakar harian atau pola penggunaan berdasarkan musim. Langkah ini meningkatkan kemampuan model dalam mengenali pola tersembunyi.

D. Model Selection

Tahap **model selection** menekankan pada pemilihan algoritma yang sesuai dengan tujuan analisis. Misalnya, digunakan *random forest* untuk memprediksi kapan kendaraan memerlukan servis berdasarkan riwayat penggunaan. Pemilihan ini mempertimbangkan akurasi, kecepatan komputasi, dan interpretabilitas.

E. Training & Validation

Dalam tahap **training & validation**, model dilatih menggunakan data historis dan diuji dengan validasi silang untuk mencegah *overfitting*. Contoh, data perjalanan tahun sebelumnya digunakan untuk melatih model prediksi perawatan, kemudian divalidasi dengan subset berbeda untuk menguji kemampuan generalisasi.

F. Testing

Tahap **testing** dilakukan dengan data uji independen. Misalnya, data perjalanan terbaru dipakai untuk menilai ketepatan prediksi jadwal servis kendaraan. Hasil evaluasi menggunakan metrik seperti *accuracy* atau *recall* memberikan gambaran keandalan model.

G. Deployment

Jika model memenuhi kriteria, tahap **deployment** dilakukan ke dalam sistem operasional perusahaan. Model diintegrasikan ke

aplikasi manajemen armada agar dapat memberikan rekomendasi servis secara otomatis dan real-time.

H. Monitoring

Tahap terakhir adalah **monitoring**, yaitu pemantauan performa model secara berkelanjutan. Jika pola perjalanan atau perilaku penggunaan berubah, model diperbarui agar tetap relevan dan akurat. Monitoring ini memastikan keberlanjutan manfaat model.

Kelebihan dan Kekurangan

a. Kelebihan

Machine Learning (ML) memiliki sejumlah **kelebihan** penting, salah satunya sifat **adaptif**, yaitu kemampuan model untuk menyesuaikan diri dengan pola baru seiring bertambahnya data. Selain itu, ML unggul dalam **mengenali pola kompleks** yang sering kali sulit atau bahkan tidak mungkin dideteksi secara manual. Keunggulan ini membuat ML relevan di berbagai sektor, mulai dari kesehatan yang menggunakan algoritma untuk menganalisis citra medis, keuangan untuk mendeteksi anomali transaksi, hingga transportasi dalam memprediksi kebutuhan perawatan kendaraan. Dengan kecepatan pemrosesan data yang tinggi, ML mampu memberikan **prediksi akurat** yang mendukung pengambilan keputusan strategis.

b. Kekurangan

Meskipun memiliki banyak manfaat, ML juga menghadapi sejumlah **kekurangan**. Model ML umumnya membutuhkan **volume data yang sangat besar** untuk dilatih agar performanya optimal, sehingga tidak selalu sesuai bagi kasus dengan data terbatas. Selain itu, banyak algoritma modern, khususnya *deep learning*, memiliki tantangan dalam **interpretabilitas**, sehingga hasil analisis sulit dijelaskan kepada pengguna non-teknis. Kompleksitas ini dapat menimbulkan hambatan dalam penerimaan hasil oleh pemangku kepentingan. Oleh karena itu, penelitian ML perlu mengutamakan **kualitas data, transparansi model**, serta memperhatikan dimensi **etika dan keberlanjutan**,

Framework Penelitian Deep Learning (DL)

Deep Learning (DL) merupakan cabang dari *Machine Learning (ML)* yang menggunakan **jaringan saraf tiruan berlapis** atau *deep neural networks* untuk mempelajari representasi data yang kompleks. DL sangat efektif dalam menangani data berskala besar seperti gambar, teks, maupun suara, karena mampu mengekstraksi fitur secara otomatis tanpa intervensi manual. Definisi ini menekankan bahwa DL tidak hanya sekadar algoritma, melainkan pendekatan komputasi yang mampu **beradaptasi** dan **belajar hierarkis** dari data.

Tahapan Framework DL

A. Data Preparation

Tahap awal dalam **framework Deep Learning (DL)** adalah **data preparation**, yaitu mengumpulkan, membersihkan, serta

melakukan anotasi pada data. Contoh nyata, dalam bidang kesehatan ribuan citra X-ray paru-paru pasien dikumpulkan, kemudian diberi label apakah menunjukkan pneumonia atau tidak. Proses ini juga meliputi verifikasi kualitas anotasi oleh ahli medis untuk mengurangi bias. Kualitas data yang baik menjadi pondasi utama keberhasilan model.

B. Model Architecture Design

Selanjutnya adalah **model architecture design**, yakni menentukan arsitektur jaringan saraf yang sesuai. Pada kasus pneumonia, dipilih *Convolutional Neural Network (CNN)* karena unggul dalam menganalisis citra medis. CNN mampu mengenali pola tekstur dan bentuk yang sulit dideteksi manusia. Dalam beberapa penelitian, kombinasi arsitektur seperti *CNN* dengan *Transformer* juga digunakan untuk meningkatkan akurasi.

C. Training with GPUs/TPUs

Tahap **training** dilakukan dengan menggunakan *GPUs* atau *TPUs* agar jutaan parameter pada CNN dapat dioptimalkan. Citra X-ray yang sudah diberi label dilatih berulang kali hingga model mampu membedakan paru sehat dan paru terinfeksi. Tahap ini memerlukan waktu lama, sumber daya komputasi besar, dan strategi pengelolaan data untuk memastikan efisiensi.

D. Regularization & Optimization

Untuk mencegah **overfitting**, digunakan teknik *dropout*, *batch normalization*, atau *weight decay*. Optimalisasi seperti *Adam optimizer* atau *stochastic gradient descent (SGD)* membantu mempercepat konvergensi. Dengan strategi ini, model lebih stabil dan mampu beradaptasi dengan data baru.

E. Evaluation

Model kemudian diuji pada citra X-ray pasien yang belum pernah dilihat sebelumnya. Evaluasi dilakukan dengan metrik *accuracy*, *precision*, *recall*, dan *AUC* untuk menilai seberapa baik model mendeteksi pneumonia. Evaluasi multimetrik memberikan gambaran lebih komprehensif terhadap performa model.

F. Deployment

Setelah kinerjanya memuaskan, model di-**deployment** ke sistem radiologi rumah sakit. Dokter dapat menggunakan model ini sebagai alat bantu diagnosis untuk mempercepat deteksi pneumonia. Tahap ini juga dilengkapi dengan **monitoring berkelanjutan** untuk memastikan model tetap relevan, serta mekanisme pembaruan data agar akurasinya konsisten.

Kelebihan dan Kekurangan

a. Kelebihan

Deep Learning (DL) memiliki sejumlah **kelebihan utama**, salah satunya kemampuan unggul dalam mengolah **data kompleks** serta mengenali pola **non-linear** yang sulit ditangkap algoritma tradisional. Keunggulan ini menjadikan DL sangat efektif untuk berbagai bidang, mulai dari pengenalan wajah, analisis citra medis, deteksi objek, hingga pemrosesan bahasa alami (*natural language processing*). Dengan representasi berlapis, DL dapat mengekstraksi fitur yang mendalam secara otomatis sehingga menghasilkan prediksi yang lebih akurat dan konsisten. Selain itu, DL mampu beradaptasi terhadap data baru, menjadikannya relevan di era *big data*.

b. **Kekurangan**

DL juga menghadapi beberapa **kekurangan** yang signifikan. Pertama, model DL membutuhkan **sumber daya komputasi besar**, termasuk perangkat keras khusus seperti *GPU* atau *TPU*, yang tidak selalu terjangkau. Kedua, DL sangat bergantung pada **dataset berukuran besar dan bervariasi** agar dapat dilatih dengan baik, sehingga sulit diterapkan pada kasus dengan data terbatas. Ketiga, terdapat tantangan **interpretabilitas hasil**, karena model berlapis dalam DL sering dianggap sebagai *black box* yang sulit dijelaskan kepada pengguna non-teknis. Oleh karena itu, penelitian dan penerapan DL perlu menekankan **efisiensi komputasi, transparansi model**, serta memperhatikan aspek **etika dan keberlanjutan**.

Perbandingan AI, ML, dan DL

A. Kompleksitas Data

Artificial Intelligence (AI) mencakup pendekatan yang luas dalam pengolahan informasi untuk pengambilan keputusan, baik dari data terstruktur maupun semi-terstruktur. **Machine Learning (ML)** biasanya lebih banyak digunakan pada **data tabular** yang terorganisir, seperti data keuangan atau survei. Sementara itu, **Deep Learning (DL)** sangat unggul dalam mengolah **data besar tidak terstruktur** seperti citra medis, teks panjang, maupun sinyal suara, karena mampu mengekstraksi representasi yang kompleks.

B. Kebutuhan Data

Dari sisi kebutuhan data, **AI** secara umum dapat bekerja dengan data dalam jumlah **moderat** karena pendekatannya masih berbasis aturan atau logika. **ML** membutuhkan **jumlah data lebih besar** agar model dapat dilatih dengan baik dan menghasilkan prediksi yang stabil. Sedangkan **DL** sangat bergantung pada **dataset yang sangat besar dan beragam** untuk mempelajari pola non-linear, sehingga performa optimal hanya tercapai dengan data masif.

C. Interpretabilitas

Model **AI** tradisional serta sebagian metode **ML** relatif lebih mudah dijelaskan dan diinterpretasikan, sehingga cocok untuk konteks yang membutuhkan transparansi. Sebaliknya, **DL** kerap

disebut sebagai *black box* karena arsitektur jaringan yang dalam dan kompleks membuat alasan di balik prediksi sulit dipahami, khususnya oleh pengguna non-teknis.

D. Komputasi

Kebutuhan komputasi pada **AI** tradisional relatif **rendah hingga menengah**, pada **ML** meningkat ke tingkat **menengah**, sedangkan **DL** menuntut komputasi sangat intensif dengan perangkat keras khusus seperti *GPU* atau *TPU*. Hal ini membuat biaya dan infrastruktur menjadi pertimbangan penting dalam implementasi.

E. Aplikasi

AI memiliki spektrum aplikasi yang **luas**, dari sistem pakar, chatbot, hingga robotika. **ML** banyak diterapkan pada data terstruktur seperti prediksi pasar saham, deteksi penipuan, dan segmentasi pelanggan. **DL** lebih berfokus pada pemrosesan **data kompleks**, misalnya pengenalan wajah, sistem penerjemahan otomatis, serta analisis citra medis yang memerlukan detail mendalam.

Studi Kasus Penerapan

A. Bisnis

- a. **Prediksi churn pelanggan** dengan ML.

Informasi Artikel

- a) **Judul:** *Improving customer retention in taxi industry using travel data analytics: A churn prediction study*
- b) **Penulis:** A.L.D. Loureiro, V.L. Miguéis, Álvaro Costa, Michel Ferreira
- c) **Tahun:** 2025
- d) **Jurnal:** *Journal of Retailing and Consumer Services*, Vol. 85, 104288
- e) **DOI:** <https://doi.org/10.1016/j.jretconser.2025.104288>

Latar Belakang

Prediksi churn pelanggan merupakan aspek strategis dalam pengelolaan bisnis, khususnya sektor transportasi. *Customer churn* menggambarkan kondisi ketika pelanggan berhenti menggunakan layanan tertentu dan beralih ke alternatif lain. Dalam industri taksi yang **non-kontraktual**, tantangan ini semakin kompleks karena tidak ada ikatan formal yang menahan pelanggan. Oleh sebab itu, penerapan *machine learning (ML)* dipandang relevan untuk memetakan perilaku perjalanan pelanggan, sehingga potensi churn dapat diprediksi secara lebih dini.

Metodologi

Penelitian ini menggunakan **data perjalanan taksi selama 52 bulan** di Coimbra, Portugal, dengan pengolahan berbasis model klasifikasi. Algoritma yang diimplementasikan mencakup **LightGBM, Random Forest (RF), dan Artificial Neural Networks (ANN)**. Untuk

meningkatkan akurasi, dikembangkan pula pendekatan **stacking ensemble**. Evaluasi kinerja dilakukan dengan metrik **F1-score, AUC, Expected Maximum Profit Criterion (EMPC)**, serta analisis *lift* untuk mengukur efektivitas prediksi pada segmen pelanggan dengan risiko churn tertinggi.

Temuan Utama

- a) **LightGBM** terbukti menjadi model paling unggul dengan EMPC tertinggi.
- b) Variabel yang berpengaruh signifikan antara lain **recency** (keterkinian penggunaan layanan), **frequency** (frekuensi pemesanan), serta faktor musiman seperti bulan liburan.
- c) Penggunaan **SHAP (Shapley Additive Explanations)** memperlihatkan interpretasi mendalam, misalnya risiko churn lebih tinggi pada pelanggan yang dominan menggunakan aplikasi digital.

Implikasi

Temuan ini menunjukkan bahwa **ML untuk prediksi churn** mampu mendukung perusahaan taksi dalam merumuskan strategi retensi yang lebih presisi. Rekomendasi strategis meliputi:

- a) **Kampanye digital yang dipersonalisasi**, seperti push notification atau promosi berbasis riwayat perjalanan.

- b) **Peningkatan kualitas layanan** melalui pengurangan waktu tunggu dan pengalaman layanan yang konsisten.
- c) **Program loyalitas adaptif** bagi pelanggan dengan penurunan intensitas penggunaan.

Kesimpulan

Secara keseluruhan, penelitian ini menegaskan bahwa penerapan **machine learning** dalam memprediksi churn dapat memperkuat **retensi pelanggan** di industri taksi. Pendekatan profit-driven yang dipadukan dengan interpretabilitas model memberikan nilai tambah dalam pengalokasian sumber daya yang lebih efektif, sekaligus mendukung penciptaan loyalitas pelanggan jangka panjang dan peningkatan keuntungan berkelanjutan.

- b. **Analisis sentimen** menggunakan DL.

Informasi Artikel

- 1) Judul: *Deep Learning Techniques for Sentiment Analysis in Code-Switched Hausa-English Tweets*
- 2) Penulis: Yusuf Aliyu, Aliza Sarlan, Kamaluddeen Usman Danyaro, Abdullahi Sani Abd Rahman, Aminu Aminu Muazu, Mustapha Yusuf Abubakar
- 3) Tahun: 2025
- 4) Jurnal: *International Journal of Information Management Data Insights*, Vol. 5, 100330
- 5) DOI: <https://doi.org/10.1016/j.jjime.2025.100330>

Latar Belakang

Analisis sentimen menjadi salah satu bidang penting dalam *Natural Language Processing (NLP)*, khususnya di era media sosial yang penuh dengan opini publik. Tantangan besar muncul ketika data berasal dari bahasa dengan sumber daya terbatas (*low-resource language*), seperti Hausa, yang sering digunakan secara *code-switched* dengan bahasa Inggris. Fenomena ini menyebabkan kesulitan dalam pemodelan sentimen karena adanya variasi linguistik dan struktur semantik yang kompleks. Penelitian ini berfokus pada peningkatan akurasi analisis sentimen melalui penerapan **Deep Learning (DL)** dan pengembangan algoritma *stemming* baru untuk bahasa Hausa.

Metodologi

Dataset terdiri dari **16.849 tweet** yang telah dilabeli positif, negatif, dan netral. Proses dimulai dengan *preprocessing* (cleaning, stemming, tokenization), kemudian diuji pada beberapa arsitektur DL: **CNN, LSTM, GRU**, serta model berbasis *transformer* seperti **mBERT, RoBERTa, XLM-R, dan AfriBERTa**. Penulis juga mengembangkan kerangka kerja **hyperparameter tuning** yang efisien untuk menyeimbangkan akurasi dan biaya komputasi. Evaluasi kinerja dilakukan

dengan menggunakan metrik **Accuracy, Precision, Recall, dan F1-score**.

Temuan Utama

- a) Model **AfriBERTa** menunjukkan performa tertinggi dengan F1-score **0.92** dan akurasi **0.919**, melampaui baseline dan model DL konvensional.
- b) Algoritma *stemming* khusus Hausa mampu mengurangi kesalahan linguistik dan meningkatkan representasi kata pada data *code-switched*.
- c) Framework tuning hyperparameter memberikan hasil efisien dengan tetap menjaga stabilitas performa model.

Implikasi

Penelitian ini memiliki implikasi signifikan dalam pengembangan **alat NLP inklusif** untuk bahasa yang kurang terwakili. Aplikasinya meliputi **monitoring media sosial, analisis opini publik, dan layanan pelanggan**. Lebih jauh, keberhasilan AfriBERTa menegaskan pentingnya model *transformer* dalam mengatasi kerumitan data multilingual. Hal ini membuka peluang untuk menciptakan teknologi NLP yang lebih adil, dapat diakses, dan relevan bagi komunitas linguistik terpinggirkan.

Kesimpulan

Secara keseluruhan, penelitian ini menegaskan bahwa penerapan **Deep Learning** berbasis *transformer* sangat efektif untuk analisis sentimen pada bahasa dengan sumber daya terbatas. Dengan kombinasi preprocessing inovatif dan tuning hyperparameter, pendekatan ini tidak hanya meningkatkan performa akademis, tetapi juga memberikan dampak praktis dalam memperkuat integrasi teknologi AI di berbagai konteks sosial dan komersial.

c. Rekomendasi produk dengan AI.

Informasi Artikel

- 1) **Judul:** *Artificial intelligence for herbicide recommendation: Case study for the use of clomazone in Brazilian soils*
- 2) **Penulis:** Hamurábi Anizio Lins, Matheus de Freitas Souza, Lucrecia Pacheco Batista, Luma Lorena Loureiro da Silva Rodrigues, Francisca Daniele da Silva, Bruno Caio Chaves Fernandes, Stefeson Bezerra de Melo, Paulo Sergio Fernandes das Chagas, Daniel Valadão Silva
- 3) **Tahun:** 2024
- 4) **Jurnal:** *Smart Agricultural Technology*, Vol. 9, 100699
- 5) **DOI:**
<https://doi.org/10.1016/j.atech.2024.100699>

Latar Belakang

Rekomendasi produk dengan **Artificial Intelligence (AI)** tidak hanya relevan pada sektor *e-commerce*, tetapi juga semakin banyak digunakan dalam **pertanian presisi**. Dalam konteks artikel ini, fokus penelitian adalah dosis herbisida *clomazone* yang sering kali disarankan hanya berdasarkan kadar liat tanah. Pendekatan tersebut terbukti menimbulkan risiko seperti overdosis atau kerusakan lingkungan. Oleh sebab itu, penggunaan **Artificial Neural Networks (ANNs)** dipandang strategis untuk menghasilkan rekomendasi berbasis data yang lebih akurat.

Metodologi

Penelitian ini menggunakan 45 sampel tanah dari berbagai wilayah Brasil dengan variasi sifat fisik dan kimia. Data ini diproses menggunakan **Multilayer Perceptron (MLP)** dalam jaringan saraf tiruan. Variabel masukan meliputi kapasitas tukar kation (CTC), kandungan kalium (K), magnesium (Mg), dan silt. Teknik **feature selection** diterapkan untuk meningkatkan akurasi model. Evaluasi kinerja dilakukan menggunakan metrik seperti R^2 , *Mean Absolute Error (MAE)*, *Root Mean Square Error (RMSE)*, serta korelasi Pearson.

Temuan Utama

- a) ANN dengan 4 variabel masukan (CTC, K, Mg, dan silt) terbukti memberikan prediksi paling akurat dengan R^2 mencapai 0.84.
- b) Faktor paling berpengaruh adalah kandungan kalium (37%) dan silt (29%), disusul Mg (22%) dan CTC (12%).
- c) Rekomendasi dosis herbisida berbasis ANN rata-rata **20% lebih rendah** dibanding panduan komersial, sehingga lebih efisien dan ramah lingkungan.

Implikasi

Hasil penelitian ini menunjukkan bahwa sistem rekomendasi berbasis AI dapat meningkatkan **efisiensi penggunaan sumber daya**, mengurangi risiko pencemaran lingkungan, dan mendukung praktik **pertanian berkelanjutan**. Secara lebih luas, konsep ini sejalan dengan penerapan sistem rekomendasi produk pada industri lain, misalnya *e-commerce*, di mana personalisasi dan presisi menjadi kunci peningkatan nilai layanan.

Kesimpulan

Secara keseluruhan, penelitian ini menegaskan bahwa **AI mampu memberikan rekomendasi produk yang lebih tepat dan berbasis data**. Dalam konteks

pertanian, ANN dapat membantu petani menentukan dosis herbisida dengan presisi, sehingga mendukung keberlanjutan, menekan biaya, dan meningkatkan produktivitas. Model ini merefleksikan esensi *product recommendation with AI*, yaitu pengambilan keputusan yang cerdas, efisien, dan berorientasi masa depan.

B. Kesehatan

- a) **Deteksi penyakit** dari citra medis dengan DL.

Informasi Artikel

a) **Judul:** *Multi-model deep learning approach for the classification of kidney diseases using medical images*

b) **Penulis:** Waleed Obaid, Abir Hussain, Tamer Rabie, Dhafar Hamed Abd, Wathiq Mansoor

c) **Tahun:** 2025

d) **Jurnal:** *Informatics in Medicine Unlocked*, Vol. 57, 101663

e) **DOI:** <https://doi.org/10.1016/j.imu.2025.101663>

Latar Belakang

Deteksi dini penyakit ginjal, termasuk *renal cell carcinoma*, kista, dan batu ginjal, menjadi prioritas penting dalam layanan kesehatan karena gejala awalnya sering tidak terdeteksi. Kondisi ini diperparah oleh keterbatasan jumlah tenaga nefrolog di seluruh dunia serta meningkatnya prevalensi penyakit ginjal. Dalam konteks tersebut, **deep**

learning (DL) hadir sebagai inovasi yang mampu mengolah citra medis secara otomatis untuk membantu diagnosis yang lebih cepat, konsisten, dan akurat. Artikel ini menitikberatkan pada sistem klasifikasi multi-model yang dirancang untuk mengoptimalkan kinerja dalam mengidentifikasi berbagai kelainan ginjal.

Metodologi

Penelitian mengandalkan dataset besar berisi **27.145 citra medis** yang mencakup empat kategori: normal, kista, tumor, dan batu ginjal. Untuk klasifikasi, digunakan **16 model CNN** populer, di antaranya *ResNet*, *InceptionV3*, *Darknet53*, *VGG16*, dan *GoogLeNet*. Upaya **data balancing** serta **augmentasi citra** dilakukan agar model tidak bias terhadap kelas mayoritas dan memiliki kemampuan generalisasi yang lebih baik. Evaluasi performa melibatkan metrik komprehensif seperti *accuracy*, *precision*, *recall*, *specificity*, *F1-score*, *Matthews Correlation Coefficient (MCC)*, dan *Kappa statistic*, yang memberikan gambaran menyeluruh terhadap keandalan sistem.

Temuan Utama

- a) Model **Darknet53** menunjukkan performa paling unggul dengan akurasi **99,69%**, *recall* **99,66%**, dan *precision* **99,77%**.

- b)** Pendekatan multi-model lebih efektif daripada metode tunggal karena mampu mengatasi variasi kompleksitas citra medis dan meningkatkan generalisasi.
- c)** Penerapan *Grad-CAM* menambah nilai interpretabilitas dengan menyoroti area penting yang digunakan model dalam proses klasifikasi, sehingga meningkatkan transparansi sistem.

Implikasi

Hasil penelitian ini memiliki dampak praktis yang signifikan, di antaranya:

- a) Mempercepat diagnosis sekaligus mengurangi beban kerja tenaga medis, khususnya radiolog.
- b) Menjamin konsistensi hasil dengan akurasi tinggi, bahkan pada citra medis dengan kualitas dan variasi yang berbeda.
- c) Menawarkan potensi integrasi dengan **explainable AI (XAI)**, yang dapat meningkatkan kepercayaan klinis dan mempercepat adopsi di fasilitas kesehatan.

Kesimpulan

Penelitian ini membuktikan bahwa **deep learning** berbasis multi-model merupakan solusi yang kuat untuk deteksi penyakit ginjal. Dengan akurasi yang mendekati sempurna serta dukungan teknik interpretabilitas, pendekatan ini memberikan dasar yang kokoh bagi pengembangan aplikasi medis berbasis AI. Ke depannya, sistem ini dapat

diperluas pada berbagai modalitas pencitraan medis dan diadaptasi pada lingkungan klinis dengan keterbatasan sumber daya, sehingga berkontribusi nyata pada peningkatan kualitas pelayanan kesehatan global.

b) **Prediksi penyebaran penyakit** dengan ML

Informasi Artikel

- a) **Judul:** Analysis of disease severity and mortality prediction using machine learning during COVID-19
- b) **Penulis:** Hodjat (Hojatollah) Hamidi, Mostafa Moradi
- c) **Tahun:** 2025
- d) **Jurnal:** Acta Psychologica, Vol. 258, 105136
- e) **DOI:** <https://doi.org/10.1016/j.actpsy.2025.105136>

Latar Belakang

Prediksi penyebaran penyakit menjadi aspek fundamental dalam strategi kesehatan global, terutama ketika menghadapi penyakit menular dengan skala pandemi. *COVID-19* memperlihatkan bahwa metode statistik tradisional sering kali tidak memadai untuk memprediksi pola penyebaran yang dinamis dan kompleks. Oleh karena itu, penerapan **Machine Learning (ML)** dipandang sebagai solusi strategis karena mampu mengolah data berskala besar, mendeteksi pola tersembunyi, serta menghasilkan prediksi yang lebih akurat. Penelitian Hamidi dan Moradi (2025) memberikan kontribusi penting dalam menunjukkan bagaimana ML dapat mengatasi

keterbatasan metode konvensional dengan memanfaatkan data klinis dan publik secara komprehensif.

Metodologi

Penelitian ini memanfaatkan dataset publik dan klinis lebih dari 1 juta pasien COVID-19 dari pemerintah Meksiko. Data diolah melalui tahapan **preprocessing** untuk menangani ketidakseimbangan kelas, **Principal Component Analysis (PCA)** untuk reduksi dimensi, dan pemilihan fitur penting. Selanjutnya, beragam algoritma diuji, meliputi *Logistic Regression*, *Random Forest*, *Support Vector Machine (SVM)*, *Gradient Boosting*, hingga *Neural Networks*. Kinerja model dievaluasi menggunakan metrik **Akurasi, Sensitivitas, Spesifisitas, dan F1-score**, sehingga dapat diperoleh gambaran yang lebih menyeluruh mengenai kekuatan dan kelemahan masing-masing pendekatan.

Temuan Utama

- a) **Logistic Regression** tampil unggul dengan akurasi 97% pada pasien selamat dan 80% pada pasien meninggal, menunjukkan keseimbangan interpretabilitas dan performa.
- b) **Faktor signifikan yang memengaruhi mortalitas** meliputi usia lanjut, komorbiditas (diabetes, hipertensi, obesitas), serta indikator klinis seperti kebutuhan intubasi dan perawatan ICU.

- c) **Hasil PCA mengidentifikasi 10 fitur utama** yang berkontribusi besar dalam memprediksi tingkat keparahan dan risiko kematian.

Implikasi

Integrasi **ML** dalam prediksi penyakit menghadirkan peluang strategis bagi sistem kesehatan. Model tidak hanya mendukung proses **triase pasien berisiko tinggi**, tetapi juga membantu rumah sakit dalam mengalokasikan sumber daya medis seperti ventilator dan kapasitas ICU secara lebih efisien. Selain itu, hasil penelitian ini memberikan landasan untuk pengembangan sistem prediksi adaptif yang dapat diterapkan pada berbagai penyakit menular di masa depan.

Kesimpulan

Secara keseluruhan, penelitian ini membuktikan bahwa penerapan **Machine Learning** tidak terbatas pada diagnosis, tetapi juga memiliki peran vital dalam prediksi mortalitas. Dengan pendekatan berbasis data yang kuat, ML mampu meningkatkan efektivitas manajemen sumber daya, mendukung mitigasi pandemi, dan memperkuat keselamatan pasien di tingkat global.

- c) **Pendukung keputusan klinis** dengan AI.

Informasi Artikel

- a) **Judul:** Development of a novel artificial intelligence clinical decision support tool for hand surgery: HandRAG

b) **Penulis:** Berk B. Ozmen, Nishant Singh, Kavach Shah, Ibrahim Berber, Damanjit Singh, Eugene Pinsky, Antonio Rampazzo, Graham S. Schwarz

c) **Tahun:** 2025

d) **Jurnal:** Journal of Hand and Microsurgery, Vol. 17, 100293

e) **DOI:** <https://doi.org/10.1016/j.jham.2025.100293>

f) **Link Artikel:**

<https://doi.org/10.1016/j.jham.2025.100293>

Latar Belakang

Clinical Decision Support (CDS) menjadi komponen penting dalam praktik kedokteran modern, terutama ketika keputusan harus dibuat dengan mempertimbangkan **anatomi kompleks, kondisi pasien yang beragam, dan teknik operasi yang terus berkembang**. Dalam konteks bedah tangan, pengambilan keputusan sering kali menghadapi keterbatasan informasi yang akurat dan real-time. Meskipun *Large Language Models (LLMs)* telah menawarkan solusi berbasis data, keterbatasan seperti **akses terbatas pada literatur ilmiah terkini** dan risiko *hallucination* masih menjadi tantangan. Artikel ini menghadirkan **HandRAG**, sebuah sistem CDS berbasis **Artificial Intelligence (AI)** yang memanfaatkan *Retrieval-Augmented Generation (RAG)* untuk memberikan rekomendasi **evidence-based** dengan **jejak sitasi langsung ke literatur ilmiah**.

Metodologi

Pengembangan HandRAG dilakukan dengan menghimpun lebih dari **4.510 artikel open-access** (2000-2024) terkait bedah tangan. Artikel diproses melalui metode **RAPTOR**, yang mengorganisasi pengetahuan dalam struktur hierarkis. Teks dibersihkan, diubah menjadi **representasi semantik** melalui *vector embeddings*, lalu dikelompokkan menggunakan *document clustering* untuk memaksimalkan presisi pencarian. Saat menerima pertanyaan klinis, sistem melakukan **ekspansi kueri**, menyeleksi **segmen literatur paling relevan**, dan menghasilkan jawaban terintegrasi melalui LLM. Evaluasi kinerja dilakukan pada **15 skenario klinis** dengan metrik **G-Eval Correctness** dan **Semantic Evaluation Metrics (SEM)** untuk menilai akurasi serta relevansi jawaban.

Temuan Utama

1. **Performa stabil** dengan skor rata-rata **0,79 (G-Eval)** dan **0,75 (SEM)**, mayoritas pada tingkat **kepercayaan moderat**.
2. Pertanyaan klinis dengan dasar literatur yang kuat, seperti *zone 2 flexor tendon repair* dan terapi **Dupuytren**, memperoleh **skor kesesuaian tertinggi**.
3. Tantangan muncul pada skenario dengan literatur terbatas, misalnya *pyogenic flexor tenosynovitis* pada pasien imunokompromais atau patologi **TFCC** pada penderita diabetes.

4. Integrasi RAG **menekan risiko *hallucination*** dengan memastikan jawaban selalu **berbasis pada referensi yang tervalidasi**.

Implikasi

Secara praktis, HandRAG berpotensi **mempercepat, menstandarkan, dan meningkatkan kualitas** keputusan medis, sekaligus menyediakan **transparansi sumber** yang memperkuat *evidence-based practice*. Selain itu, sistem ini dapat dimanfaatkan dalam **pendidikan klinis**, memberi akses mudah pada literatur terkini. Namun, implementasi klinis skala luas tetap membutuhkan **validasi empiris oleh praktisi, pembaruan rutin basis data**, serta kepatuhan pada **regulasi medis** dan aspek **keselamatan pasien**.

Kesimpulan

HandRAG menunjukkan bagaimana kombinasi **AI, RAG, dan LLM** dapat menjadi solusi CDS yang **akurat, transparan, dan kontekstual** di bidang bedah tangan. Dengan pengembangan berkelanjutan, validasi klinis, dan tata kelola yang baik, pendekatan ini berpotensi menjadi **model rujukan** untuk mendukung praktik medis di berbagai disiplin lain.

C. Sosial

- a) **Analisis media sosial** untuk opini publik.

Informasi Artikel

- a) **Judul:** Analyzing social media discourse of avian influenza outbreaks
- b) **Penulis:** Marzieh Soltani, Shayan Sharif, Rozita Dara
- c) **Tahun:** 2025
- d) **Jurnal:** *Natural Language Processing Journal*, Vol. 12, 100176
- e) **DOI:** <https://doi.org/10.1016/j.nlp.2025.100176>

Latar Belakang

Analisis media sosial telah berkembang menjadi instrumen kritis dalam memahami pola pikir dan persepsi publik terhadap isu kesehatan global. Wabah *avian influenza* tidak hanya mengancam industri unggas dan keanekaragaman hayati, tetapi juga menimbulkan implikasi serius bagi **kesehatan manusia, keamanan pangan, dan stabilitas ekonomi internasional**. Media sosial seperti X (*Twitter*) menyediakan data real-time yang kaya akan **emosi, narasi, dan opini masyarakat**, menjadikannya sumber yang bernilai dalam mendeteksi dinamika wacana publik. Dengan memanfaatkan **Natural Language Processing (NLP)** dan *large language models (LLMs)*, penelitian ini berusaha menggali pemetaan persepsi masyarakat secara komprehensif, baik dalam aspek psikologis, sosial, maupun kebijakan publik.

Metodologi

Penelitian ini menganalisis lebih dari **dua juta postingan X** yang dikumpulkan antara Juli 2021 hingga April 2023. Untuk mencapai hasil yang valid, digunakan kombinasi model canggih: **DistilBERT** untuk menyaring konten relevan, **Mixtral-8x7B** untuk mengklasifikasi tema diskusi, **BERTopic** untuk memodelkan topik secara sistematis, dan **RoBERTa** untuk menelaah sentimen serta emosi. Pendekatan multi-model ini menghasilkan kerangka analisis yang lebih akurat dalam mengidentifikasi isu utama seperti penyebaran penyakit lintas spesies, dampak ekonomi, respon kebijakan pemerintah, serta narasi yang berkembang di kalangan pengguna.

Temuan Utama

- a) Lebih dari **70% percakapan publik** berkonotasi negatif, mencakup kekhawatiran atas **kenaikan harga telur dan ayam**, risiko kesehatan manusia, hingga kerusakan ekologi pada satwa liar.
- b) **Sentimen dominan** berupa *fear*, *anger*, dan *sadness*, dengan intensitas meningkat signifikan ketika terjadi lonjakan kasus atau berita besar terkait wabah.
- c) **Aktor sentral** seperti media arus utama, aktivis lingkungan, serta akun berpengaruh memperlihatkan peran dominan dalam mengarahkan narasi publik dan memperkuat penyebaran informasi.

Implikasi

Studi ini menegaskan fungsi **media sosial sebagai barometer opini publik sekaligus sistem peringatan dini** dalam konteks krisis kesehatan. Integrasi NLP berbasis LLMs memberikan peluang untuk:

1. Merancang strategi komunikasi krisis yang **proaktif dan adaptif** terhadap perubahan opini publik.
2. Mengidentifikasi secara dini **misinformasi** dan narasi keliru yang berpotensi memperburuk kepanikan sosial.
3. Menyediakan dasar analitis untuk pengambilan kebijakan yang lebih responsif dan berbasis data real-time.

Kesimpulan

Secara menyeluruh, penelitian ini memperlihatkan bahwa penerapan **kecerdasan buatan dalam analisis wacana digital** mampu memberikan wawasan strategis untuk pengelolaan krisis kesehatan global. Media sosial tidak hanya berfungsi sebagai ruang ekspresi masyarakat, tetapi juga sebagai instrumen analitis yang dapat membantu otoritas dalam **memetakan dinamika opini, merumuskan kebijakan responsif, serta meningkatkan efektivitas komunikasi publik** dalam menghadapi ancaman pandemi.

b) **Prediksi kriminalitas** berbasis data.

Informasi Artikel

- a) Judul: Using machine learning to conduct crime linking of residential burglary
- b) Penulis: Eric Halford, Ian Gibson
- c) Tahun: 2025
- d) Jurnal: *International Journal of Law, Crime and Justice*, Vol. 80, 100716
- e) DOI: <https://doi.org/10.1016/j.ijlcj.2024.100716>

Latar Belakang

Prediksi kriminalitas berbasis data kini menjadi salah satu fondasi penting dalam kriminologi modern. Fokus utamanya bukan hanya untuk mengidentifikasi pelaku, tetapi juga untuk mendukung strategi pencegahan kejahatan yang lebih efektif. Pendekatan tradisional seperti *crime linkage analysis* sering terbatas dalam mengolah data kompleks dan beragam. Kondisi ini menimbulkan kebutuhan terhadap metode analitik canggih yang mampu membaca pola secara mendalam. Kehadiran **machine learning (ML)**, khususnya algoritma *Random Forest*, menawarkan peluang untuk menghasilkan analisis yang lebih presisi dalam memetakan keterkaitan kasus, khususnya pada tindak pencurian rumah tinggal (*residential burglary*). Dengan kemampuan mengolah variabel multi-dimensi, pendekatan ini memperluas cakrawala pemahaman atas perilaku kriminal.

Metodologi

Penelitian menggunakan **200 pasang kasus pencurian rumah di Inggris** dengan total 67 variabel yang dianalisis. Variabel mencakup aspek **spasial** (jarak antar-kejahatan), **temporal** (hari dan waktu kejadian), **modus operandi** (cara masuk), hingga **jenis properti yang dicuri**. Algoritma *Random Forest* dipilih karena kemampuannya dalam mengatasi **overfitting**, menyeimbangkan data yang tidak seimbang, dan membaca hubungan non-linear antar variabel. Evaluasi model dilakukan dengan berbagai metrik, antara lain akurasi, F1-score, ROC-AUC, dan *Matthews Correlation Coefficient* (MCC). Hasil penelitian menunjukkan akurasi prediksi mendekati **90%**, menandakan performa yang sangat andal. Lebih lanjut, teknik *feature importance* dan *permutation importance* digunakan untuk mengidentifikasi faktor-faktor kunci dalam prediksi keterkaitan kasus.

Temuan Utama

- a) **Jarak antar-kejahatan yang pendek** merupakan indikator paling konsisten dalam menghubungkan kasus.
- b) **Faktor temporal**, terutama hari Senin, Jumat, dan rentang waktu 20.00–00.00, terbukti signifikan.
- c) **Karakteristik rumah**, seperti tipe *detached* dan *new build homes*, muncul sebagai faktor prediktif dominan.

- d) **Algoritma *Random Forest*** menghasilkan tingkat akurasi tinggi sekaligus menyajikan *decision tree* yang mudah dipahami, sehingga mendukung penggunaan praktis oleh aparat penegak hukum.

Implikasi

Hasil penelitian ini memiliki dampak luas dalam konteks praktis, antara lain:

- a) Mempercepat proses identifikasi **pelaku berulang** berdasarkan pola perilaku kriminal.
- b) Memungkinkan pemetaan **hotspot kejahatan** yang lebih tepat melalui analisis spasial-temporal.
- c) Menjadi acuan dalam merancang strategi pencegahan, seperti **patroli adaptif** sesuai pola risiko wilayah dan waktu.

Kesimpulan

Studi ini menegaskan bahwa penerapan **machine learning** pada prediksi kriminalitas berbasis data mampu meningkatkan kualitas analisis sekaligus memperkaya teori dan praktik kriminologi. Walaupun ruang lingkup penelitian masih terbatas pada pencurian rumah, pendekatan ini membuka peluang untuk diaplikasikan pada jenis kejahatan lain. Dengan demikian, hasilnya tidak hanya relevan bagi penegakan hukum berbasis teknologi cerdas, tetapi juga mendukung pengambilan kebijakan kriminal yang berbasis bukti (*evidence-based policy*),

memperkuat upaya pencegahan, dan meningkatkan efektivitas sistem peradilan pidana.

D. Teknologi

- a) **Kendaraan otonom** dengan DL dan RL.

Informasi Artikel

- a) **Judul:** Concurrent shape and controller optimization of autonomous underwater vehicles for path following tasks using evolutionary algorithm-driven reinforcement learning
- b) **Penulis:** Cagatay Sariman, Ahmed Hallawa, Anke Schmeink
- c) **Tahun:** 2025
- d) **Jurnal:** *Ocean Engineering*, Vol. 336, 121841
- e) **DOI:** <https://doi.org/10.1016/j.oceaneng.2025.121841>

Latar Belakang

Autonomous Underwater Vehicles (AUVs) semakin diakui sebagai teknologi strategis untuk eksplorasi laut, pemetaan dasar laut, serta misi penyelamatan dan militer. Kendati demikian, pengembangannya menghadapi tantangan besar, terutama dalam hal **dinamika fluida yang kompleks** dan **ketidakpastian kondisi bawah laut**. Faktor-faktor tersebut membuat penciptaan kontroler yang **robust** dan **stabil** menjadi persoalan kritis. Selain aspek

kontrol, penelitian menyoroti bahwa **morfologi fisik AUV** sangat menentukan efisiensi energi, manuverabilitas, serta daya tahan operasional. Oleh sebab itu, integrasi *deep learning (DL)* dan *reinforcement learning (RL)* diusulkan sebagai pendekatan mutakhir untuk mengatasi keterbatasan metode tradisional. Kerangka ini tidak hanya meningkatkan kemampuan adaptasi, tetapi juga memastikan optimasi menyeluruh antara desain fisik dan strategi navigasi.

Metodologi

Penelitian ini menggunakan kerangka **UR-EARL (Evolutionary Algorithm-driven Reinforcement Learning)**. Morfologi AUV dioptimalkan melalui **Lindenmayer-System-based evolutionary algorithm** yang memungkinkan eksplorasi bentuk inovatif dengan efisiensi tinggi. Sementara itu, sistem kontrol dilatih menggunakan algoritma *Twin Delayed Deep Deterministic Policy Gradient (TD3)*, yang dikenal mampu mengurangi bias estimasi dan meningkatkan stabilitas pembelajaran. Proses ini dilakukan secara **ko-optimisasi**, yakni menggabungkan evolusi desain dan kontrol dalam satu siklus adaptif. Untuk meningkatkan efektivitas, strategi **step-by-step optimization** digunakan, memecah masalah besar menjadi beberapa tahap pembelajaran. Simulasi dilaksanakan dalam lingkungan *OpenAI Gym* dengan arus

non-uniform yang dinamis, sehingga meniru kondisi nyata laut dalam secara lebih representatif.

Temuan Utama

- a) Ko-optimalisasi meningkatkan performa AUV, menghasilkan desain **kompak, padat, dan hemat energi**.
- b) Algoritma TD3 terbukti lebih unggul dibanding pendekatan RL konvensional, karena stabil dan mampu beradaptasi secara progresif.
- c) AUV mampu menavigasi jalur 3D secara presisi, meski berada dalam kondisi arus turbulen.
- d) Bentuk optimal menyerupai ikan dengan rasio panjang-lebar yang mendukung efisiensi hidrodinamis, stabilitas, dan kelincahan.

Implikasi

Studi ini memiliki implikasi luas dalam pengembangan **kendaraan otonom berbasis kecerdasan buatan**.

Aplikasi praktis meliputi:

- a) **Inspeksi infrastruktur energi bawah laut**, seperti pipa dan kabel komunikasi.
- b) **Eksplorasi laut dalam**, untuk pemetaan sumber daya mineral dan energi.
- c) **Pemantauan ekosistem laut**, mendukung konservasi biodiversitas dan mitigasi perubahan iklim.
- d) **Operasi penyelamatan dan keamanan maritim**, termasuk patroli di wilayah strategis.

Dengan demikian, pendekatan ini memperkuat relevansi DL dan RL dalam mendukung kebutuhan nyata industri maritim dan keamanan global.

Kesimpulan

Penelitian ini menunjukkan bahwa **integrasi DL dan RL melalui ko-optimalisasi morfologi dan kontrol** menghasilkan AUV yang lebih adaptif, stabil, serta efisien. Pendekatan ini membuka jalan baru bagi desain kendaraan otonom yang mampu menghadapi tantangan lingkungan ekstrem. Selain relevan di ranah maritim, kerangka ini dapat diperluas ke kendaraan otonom darat maupun udara. Oleh karena itu, DL dan RL dapat dipandang sebagai **fondasi utama bagi evolusi teknologi transportasi otonom lintas domain**.

b) **Penerjemahan mesin** dengan AI

Informasi Artikel

a) **Judul:** Translating the machine; An assessment of clinician understanding of ophthalmological artificial intelligence outputs

b) **Penulis:** Oskar Wysocki, Sammie Mak, Hannah Frost, Donna M. Graham, Dónal Landers, Tariq Aslam

c) **Tahun:** 2025

d) **Jurnal:** International Journal of Medical Informatics, Vol. 201, 105958

e) **DOI:** <https://doi.org/10.1016/j.ijmedinf.2025.105958>

Latar Belakang

Penerjemahan mesin dengan AI telah mengalami perkembangan pesat sebagai jawaban atas kebutuhan komunikasi lintas bahasa di era globalisasi. Teknologi ini bukan sekadar alat bantu, melainkan katalis yang mempercepat interaksi pengetahuan, ekonomi, dan sosial. Pendekatan berbasis *machine learning* dan *deep learning* memungkinkan sistem untuk memahami konteks, struktur sintaksis, dan makna semantik secara lebih mendalam dibandingkan metode tradisional berbasis aturan. Relevansi teknologi ini kian menonjol pada sektor akademik, bisnis, hingga diplomasi, di mana kecepatan serta akurasi penerjemahan berkontribusi pada strategi komunikasi global.

Metodologi

Model utama yang digunakan adalah *Neural Machine Translation (NMT)*. Prosesnya dimulai dengan mengubah bahasa sumber menjadi representasi vektor menggunakan jaringan saraf berlapis, kemudian menghasilkan teks bahasa target dengan mempertimbangkan keseluruhan konteks. Untuk meningkatkan performa, dilakukan *fine-tuning* terhadap korpus khusus, seperti dokumen medis, hukum, maupun teknis. Evaluasi kualitas melibatkan metrik otomatis seperti BLEU (*Bilingual Evaluation Understudy*), METEOR, serta uji berbasis manusia seperti *fluency test* dan *adequacy test*. Pendekatan kombinatorik ini memberikan

gambaran objektif sekaligus subjektif mengenai kualitas terjemahan.

Temuan Utama

- a)** *Neural Machine Translation* secara konsisten menghasilkan terjemahan yang lebih kohesif dan alami dibandingkan pendekatan statistik.
- b)** Kesulitan tetap muncul pada idiom, istilah budaya, dan nuansa emosional yang kompleks karena keterbatasan padanan semantik.
- c)** Integrasi *attention mechanism* meningkatkan akurasi sekaligus transparansi model, sehingga pengguna dapat memahami fokus algoritma dalam menghasilkan terjemahan.

Implikasi

Kontribusi penerjemahan mesin dengan AI sangat luas dan strategis. Di bidang akademik, teknologi ini memperluas akses terhadap literatur multibahasa dan memperkuat kolaborasi riset global. Pada ranah bisnis, hambatan komunikasi lintas negara dapat ditekan, mempercepat interaksi perdagangan dan investasi. Dalam konteks sosial-politik, AI mendukung diplomasi serta dialog antarbudaya yang lebih inklusif. Implikasi penting meliputi:

- a) Peningkatan kolaborasi riset internasional dan publikasi multibahasa.

- b) Optimalisasi interaksi ekonomi global dan tata kelola pemerintahan lintas negara.
- c) Pemberdayaan masyarakat dengan keterbatasan bahasa melalui akses informasi yang lebih luas.

Kesimpulan

Secara keseluruhan, penerjemahan mesin berbasis AI adalah **inovasi transformatif** yang mendukung percepatan komunikasi global dan transfer pengetahuan. Walaupun tantangan masih ada, khususnya pada penerjemahan idiom dan konteks budaya, kemajuan teknologi terus mendekatkan hasil terjemahan ke kualitas manusia. Untuk memastikan akurasi dan keadilan bahasa, evaluasi kritis dan pengawasan etis perlu dikedepankan. Dengan pendekatan tersebut, penerjemahan mesin dapat menjadi jembatan yang kokoh dalam membangun interaksi global yang inklusif, berkelanjutan, dan berorientasi pada kolaborasi.

- d) **Optimisasi jaringan** menggunakan ML.

Informasi Artikel

a) Judul: Optimizing solar photovoltaic power forecasting via multi-architecture machine learning framework with multiple hyperparameter optimization techniques

b) Penulis: Muhammad Faizan Tahir, Anthony Tzes, Tarek H.M. El-Fouly, Mohamed Shawky El Moursi, Dongliang Xiao, Nauman Ali Larik

c) Tahun: 2025

d) Jurnal: *Energy Strategy Reviews*, Vol. 61, 101859

e) DOI: <https://doi.org/10.1016/j.esr.2025.101859>

Latar Belakang

Optimisasi jaringan dengan pendekatan *machine learning* (ML) menjadi solusi strategis dalam menghadapi kompleksitas sistem energi modern. Khususnya pada jaringan berbasis *photovoltaic* (PV), prediksi daya yang akurat sangat penting karena produksi listrik bergantung pada variabilitas cuaca, intensitas radiasi matahari, serta kondisi lingkungan yang dinamis. Tanpa model prediksi yang handal, jaringan berisiko mengalami ketidakstabilan, fluktuasi tegangan, hingga inefisiensi distribusi energi. Penelitian ini berangkat dari kebutuhan mendesak untuk mengembangkan model prediksi berbasis ML yang lebih adaptif, presisi, dan mampu beroperasi optimal di iklim ekstrem, seperti di Uni Emirat Arab.

Metodologi

Studi ini menyusun kerangka kerja multi-arsitektur dengan mengevaluasi beberapa algoritma ML, antara lain **Artificial Neural Networks (ANN)**, **Support Vector Machines (SVM)**, **Regression Trees (RT)**, **Gaussian Process Regression (GPR)**, serta **Linear Regression (LR)**. Setiap algoritma diuji dalam berbagai konfigurasi untuk memetakan kinerja berdasarkan akurasi, efisiensi

pelatihan, dan kecepatan prediksi. Untuk meningkatkan performa, penelitian ini menerapkan teknik optimisasi hiperparameter, seperti *Tree-structured Parzen Estimator (TPE)*, *Bayesian Optimization (BO)*, *Random Search with Successive Halving (RSHA)*, dan *Grey Wolf Optimizer (GWO)*. Evaluasi komprehensif dilakukan dengan metrik **Root Mean Square Error (RMSE)**, **Mean Absolute Error (MAE)**, serta analisis trade-off antara akurasi, kompleksitas model, dan waktu komputasi.

Temuan Utama

- a) Konfigurasi **wide ANN** terbukti paling unggul dengan menghasilkan RMSE terendah, menandakan kemampuannya menangkap pola non-linear yang kompleks.
- b) Teknik TPE menjadi metode optimisasi hiperparameter yang paling efektif, mengungguli BO, RSHA, maupun GWO dalam hal peningkatan akurasi.
- c) Performa model sangat dipengaruhi oleh kombinasi arsitektur dan strategi optimisasi, menegaskan bahwa pemilihan algoritma saja tidak cukup tanpa tuning parameter yang tepat.
- d) Studi ini memperkaya literatur dengan membuktikan efektivitas evaluasi lintas-model dalam kondisi iklim ekstrem, yang sebelumnya masih jarang dikaji.

Implikasi

Penelitian ini memiliki implikasi luas dalam konteks pengelolaan energi modern. Pertama, penerapan kerangka kerja ini dapat meningkatkan **efisiensi operasional jaringan listrik**, khususnya dalam integrasi energi terbarukan skala besar. Kedua, hasil penelitian mendukung kebijakan transisi energi dengan mengurangi ketergantungan pada sumber daya fosil. Ketiga, model ini membuka peluang adaptasi pada sektor lain, seperti manajemen beban listrik perkotaan dan integrasi kendaraan listrik, yang juga membutuhkan prediksi daya akurat. Dengan demikian, kontribusinya tidak hanya bersifat teknis, tetapi juga strategis dalam mendukung keberlanjutan energi global.

Kesimpulan

Secara komprehensif, penelitian ini membuktikan bahwa kombinasi multi-arsitektur ML dengan optimisasi hiperparameter mampu memberikan peningkatan signifikan pada akurasi prediksi daya PV. Temuan ini menegaskan bahwa **optimisasi jaringan energi berbasis ML** adalah pilar penting dalam membangun sistem energi yang lebih tangguh, berkelanjutan, dan adaptif terhadap tantangan masa depan.

Soal Uraian

1. Jelaskan apa yang dimaksud dengan metodologi penelitian data sains dan mengapa integritas penting dalam penerapannya.
2. Uraikan tahapan utama dalam *data science lifecycle*.
3. Apa perbedaan mendasar antara metodologi penelitian AI, ML, dan DL?
4. Gambarkan dan jelaskan alur framework penelitian AI pada kasus nyata.
5. Sebutkan tahapan penting dalam ML pipeline dan contoh aplikasinya.
6. Bagaimana prinsip kerja *neural network* dalam DL, dan mengapa dibutuhkan banyak data serta komputasi besar?
7. Buatlah tabel perbandingan AI, ML, dan DL dari sisi lingkup, kompleksitas, kebutuhan data, dan contoh penerapan.
8. Berikan contoh studi kasus penerapan data sains dalam bidang kesehatan dan framework penelitian yang tepat.
9. Mengapa *data cleaning* dan *data preparation* krusial dalam data sains? Jelaskan dengan contoh.
10. Apa tantangan utama dalam mengimplementasikan metodologi data sains di industri, dan bagaimana solusinya?

Tugas Kelompok

Judul Tugas:

Analisis dan Penerapan Framework Penelitian Data Sains (AI, ML, DL) pada Studi Kasus Nyata

Deskripsi Tugas:

Setiap kelompok memilih satu studi kasus nyata (bisnis, kesehatan, sosial, atau teknologi), kemudian:

1. Mendeskripsikan masalah penelitian.
2. Menentukan metodologi (AI, ML, atau DL).
3. Menyusun framework penelitian: identifikasi masalah, pengumpulan data, pembersihan data, pemilihan metode, pelatihan/pengujian model, evaluasi hasil, implementasi solusi.
4. Membuat ilustrasi visual framework (flowchart/diagram/infografis).
5. Membandingkan metodologi terpilih dengan alternatif lain.

Output:

1. Laporan kelompok (5-10 halaman) berisi teori, analisis kasus, framework, ilustrasi, perbandingan.
2. Presentasi singkat (10-15 menit).

Machine Learning

A. CPL

Mengenal konsep *machine learning*.

B. CPMK

Mampu menjelaskan konsep *machine learning*.

C. Sub-CPMK

Menyebutkan tahapan *machine learning*.

Machine learning merupakan bidang yang berkembang sangat pesat dalam ranah **kecerdasan buatan** (*artificial intelligence*). Teknologi ini menjadi fondasi bagi berbagai aplikasi modern, mulai dari rekomendasi produk di e-commerce, sistem pengenalan wajah pada perangkat keamanan, hingga prediksi cuaca dan analisis kesehatan. Pemahaman mendalam tentang *machine learning* penting bagi mahasiswa teknologi informasi dan siapa pun yang ingin memahami bagaimana data diubah menjadi **pengetahuan yang berguna**. Pembahasan ini disejajarkan dengan pemahaman dari literatur *Principles of Data Science* sehingga mahasiswa dapat melihat kaitan langsung antara pembelajaran mesin dengan prinsip pengolahan data dalam skala luas.

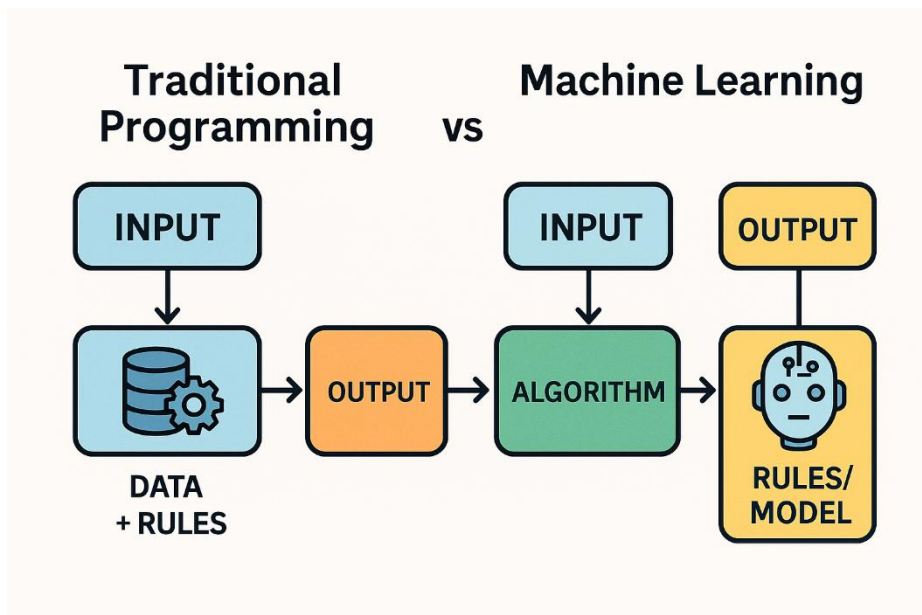
Definisi dan Konsep Dasar

Machine learning adalah **cabang dari kecerdasan buatan** yang memungkinkan komputer **belajar dari data** tanpa diprogram

secara eksplisit. Berbeda dengan pemrograman tradisional, di mana aturan ditentukan langsung oleh manusia, *machine learning* memungkinkan komputer menemukan **pola tersembunyi** dari data untuk membuat prediksi atau keputusan.

Proses ini memanfaatkan pendekatan matematis dan algoritmik yang terus disempurnakan. Dalam *Principles of Data Science*, inti *machine learning* adalah bagaimana data yang beragam diolah menjadi representasi sederhana yang dapat digunakan dalam prediksi dan pengambilan keputusan. Karakteristik utama:

1. **Berbasis data** kualitas hasil bergantung pada data.
2. **Adaptif** model dapat diperbarui dengan data baru.
3. **Prediktif** digunakan untuk memperkirakan hasil dari pola historis.
4. **Otomatisasi** proses dapat berjalan dengan intervensi minimal.



Gambar 34. *Traditional Programming vs Machine Learning.*

Gambar diatas menggambarkan perbedaan mendasar antara **Traditional Programming** dan **Machine Learning**, dua pendekatan yang menjadi fondasi penting dalam teknologi modern. Pada sisi **Traditional Programming**, alur dimulai dengan **Input** berupa data dan aturan (*rules*). Programmer secara eksplisit merancang aturan logis untuk mengolah data sehingga menghasilkan **Output** yang deterministik. Dengan demikian, logika sepenuhnya ditentukan oleh manusia, sedangkan sistem hanya mengeksekusi sesuai instruksi.

Sebaliknya, pada **Machine Learning**, alurnya berbalik. Sistem menerima **Input** berupa data serta **Output** (label/target) yang sudah diketahui. Data tersebut diproses oleh **Algorithm** sehingga terbentuk **Model** atau *rules* yang dihasilkan secara otomatis. Model inilah yang kemudian digunakan untuk memprediksi atau menghasilkan **Output** baru tanpa perlu penulisan aturan manual. Dengan kata lain, **Machine Learning** memungkinkan sistem *belajar* dari pola data dan beradaptasi dalam pengambilan keputusan.

Insight penting *Traditional Programming* paling sesuai untuk masalah dengan aturan yang jelas dan konsisten, sedangkan *Machine Learning* unggul ketika aturan sulit diformulasikan tetapi tersedia banyak data.

Gambar ini menegaskan bahwa **Machine Learning menghadirkan fleksibilitas serta adaptabilitas**, menjadikannya kunci dalam era *big data* dan kecerdasan buatan. Pemahaman visual ini memperlihatkan pergeseran peran manusia: dari menuliskan aturan secara detail menuju penyediaan data berkualitas serta pemilihan algoritma yang tepat.

Tahap Pengumpulan Data

Tahap awal siklus *machine learning* adalah **pengumpulan data**. Data dapat berupa teks, angka, gambar, suara, atau gabungan. Sumber data meliputi sistem informasi organisasi, sensor IoT (*Internet of Things*), media sosial, atau repositori terbuka. Literatur *Principles of Data Science* menegaskan pentingnya proses pengumpulan yang tepat agar model tidak bias. Kriteria data:

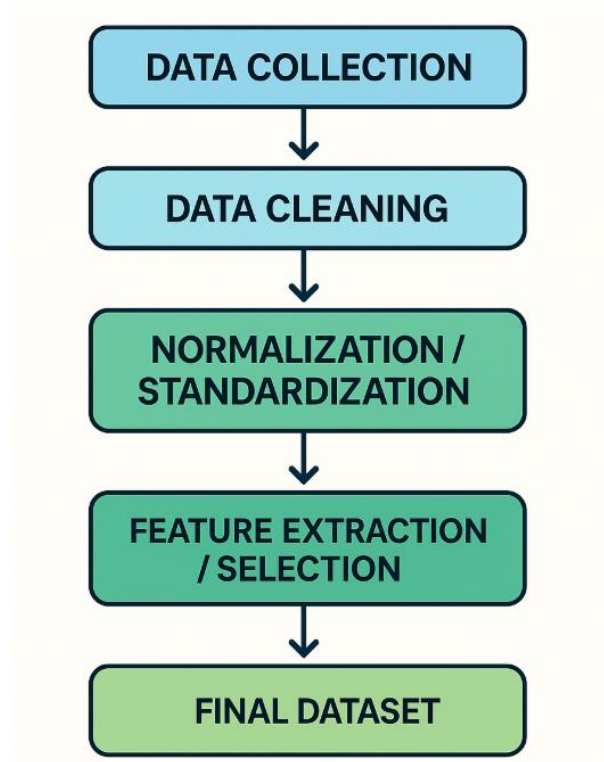
- a) **Relevansi** sesuai tujuan.
- b) **Kualitas** akurat dan bebas kesalahan.
- c) **Kuantitas** cukup banyak untuk melatih model.
- d) **Keberagaman** variasi data untuk meningkatkan generalisasi.

Tahap Praproses Data

Data mentah perlu diproses agar siap digunakan. Tahap ini meliputi:

1. **Pembersihan** menghapus duplikat, memperbaiki nilai hilang, dan mengatasi inkonsistensi.
2. **Transformasi** menormalkan angka atau mengubah teks menjadi vektor.
3. **Reduksi** menyaring atribut tidak relevan.
4. **Pembagian** memisahkan menjadi **training set**, **validation set**, dan **test set**.

Menurut *Principles of Data Science*, praproses yang buruk dapat menyebabkan kesalahan interpretasi sehingga pola palsu dianggap benar oleh algoritma.



Gambar 35. Tahapan Pengumpulan & Praproses Data

Pengumpulan dan praproses data yang disebut *data preprocessing pipeline*. Tahap pertama adalah **data collection**, yakni menghimpun data dari sumber beragam seperti sensor, database, berkas digital, maupun *api*. Data mentah hasil koleksi ini kerap belum siap dipakai karena berisi ketidakteraturan dan inkonsistensi.

Langkah berikutnya, **Data Cleaning**, berfokus pada penghapusan duplikasi, penanganan *missing values*, dan deteksi *outliers* yang berpotensi merusak integritas analisis. Setelah data bersih, dilakukan **Normalization/Standardization**, yaitu proses penyamaan skala antar variabel agar setiap atribut dapat diperlakukan secara adil tanpa dominasi akibat perbedaan satuan atau rentang nilai.

Tahap selanjutnya adalah **Feature Extraction/Selection**. Di sini, fitur-fitur paling relevan dipilih atau diturunkan dari data mentah untuk memperkuat representasi informasi. Keberhasilan model *Machine Learning* sangat dipengaruhi kualitas fitur yang digunakan. Akhir dari pipeline ini menghasilkan **Final Dataset**, kumpulan data siap pakai untuk melatih dan menguji model.

Insight penting kualitas model *Machine Learning* bergantung pada kualitas data yang melandasinya. Oleh sebab itu, **proporsi waktu terbesar dalam proyek berbasis data biasanya tercurah pada tahap preprocessing**.

A. Pemilihan Algoritma

Pemilihan algoritma menentukan pendekatan model. Tiga jenis utama adalah:

1. Supervised Learning

Menggunakan data berlabel, misalnya pelanggan aktif/tidak aktif. Algoritma **Linear Regression, Logistic Regression, Decision Tree, Random Forest, SVM**.

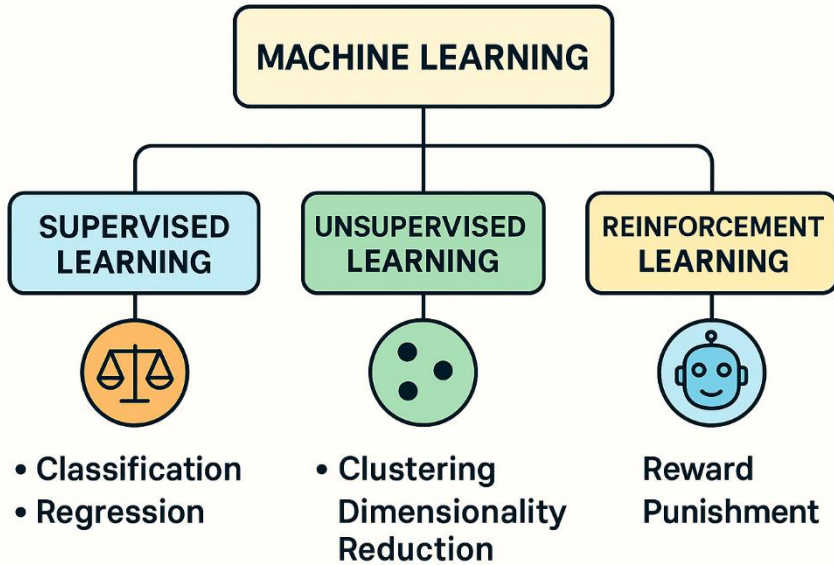
2. Unsupervised Learning

Data tidak berlabel. Tujuannya menemukan pola. Algoritma: **K-Means, Hierarchical Clustering, PCA**.

3. Reinforcement Learning

Agen belajar dari interaksi dengan lingkungan melalui **reward** dan **penalty**. Digunakan dalam robotika, permainan (*game*), dan sistem rekomendasi dinamis.

Dalam *Principles of Data Science*, pemilihan algoritma disesuaikan dengan jenis masalah, sifat data, dan tujuan akhir.



Gambar 36. Jenis-Jenis Algoritma *Machine Learning*

Tiga kategori utama dalam **algoritma Machine Learning**, yaitu **Supervised Learning**, **Unsupervised Learning**, dan **Reinforcement Learning**. Masing-masing kategori memiliki ciri khas, metode pembelajaran, serta contoh penerapan yang berbeda.

Pada **Supervised Learning**, sistem dilatih menggunakan data berlabel. Setiap input disertai output yang diketahui, sehingga algoritma mampu belajar membuat prediksi akurat. Contoh nyata: *classification* untuk memilah email *spam* dan *not spam*, serta *regression* untuk memprediksi harga rumah berdasarkan fitur tertentu.

Sebaliknya, **Unsupervised Learning** beroperasi tanpa label. Algoritma mencari pola tersembunyi atau struktur alami dalam data.

Penerapannya meliputi *clustering*—misalnya mengelompokkan pelanggan berdasarkan perilaku belanja—dan *dimensionality reduction* untuk menyederhanakan data kompleks agar lebih mudah dianalisis.

Adapun **Reinforcement Learning** didasarkan pada mekanisme *reward* dan *punishment* hasil interaksi dengan lingkungan. Sistem belajar menentukan strategi optimal melalui percobaan berulang. Contohnya adalah pelatihan robot agar mampu berjalan stabil atau agen AI yang menguasai permainan catur dan video game.

Insight penting setiap jenis algoritma memiliki keunggulan tersendiri. *Supervised Learning* unggul saat data historis berlabel tersedia, *Unsupervised Learning* efektif untuk eksplorasi pola tersembunyi, sementara *Reinforcement Learning* sangat kuat dalam pengambilan keputusan dinamis.

B. Proses Pelatihan Model

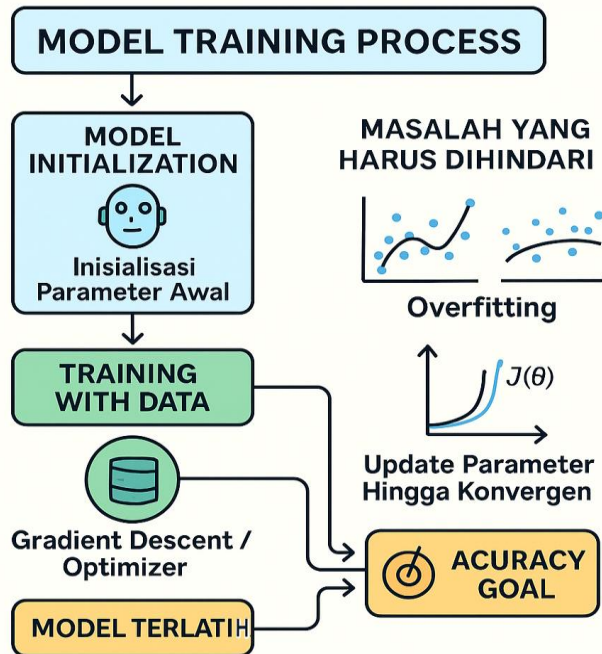
Pelatihan dilakukan dengan memberikan data latih agar model belajar pola. Tahapan:

1. **Inisialisasi model** dengan parameter awal.
2. **Optimisasi** menggunakan metode seperti *gradient descent*.
3. **Iterasi** hingga akurasi tercapai.

Masalah yang harus dihindari:

- a) **Overfitting**: model terlalu sesuai dengan data latih.
- b) **Underfitting**: model gagal menangkap pola.

Principles of Data Science menyarankan penggunaan data validasi dan uji untuk mengontrol kedua masalah tersebut.



Gambar 37. Pelatihan Model

Proses pelatihan model *Machine Learning*. Tahap pertama adalah **Model Initialization**, yaitu menginisialisasi parameter awal. Model pada tahap ini belum memiliki kemampuan prediksi yang baik dan masih berupa kerangka dengan nilai acak.

Tahap berikutnya adalah **Training with Data**, di mana model diberikan data latih agar dapat mempelajari pola yang terkandung di dalamnya. Proses ini kemudian dilanjutkan dengan **Optimization** menggunakan metode seperti *gradient descent*. Pada tahap optimisasi, parameter model diperbarui secara iteratif untuk meminimalkan fungsi kerugian (*loss function*).

Selama pelatihan, dilakukan **iterasi berulang** hingga model mencapai titik konvergensi dan memenuhi target akurasi. Hasil akhir

dari rangkaian ini adalah **model terlatih** yang siap digunakan untuk memprediksi data baru.

Namun, ada dua masalah besar yang perlu dihindari. Pertama, **overfitting**, yaitu kondisi ketika model terlalu menyesuaikan diri dengan data latih sehingga gagal melakukan generalisasi pada data baru. Kedua, **underfitting**, yaitu saat model terlalu sederhana sehingga tidak mampu menangkap pola penting dalam data. *Principles of Data Science* menyarankan penggunaan data validasi dan uji untuk memantau serta mengontrol kedua masalah tersebut.

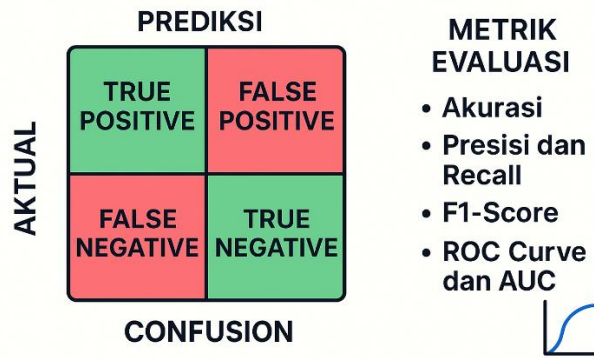
Insight penting keberhasilan pelatihan model tidak hanya bergantung pada algoritma yang digunakan, tetapi juga pada **strategi pengendalian kualitas data serta evaluasi berlapis**. Dengan menjaga keseimbangan, model dapat mencapai kinerja optimal sekaligus tetap andal saat dihadapkan pada data nyata.

C. Evaluasi Model

Model diuji dengan data uji. Metrik evaluasi:

- a) **Akurasi**
- b) **Presisi dan Recall**
- c) **F1-Score**
- d) **Confusion Matrix**
- e) **ROC Curve dan AUC**

Literatur *Principles of Data Science* menekankan bahwa metrik dipilih sesuai konteks dan tujuan praktis.



Gambar 38. Metrik evaluasi

Confusion matrix dalam mengevaluasi performa model klasifikasi. Matriks ini menampilkan empat komponen inti: **true positive (tp)**, **true negative (tn)**, **false positive (fp)**, dan **false negative (fn)**. Keempat nilai inilah yang menjadi fondasi bagi perhitungan berbagai **metrik evaluasi**. Beberapa metrik utama yang diturunkan dari confusion matrix meliputi:

1. **Akurasi**

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Mengukur rasio prediksi benar terhadap total prediksi.

2. **Presisi**

$$Precision = \frac{TP}{TP + FP}$$

Tingkat ketepatan prediksi positif, dihitung dengan $TP / (TP + FP)$.

3. **Recall**

$$Recall = \frac{TP}{TP + FN}$$

Kemampuan model menemukan seluruh kasus positif, dihitung dengan $TP / (TP + FN)$.

4. **F1-score**

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Rata-rata antara presisi dan recall, sangat berguna saat terjadi ketidakseimbangan data.

5. **Roc curve & auc**

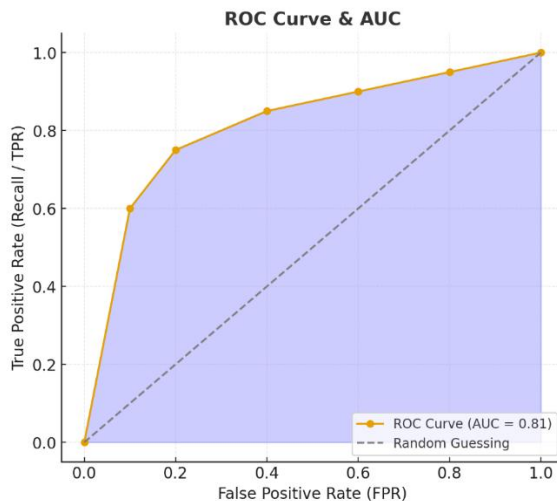
False Positive Rate (FPR)

$$FPR = \frac{FP}{FP + TN}$$

True Positive Rate (TPR atau Recall)

$$TPR = \frac{TP}{TP + FN}$$

Grafik hubungan *true positive rate* dan *false positive rate*, di mana auc memberikan ukuran kinerja keseluruhan model.



Gambar 39. Roc Curve & Auc

Gambar di atas menampilkan **ROC Curve (Receiver Operating Characteristic Curve)** yang menjadi alat penting untuk mengevaluasi model klasifikasi di berbagai *threshold*. Pada sumbu horizontal terdapat **False Positive Rate (FPR)** yang menunjukkan proporsi kesalahan negatif yang salah diklasifikasikan sebagai positif. Pada sumbu vertikal terdapat **True Positive Rate (TPR)** atau *recall*, yang menekankan seberapa banyak kasus positif berhasil dideteksi. Garis putus-putus abu-abu merepresentasikan baseline *random guessing*, sementara kurva berwarna menampilkan kinerja nyata model.

AUC (Area Under the Curve) adalah indikator numerik yang mengukur luas area di bawah kurva ROC. Skala nilainya antara 0 hingga 1, di mana **semakin mendekati 1 maka performa semakin baik**. Dalam contoh grafik, nilai **AUC $\approx$ 0.81** menunjukkan bahwa model memiliki kemampuan diskriminatif yang cukup kuat, meskipun masih bisa ditingkatkan.

Selain itu, **model harus diuji dengan data uji** agar hasil evaluasi objektif. Literatur *principles of data science* menekankan bahwa pemilihan metrik tidak bersifat universal, melainkan harus **disesuaikan dengan konteks serta tujuan praktis**. Misalnya, pada diagnosis medis, **recall** lebih penting untuk meminimalkan kasus positif yang terlewat, sementara dalam sistem rekomendasi produk, **presisi** menjadi prioritas untuk menjaga relevansi.

Insight penting tidak ada satu metrik yang selalu dominan. **Mengombinasikan beberapa metrik memberikan gambaran menyeluruh** sehingga evaluasi model lebih adil, seimbang, dan relevan dengan kebutuhan nyata.

D. Deployment Model

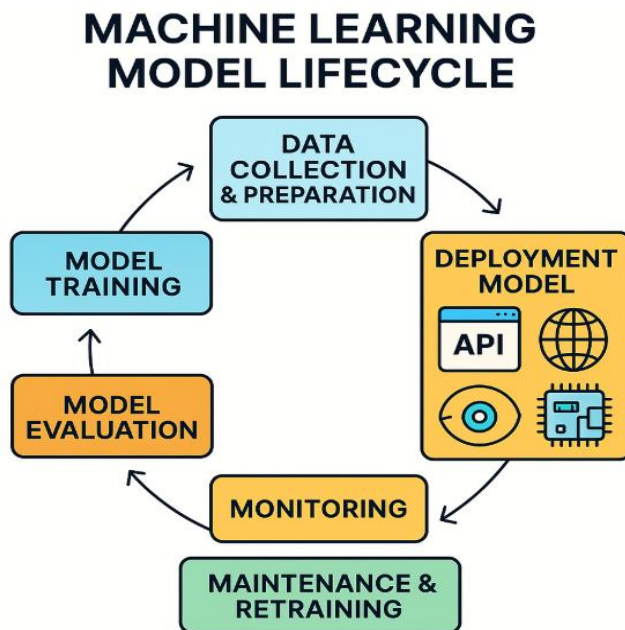
Model yang baik kemudian di-*deploy* pada sistem nyata, misalnya prediksi persediaan gudang. Deployment dapat berupa API, integrasi aplikasi web, atau sistem tersemat. Aspek penting:

- a) **Skalabilitas**
- b) **Keandalan**
- c) **Keamanan**

E. Pemeliharaan dan Pembaruan Model

Model bukan sistem statis. Dunia nyata berubah sehingga model perlu diperbarui. Kegiatan meliputi:

- a) **Monitoring performa**
- b) **Retraining** dengan data baru
- c) **Adaptasi** sesuai kebutuhan



Gambar 40. Siklus Hidup *Machine Learning Model*

Menggambarkan **siklus hidup model Machine Learning** yang menunjukkan tahapan berulang dari pengembangan hingga pemeliharaan. Proses dimulai dengan **Data Collection & Preparation**, yaitu pengumpulan, pembersihan, serta praproses data agar siap dipakai dalam pelatihan. Tahap ini menjadi fondasi kualitas model. Selanjutnya adalah **Model Training**, di mana algoritma *Machine Learning* dilatih menggunakan data latih. Setelah itu dilakukan **Model Evaluation** dengan data validasi dan uji untuk menilai performa model secara objektif. Jika performa memadai, model melangkah ke tahap berikutnya.

Tahap penting berikutnya adalah **Deployment Model**. Pada fase ini, model terlatih diterapkan pada sistem nyata. Contoh penerapan meliputi prediksi persediaan gudang, sistem rekomendasi, atau deteksi anomali. Deployment dapat berupa **API**, integrasi pada **aplikasi web**, atau diimplementasikan pada **embedded system**. Faktor yang harus diperhatikan meliputi **skalabilitas**, **keandalan**, dan **keamanan**. Selain itu, literatur menekankan pentingnya menjaga **etika** dan **privasi data** saat model di-deploy.

Setelah implementasi, dilakukan **Monitoring** untuk mengamati kinerja model dengan data nyata. Jika terjadi *data drift* atau *concept drift*, maka dilakukan **Maintenance & Retraining** agar model tetap adaptif. **Insight penting** Deployment bukan akhir siklus, melainkan awal dari fase produksi. **Keberhasilan sistem AI terletak pada proses berulang yang mencakup pelatihan, evaluasi, deployment, monitoring, dan retraining.**

Tantangan dalam *Machine Learning*

A. Kualitas Data

Keberhasilan model bergantung pada **kualitas data**. Tantangan umum meliputi ketidaklengkapan, ketidakseimbangan kelas, *label noise*, serta *data leakage* yang merusak validitas evaluasi. Di fase produksi, *data drift* dapat menurunkan performa. Strategi perbaikan: audit dan dokumentasi data, *data governance*, *versioning*, serta pemisahan *train-validation-test* yang ketat disertai *cross-validation* saat perlu.

B. Kompleksitas Algoritma

Algoritma modern—dari *gradient boosting* hingga *deep learning*—menuntut **sumber daya komputasi**, penalaan hiperparameter, dan praktik **reproducibility**. Risiko **overfitting/underfitting** harus diatasi melalui *regularization*, *early stopping*, dan rekayasa fitur yang tepat. Penerapan praktik **MLOps** (pipeline otomatis, pelacakan eksperimen, dan *model registry*) membantu menjaga konsistensi kualitas dari eksperimen hingga rilis.

C. Etika dan Privasi

Dimensi **etika dan privasi** menuntut *privacy by design*, *data minimization*, dan teknik seperti *anonymization*, *differential privacy*, atau *federated learning* bila relevan. Lakukan penilaian risiko, pengujian **fairness** untuk mendeteksi bias, sertakan persetujuan yang sah, serta patuhi regulasi perlindungan data. Dokumentasi keputusan penting bagi akuntabilitas publik.

D. Transparansi Model (*Black Box*)

Model berkapasitas tinggi sering dipersepsikan **black box**. Bangun **keterjelasan** melalui *explainability* (mis. *SHAP*, *LIME*), *model cards*, dan log keputusan. Transparansi meningkatkan **kepercayaan** dan mempermudah audit—terutama di domain berisiko tinggi.

Insight penting: Keunggulan *machine learning* lahir dari **metodologi yang ketat, governance data yang kuat**, dan **kepatuhan etika**. Menggabungkan kontrol teknis (validasi, regularisasi, MLOps) dengan kontrol sosial (privasi, fairness, transparansi) memastikan sistem yang **andal, adil, dan dapat dipertanggungjawabkan**.

Soal Uraian

1. Jelaskan perbedaan utama antara traditional programming dan *machine learning* dengan memberikan contoh penerapannya dalam kehidupan sehari-hari.
2. Mengapa tahap pengumpulan dan praproses data sangat penting dalam *machine learning*? Berikan contoh kasus jika tahap ini diabaikan.
3. Sebutkan dan jelaskan tiga jenis utama algoritma machine learning beserta satu contoh penggunaan nyata dari masing-masing jenis.
4. Apa yang dimaksud dengan supervised learning? Berikan ilustrasi sederhana yang mudah dipahami.

5. Mengapa pemilihan algoritma yang tepat berpengaruh pada performa model *machine learning*? Jelaskan dengan contoh aplikatif.
6. Bagaimana cara mengukur performa model machine learning menggunakan *confusion matrix*? Sertakan penjelasan tentang metrik akurasi, presisi, dan recall.
7. Uraikan tahapan *lifecycle machine learning* mulai dari mendefinisikan masalah hingga deployment model.
8. Mengapa model *machine learning* yang sudah di-*deploy* perlu dilakukan maintenance secara berkala? Berikan minimal dua alasan yang logis.
9. Bandingkan kelebihan dan kekurangan antara *unsupervised learning* dan *reinforcement learning* dalam konteks penerapannya.
10. Berikan contoh aplikasi machine learning dalam bidang kesehatan atau transportasi, lalu jelaskan bagaimana tahapan data → model → evaluasi → deployment dapat diterapkan pada kasus tersebut.

Tugas Kelompok

Mahasiswa mampu menerapkan konsep dasar *machine learning* dalam studi kasus nyata berbasis *data sains*, mulai dari pengumpulan data, praproses, pemilihan algoritma, pelatihan, evaluasi, deployment, hingga pemeliharaan model, serta mampu mengaitkan dengan prinsip-prinsip yang terdapat pada literatur *Principles of Data Science* untuk menjaga relevansi akademik.

Instruksi Tugas

1. Pemilihan Studi Kasus

Pilih salah satu studi kasus dari bidang *data sains*, misalnya:

- a. Prediksi harga rumah berdasarkan data properti dengan menggunakan variabel lokasi, ukuran, dan fasilitas.
- b. Analisis sentimen ulasan produk pada e-commerce untuk mengetahui persepsi pelanggan.
- c. Prediksi permintaan transportasi daring berdasarkan waktu, cuaca, dan lokasi geografis.
- d. Klasifikasi penyakit dari data kesehatan sederhana, seperti gejala dan hasil pemeriksaan laboratorium.

2. Pengumpulan dan Praproses Data

- a. Identifikasi sumber data yang relevan (dataset publik, sumber resmi, atau simulasi data).
- b. Lakukan pembersihan data dari duplikasi, nilai kosong, atau data bias.
- c. Transformasi data agar dapat digunakan (misalnya normalisasi angka, encoding kategori, atau konversi teks menjadi vektor).
- d. Jelaskan secara kritis mengapa tahapan ini sangat penting menurut prinsip *data quality* dalam *Principles of Data Science*.

3. Pemilihan Algoritma

- a. Tentukan algoritma yang sesuai dengan permasalahan (contoh: *Linear Regression*, *Decision Tree*, *K-Means*).
- b. Berikan alasan pemilihan algoritma dengan mengaitkan ke karakteristik data dan tujuan studi kasus.

- c. Bandingkan secara singkat dua algoritma berbeda dan jelaskan mengapa salah satunya lebih sesuai.

4. **Pelatihan dan Evaluasi Model**

- a. Latih model menggunakan *training set* dengan pendekatan validasi silang (*cross-validation*) bila memungkinkan.
- b. Uji performa dengan *test set* menggunakan metrik seperti akurasi, presisi, recall, dan F1-score.
- c. Sajikan hasil dalam bentuk tabel atau grafik sederhana dan lakukan interpretasi.

5. **Deployment Sederhana**

- a. Simulasikan penggunaan model pada data baru, misalnya prediksi harga rumah untuk data properti yang belum pernah digunakan.
- b. Jelaskan bagaimana model dapat diintegrasikan dengan sistem informasi atau aplikasi sederhana.
- c. Diskusikan potensi masalah saat deployment seperti keterbatasan komputasi atau keterlambatan prediksi.

6. **Pemeliharaan Model**

- a. Diskusikan strategi pembaruan model ketika data baru tersedia (misalnya retraining berkala).
- b. Jelaskan bagaimana monitoring dilakukan untuk menjaga akurasi, termasuk metrik yang dipantau.
- c. Kaitkan dengan konsep *model lifecycle* dalam *Principles of Data Science*.

Format Laporan

- a. **Pendahuluan:** latar belakang studi kasus, urgensi, dan tujuan.

- b. **Metodologi:** langkah-langkah pengumpulan data, praproses, pemilihan algoritma, pelatihan, evaluasi.
- c. **Hasil dan Pembahasan:** tabel, grafik, serta interpretasi hasil.
- d. **Simulasi Deployment:** implementasi sederhana dan evaluasi tantangan.
- e. **Kesimpulan:** rangkuman temuan dan rekomendasi pengembangan lebih lanjut.

Laporan dikumpulkan dalam bentuk **PDF** dengan panjang 12-15 halaman, menggunakan bahasa akademik sesuai KBBI dan PUEBI.

Supervised Learning

A. CPL

Memahami model *Supervised Learning* sebagai salah satu pendekatan utama dalam *machine learning* yang berfokus pada pemanfaatan data berlabel untuk menghasilkan prediksi atau klasifikasi yang akurat.

B. CPMK

Mahasiswa mampu membandingkan algoritma *machine learning* dalam kategori *Supervised Learning* berdasarkan karakteristik, kelebihan, dan kelemahannya.

C. Sub-CPMK

Mahasiswa mampu menentukan algoritma *Supervised Learning* yang tepat untuk suatu penerapan nyata sesuai dengan karakteristik data dan tujuan sistem.

Supervised Learning merupakan salah satu pendekatan utama dalam **machine learning** yang banyak digunakan dalam berbagai aplikasi modern. Model ini berfokus pada pemanfaatan data berlabel, di mana terdapat pasangan antara **input (X)** dan **output (Y)**. Tujuannya adalah menemukan pola yang mampu memetakan input ke output secara akurat. Dalam praktik, pendekatan ini berperan penting dalam prediksi harga rumah, klasifikasi email spam, analisis kesehatan, serta pengenalan wajah. Pembahasan berikut menguraikan data terstruktur dan tidak terstruktur, definisi *Supervised Learning*, jenis algoritma, tahapan pemodelan, teknik evaluasi, serta

tren algoritma masa depan secara sistematis dengan integrasi teori dan penerapan nyata.

Data Terstruktur dan Tidak Terstruktur

A. Definisi Data Terstruktur

Data terstruktur adalah data berbentuk tabel dengan baris dan kolom. Contoh umum meliputi data transaksi penjualan, nilai ujian, atau catatan demografi pelanggan. Kelebihan data ini adalah mudah diolah karena tiap kolom memiliki tipe data jelas seperti angka, kategori, atau tanggal.

Tabel 8. Penjualan (Data Terstruktur)

Tanggal	No. Faktur	Pelanggan	Produk	Kategori	Qty	Harga Satuan
2025-08-01	INV-1000	Andi	Laptop Pro 14	Laptop	4	18,500,000
2025-08-02	INV-1001	Budi	Mouse Wireless	Aksesoris	5	350,000
2025-08-03	INV-1002	Citra	Keyboard Mechanical	Aksesoris	3	950,000
2025-08-04	INV-1003	Dewi	Monitor 27"	Monitor	5	3,600,000
2025-08-05	INV-1004	Eka	Headset ANC	Audio	5	1,250,000
2025-08-06	INV-1005	Fajar	SSD 1TB	Storage	2	1,450,000
2025-08-07	INV-1006	Gita	Router WiFi 6	Networking	3	980,000
2025-08-08	INV-1007	Hadi	Webcam HD	Aksesoris	3	550,000
2025-08-09	INV-1008	Indra	Printer InkJet	Printer	3	2,200,000
2025-08-10	INV-1009	Joko	Smartwatch S	Wearable	5	3,100,000

B. Definisi Data Tidak Terstruktur

Data tidak terstruktur mencakup teks bebas, gambar, audio, dan video. Data ini memerlukan proses tambahan seperti *natural language processing* untuk teks atau ekstraksi fitur visual untuk gambar. Contohnya, ulasan pelanggan harus diubah menjadi representasi vektor agar dapat diproses oleh algoritma.

Tabel 9. Medis/Chat (Tidak Terstruktur)

ID	Timestamp	Sumber	Penulis	Teks
REC-001	10/08/2025 00:00	EHR Catatan	dr. Maya	Pasien datang dengan keluhan batuk berdahak 3 hari, demam malam. Diberikan amoksisilin 500 mg, anjurkan istirahat & minum hangat.
REC-002	10/08/2025 06:00	EHR Catatan	dr. Surya	Riwayat asma sejak kecil, inhaler habis. Saturasi 96%, wheezing (+). Edukasi teknik inhalasi.
REC-003	10/08/2025 12:00	EHR Catatan	dr. Lestari	Nyeri kepala berdenyut, skala 6/10. Tekanan darah 140/90. Observasi 24 jam & CT bila memburuk.
REC-004	10/08/2025 18:00	EHR Lab/Visit	dr. Bima	Gula darah puasa 135 mg/dL. Sarankan diet rendah gula, kontrol 2 minggu.
REC-005	11/08/2025 00:00	EHR Catatan	dr. Hana	Pasien mengeluh sulit tidur, stress pekerjaan. Rekomendasi sleep hygiene & konseling.
REC-006	11/08/2025 06:00	Chat	Pasien	Chat 10:14 – "Dok, dada terasa sesak pas naik tangga" tanya alergi/riwayat jantung.

ID	Timestamp	Sumber	Penulis	Teks
REC-007	11/08/2025 12:00	Chat	Orang Tua Pasien	Chat 20:05 – "Anak saya demam 38,5 sejak kemarin, makan masih mau." – sarankan kompres & cairan.
REC-008	11/08/2025 18:00	EHR Catatan	dr. Wira	Pasien pasca operasi 2 hari, luka kering, nyeri ringan. Ganti balutan, analgetik bila perlu.
REC-009	12/08/2025 00:00	EHR Catatan	dr. Nia	Keluhan mual sejak pagi, tidak muntah. Tanda vital stabil. Beri antasida, evaluasi gejala.
REC-010	12/08/2025 06:00	Chat	Pasien	Chat 07:32 – "Obat batuknya bikin ngantuk nggak?" – jelaskan efek samping potensial.

C. Signifikansi Perbedaan Data

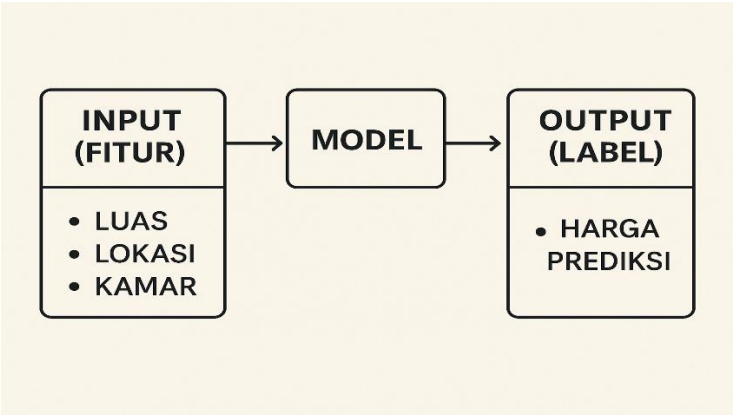
Perbedaan jenis data menentukan strategi pemodelan. Algoritma regresi sederhana lebih sesuai untuk data terstruktur, sedangkan data tidak terstruktur perlu tahap ekstraksi fitur. Hal ini menekankan pentingnya **kualitas dan kesiapan data** sebelum digunakan dalam *Supervised Learning*.

Pengertian Model Supervised Learning

Supervised Learning adalah pendekatan yang menggunakan data berlabel, di mana setiap input memiliki output yang diketahui. Model membangun fungsi $f(x) \rightarrow y$ untuk memprediksi keluaran berdasarkan masukan baru. Contoh nyata adalah sistem prediksi cuaca: suhu, kelembapan, dan tekanan sebagai input, sementara kondisi cuaca (hujan/cerah) sebagai output. fitur *luas (m^2)*, *lokasi*, *jumlah kamar*, dan *harga (prediksi/label)*:

Tabel 10. Contoh Dataset Rumah

No	Luas (m ²)	Lokasi	Kamar	Harga (Juta Rp)
1	60	Jakarta	2	850
2	120	Bandung	3	950
3	90	Surabaya	3	1,100
4	150	Jakarta	4	2,300
5	75	Yogyakarta	2	700



Gambar 41. Fitur dan Label

Karakteristik Utama

A. Data Berlabel dengan Pasangan *Input-Output*

Dalam pembelajaran terawasi (*supervised learning*), salah satu **karakteristik utama** adalah keberadaan **data berlabel**. Setiap data pelatihan memiliki pasangan yang jelas antara *input* (masukan) dan *output* (keluaran). Keterhubungan ini memberikan landasan bagi algoritma untuk **mengidentifikasi pola** dalam data dan melakukan **generalisasi** ketika menghadapi data baru. Tanpa label yang tepat, model akan kesulitan membangun hubungan sebab-akibat yang valid.

B. Tujuan Prediksi Nilai Kontinu maupun Kategori

Pembelajaran terawasi berfokus pada **prediksi yang terarah**. Prediksi ini bisa berbentuk **nilai kontinu** (*regression*) seperti perkiraan harga rumah, atau **kelas kategori** (*classification*) seperti membedakan email *spam* dan *non-spam*. Fleksibilitas ini menjadikan pembelajaran terawasi sangat relevan untuk berbagai bidang seperti ekonomi, kesehatan, hingga teknologi digital, karena mampu **menyediakan solusi praktis** yang berbasis data.

C. Evaluasi Kinerja dengan Membandingkan Prediksi dan Keluaran Aktual

Ciri penting lainnya adalah adanya **mekanisme evaluasi kinerja yang terukur**. Evaluasi dilakukan dengan membandingkan **prediksi model** terhadap **keluaran aktual**, sehingga kualitas model dapat dinilai secara objektif. Beberapa ukuran yang digunakan meliputi *accuracy*, *precision*, *recall*, hingga *mean squared error*. Evaluasi ini penting untuk memastikan bahwa model tidak hanya bekerja baik pada data pelatihan, tetapi juga **tetap konsisten dan andal** saat diuji pada data baru.

Jenis-Jenis Algoritma *Supervised Learning*

A. Regresi

Algoritma **regresi** merupakan salah satu pendekatan penting dalam *supervised learning* yang digunakan ketika **output** atau variabel target berupa **nilai kontinu**. Dalam konteks ini, model bertugas untuk **memprediksi nilai numerik** dengan memanfaatkan pola yang terdeteksi dari data historis. Pemahaman terhadap regresi

menjadi krusial karena banyak permasalahan di dunia nyata—seperti prediksi harga, estimasi permintaan, hingga analisis tren—yang membutuhkan pendekatan ini. Berikut adalah analisis komprehensif mengenai beberapa jenis algoritma regresi yang umum digunakan:

1. *Linear Regression*

Linear Regression adalah metode regresi paling dasar dan mudah dipahami. Prinsip utamanya adalah adanya **hubungan linier** antara variabel input (*independent variables*) dan variabel output (*dependent variable*). Persamaan umum:

$$Y = \beta_0 + \beta_1 X + \epsilon$$

Di mana **Y** adalah variabel output, **X** variabel input, **β_0** intercept, **β_1** koefisien regresi, dan **ϵ** error. Kelebihannya ialah **sederhana dan interpretable**, cocok untuk data yang memang berpola linier. Namun, jika pola data bersifat non-linear, akurasi menurun drastis.

2. *Polynomial Regression*

Ketika pola data tidak sesuai garis lurus, digunakan **Polynomial Regression**. Formula umumnya:

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \dots + \beta_n X^n + \epsilon$$

Dengan menambahkan pangkat variabel input, model mampu menangkap **pola non-linear** yang lebih kompleks. Hal ini menjadikannya fleksibel, tetapi juga menimbulkan risiko **overfitting** jika derajat polinomial terlalu tinggi. Oleh karena itu, penggunaannya perlu dikombinasikan dengan teknik validasi model agar hasil lebih seimbang.

3. *Support Vector Regression (SVR)*

Support Vector Regression (SVR) merupakan turunan dari *Support Vector Machine (SVM)* yang difokuskan untuk regresi. SVR meminimalkan fungsi:

$$\begin{aligned} &\text{minimize } \frac{1}{2} \|w\|^2 \\ &\text{subject to } |y_i - (w \cdot x_i + b)| \leq \epsilon \end{aligned}$$

Konsep utamanya adalah **margin** yang menentukan toleransi kesalahan. SVR unggul dalam menghadapi **data berdimensi tinggi** serta relatif tahan terhadap **outlier**. Akan tetapi, ia membutuhkan **komputasi besar** dan penyetelan parameter yang tepat, sehingga tidak selalu efisien untuk dataset yang sangat besar.

B. Klasifikasi

Klasifikasi merupakan salah satu **teknik penting** dalam pembelajaran mesin yang digunakan ketika **output** berupa **kategori**. Penerapan metode ini sangat luas, mulai dari deteksi spam, diagnosis medis, analisis sentimen, hingga pengenalan wajah. Formula ditulis dalam bentuk teks linear dengan simbol yang lebih konsisten.

1. Logistic Regression

Logistic Regression berfungsi pada **klasifikasi biner**. Model ini mengubah kombinasi linier fitur menjadi probabilitas melalui fungsi sigmoid:

Formula utama:

$$P(y = 1 | x) = \frac{1}{1 + \exp(-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_n x_n))}$$

Dimana:

$P(y = 1 | x)$: probabilitas output y bernilai 1.

β_0 : intercept (konstanta).

$\beta_1, \dots, \beta_n$: koefisien yang menunjukkan pengaruh variabel independen.

$x_1, \dots, x_n$: variabel input (fitur).

Jika nilai probabilitas $P(y = 1 | x) \geq 0.5$, maka diprediksi kelas **1**, sebaliknya jika < 0.5 , diprediksi kelas **0**.

Hubungan dengan logit:

$$\text{logit}(p) = \ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x_1 + \dots + \beta_n x_n$$

Interpretasi probabilitas memungkinkan penentuan kelas dengan ambang batas tertentu, misalnya 0.5.

2. Decision Tree

Decision Tree adalah salah satu algoritma klasifikasi yang bekerja dengan membagi data ke dalam cabang-cabang berdasarkan aturan keputusan (*if-then rule*). Struktur pohon ini terdiri atas **root node** (simpul awal), **internal node** (simpul percabangan), dan **leaf node** (simpul akhir yang berisi label kelas).

Prinsip Dasar

Proses klasifikasi dilakukan dengan memilih atribut yang paling baik dalam memisahkan data. Kriteria pemilihan atribut biasanya menggunakan ukuran seperti:

1. **Information Gain** (berbasis *Entropy*).

Rumus *Entropy*:

$$\text{Entropy}(S) = - \sum_{i=1}^c p_i \log_2(p_i)$$

di mana p_i adalah probabilitas data masuk ke kelas ke- i .

Rumus *Information Gain*:

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

dengan S_v adalah subset dari S berdasarkan nilai atribut A .

2. **Gini Index** (digunakan pada algoritma seperti CART).

Rumus:

$$Gini(S) = 1 - \sum_{i=1}^c p_i^2$$

3. Random Forest

Random Forest adalah algoritma klasifikasi berbasis ansambel yang menggabungkan banyak *Decision Tree* untuk meningkatkan akurasi prediksi dan mengurangi risiko *overfitting*. Algoritma ini bekerja dengan prinsip **bagging** (*bootstrap aggregating*), yaitu membuat beberapa pohon keputusan dari sampel data yang diambil secara acak dengan pengembalian (*bootstrap sample*).

Prinsip Dasar

1. Dari dataset berukuran N , diambil beberapa subset acak dengan metode bootstrap.
2. Untuk setiap subset, dibangun sebuah *Decision Tree* dengan pemilihan fitur secara acak pada setiap node.
3. Hasil klasifikasi ditentukan dengan **voting mayoritas** (untuk klasifikasi) atau **rata-rata** (untuk regresi).

Formula Voting

Jika terdapat M pohon dalam hutan, prediksi kelas $\hat{y}$ diberikan oleh:

$$\hat{y} = \text{mode}\{h_1(x), h_2(x), \dots, h_M(x)\}$$

di mana:

$h_j(x)$ adalah prediksi dari pohon ke- j .

mode berarti kelas yang paling sering dipilih.

4. Naïve Bayes

Naïve Bayes adalah algoritma klasifikasi berbasis teori **Teorema Bayes**, dengan asumsi bahwa setiap fitur dianggap **independen** satu sama lain (*naïve assumption*). Meskipun asumsi ini sering kali tidak sepenuhnya benar dalam dunia nyata, metode ini terbukti **sederhana, cepat, dan efektif**, khususnya untuk dataset berukuran besar seperti teks.

Prinsip Dasar

Teorema Bayes menyatakan:

$$P(C | X) = \frac{P(X | C) P(C)}{P(X)}$$

Dimana:

$P(C | X)$: probabilitas kelas C diberikan data X .

$P(X | C)$: probabilitas kemunculan data X jika kelasnya C .

$P(C)$: probabilitas awal (prior) dari kelas C .

$P(X)$: probabilitas total data X .

Dengan asumsi independensi antar fitur $x_1, x_2, \dots, x_n$:

$$P(C | x_1, x_2, \dots, x_n) \propto P(C) \prod_{i=1}^n P(x_i | C)$$

Sehingga prediksi kelas ditentukan oleh:

$$\hat{C} = \arg \max_C P(C) \prod_{i=1}^n P(x_i | C)$$

5. K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) adalah algoritma klasifikasi berbasis **instance-based learning** atau **lazy learning**, karena tidak membangun model eksplisit saat pelatihan, melainkan menyimpan seluruh data latih. Untuk mengklasifikasikan data baru, algoritma mencari **k tetangga terdekat** berdasarkan ukuran jarak, lalu menentukan kelas mayoritas dari tetangga tersebut.

Prinsip Dasar

1. Tentukan parameter k (jumlah tetangga).
2. Hitung jarak antara data uji x dan seluruh data latih.
3. Pilih k data latih dengan jarak terdekat.
4. Tentukan kelas berdasarkan **voting mayoritas** dari k tetangga.

Formula jarak (Euclidean Distance):

$$d(x, x_i) = \sqrt{\sum_{j=1}^n (x_j - x_{ij})^2}$$

Prediksi kelas ditentukan oleh:

$$\hat{y} = \arg \max_c \sum_{i \in N_k(x)} \mathbf{1}(y_i = c)$$

Dimana:

$N_k(x)$: himpunan k tetangga terdekat dari data x .

$\mathbf{1}(y_i = c)$: fungsi indikator, bernilai 1 jika data i termasuk kelas c .

6. Support Vector Machine (SVM)

Support Vector Machine (SVM) adalah algoritma klasifikasi yang bekerja dengan mencari **hyperplane optimal** yang memisahkan data dari dua kelas dengan margin terbesar. Data yang paling dekat dengan hyperplane disebut **support vectors**, dan titik-titik inilah yang menentukan posisi serta orientasi hyperplane.

Prinsip Dasar

Jika terdapat dataset (x_i, y_i) dengan label $y_i \in \{-1, +1\}$, maka tujuan SVM adalah menemukan hyperplane:

$$f(x) = w^T x + b$$

sehingga:

$$y_i(w^T x_i + b) \geq 1, \forall i$$

Masalah optimisasi yang harus diselesaikan adalah:

$$\min_{w,b} \frac{1}{2} \|w\|^2$$

dengan syarat di atas.

Untuk data yang tidak sepenuhnya linear separable, digunakan

Soft Margin SVM dengan menambahkan variabel slack ξ_i :

$$\min_{w,b,\xi} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \xi_i$$

dengan C sebagai parameter regularisasi.

Selain itu, melalui **Kernel Trick**, SVM dapat memetakan data ke ruang berdimensi lebih tinggi untuk menangani data non-linear. Kernel yang umum digunakan antara lain *linear kernel*, *polynomial kernel*, dan *radial basis function (RBF)*.

C. Pertimbangan Pemilihan

Faktor utama: ukuran dataset, kompleksitas hubungan antar variabel, kebutuhan akurasi, dan efisiensi komputasi. Diskusi kritis diperlukan agar algoritma sesuai dengan masalah nyata.

Tahapan dalam Model *Supervised Learning*

A. Pengumpulan Data

Tahap awal dalam *supervised learning* adalah **pengumpulan data**. Proses ini tidak hanya sekadar mengumpulkan informasi, tetapi juga memastikan data memiliki **relevansi, representasi, dan kualitas** yang memadai. Data dapat bersumber dari **internal** organisasi, seperti catatan transaksi atau sensor, maupun dari **eksternal**, misalnya data publik, survei, maupun *open datasets*. Jika sejak awal data mengandung bias atau ketidakseimbangan, hal ini akan berdampak langsung pada performa model.

B. Praproses Data

Langkah selanjutnya adalah **praproses data**. Kegiatan ini meliputi **pembersihan** dari anomali, duplikasi, atau nilai kosong, **normalisasi** agar setiap fitur berada pada skala yang sebanding, serta **encoding** untuk variabel kategori sehingga dapat dibaca oleh algoritme. Tahap ini krusial karena kualitas praproses menentukan sejauh mana model mampu mengenali pola yang tersembunyi dalam data.

C. Pelatihan Model

Berikutnya adalah **pelatihan model** menggunakan **data latih**. Proses ini memanfaatkan berbagai algoritme pembelajaran mesin dengan teknik optimisasi, salah satunya *gradient descent*. Tujuannya adalah menemukan parameter terbaik yang meminimalkan error. Model yang seimbang harus mampu menggeneralisasi, bukan sekadar menghafal data, sehingga dapat memprediksi kasus baru secara akurat.

D. Evaluasi Model

Tahap evaluasi bertujuan menilai kinerja model menggunakan **data uji**. Evaluasi berfokus pada kemampuan model dalam menghadapi risiko **overfitting** maupun **underfitting**. Beragam metrik dapat digunakan, seperti **akurasi, presisi, recall, dan F1-score**. Dengan evaluasi menyeluruh, peneliti dapat menentukan apakah model siap dilanjutkan atau perlu disesuaikan.

E. Deployment

Tahap terakhir adalah **deployment**, yaitu penerapan model ke dalam **sistem nyata** agar memberikan prediksi secara langsung. Deployment tidak berhenti pada implementasi saja, melainkan juga mencakup **monitoring berkala**, pembaruan, dan penyesuaian model sesuai perubahan dinamika data di lapangan.

Teknik Evaluasi di *Supervised Learning*

A. Regresi

Supervised learning, khususnya pada kasus **regresi**, dibutuhkan metode evaluasi yang sistematis untuk menilai sejauh mana model mampu memprediksi nilai target dengan tepat. Evaluasi yang baik berfungsi menjaga **validitas akademik** model, meningkatkan transparansi, serta memastikan hasil prediksi dapat diinterpretasikan dengan benar dalam konteks praktis. Tiga teknik evaluasi utama yang sering digunakan adalah **mse**, **rmse**, dan **r²**.

Mean Squared Error (MSE)

Mean Squared Error (MSE) menghitung rata-rata dari kuadrat selisih antara nilai aktual dengan nilai prediksi. Rumusnya ditulis:

$$MSE = \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{n}$$

Dimana:

y_i = nilai aktual (observasi ke-i)

ŷ_i = nilai prediksi (estimasi ke-i)

n = jumlah data

MSE digunakan untuk mengukur seberapa jauh nilai prediksi menyimpang dari nilai aktual. Semakin kecil nilai **MSE**, semakin baik model regresi dalam melakukan prediksi.

Root Mean Squared Error (RMSE)

Root Mean Squared Error (RMSE) adalah akar dari Mean Squared Error. Ukuran ini sering digunakan karena hasilnya berada

pada **skala yang sama dengan data aslinya**, sehingga lebih mudah diinterpretasikan. Rumus:

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{n}}$$

Dimana:

y_i = nilai aktual (observasi ke-i)

$\hat{y}_i$ = nilai prediksi (estimasi ke-i)

n = jumlah data

Semakin kecil nilai **RMSE**, semakin baik kualitas prediksi model karena tingkat kesalahan (error) semakin rendah.

Koefisien Determinasi (R^2)

Koefisien Determinasi atau *Coefficient of Determination* (R^2) digunakan untuk mengukur **proporsi variasi nilai aktual (y)** yang dapat dijelaskan oleh **model prediksi ($\hat{y}$)**. Dengan kata lain, R^2 menunjukkan seberapa baik model regresi mampu merepresentasikan hubungan antara variabel independen dan variabel dependen.

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}$$

Nilai **R^2** berada pada rentang **0 hingga 1**:

Jika **$R^2 = 0$** , artinya model sama sekali tidak mampu menjelaskan variasi data.

Jika **R^2 mendekati 1**, artinya model memiliki kemampuan yang sangat baik dalam menjelaskan variasi data.

Semakin tinggi nilai **R²**, semakin besar proporsi variasi data yang dapat dijelaskan oleh model, sehingga kualitas representasi model menjadi lebih baik.

B. Klasifikasi

Evaluasi dalam *supervised learning* khususnya pada **klasifikasi** berfungsi sebagai dasar untuk mengukur seberapa efektif model dalam melakukan prediksi. Setiap metrik tidak hanya memberi angka, tetapi juga sudut pandang berbeda mengenai kinerja model, sehingga pemahaman yang komprehensif sangat diperlukan.

1. Accuracy

Accuracy adalah rasio jumlah prediksi benar terhadap keseluruhan data. Formula:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Walaupun sederhana, metrik ini bisa menyesatkan bila distribusi kelas tidak seimbang.

2. Precision

Precision menilai ketepatan prediksi positif. Formula:

$$Precision = \frac{TP}{TP + FP}$$

Sangat relevan ketika konsekuensi dari *false positive* cukup serius, misalnya pada sistem deteksi spam atau persetujuan transaksi finansial.

3. Recall

Recall menilai sejauh mana model berhasil menemukan seluruh data positif. Formula:

$$Recall = \frac{TP}{TP + FN}$$

Metrik ini penting dalam konteks kesehatan, seperti skrining penyakit, di mana kehilangan satu kasus positif bisa berakibat fatal.

4. F1-Score

F1-score menggabungkan *precision* dan *recall* dalam bentuk rata-rata harmonis. Formula:

$$F1-Score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Digunakan saat data tidak seimbang, karena memberikan gambaran keseimbangan antara mendeteksi positif sekaligus mengurangi kesalahan.

5. Confusion Matrix

Confusion Matrix menampilkan tabel distribusi hasil prediksi:

	Prediksi Positif	Prediksi Negatif
Aktual Positif	<i>TP</i>	<i>FN</i>
Aktual Negatif	<i>FP</i>	<i>TN</i>

- TP (True Positive):** Positif terprediksi benar
- TN (True Negative):** Negatif terprediksi benar
- FP (False Positive):** Negatif terprediksi positif
- FN (False Negative):** Positif terprediksi negatif

Matriks ini memungkinkan analisis detail terhadap jenis kesalahan model, misalnya apakah model lebih sering salah mendeteksi positif atau negatif.

C. Pentingnya Evaluasi

Evaluasi harus disesuaikan dengan kebutuhan praktis. Misalnya, dalam kesehatan, **recall** lebih krusial daripada akurasi agar semua kasus positif teridentifikasi.

Algoritma *Supervised Learning* untuk AI Masa Depan

A. Deep Learning

Deep Learning menggunakan *jaringan saraf berlapis* untuk mengolah data dalam bentuk gambar, teks, maupun suara. Keunggulannya terletak pada kemampuannya menangkap pola yang sangat kompleks. Misalnya, dalam pengenalan wajah, *deep learning* mampu mengenali detail kecil yang sulit diproses oleh algoritma tradisional. Dengan demikian, metode ini sangat cocok digunakan pada data yang bersifat besar, tidak terstruktur, dan dinamis.

B. Ensemble Learning

Ensemble Learning menggabungkan beberapa model sederhana untuk meningkatkan akurasi prediksi. Contohnya, *Random Forest* dan *Gradient Boosting* terbukti efektif dalam berbagai kompetisi data. Strategi ini bekerja dengan prinsip bahwa gabungan banyak model lemah dapat menghasilkan kinerja yang lebih baik dibanding satu model tunggal. Keunggulannya adalah stabilitas hasil dan daya generalisasi yang tinggi.

C. Hybrid Models

Hybrid Models mengombinasikan pendekatan klasik dan modern. Misalnya, integrasi antara *deep learning* dengan regresi tradisional dapat memberikan fleksibilitas lebih besar. Pendekatan ini memungkinkan penyesuaian metode terhadap karakteristik data, sehingga hasil prediksi menjadi lebih efisien sekaligus akurat.

D. Tren Mendatang

Dengan pertumbuhan pesat **big data**, kebutuhan terhadap algoritma yang **adaptif** dan **efisien** semakin meningkat. Oleh karena itu, mahasiswa perlu menguasai algoritma dasar sekaligus memahami tren terbaru, agar mampu mengembangkan riset dan aplikasi profesional yang relevan dengan perkembangan teknologi AI masa depan.

Soal Uraian

1. Jelaskan perbedaan antara data terstruktur dan data tidak terstruktur serta berikan contohnya dalam konteks machine learning.
2. Apa yang dimaksud dengan Supervised Learning? Sertakan contoh penerapannya dalam kehidupan sehari-hari.
3. Bandingkan dua algoritma klasifikasi yang berbeda, misalnya K-Nearest Neighbors (KNN) dengan Support Vector Machine (SVM), dari sisi cara kerja dan kelebihan/kekurangannya.
4. Sebutkan dan jelaskan dua algoritma regresi yang umum digunakan pada supervised learning.

5. Gambarkan tahapan utama dalam proses pembangunan model supervised learning mulai dari pengumpulan data hingga deployment.
6. Mengapa evaluasi model sangat penting dalam supervised learning? Sertakan contoh metrik evaluasi yang digunakan pada regresi dan klasifikasi.
7. Apa perbedaan utama antara akurasi (accuracy) dan F1-score? Pada kasus seperti apa F1-score lebih tepat digunakan dibanding akurasi?
8. Jelaskan konsep overfitting dan underfitting dalam supervised learning. Bagaimana cara mengatasinya?
9. Menurut Anda, mengapa deep learning dan ensemble learning dianggap sebagai algoritma yang relevan untuk kecerdasan buatan masa depan?
10. Misalkan Anda diminta membangun sistem prediksi penyakit dari data medis pasien. Algoritma supervised learning apa yang akan Anda pilih? Jelaskan alasan pemilihannya.

Tugas Kelompok

Setiap kelompok diminta untuk menerapkan algoritma *supervised learning* pada sebuah dataset sederhana guna menunjukkan pemahaman tentang **konsep, tahapan, dan evaluasi model**.

Langkah-Langkah

1. Pemilihan Dataset

Pilih dataset dari sumber terbuka seperti *UCI Machine*

Learning Repository, Kaggle, atau dataset sederhana. Contoh yang direkomendasikan:

1. Prediksi harga rumah.
2. Klasifikasi bunga Iris.
3. Klasifikasi email spam.

2. **Preprocessing Data**

1. Lakukan pembersihan data (hapus nilai hilang atau duplikasi).
2. Lakukan normalisasi atau standarisasi numerik.
3. Lakukan encoding untuk variabel kategorikal.

3. **Penerapan Algoritma**

Terapkan minimal **dua algoritma supervised learning** berbeda, misalnya:

1. *Logistic Regression vs Decision Tree*.
2. *Linear Regression vs Random Forest*.

4. **Evaluasi Model**

Gunakan metrik evaluasi yang sesuai dengan jenis masalah:

1. Untuk klasifikasi: **Accuracy** dan **F1-score**.
2. Untuk regresi: **MSE** dan **R²**.

5. **Laporan Perbandingan**

Sajikan analisis perbandingan hasil antara kedua algoritma, meliputi:

1. Kelebihan dan kekurangan.
2. Nilai metrik evaluasi.
3. Interpretasi hasil yang diperoleh.

6. **Presentasi Kelompok**

Buat presentasi singkat sebanyak **5-7 slide** untuk dipaparkan di kelas.

Output yang Dikumpulkan

1. **Laporan Kelompok (3-5 halaman)** berisi:

1. Deskripsi dataset.
2. Tahapan preprocessing.
3. Penjelasan algoritma yang digunakan.
4. Hasil evaluasi model.
5. Analisis perbandingan dan kesimpulan.

2. **File Kode Program:**

Gunakan *Python* (Jupyter Notebook) atau *R*.

Unsupervised Learning

A. CPL

Memahami model *unsupervised learning*.

B. CPMK

Membandingkan algoritma *machine learning* pada *unsupervised learning*.

C. Sub-CPMK

Menentukan algoritma *machine learning* pada *unsupervised learning* serta penerapan yang tepat.

Unsupervised learning adalah salah satu pendekatan utama dalam **machine learning** yang berperan penting dalam menemukan pola atau struktur tersembunyi dari data tanpa menggunakan label. Berbeda dengan *supervised learning* yang membutuhkan pasangan input-output, pendekatan ini memanfaatkan data mentah untuk mengidentifikasi keteraturan, mengelompokkan data, melakukan reduksi dimensi, atau menemukan aturan asosiasi.

Bab ini disusun dengan gaya akademik yang terstruktur, menguraikan definisi, jenis algoritma, tahapan, teknik evaluasi, hingga arah perkembangan modern yang relevan dengan **artificial intelligence (AI)**, serta menekankan integrasi antara teori dan penerapan nyata.

Pengertian Model Unsupervised Learning

Unsupervised learning merupakan salah satu pendekatan dalam *machine learning* yang digunakan untuk **menganalisis data tanpa label**. Artinya, sistem tidak diberikan informasi mengenai kategori atau jawaban benar, melainkan harus menemukan pola atau struktur tersembunyi dalam data secara mandiri. Data yang diproses bisa berupa **numerik, kategorikal**, hingga bentuk yang lebih kompleks seperti **teks, gambar, maupun suara**. Tujuan Utama Metode ini memiliki beberapa tujuan pokok yang sangat relevan dalam praktik analisis data modern:

1. **Mengelompokkan data berdasarkan kemiripan (clustering).**

Proses ini membantu menemukan kelompok atau klaster dalam data yang memiliki karakteristik serupa. Contoh penerapannya terlihat dalam segmentasi pelanggan atau analisis perilaku pengguna.

2. **Mereduksi jumlah variabel (dimensionality reduction).**

Dengan mengurangi dimensi, analisis menjadi lebih sederhana tanpa kehilangan informasi penting. Hal ini umum digunakan dalam visualisasi data serta mempercepat proses komputasi.

3. **Menemukan aturan asosiasi (association rules).**

Aturan ini menjelaskan keterkaitan antarvariabel, seperti hubungan antarproduk dalam keranjang belanja konsumen (*market basket analysis*).

Keunggulan **unsupervised learning** terletak pada kemampuannya dalam **mengungkap pola tersembunyi** yang tidak terlihat oleh analisis tradisional. Misalnya, perusahaan dapat memahami kebutuhan pelanggan secara lebih personal, ilmuwan dapat menganalisis data genetik, dan sistem keamanan dapat mendeteksi anomali.

Perbedaan dengan Supervised Learning

1. Supervised Learning

Supervised learning adalah metode yang **membutuhkan data berlabel**. Artinya, setiap data latih memiliki pasangan antara *input* dan *output* yang benar. Tujuan utamanya adalah **memprediksi hasil tertentu** berdasarkan pola yang sudah dipelajari dari data sebelumnya. Contoh penerapannya adalah dalam **klasifikasi email** (spam atau bukan spam), **prediksi harga rumah**, hingga **pengenalan wajah**.

Kelebihan utama pendekatan ini adalah akurasi yang tinggi dalam menghasilkan prediksi, karena sistem memiliki rujukan jelas berupa label. Namun, kekurangannya terletak pada kebutuhan data berlabel yang besar dan mahal untuk disiapkan.

2. Unsupervised Learning

Berbeda dengan pendekatan sebelumnya, **unsupervised learning tidak memerlukan label** pada data. Fokus utamanya adalah **mengeksplorasi pola tersembunyi** dan struktur alami

dalam data. Metode ini sering digunakan untuk **clustering**, **reduksi dimensi**, dan **market basket analysis**.

Keunggulan pendekatan ini adalah kemampuannya menemukan wawasan baru tanpa instruksi eksplisit. Meski demikian, hasilnya terkadang lebih sulit diinterpretasikan karena tidak ada label yang jelas sebagai acuan.

Contoh Aplikasi Nyata

1. Segmentasi pelanggan

Salah satu penerapan paling umum adalah **segmentasi pelanggan**. Perusahaan dapat **mengelompokkan konsumen berdasarkan pola pembelian**, preferensi, atau perilaku mereka. Dengan demikian, strategi pemasaran dapat disesuaikan sehingga lebih **personalized** dan efektif. Misalnya, pelanggan yang sering membeli produk tertentu dapat ditawarkan promosi spesifik yang relevan.

2. Deteksi anomali

Dalam bidang keamanan siber, **deteksi anomali** menjadi kunci. Sistem dapat **mengidentifikasi aktivitas tidak biasa dalam jaringan komputer**, seperti akses mencurigakan atau transaksi abnormal. Hal ini membantu mencegah ancaman seperti peretasan atau penipuan keuangan.

3. Rekomendasi produk

Aplikasi lain yang sering ditemui adalah **rekomendasi produk**. Melalui analisis keterkaitan antarproduk, sistem dapat menemukan barang yang **sering dibeli bersama**. Contoh paling populer adalah fitur rekomendasi pada *e-commerce*, yang membantu meningkatkan pengalaman pengguna sekaligus mendorong penjualan.

4. Analisis gambar medis

Dalam bidang kesehatan, **analisis gambar medis** memainkan peran penting. Dengan mengelompokkan citra berdasarkan **kemiripan visual**, sistem dapat mendukung dokter dalam **diagnosis penyakit**. Misalnya, citra hasil MRI atau CT-Scan dapat dikelompokkan untuk mempermudah deteksi kelainan..

Jenis-Jenis Algoritma untuk Model *Unsupervised Learning*

A. Clustering

Clustering merupakan salah satu teknik penting dalam *unsupervised learning* yang bertujuan untuk **membagi data menjadi kelompok** berdasarkan tingkat kesamaan atau kedekatan karakteristik. Metode ini sangat bermanfaat dalam berbagai bidang, mulai dari pemasaran, analisis sosial, hingga penelitian ilmiah. Berikut beberapa algoritma *clustering* yang banyak digunakan dalam praktik.:

1. K-Means

K-Means adalah algoritma yang **populer, sederhana, dan efisien** untuk data berukuran besar. Prinsip kerjanya adalah

membagi data ke dalam sejumlah klaster yang telah ditentukan sebelumnya. Algoritma ini bekerja optimal pada data dengan klaster berbentuk **bulat atau cenderung simetris**. Contoh penerapannya adalah dalam **segmentasi pelanggan** atau **pengelompokan dokumen**.

Prinsip Kerja

Secara garis besar, algoritma K-Means bekerja melalui langkah-langkah berikut:

1. Menentukan jumlah klaster (k) yang akan dibentuk.
2. Memilih posisi awal centroid secara acak.
3. Menghitung jarak setiap titik data ke centroid terdekat dan mengelompokkan titik tersebut.
4. Memperbarui posisi centroid berdasarkan rata-rata titik data dalam tiap klaster.
5. Mengulang proses hingga terjadi **konvergensi**, yaitu posisi centroid stabil atau perubahan sangat kecil.

Algoritma ini paling optimal digunakan pada data dengan **klaster berbentuk bulat atau cenderung simetris**, sehingga pemisahan antar kelompok lebih jelas.

Formula

Tujuan K-Means adalah **meminimalkan jumlah kuadrat jarak antara setiap titik data dengan centroid klasternya**. Fungsi objektif dapat dituliskan sebagai:

$$J = \sum_{i=1}^k \sum_{x_j \in C_i} |x_j - \mu_i|^2$$

Dimana:

J : ukuran total kesalahan yang diusahakan untuk diperkecil.

k : jumlah kelompok atau klaster yang dipilih sejak awal.

x_j : titik data ke- j dalam kumpulan data.

C_i : kumpulan anggota yang termasuk dalam klaster ke- i .

μ_i : pusat atau titik rata-rata (centroid) dari klaster ke- i .

$|x_j - \mu_i|^2$: ukuran jarak kuadrat yang menunjukkan seberapa dekat suatu data dengan pusat klasternya.

2. Hierarchical clustering

Berbeda dari K-Means, **Hierarchical Clustering** menghasilkan **struktur hierarki** yang dapat divisualisasikan dengan **dendrogram**. Struktur ini membantu pengguna memahami hubungan antarobjek secara bertingkat, dari yang paling mirip hingga paling berbeda. Metode ini bermanfaat dalam **analisis genetik, biologi**, atau konteks di mana interpretasi hubungan antarentitas sangat penting.

Prinsip Kerja

Proses *hierarchical clustering* biasanya dilakukan dengan dua pendekatan:

1. Agglomerative

Memulai dari setiap data sebagai klaster individu, kemudian menggabungkannya secara bertahap hingga terbentuk satu klaster besar.

2. **Divisive**

Memulai dari satu klaster besar, kemudian membaginya menjadi klaster-klaster kecil.

Algoritma ini mengandalkan ukuran kedekatan atau *similarity measure*, seperti jarak Euclidean, Manhattan, atau Cosine.

Formula

Salah satu metode pengukuran jarak antar klaster adalah **metode linkage**. Misalnya, pada *single linkage* (jarak terdekat antar klaster), formula matematis dapat dituliskan sebagai:

$$d(A, B) = \min_{x \in A, y \in B} d(x, y)$$

Dimana:

$d(A, B)$: jarak antara klaster A dan B.

x, y : titik data dalam masing-masing klaster.

$d(x, y)$: jarak antara dua titik data (umumnya jarak Euclidean).

Selain *single linkage*, terdapat juga metode lain seperti *complete linkage* (menggunakan jarak terjauh) dan *average linkage* (menggunakan rata-rata jarak antar anggota klaster).

3. **DBSCAN**

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) adalah algoritma berbasis kepadatan. Keunggulannya terletak pada kemampuannya **menangani data**

dengan bentuk kluster tidak beraturan serta **mendeteksi outlier**. Algoritma ini sangat efektif dalam **analisis spasial**, seperti mendeteksi pola keramaian di suatu wilayah atau mengidentifikasi perilaku anomali.

Prinsip Kerja

DBSCAN bekerja dengan dua parameter utama:

1. **E (epsilon)**

Radius atau jarak maksimum untuk menentukan kedekatan antar titik.

2. **Minpts**

Jumlah minimum titik dalam radius ϵ agar suatu titik dapat dianggap sebagai bagian inti kluster (*core point*).

Berdasarkan kedua parameter tersebut, titik data dapat dikategorikan menjadi tiga jenis:

a. **Core point**

Memiliki jumlah tetangga $\geq$ minpts dalam radius ϵ .

b. **Border point**

Berada dalam radius ϵ dari core point, tetapi jumlah tetangganya $<$ minpts.

c. **Noise (outlier)**

Tidak termasuk core point maupun border point.

Formula

Definisi tetangga ϵ dari suatu titik x dapat dirumuskan sebagai:

$$N_{\epsilon}(x) = \{y \in D \mid d(x, y) \leq \epsilon\}$$

Dimana:

$N_\epsilon(x)$: himpunan titik dalam radius ϵ dari titik x .

D : keseluruhan himpunan data.

$d(x,y)$: jarak antara titik x dan y .

ϵ : parameter jarak maksimum.

Syarat **core point** dapat dituliskan:

$$|N_\epsilon(x)| \geq MinPts$$

Artinya, jika jumlah tetangga sebuah titik dalam radius ϵ minimal sebesar $MinPts$, maka titik tersebut menjadi pusat kluster.

B. Dimensionality Reduction

Dimensionality reduction adalah teknik penting dalam *machine learning* dan analisis data yang bertujuan untuk **mengurangi jumlah variabel** tanpa kehilangan informasi utama. Dengan mengurangi dimensi, data menjadi lebih mudah dianalisis, divisualisasikan, dan diproses secara komputasional. Hal ini sangat relevan ketika data memiliki banyak fitur yang kompleks atau berukuran besar:

1. PCA (Principal Component Analysis)

PCA merupakan metode yang paling populer dalam **reduksi dimensi**. Teknik ini bekerja dengan cara **mentransformasikan variabel ke dalam komponen utama baru** yang saling bebas dan mewakili sebagian besar variasi data. Kelebihan PCA adalah kesederhanaan perhitungannya dan efektivitasnya dalam merangkum informasi. PCA banyak digunakan dalam bidang keuangan, pengenalan wajah, serta bioinformatika.

Prinsip Kerja

PCA bekerja dengan cara:

1. Menghitung matriks kovarians dari data.
2. Menentukan **eigenvalue** dan **eigenvector** dari matriks tersebut.
3. Mengurutkan komponen berdasarkan eigenvalue terbesar (variasi terbesar).
4. Memproyeksikan data ke dalam ruang berdimensi lebih rendah menggunakan komponen utama tersebut.

Formula

Transformasi PCA dapat dirumuskan sebagai berikut:

$$Z = XW$$

Dimana:

X: matriks data asli berukuran $(n \times p)$, dengan n sampel dan p variabel.

W: matriks berisi vektor eigen dari matriks kovarians data.

Z: matriks data baru dalam ruang komponen utama.

Sedangkan perhitungan matriks kovarians ditulis sebagai:

$$C = \frac{1}{n-1} X^T X$$

Eigenvector dari matriks kovarians **C** digunakan sebagai arah komponen utama.

2. T-SNE

T-SNE lebih banyak digunakan untuk **visualisasi data berdimensi tinggi** ke dalam bentuk peta dua atau tiga dimensi. Keunggulan utamanya adalah kemampuannya dalam **menjaga**

struktur lokal data, sehingga pola kluster dapat terlihat lebih jelas. Metode ini sangat populer dalam eksplorasi data kompleks seperti genomik atau representasi kata (*word embeddings*).

Prinsip Kerja

T-SNE bekerja dengan cara:

1. Mengukur kemiripan antar titik dalam ruang berdimensi tinggi dengan distribusi probabilitas.
2. Mentransformasikan data ke ruang berdimensi rendah sambil menjaga hubungan probabilitas tersebut.
3. Meminimalkan perbedaan distribusi menggunakan ukuran

Kullback-Leibler divergence.

Formula

Probabilitas kemiripan antar titik x_i dan x_j dalam ruang berdimensi tinggi didefinisikan sebagai:

$$p_{j|i} = \frac{\exp(-|x_i - x_j|^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-|x_i - x_k|^2 / 2\sigma_i^2)}$$

Selanjutnya, probabilitas gabungan dituliskan:

$$p_{\{ij\}} = \frac{1}{2n} (p_{\{j|i\}} + p_{\{i|j\}})$$

Untuk ruang berdimensi rendah (misalnya y_i dan y_j), distribusi probabilitas didefinisikan menggunakan distribusi *Student-t*:

$$q_{ij} = \frac{(1 + |y_i - y_j|^2)^{-1}}{\sum_{k \neq l} (1 + |y_k - y_l|^2)^{-1}}$$

Fungsi objektif T-SNE adalah meminimalkan **Kullback-Leibler divergence** antara distribusi p_{ij} dan q_{ij} :

$$KL(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

3. Autoencoder

Autoencoder adalah bentuk **jaringan saraf tiruan** yang dirancang untuk **mengompresi data** ke representasi dengan dimensi lebih rendah, lalu merekonstruksinya kembali. Pendekatan ini fleksibel karena dapat menangani data non-linear. Autoencoder banyak diterapkan dalam kompresi gambar, deteksi anomali, dan pemrosesan bahasa alami.

Prinsip Kerja

Struktur autoencoder terdiri dari tiga bagian utama:

1. Encoder

Mentransformasikan input ke representasi berdimensi rendah (*latent space*).

2. Latent Representation

Hasil kompresi data yang berisi fitur penting.

3. Decoder

Merekonstruksi kembali data dari representasi laten ke bentuk asli.

Tujuan pelatihan autoencoder adalah **meminimalkan selisih antara input dan output rekonstruksi**.

Formula

Jika diberikan data input x , maka proses encoding dan decoding dapat dituliskan sebagai:

$$h = f(Wx + b)$$

$$\hat{x} = g(W'h + b')$$

Dimana:

x: data input.

h: representasi laten hasil *encoder*.

$\hat{x}$: data hasil rekonstruksi dari *decoder*.

W, W': bobot pada layer encoder dan decoder.

b, b': bias pada masing-masing layer.

f, g: fungsi aktivasi non-linear.

Fungsi objektif autoencoder adalah meminimalkan **rekonstruksi error**, misalnya dengan menggunakan *Mean Squared Error (MSE)*:

$$L(x, \hat{x}) = |x - \hat{x}|^2$$

C. Association Rule Learning

Association Rule Learning adalah salah satu metode penting dalam *unsupervised learning* yang bertujuan untuk **menemukan aturan asosiasi antar item** dalam sebuah dataset. Teknik ini sangat berguna untuk memahami keterkaitan antarvariabel sehingga dapat digunakan dalam pengambilan keputusan berbasis data. Salah satu penerapan paling terkenal adalah dalam *market basket analysis*, yaitu analisis kebiasaan belanja konsumen.:

1. Apriori

Apriori merupakan algoritma klasik yang digunakan dalam menemukan aturan asosiasi. Prinsip kerjanya adalah **mengidentifikasi itemset yang sering muncul bersama** dalam dataset. Dari itemset tersebut, algoritma membentuk aturan asosiasi yang menunjukkan kemungkinan keterkaitan antarproduk. Contohnya, jika banyak pelanggan membeli *roti* dan *mentega* secara bersamaan, maka peluang mereka membeli *selai* juga tinggi.

Kelebihan Apriori adalah kesederhanaannya, namun kekurangannya adalah membutuhkan waktu komputasi yang cukup besar pada dataset berukuran besar karena algoritma ini harus melakukan pencarian berulang kali.

Prinsip Kerja

Apriori menggunakan dua konsep utama:

1. **Support**

Menunjukkan seberapa sering sebuah itemset muncul dalam dataset.

2. **Confidence**

Mengukur kekuatan aturan asosiasi dengan menghitung seberapa sering item B muncul ketika item A juga muncul.

Algoritma ini bekerja secara iteratif, dimulai dari itemset tunggal, kemudian membangun kandidat itemset yang lebih besar hingga ditemukan aturan dengan *support* dan *confidence* yang memenuhi ambang batas minimum.

Formula

Support dari suatu aturan $A \Rightarrow B$:

$$Support(A \Rightarrow B) = \frac{\text{Jumlah transaksi yang mengandung A dan B}}{\text{Total jumlah transaksi}}$$

Confidence dari aturan $A \Rightarrow B$:

$$Confidence(A \Rightarrow B) = \frac{Support(A \cup B)}{Support(A)}$$

Lift sebagai ukuran tambahan untuk menilai kekuatan aturan:

$$Lift(A \Rightarrow B) = \frac{Confidence(A \Rightarrow B)}{Support(B)}$$

2. FP-Growth

Sebagai alternatif yang lebih efisien, hadir **FP-Growth (Frequent Pattern Growth)**. Algoritma ini menggunakan struktur data khusus berupa *FP-tree* untuk **menyimpan pola frekuensi item secara kompak**. Dengan cara ini, pencarian aturan asosiasi dapat dilakukan lebih cepat tanpa harus menghasilkan kandidat itemset sebanyak pada Apriori. FP-Growth sangat efektif dalam **mengolah dataset besar** dengan pola berulang yang kompleks.

Prinsip Kerja

FP-Growth bekerja melalui dua tahap utama:

1. Membangun FP-Tree

Dari dataset dengan mencatat frekuensi item dan urutan kemunculannya.

2. Menambang frequent itemset

Dari FP-Tree dengan menggunakan pendekatan rekursif, tanpa harus menghasilkan semua kandidat itemset seperti pada Apriori.

Dengan demikian, FP-Growth lebih hemat waktu dan memori ketika berhadapan dengan data berskala besar.

Formula

Seperti pada aturan asosiasi, FP-Growth menggunakan ukuran utama yaitu **support** untuk menentukan frequent itemset. Rumus *support* dapat dituliskan sebagai:

$$Support(X) = \frac{\text{Jumlah transaksi yang mengandung X}}{\text{Total jumlah transaksi}}$$

Untuk membentuk aturan asosiasi dari frequent itemset, digunakan pula ukuran **confidence**:

$$Confidence(A \Rightarrow B) = \frac{Support(A \cup B)}{Support(A)}$$

Tabel 11. Perbandingan Algoritma

Kategori	Contoh Algoritma	Kelebihan	Kekurangan
<i>Clustering</i>	K-Means, DBSCAN	Mudah digunakan, hasil mudah divisualisasikan	Sensitif terhadap parameter dan skala data
<i>Dimensionality Reduction</i>	PCA, t-SNE, Autoencoder	Mengurangi kompleksitas data, visualisasi	Risiko kehilangan informasi penting
<i>Association Rule</i>	Apriori, FP-Growth	Efektif untuk rekomendasi produk	Performa menurun pada dataset sangat besar

Tahapan di Model Unsupervised Learning

Penerapan **unsupervised learning**, terdapat beberapa tahapan yang harus dilalui agar proses analisis data dapat berjalan secara efektif. Tahapan ini memastikan bahwa data yang digunakan siap diproses, algoritma yang dipilih sesuai, serta hasil analisis dapat dipertanggungjawabkan secara akademik maupun praktis. Berikut penjelasan langkah-langkah umum dalam membangun model *unsupervised learning*:

A. Pengumpulan data

Tahap awal adalah **pengumpulan data** dari berbagai sumber. Data dapat berasal dari **basis data internal**, perangkat **sensor**

IoT, maupun **sumber publik** seperti data penelitian atau dataset terbuka. Kualitas data yang dikumpulkan akan sangat menentukan kualitas model.

B. Preprocessing

Data mentah perlu melalui tahap **preprocessing**, yaitu pembersihan dari nilai yang hilang atau tidak relevan, **normalisasi** agar skala variabel seragam, serta **transformasi** untuk menyesuaikan format data. Tahap ini penting agar algoritma dapat bekerja secara optimal.

C. Pemilihan algoritma

Pemilihan **algoritma** harus disesuaikan dengan tujuan analisis. Misalnya, **clustering** digunakan untuk segmentasi pelanggan, sedangkan **PCA (Principal Component Analysis)** dipakai untuk **reduksi dimensi** agar data lebih sederhana.

D. Pelatihan model

Pada tahap ini, algoritma dijalankan pada data **tanpa label** untuk menemukan pola, struktur, atau keterkaitan. Proses pelatihan akan menghasilkan output berupa klaster, representasi baru, atau aturan asosiasi..

E. Evaluasi hasil

Langkah terakhir adalah **evaluasi hasil**. Evaluasi dapat dilakukan dengan **metrik internal** (misalnya *silhouette score*), **metrik eksternal** jika ada label pembanding, atau melalui

visualisasi data. Evaluasi bertujuan memastikan hasil relevan, tidak bias, dan bermanfaat untuk pengambilan keputusan..

Teknik Evaluasi di Unsupervised Learning

Proses evaluasi model menjadi tantangan tersendiri karena data yang digunakan tidak memiliki label. Oleh karena itu, dibutuhkan **metrik evaluasi internal** untuk menilai kualitas hasil yang diperoleh. Tujuan utama dari evaluasi ini adalah memastikan bahwa hasil klasterisasi atau reduksi dimensi benar-benar merepresentasikan struktur alami data secara relevan, konsisten, dan tidak bias.

A. Internal Evaluation Metrics

a) Silhouette score

Silhouette score merupakan metrik yang digunakan untuk **menilai kesesuaian data dalam klaster**. Nilai skor ini berkisar antara -1 hingga 1. Semakin mendekati angka 1, semakin baik data ditempatkan dalam klasternya. Jika nilainya mendekati 0, berarti titik data berada di perbatasan dua klaster. Nilai negatif menandakan bahwa data salah ditempatkan. Metrik ini sangat berguna untuk memvalidasi hasil klasterisasi secara umum.

Prinsip Kerja

Perhitungan Silhouette Score mempertimbangkan dua aspek utama:

1. Cohesion ($a(i)$)

Jarak rata-rata antara titik data dengan semua anggota lain dalam kluster yang sama.

2. **Separation (b(i))**

Jarak rata-rata antara titik data dengan titik-titik dalam kluster terdekat lainnya.

Dengan membandingkan kedua nilai ini, Silhouette Score dapat menggambarkan seberapa baik pemisahan antar kluster sekaligus kepadatan dalam kluster.

Formula

Rumus Silhouette Score untuk suatu titik data i dapat dituliskan sebagai:

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$

Dimana:

s(i): nilai silhouette untuk titik data ke- i .

a(i): rata-rata jarak antara titik i dengan semua titik dalam kluster yang sama (cohesion).

b(i): rata-rata jarak antara titik i dengan semua titik dalam kluster terdekat (separation).

max(a(i), b(i)): digunakan untuk normalisasi nilai agar berada pada rentang -1 hingga 1.

b) **Davies-bouldin index**

Davies-Bouldin Index (DBI) digunakan untuk mengukur **kualitas kluster berdasarkan kemiripan antar kluster**. Semakin rendah nilai DBI, semakin baik hasil klusterisasi karena

menunjukkan bahwa kluster lebih terpisah dan padat. Metrik ini membantu peneliti dalam membandingkan performa beberapa algoritma klusterisasi pada dataset yang sama.

Prinsip Kerja

Perhitungan DBI melibatkan dua komponen utama:

1. **Within-cluster scatter (S_i)**

Mengukur seberapa rapat titik-titik dalam kluster ke- i .

2. **Between-cluster separation (M_{ij})**

Jarak antara pusat kluster ke- i dan pusat kluster ke- j .

Setiap pasangan kluster dibandingkan untuk menghitung rasio kedekatan relatifnya. Nilai DBI kemudian merupakan rata-rata dari rasio-rasio maksimum untuk semua kluster.

Formula

Davies-Bouldin Index didefinisikan sebagai:

$$DBI = \frac{1}{k} \sum_{i=1}^k \max_{j \neq i} \left(\frac{S_i + S_j}{M_{ij}} \right)$$

Dimana:

k: jumlah kluster.

S_i : rata-rata jarak titik-titik dalam kluster ke- i terhadap pusat kluster c_i .

M_{ij} : jarak antara pusat kluster c_i dan pusat kluster c_j .

Rasio $(S_i + S_j)/M_{ij}$: menggambarkan tingkat kemiripan antara dua kluster.

c) **Dunn index**

Dunn Index menilai **pemisahan antar klaster** dengan membandingkan jarak minimum antar klaster dengan jarak maksimum dalam klaster. Nilai Dunn Index yang tinggi menunjukkan klaster terpisah dengan baik dan memiliki kepadatan internal yang konsisten. Hal ini menjadikan metrik ini efektif untuk memastikan kualitas pemisahan data.

Prinsip Kerja

Dunn Index memperhatikan dua aspek utama:

1. **Jarak minimum antar klaster**

Menunjukkan seberapa jauh klaster yang berbeda dipisahkan satu sama lain.

2. **Jarak maksimum dalam klaster**

Menggambarkan seberapa rapat titik-titik dalam satu klaster. Dengan membandingkan keduanya, Dunn Index memberikan ukuran keseimbangan antara pemisahan dan kepadatan klaster.

Formula

Dunn Index dapat dirumuskan sebagai berikut:

$$D = \frac{\min_{1 \leq i < j \leq k} d(C_i, C_j)}{\max_{1 \leq l \leq k} d_{iam}(C_l)}$$

Dimana:

D: nilai Dunn Index.

k: jumlah klaster.

$d(C_i, C_j)$: jarak antara klaster ke- i dan klaster ke- j (biasanya menggunakan jarak minimum antar anggota).

$diam(C_l)$: diameter klaster ke- l , yaitu jarak maksimum antara dua titik dalam klaster tersebut.

B. External Evaluation Metrics

a) **Adjusted rand index (ARI)**

Adjusted Rand Index (ARI) digunakan untuk **membandingkan hasil klasterisasi dengan label sebenarnya**. Metrik ini memperhitungkan semua pasangan titik data dan menilai apakah pasangan tersebut ditempatkan dalam klaster yang sama atau berbeda, dibandingkan dengan label asli. Nilai ARI berkisar antara -1 hingga 1, dengan:

- a. Nilai mendekati **1** menunjukkan kesesuaian tinggi dengan label.
- b. Nilai mendekati **0** menunjukkan hasil acak.
- c. Nilai negatif menunjukkan hasil klasterisasi yang lebih buruk daripada acak.

Keunggulan ARI adalah kemampuannya dalam menyesuaikan nilai terhadap peluang acak, sehingga hasil evaluasi lebih objektif.

Prinsip Kerja

Nilai ARI berada dalam rentang -1 hingga 1:

- a. Nilai mendekati **1** menunjukkan kesesuaian tinggi dengan label.

- b. Nilai mendekati **0** menunjukkan hasil yang setara dengan acak.
- c. Nilai negatif menunjukkan klusterisasi lebih buruk daripada acak.

Formula

Secara matematis, ARI dapat dituliskan sebagai:

$$ARI = \frac{RI - E[RI]}{\max(RI) - E[RI]}$$

Dimana:

RI (Rand Index): proporsi kesepakatan antara hasil klusterisasi dengan label sebenarnya.

E[RI]: nilai harapan dari Rand Index jika klusterisasi dilakukan secara acak.

max (RI): nilai maksimum Rand Index.

Rand Index (RI) sendiri dihitung berdasarkan jumlah pasangan titik data yang konsisten antara kluster hasil dan label asli.

b) Normalized mutual information (NMI)

Normalized Mutual Information (NMI) mengukur **informasi bersama** antara label kluster dan label sebenarnya. Metrik ini berasal dari teori informasi dan digunakan untuk menilai sejauh mana pengetahuan tentang kluster dapat memberikan informasi mengenai label asli. Nilai NMI berada pada rentang 0 hingga 1:

- a. Nilai mendekati **1** menunjukkan kesesuaian tinggi.
- b. Nilai mendekati **0** menunjukkan hampir tidak ada informasi yang dibagi.

Kelebihan NMI adalah sifatnya yang **simetris**, sehingga hasil evaluasi tidak bergantung pada urutan label klaster.

Prinsip Kerja

Nilai NMI berada pada rentang 0 hingga 1:

- a. Nilai mendekati **1** menunjukkan kesesuaian yang tinggi antara hasil klasterisasi dan label asli.
- b. Nilai mendekati **0** menunjukkan bahwa hampir tidak ada informasi yang dibagi.

Dengan demikian, NMI membantu menilai kualitas klasterisasi berdasarkan informasi yang tumpang tindih antara label prediksi dan label sebenarnya.

Formula

Secara matematis, NMI didefinisikan sebagai:

$$NMI(U, V) = \frac{2 \cdot I(U; V)}{H(U) + H(V)}$$

Dimana:

U: hasil klasterisasi.

V: label sebenarnya.

I(U; V): *mutual information* antara U dan V.

H(U): entropi dari distribusi klaster U.

H(V): entropi dari distribusi label V.

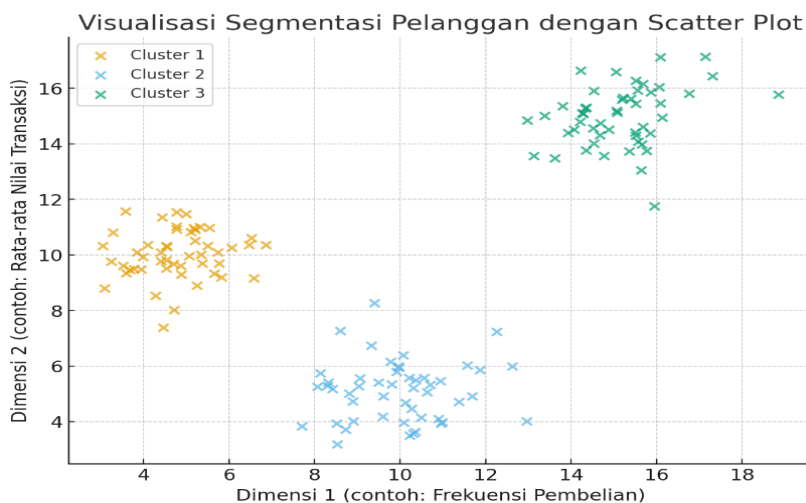
Nilai NMI yang tinggi menunjukkan bahwa informasi dari hasil klasterisasi sejalan dengan label asli, sedangkan nilai rendah menandakan klasterisasi kurang relevan.

C. Visualisasi Data

Analisis **unsupervised learning**, hasil klasterisasi atau reduksi dimensi sering kali sulit dipahami hanya dari angka atau tabel. Oleh karena itu, **visualisasi data** menjadi langkah penting untuk membantu peneliti maupun praktisi memahami pola, distribusi, dan keterkaitan antarvariabel. Dengan visualisasi yang tepat, interpretasi data dapat dilakukan dengan lebih mudah, akurat.

a) Scatter plot

Salah satu teknik dasar yang banyak digunakan adalah **scatter plot**. Visualisasi ini menampilkan distribusi titik data berdasarkan dua atau tiga variabel yang dipilih. Dalam konteks klasterisasi, scatter plot memudahkan kita untuk melihat **distribusi cluster**, kepadatan, serta pemisahan antar kelompok. Misalnya, dalam analisis segmentasi pelanggan, scatter plot dapat memperlihatkan perbedaan perilaku konsumen dalam dua dimensi utama, sehingga setiap kelompok dapat diidentifikasi lebih jelas.

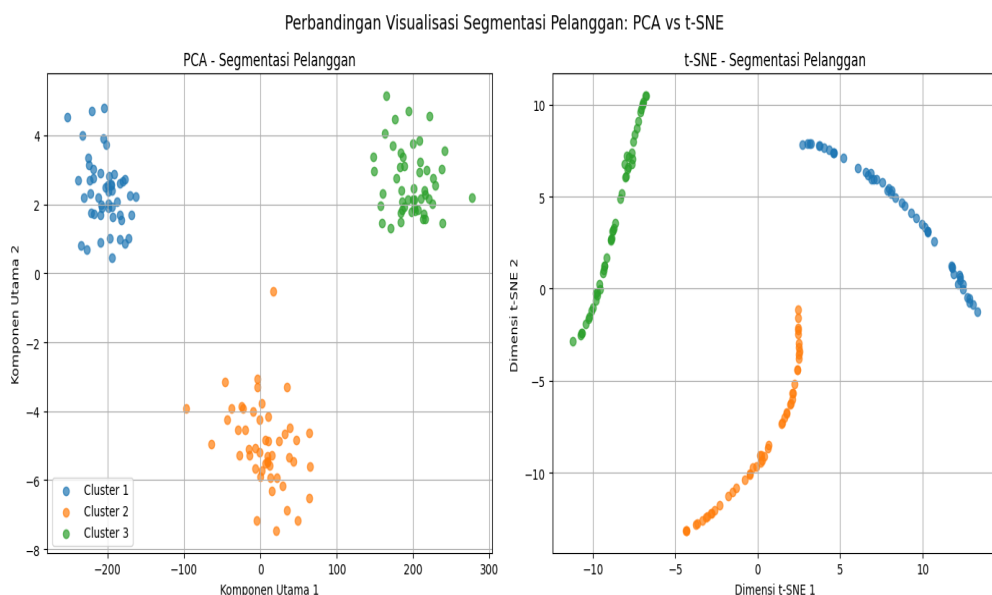


Gambar 42. Visualisasi Scatter Plot

b) Peta dua dimensi dari PCA atau t-SNE

Untuk dataset berdimensi tinggi, penggunaan metode reduksi dimensi seperti **PCA (Principal Component Analysis)** atau **T-SNE (t-distributed Stochastic Neighbor Embedding)** sangat efektif. Kedua metode ini memungkinkan data kompleks diproyeksikan ke dalam **peta dua dimensi** sehingga lebih mudah diinterpretasikan.

- PCA** bekerja dengan mentransformasikan variabel ke dalam komponen utama baru yang mewakili sebagian besar variasi data.
- T-SNE** lebih unggul dalam menjaga **struktur lokal data**, sehingga pola kluster yang rumit dapat divisualisasikan dengan lebih jelas.



Gambar 43.Perbandingan Visualisasi PCA Vs t-SNE

Jenis Algoritma *Unsupervised Learning* untuk AI di Masa Depan

A. Deep Autoencoder

Deep Autoencoder merupakan model yang efektif untuk **kompresi data kompleks** dan **deteksi anomali**. Algoritma ini bekerja dengan cara merekonstruksi data input sehingga dapat mengidentifikasi perbedaan signifikan antara data normal dan anomali. Aplikasinya meliputi keamanan siber hingga diagnosis medis.

B. Generative Adversarial Networks (GANs)

Generative Adversarial Networks atau **GANs** melibatkan dua jaringan: **generator** yang menghasilkan data baru dan **discriminator** yang mengevaluasi keaslian data. Dengan mekanisme ini, GANs dapat menciptakan **data sintetis realistis**, misalnya dalam pengembangan gambar, suara, dan simulasi lingkungan virtual..

C. Self-Organizing Maps (SOM)

Self-Organizing Maps membantu merepresentasikan **data berdimensi tinggi** ke dalam bentuk **peta dua dimensi**. Hal ini memudahkan analisis visual dan penemuan pola tersembunyi. SOM sering dimanfaatkan dalam riset pasar, pengenalan pola, serta visualisasi data besar.

D. Graph-Based Clustering

Algoritma ini mengelompokkan data berdasarkan struktur **graf**, sehingga sangat berguna untuk menganalisis hubungan dalam jejaring sosial, kolaborasi akademik, maupun interaksi

biologis. **Graph-based clustering** mampu mengungkap konektivitas kompleks yang sulit ditangkap oleh metode tradisional..

Soal Uraian

1. Jelaskan pengertian *unsupervised learning* dan perbedaannya dengan *supervised learning*.
2. Sebutkan dan jelaskan minimal tiga algoritma *clustering* yang digunakan dalam *unsupervised learning*.
3. Mengapa *dimensionality reduction* penting dalam *unsupervised learning*? Berikan contohnya.
4. Uraikan langkah-langkah utama dalam membangun model *unsupervised learning*.
5. Sebutkan dan jelaskan dua metrik evaluasi internal yang digunakan untuk mengukur kualitas hasil *clustering*.
6. Bandingkan metode K-Means dan Hierarchical Clustering dalam hal kelebihan dan kekurangan.
7. Bagaimana cara kerja algoritma Apriori dalam menemukan *association rules*?
8. Jelaskan peran *Generative Adversarial Networks (GANs)* dalam pengembangan *artificial intelligence* di masa depan.
9. Mengapa visualisasi penting dalam mengevaluasi hasil *unsupervised learning*? Berikan contoh bentuk visualisasi yang umum digunakan.
10. Menurut Anda, algoritma *unsupervised learning* mana yang paling relevan untuk digunakan dalam *big data* dan mengapa?

Tugas Kelompok

Model *unsupervised learning*

Buatlah implementasi *machine learning* untuk model *unsupervised learning* dengan studi kasus sederhana dan mudah dipahami.

Petunjuk tugas:

1. Studi Kasus

Segmentasi pelanggan berdasarkan data transaksi sederhana (misalnya: jumlah pembelian, frekuensi kunjungan, dan total belanja).

2. Langkah-Langkah:

- a. Kumpulkan dataset sederhana (bisa berupa data simulasi atau dataset publik kecil).
- b. Lakukan preprocessing data (normalisasi/standarisasi).
- c. Terapkan algoritma *K-Means Clustering* untuk membagi pelanggan ke dalam beberapa kelompok.
- d. Visualisasikan hasil clustering dalam bentuk scatter plot (misalnya kelompok pelanggan dengan warna berbeda).
- e. Interpretasikan hasil cluster (contoh: pelanggan dengan belanja tinggi, pelanggan reguler, pelanggan jarang belanja).

3. Output yang Diharapkan:

- a. Notebook Python (Jupyter/Google Colab) berisi kode implementasi.
- b. Laporan singkat berisi penjelasan tahapan pengerjaan, hasil visualisasi, dan analisis kelompok pelanggan.

Tujuan tugas ini adalah agar mahasiswa dapat memahami bagaimana *unsupervised learning* diterapkan pada kasus nyata dengan data sederhana dan memperoleh keterampilan praktis dalam mengolah serta menginterpretasikan data.

Model *Unsupervised Learning*

Judul:

Implementasi *Machine Learning* untuk Model *Unsupervised Learning*
(Studi Kasus Sederhana)

Tujuan Pembelajaran:

1. Menerapkan tahapan *unsupervised learning* dari pra-pemrosesan hingga evaluasi.
2. Membandingkan beberapa algoritma dan memilih yang paling sesuai untuk kasus sederhana.
3. Menghasilkan visualisasi dan laporan yang mudah dipahami non-teknis.

Studi Kasus (pilih salah satu):

1. **Segmentasi Pelanggan** dengan *K-Means* pada data sederhana (mis. pendapatan dan skor belanja).
2. **Deteksi Anomali Transaksi** dengan *DBSCAN* pada data jumlah transaksi per hari.
3. **Asosiasi Produk** dengan *Apriori* pada data keranjang belanja mini (beberapa item sehari-hari).
4. **Reduksi Dimensi & Visualisasi** dengan **PCA** dan/atau **t-SNE** pada data 2-4 fitur, lalu jelaskan pola yang muncul.

Sumber Data yang Disarankan (mudah & kecil):

1. Dataset *UCI Machine Learning Repository*, *Kaggle*, atau dataset sederhana yang kalian buat sendiri (10–200 baris sudah cukup).
2. Alternatif: dataset publik kecil yang umum dikenal (mis. *Iris* untuk reduksi dimensi; atau data belanja sederhana yang kalian susun dari survei kelas).

Alat/Perangkat:

Python (Jupyter/Colab), pandas, numpy, scikit-learn (untuk K-Means, DBSCAN, PCA, t-SNE), mlxtend (Apriori) bila diperlukan, dan matplotlib untuk visualisasi.

Langkah Implementasi yang Wajib:

1. **Definisi Masalah:** nyatakan pertanyaan riset (contoh: “Bagaimana mengelompokkan pelanggan ke 3 segmen?”).
2. **Import Data & Explorasi Singkat:** cek ukuran, tipe data, *missing values*.
3. **Pra-pemrosesan:** pilih fitur, tangani *missing values*, lakukan normalisasi/standarisasi jika perlu.
4. **Pemodelan:**
 - a. Clustering (K-Means/DBSCAN/hierarki) **atau** Apriori **atau** Reduksi Dimensi (PCA/t-SNE).
 - b. Jika *clustering*: coba beberapa nilai parameter (mis. *k* pada K-Means, *eps* & *min_samples* pada DBSCAN).
5. **Evaluasi & Validasi:** gunakan minimal 1 metrik internal (mis. *Silhouette Score* atau *Davies-Bouldin*), dan sertakan interpretasi.

6. **Visualisasi:** tampilkan plot hasil (scatter 2D/3D, *cluster centers*, *elbow/silhouette plot*, peta 2D PCA/t-SNE, atau grafik *association rules*).
7. **Kesimpulan & Rekomendasi:** jelaskan temuan, batasan, serta saran perbaikan/lanjutan.

Output/Dokumen yang Dikumpulkan:

1. **Notebook** berisi kode yang *clean & runnable* dengan komentar seperlunya.
2. **Laporan singkat (2-4 halaman):** latar belakang, metode, hasil (tabel/gambar), evaluasi, kesimpulan.
3. **Poster/Slide 1-3 halaman:** ringkasan visual untuk presentasi 5-7 menit.

Reinforcement Learning

A. CPL

Memahami model Reinforcement Learning.

B. CPMK

Membandingkan Algoritma Machine Learning pada Reinforcement Learning.

C. Sub-CPMK

Menentukan algoritma machine learning Reinforcement Learning dan penerapannya yang tepat.

Reinforcement Learning (RL) adalah salah satu cabang dalam **machine learning** yang berfokus pada bagaimana agen (*agent*) belajar untuk mengambil keputusan optimal melalui interaksi dengan lingkungan (*environment*). Proses belajar terjadi berdasarkan prinsip *trial and error* di mana agen menerima **reward** (hadiah) atau **punishment** (hukuman) dari tindakannya. Tujuan utama RL adalah memaksimalkan akumulasi reward jangka panjang.

Pembahasan dalam bab ini akan menguraikan konsep dasar RL, jenis algoritma yang digunakan, tahapan pelatihan agen, teknik evaluasi, hingga perkembangan algoritma RL modern yang relevan dengan masa depan **artificial intelligence (AI)**. Penjelasan disusun secara komprehensif agar pembaca dapat memahami RL tidak hanya sebagai teori, tetapi juga sebagai pendekatan praktis yang mendasari banyak teknologi modern, termasuk robotika, kendaraan otonom, dan permainan komputer cerdas.

Pengertian Model Reinforcement Learning

RL memandang proses pembelajaran sebagai interaksi antara dua komponen utama: **agen** dan **lingkungan**. Agen berfungsi sebagai pengambil keputusan, sementara lingkungan memberikan respon balik terhadap tindakan yang diambil agen. Pada setiap langkah, agen mengamati kondisi lingkungan (*state*), memilih tindakan (*action*), lalu menerima umpan balik berupa **reward** atau **punishment**. Karakteristik penting RL:

1. Belajar dari pengalaman

Karakteristik pertama RL adalah **belajar dari pengalaman**. Agen tidak diberi label yang sudah ditentukan sebelumnya, melainkan harus mengumpulkan pengetahuan melalui interaksi berulang dengan lingkungannya. Proses ini mencerminkan prinsip *learning by doing*, di mana agen menggunakan setiap pengalaman sebagai referensi untuk memperbaiki strategi yang dimilikinya. Hal ini menjadikan RL berbeda dari pendekatan lain karena lebih fleksibel dalam menghadapi masalah nyata yang penuh ketidakpastian.

2. Trial and error

Karakteristik kedua adalah **trial and error**. Agen harus mencoba berbagai tindakan sebelum akhirnya menemukan strategi yang paling optimal. Dalam tahap ini, kesalahan bukanlah kegagalan melainkan bagian dari proses belajar. Melalui *exploration* dan *exploitation*, agen dapat membangun pemahaman yang lebih komprehensif tentang lingkungannya. Konsep ini relevan dengan

kehidupan nyata, di mana keberhasilan sering kali muncul setelah serangkaian percobaan dan koreksi.

3. **Reward jangka panjang**

Karakteristik ketiga adalah fokus pada **reward jangka panjang**. Agen tidak hanya mengejar keuntungan langsung dari setiap tindakan, tetapi juga mempertimbangkan dampak kumulatif di masa depan. Pendekatan ini mencerminkan prinsip *long-term optimization*, yang penting dalam konteks pengambilan keputusan strategis. Dengan demikian, RL mampu menghasilkan kebijakan yang lebih stabil, adaptif, dan berorientasi pada keberlanjutan..

Perbedaan dengan Pendekatan Lain

Bidang **machine learning**, terdapat berbagai pendekatan yang digunakan untuk menyelesaikan masalah. Tiga di antaranya yang paling sering dibahas adalah *supervised learning*, *unsupervised learning*, dan *reinforcement learning*. Masing-masing pendekatan memiliki karakteristik, tujuan, dan mekanisme belajar yang berbeda. Pemahaman atas perbedaan ini sangat penting bagi pembaca agar dapat memilih metode yang paling sesuai dengan konteks permasalahan.

A. Supervised learning

Supervised learning membutuhkan **data berlabel** yang berisi pasangan input dan output. Model dilatih untuk mempelajari hubungan antara keduanya sehingga dapat memprediksi

output baru berdasarkan input yang belum pernah dilihat. Contoh penerapannya adalah klasifikasi gambar, deteksi spam email, dan prediksi harga rumah. Kelebihan pendekatan ini adalah tingkat akurasi tinggi apabila tersedia data berlabel yang cukup. Namun, kelemahannya terletak pada ketergantungan besar terhadap kualitas dan jumlah data berlabel.

B. Unsupervised learning

Berbeda dengan supervised, **unsupervised learning** bekerja dengan **data tanpa label**. Tujuannya adalah menemukan pola tersembunyi atau struktur dalam data. Contoh metode populer adalah *clustering* (pengelompokan) dan *dimensionality reduction*. Penerapannya dapat ditemukan pada segmentasi pasar, analisis perilaku konsumen, atau pengelompokan dokumen. Kelebihannya adalah mampu menganalisis data dalam jumlah besar tanpa memerlukan label, tetapi interpretasi hasil sering kali lebih menantang.

C. Reinforcement learning

Reinforcement learning berbeda dari dua pendekatan sebelumnya karena agen belajar melalui **interaksi dan pengalaman** dengan lingkungannya. Umpan balik diberikan dalam bentuk *reward* atau *punishment*, bukan label eksplisit. Tujuannya adalah memaksimalkan **reward jangka panjang** dengan memilih tindakan yang optimal. Contoh penerapan

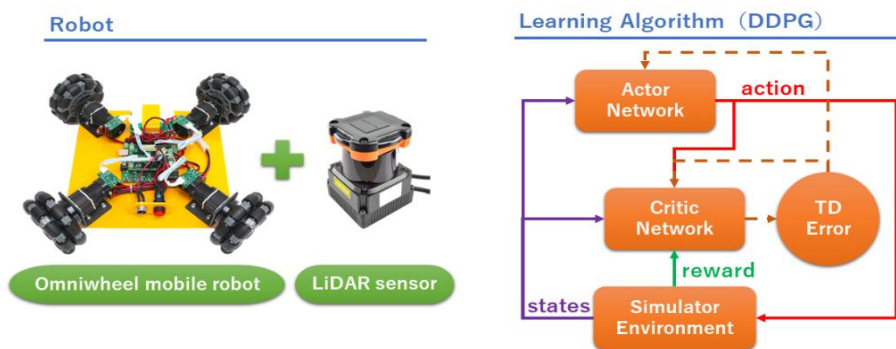
mencakup robotika, *game playing*, hingga kendaraan otonom. Pendekatan ini unggul dalam situasi yang dinamis dan penuh ketidakpastian.

Contoh Aplikasi

Reinforcement Learning (RL) merupakan pendekatan **machine learning** yang semakin banyak digunakan dalam berbagai bidang. Keunggulannya terletak pada kemampuannya untuk belajar dari interaksi dengan lingkungan dan menyesuaikan strategi secara dinamis. Dengan mekanisme umpan balik berbasis *reward*, RL memungkinkan pengembangan sistem yang lebih adaptif, cerdas, dan berorientasi pada hasil jangka panjang.

1. Robotika

Dalam bidang **robotika**, RL digunakan untuk melatih robot agar mampu bergerak secara mandiri dan menghindari rintangan. Robot belajar melalui proses *trial and error* sehingga dapat menemukan pola pergerakan yang paling efisien. Contoh penerapan adalah robot pembersih lantai yang mampu menavigasi ruangan tanpa menabrak perabotan.



Gambar 44. Robotika Navigasi dan Hindari Hambatan

Pembahasan mengenai *Mobile Robot Obstacle Avoidance by Reinforcement Learning* memberikan gambaran komprehensif tentang bagaimana robot dapat bergerak secara otonom sekaligus menghindari hambatan di lingkungannya. Pada ilustrasi terlihat kombinasi **omniwheel mobile robot** dengan **LiDAR sensor**. Sensor berperan sebagai mata yang memindai kondisi sekitar, sementara algoritma pembelajaran berfungsi sebagai otak pengendali.

Algoritma inti yang digunakan adalah *Deep Deterministic Policy Gradient (DDPG)*, sebuah metode dalam *Reinforcement Learning* (RL) yang dirancang untuk menangani ruang aksi kontinu. Prosesnya bekerja ketika robot menerima **states** dari lingkungan, lalu jaringan *Actor* menghasilkan sebuah **action**. Aksi ini dievaluasi oleh *Critic Network* dengan membandingkan hasil prediksi dan kondisi nyata menggunakan **TD Error**. Evaluasi ini melahirkan **reward**, yang memperkuat pemilihan aksi optimal di masa depan. Dengan demikian, terbentuklah pola belajar berbasis *trial and error* yang sistematis dan adaptif.

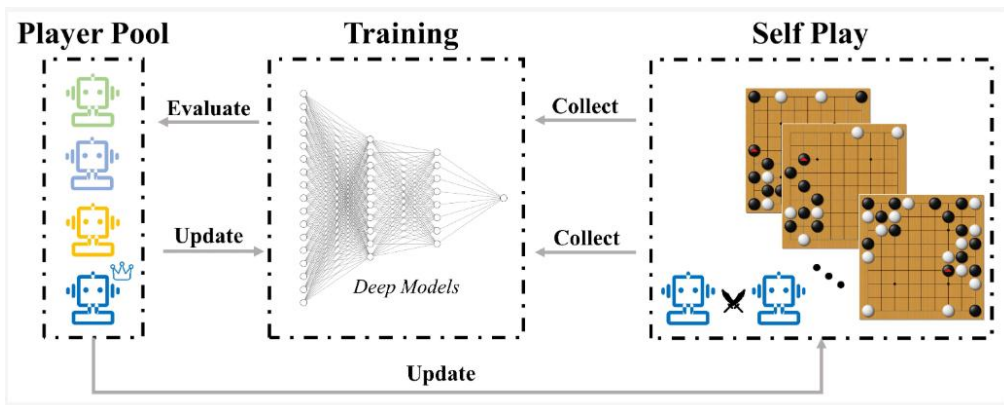
Implikasi penerapan ini cukup luas. Robot yang dilengkapi kemampuan adaptif tidak hanya mampu menghindari tabrakan, tetapi juga menyesuaikan strategi gerak terhadap dinamika lingkungan yang berubah. Dengan kata lain, **robotika berbasis RL memperlihatkan peningkatan nyata dalam efisiensi dan fleksibilitas navigasi** pada berbagai konteks, mulai dari rumah tangga hingga aplikasi industri.

Secara akademik, pendekatan ini menunjukkan pentingnya integrasi **perangkat keras** (robot dan sensor) dengan **perangkat**

lunak cerdas (algoritma RL). Sinergi keduanya melahirkan sistem otonom yang tangguh, menjadikan RL salah satu fondasi utama dalam pengembangan robotika masa depan.

2. Permainan (Games)

Reinforcement Learning telah mencapai popularitas besar dalam bidang **permainan**. Salah satu contoh paling terkenal adalah *AlphaGo* yang dikembangkan oleh DeepMind. Agen RL dalam *AlphaGo* mampu mengalahkan pemain profesional dalam permainan Go, sebuah pencapaian yang sebelumnya dianggap sangat sulit. Keberhasilan ini menunjukkan bahwa RL mampu menangani kompleksitas strategi dalam permainan yang memiliki kemungkinan langkah hampir tak terbatas.



Gambar 45. LR dalam Permainan Go

Reinforcement Learning (RL) telah menjadi salah satu terobosan besar dalam bidang kecerdasan buatan, khususnya pada permainan strategi seperti Go. Proses ini terlihat jelas pada ilustrasi yang menampilkan tiga komponen utama: **Player Pool**, **Training**, dan **Self Play**. Masing-masing elemen ini memiliki

peran yang saling melengkapi dalam menciptakan agen cerdas yang mampu berkembang secara mandiri.

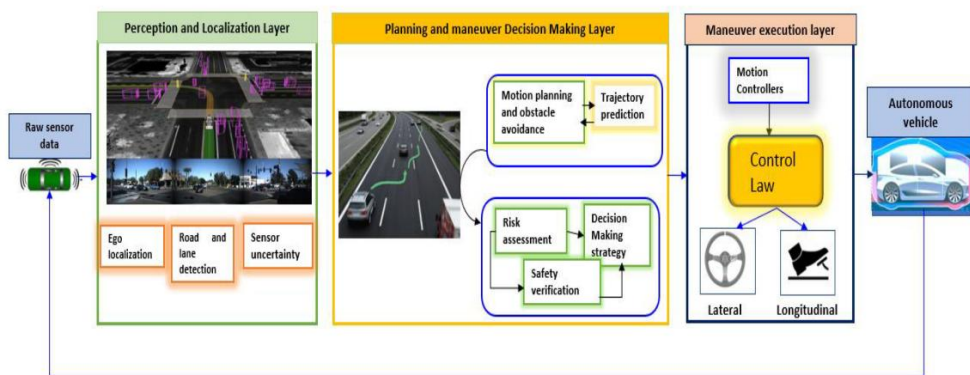
Pada tahap **Self Play**, agen bermain melawan dirinya sendiri. Dari proses ini, data permainan dikumpulkan untuk memperkaya pengalaman agen. Proses *collect* inilah yang menjadi dasar bagi pembaruan strategi di tahap berikutnya. Data tersebut kemudian digunakan dalam bagian **Training**, di mana model jaringan saraf dalam (*Deep Models*) berfungsi untuk menganalisis pola, menentukan strategi, dan meningkatkan pengambilan keputusan.

Setelah model diperbarui, hasil pelatihan dikirim kembali ke **Player Pool** untuk dievaluasi. Pada tahap ini, agen dibandingkan dengan versi sebelumnya untuk memastikan apakah peningkatan benar-benar signifikan. Evaluasi berperan penting menjaga **integritas** dan konsistensi perkembangan agen. Jika performa lebih baik, model baru menggantikan model lama dan kembali digunakan dalam siklus pembelajaran.

Kekuatan *Reinforcement Learning* terletak pada kemampuannya menciptakan loop berkelanjutan antara *self play*, pelatihan, dan evaluasi. Dengan pendekatan ini, agen tidak hanya mengandalkan data statis, tetapi juga mampu menciptakan pengalaman baru melalui eksplorasi berulang. Hal ini menjadikan RL **salah satu pendekatan paling efektif** dalam membangun sistem otonom yang adaptif dan terus berkembang.

3. Kendaraan Otonom

Pada **kendaraan otonom**, RL digunakan untuk mengatur kecepatan, arah, dan pengambilan keputusan berdasarkan kondisi jalan. Mobil belajar menyesuaikan strategi berkendara agar tetap aman, efisien, dan nyaman. Dengan memanfaatkan data sensor dan umpan balik dari lingkungan, kendaraan otonom dapat menghadapi situasi nyata seperti kemacetan atau perubahan cuaca.



Gambar 46. Kendaraan Otonom

Sistem kendaraan otonom bekerja melalui tiga lapisan penting yang saling terhubung untuk memastikan kendaraan dapat bergerak aman dan efisien. Lapisan pertama adalah **perception and localization layer**. Pada bagian ini, data mentah dari sensor digunakan untuk *ego localization*, deteksi jalan serta jalur, dan memperhitungkan ketidakpastian sensor. Insight utama pada tahap ini adalah bahwa **akurasi data sensor** menjadi fondasi utama bagi pengambilan keputusan berikutnya.

Lapisan kedua adalah **planning and maneuver decision making layer**. Di sinilah kendaraan melakukan *motion planning* dan *obstacle avoidance*. Selain itu, sistem juga membuat prediksi

lintasan (*trajectory prediction*), melakukan penilaian risiko (*risk assessment*), serta verifikasi keselamatan. **Proses perencanaan yang adaptif** sangat penting karena kondisi jalan selalu berubah, seperti kendaraan lain yang tiba-tiba berpindah jalur atau adanya rintangan mendadak. Pada konteks ini, *reinforcement learning* (RL) dapat digunakan untuk menyesuaikan strategi pengambilan keputusan. Sebagai contoh, pada kendaraan otonom, RL digunakan untuk mengatur kecepatan, arah, dan strategi manuver berdasarkan kondisi lingkungan secara real-time.

Lapisan terakhir adalah **maneuver execution layer**, yang mengubah strategi menjadi aksi konkret melalui pengendali gerak (*motion controllers*). Kendaraan kemudian melakukan kontrol lateral dan longitudinal dengan mengikuti *control law*. Insight penting pada tahap ini adalah bahwa **stabilitas sistem kontrol** memastikan manuver yang direncanakan dapat dijalankan dengan halus dan aman.

Keseluruhan arsitektur ini memperlihatkan bagaimana integrasi antara sensor, algoritma, dan pengendali membentuk sistem otonom yang andal dan konsisten.

Jenis-Jenis Algoritma pada Reinforcement Learning

A. Value-Based Methods

Metode **Value-Based** merupakan pendekatan yang berfokus pada pembelajaran **fungsi nilai (value function)**. Fungsi ini bertujuan untuk memperkirakan seberapa baik suatu tindakan (*action*) dalam keadaan (*state*) tertentu. Dengan kata lain, metode ini

membantu agen menentukan langkah terbaik berdasarkan estimasi nilai dari setiap kemungkinan tindakan. Dalam konteks *reinforcement learning*, fokus utamanya adalah pada pencarian strategi optimal untuk memaksimalkan *reward* jangka panjang.

1. Q-learning

Q-Learning adalah salah satu algoritma paling populer dalam keluarga *value-based methods*. Algoritma ini bekerja dengan cara memperbarui **tabel Q** berdasarkan pengalaman yang diperoleh agen ketika berinteraksi dengan lingkungan. Setiap pasangan *state-action* akan diberi nilai yang menunjukkan ekspektasi *reward* di masa depan. Meskipun relatif sederhana, algoritma ini sangat efektif dalam berbagai permasalahan optimasi, seperti pengendalian robot maupun sistem rekomendasi dasar. Keunggulan utamanya adalah kemampuannya belajar tanpa perlu mengetahui model lingkungan terlebih dahulu. Dalam algoritma **Q-Learning**, pembaruan nilai Q dilakukan dengan menggunakan persamaan berikut:

Formula

Notasi dasar

Ruang keadaan: $\mathcal{S}$, ruang aksi: $\mathcal{A}$

Fungsi nilai-aksi: $Q(s, a)$

Faktor diskonto: $\gamma \in [0, 1)$

Laju belajar: α

Hadiah: r

Persamaan Optimalitas Bellman untuk nilai-aksi

$$Q^*(s, a) = \mathbb{E} \left[r_{t+1} + \gamma \max_{a' \in \mathcal{A}} Q^*(s_{t+1}, a') \mid s_t = s, a_t = a \right].$$

Aturan pembaruan (*update rule*) Q-Learning (off-policy)

Bentuk inkremental:

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha_t \left(\underset{\text{target}}{r_{t+1} + \gamma \max_{a'} Q_t(s_{t+1}, a')} - Q_t(s_t, a_t) \right).$$

Sebagai interpolasi menuju *target*:

$$Q_{t+1}(s_t, a_t) = (1 - \alpha_t) Q_t(s_t, a_t) + \alpha_t (r_{t+1} + \gamma \max_{a'} Q_t(s_{t+1}, a')).$$

Temporal-Difference (TD) error

$$\delta_t = r_{t+1} + \gamma \max_{a'} Q_t(s_{t+1}, a') - Q_t(s_t, a_t), \quad Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha_t \delta_t.$$

Probabilitas aksi:

$$\pi_\varepsilon(a | s) = (1 - \varepsilon) \mathbf{1}_{a \in \operatorname{argmax}_{a'} Q(s, a')} + \frac{\varepsilon}{|\mathcal{A}|}.$$

Atau pemilihan aksi ringkas:

$$a_t = \begin{cases} \operatorname{argmax}_{a \in \mathcal{A}} Q(s_t, a), & \text{dengan probabilitas } 1 - \varepsilon, \\ \text{pilih aksi seragam dari } \mathcal{A}, & \text{dengan probabilitas } \varepsilon. \end{cases}$$

Kondisi laju belajar (*Robbins-Monro*) untuk konvergensi tabular

Jika setiap pasangan (s, a) dikunjungi tak hingga kali dan laju belajar memenuhi

$$\sum_{t=0}^{\infty} \alpha_t = \infty \text{ dan } \sum_{t=0}^{\infty} \alpha_t^2 < \infty,$$

serta $0 \leq \gamma < 1$, maka Q_t konvergen ke Q^* hampir pasti (untuk MDP terhingga).

Penanganan keadaan terminal

Untuk keadaan terminal s_{t+1} (episode berakhir), gunakan

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha_t (r_{t+1} - Q_t(s_t, a_t)),$$

karena tidak ada *bootstrapping* lanjut ($\max_{a'} Q(\cdot)$) setelah terminal.

Kebijakan deterministik:

$$\pi^*(s) = \operatorname{argmax}_{a \in \mathcal{A}} Q^*(s, a).$$

Kebijakan serakah terhadap Q_t :

$$\pi_t(s) = \operatorname{argmax}_{a \in \mathcal{A}} Q_t(s, a).$$

Variasi umum: *Double Q-Learning* (opsional)

Untuk mengurangi *overestimation bias*:

$$\begin{aligned} Q_{t+1}^A(s_t, a_t) &= Q_t^A(s_t, a_t) + \alpha_t \left(r_{t+1} + \gamma Q_t^B(s_{t+1}, \operatorname{argmax}_{a'} Q_t^A(s_{t+1}, a')) - Q_t^A(s_t, a_t) \right), \\ Q_{t+1}^B(s_t, a_t) &= Q_t^B(s_t, a_t) + \alpha_t \left(r_{t+1} + \gamma Q_t^A(s_{t+1}, \operatorname{argmax}_{a'} Q_t^B(s_{t+1}, a')) - Q_t^B(s_t, a_t) \right). \end{aligned}$$

Mean Squared TD Error (diagnostik)

$$\text{MSTDE} = \mathbb{E}[\delta_t^2].$$

2. Deep Q-Network (DQN)

Seiring meningkatnya kompleksitas lingkungan dengan ruang keadaan (*state space*) yang besar, keterbatasan Q-Learning menjadi nyata. Untuk mengatasi hal ini, dikembangkanlah **Deep Q-Network (DQN)**. Algoritma ini memanfaatkan **neural networks** untuk memperkirakan fungsi nilai secara lebih efisien. Dengan menggunakan *deep learning*, DQN mampu menangani permasalahan yang sebelumnya tidak dapat diselesaikan hanya dengan tabel Q. Misalnya, DQN telah berhasil diaplikasikan dalam permainan video yang kompleks, di mana agen mampu mengembangkan strategi yang mendekati manusia.

Formula

Aproksimasi fungsi nilai-aksi dengan jaringan saraf

$$Q(s, a; \theta) \approx Q^*(s, a)$$

dengan parameter jaringan saraf θ .

Target pembelajaran (loss target)

Pada iterasi ke- i :

$$y_i = r_{i+1} + \gamma \max_{a'} Q(s_{i+1}, a'; \theta^-)$$

dengan θ^- parameter jaringan *target* (salinan θ yang diperbarui secara periodik).

Fungsi kerugian (*loss function*)

$$L_i(\theta_i) = \mathbb{E}_{(s,a,r,s') \sim D} [(y_i - Q(s, a; \theta_i))^2]$$

dengan D adalah *experience replay buffer*.

Aturan pembaruan parameter jaringan

Gradient descent:

$$\theta_{i+1} = \theta_i + \alpha \nabla_{\theta_i} L_i(\theta_i)$$

atau lebih eksplisit:

$$\theta_{i+1} = \theta_i + \alpha (y_i - Q(s, a; \theta_i)) \nabla_{\theta_i} Q(s, a; \theta_i).$$

Definisi *Experience Replay*

Sampel pengalaman disimpan sebagai tupel:

$$(s_t, a_t, r_{t+1}, s_{t+1}) \in D.$$

Mini-batch acak diambil dari D untuk melatih jaringan, mengurangi korelasi antar-sampel.

$$a_t = \begin{cases} \arg \max_{a'} Q(s_t, a'; \theta), & \text{dengan probabilitas } 1 - \varepsilon, \\ \text{pilih aksi acak dari } \mathcal{A}, & \text{dengan probabilitas } \varepsilon. \end{cases}$$

Mekanisme jaringan target (*target network*)

Parameter jaringan target diperbarui setiap C langkah:

$$\theta^- \leftarrow \theta.$$

B. Policy-Based Methods

Policy didefinisikan sebagai distribusi probabilitas yang memetakan **state** ke **action**. Dengan kata lain, metode ini mencari

parameter optimal dari fungsi kebijakan $\pi_{\theta}(a | s)$ sehingga ekspektasi total hadiah yang diterima agen dapat dimaksimalkan. Keunggulan utama dari pendekatan ini adalah kemampuannya untuk menghasilkan aksi secara langsung tanpa harus melalui pencarian nilai maksimum dari fungsi nilai.

1. Reinforce

Algoritma **REINFORCE** merupakan salah satu metode klasik dalam keluarga *policy gradient*. Algoritma ini menggunakan pendekatan *Monte Carlo* untuk menghitung gradien kebijakan berdasarkan episode penuh. Pembaruan kebijakan dilakukan dengan memperkuat aksi yang menghasilkan hadiah lebih besar. Meskipun sederhana, REINFORCE memiliki kelemahan berupa varians tinggi sehingga konvergensi bisa lambat.

Formula

Fungsi objektif (expected return)

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_{\theta}}[R(\tau)]$$

dengan $\tau = (s_0, a_0, s_1, a_1, \dots)$ adalah lintasan, dan $R(\tau) = \sum_{t=0}^T \gamma^t r_t$.

Gradien kebijakan (*Policy Gradient Theorem*)

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{\pi_{\theta}}[\nabla_{\theta} \log \pi_{\theta}(a_t | s_t) R_t]$$

dengan $R_t = \sum_{k=t}^T \gamma^{k-t} r_k$ adalah *return* mulai dari waktu t .

Aturan pembaruan parameter

$$\theta \leftarrow \theta + \alpha \nabla_{\theta} J(\theta)$$

atau lebih eksplisit:

$$\theta \leftarrow \theta + \alpha \sum_{t=0}^T \nabla_{\theta} \log \pi_{\theta}(a_t | s_t) R_t.$$

Versi dengan *baseline* (mengurangi varians)

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{\pi_{\theta}} [\nabla_{\theta} \log \pi_{\theta}(a_t | s_t) (R_t - b(s_t))]$$

dengan $b(s_t)$ biasanya dipilih sebagai *state-value function* $V^{\pi}(s_t)$.

Loss function yang setara

Sering ditulis sebagai minimisasi *loss*:

$$L(\theta) = -\mathbb{E}_{\pi_{\theta}} [\log \pi_{\theta}(a_t | s_t) R_t].$$

2. Proximal policy optimization (PPO)

Proximal Policy Optimization (PPO) adalah pengembangan metode gradien kebijakan yang dirancang untuk menjaga **stabilitas pembaruan**. Dengan memperkenalkan fungsi objektif terklip (*clipped objective function*), PPO mampu menghindari perubahan kebijakan yang terlalu drastis. Algoritma ini menjadi sangat populer dalam penelitian maupun aplikasi praktis karena keseimbangan antara kesederhanaan, stabilitas, dan efisiensi.

Formula

Rasio probabilitas (*probability ratio*)

$$r_t(\theta) = \frac{\pi_{\theta}(a_t | s_t)}{\pi_{\theta_{\text{old}}}(a_t | s_t)}.$$

Clipped Surrogate Objective (maksimisasi)

$$\mathcal{L}_{\text{CLIP}}(\theta) = \mathbb{E}_t \left[\min \left(r_t(\theta) \hat{A}_t, \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon) \hat{A}_t \right) \right].$$

$$\delta_t = r_t + \gamma V_\phi(s_{t+1}) - V_\phi(s_t), \hat{A}_t = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}.$$

Value loss (kuadrat galat nilai)

$$\mathcal{L}_{VF}(\phi) = \mathbb{E}_t[(V_\phi(s_t) - \hat{V}_t)^2] \text{ (sering: } \hat{V}_t = \hat{A}_t + V_\phi(s_t) \text{ atau *MC return*.)}$$

Entropy bonus (eksplorasi)

$$\mathcal{L}_{ENT}(\theta) = \mathbb{E}_t[\mathcal{H}[\pi_\theta(\cdot | s_t)]].$$

Objektif total (untuk dimaksimalkan)

$$\mathcal{L}_{PPO}(\theta, \phi) = \mathcal{L}_{CLIP}(\theta) - c_1 \mathcal{L}_{VF}(\phi) + c_2 \mathcal{L}_{ENT}(\theta).$$

Dalam implementasi berbasis minimisasi, gunakan

$$\text{Loss} = -\mathcal{L}_{PPO}$$

Alternatif: penalti KL (*KL-penalty objective*)

$$\mathcal{L}_{KL-PEN}(\theta) = \mathbb{E}_t[r_t(\theta) \hat{A}_t - \beta \text{KL}(\pi_{\theta_{\text{old}}}(\cdot | s_t) \parallel \pi_\theta(\cdot | s_t))].$$

Batas langkah rata-rata (target-KL, kriteria *early stop*)

$$\bar{\text{KL}}(\theta_{\text{old}}, \theta) = \mathbb{E}_t[\text{KL}(\pi_{\theta_{\text{old}}}(\cdot | s_t) \parallel \pi_\theta(\cdot | s_t))] \leq \text{KL}_{\text{target}}.$$

Pembaruan parameter (skema umum)

Kebijakan:

$$\theta \leftarrow \arg \max_{\theta} \hat{\mathbb{E}}[\min(r_t(\theta) \hat{A}_t, \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon) \hat{A}_t)].$$

Kritis nilai:

$$\phi \leftarrow \arg \min_{\phi} \hat{\mathbb{E}}[(V_\phi(s_t) - \hat{V}_t)^2].$$

Normalisasi *advantage* (opsional tapi umum)

$$\tilde{A}_t = \frac{\hat{A}_t - \mu_A}{\sigma_A}, \mu_A = \text{mean}(\hat{A}), \sigma_A = \text{std}(\hat{A}).$$

C. Actor-Critic Methods

Pada pendekatan ini, **aktor** merepresentasikan kebijakan $\pi_\theta(a | s)$ yang dioptimalkan secara langsung, sedangkan **kritikus**

memperkirakan fungsi nilai, biasanya $V^\pi(s)$ atau fungsi keunggulan (*advantage function*). Proses pembaruan parameter dilakukan dengan memanfaatkan umpan balik dari kritikus sehingga kebijakan dapat diperbaiki secara lebih stabil. Hal ini mengurangi varians tinggi yang umumnya muncul pada metode *policy-based* murni seperti *REINFORCE*.

1. Advantage Actor Critic (A2C)

Advantage Actor-Critic (A2C) merupakan salah satu varian yang populer. Algoritma ini menghitung **advantage** $A(s, a) = Q(s, a) - V(s)$ untuk mengukur seberapa baik suatu tindakan dibandingkan dengan rata-rata nilai keadaan tersebut. Dengan memanfaatkan keunggulan relatif ini, A2C mampu mempercepat proses pembelajaran sekaligus menjaga stabilitas konvergensi.

Formula

Kebijakan dan nilai

Kebijakan parameterik: $\pi_\theta(a | s)$

Fungsi nilai: $V_\phi(s)$

Return n-langkah (target nilai)

Untuk horizon n dan diskonto $\gamma \in [0,1)$:

$$G_t^{(n)} = \sum_{k=0}^{n-1} \gamma^k r_{t+k} + \gamma^n V_\phi(s_{t+n}).$$

Advantage (estimasi)

$$\hat{A}_t = G_t^{(n)} - V_\phi(s_t).$$

(Opsional lain: normalisasi $\tilde{A}_t = (\hat{A}_t - \mu_A) / \sigma_A$).

Gradien aktor (policy gradient)

Bentuk ekspektasi:

$$\nabla_{\theta} J(\theta) = \mathbb{E}[\nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \hat{A}_t].$$

Pembaruan stokastik (satu sampel atau mini-batch):

$$\theta \leftarrow \theta + \alpha_{\pi} \nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \hat{A}_t.$$

Entropy bonus (eksplorasi)

$$\mathcal{H}(\pi_{\theta}(\cdot | s_t)) = - \sum_a \pi_{\theta}(a | s_t) \log \pi_{\theta}(a | s_t)$$

(diskret; kontinu gunakan entropi diferensial).

Loss kritikus (mean squared value error)

$$\mathcal{L}_V(\phi) = \mathbb{E}[(G_t^{(n)} - V_{\phi}(s_t))^2].$$

Objektif total A2C (untuk dimaksimalkan)

$$\mathcal{L}_{A2C}(\theta, \phi) = \mathbb{E}[\underbrace{\log \pi_{\theta}(a_t | s_t) \hat{A}_t}_{\text{aktor}} - c_v (G_t^{(n)} - V_{\phi}(s_t))^2 + c_H \mathcal{H}(\pi_{\theta}(\cdot | s_t))].$$

Pada implementasi berbasis minimisasi, gunakan Loss = $-\mathcal{L}_{A2C}$.

Bentuk kontinu (Gaussian policy; opsional)

Dengan $\pi_{\theta}(a | s) = \mathcal{N}(a; \mu_{\theta}(s), \Sigma_{\theta}(s))$:

$$\log \pi_{\theta}(a | s) = -\frac{1}{2}[(a - \mu_{\theta}(s))^{\top} \Sigma_{\theta}(s)^{-1} (a - \mu_{\theta}(s)) + \log |\Sigma_{\theta}(s)| + d \log (2\pi)].$$

Gradien aktor tetap $\nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \hat{A}_t$.

Skema sinkron (A2C klasik)

Kumpulkan rollout dari beberapa *worker* secara **sinkron**, lalu lakukan langkah optimisasi bersama:

$$\{\theta, \phi\} \leftarrow \arg \max_{\theta, \phi} \hat{\mathbb{E}}[\mathcal{L}_{A2C}(\theta, \phi)].$$

2. Deep Deterministic Policy Gradient (DDPG)

Deep Deterministic Policy Gradient (DDPG) adalah pengembangan Actor-Critic yang ditujukan untuk ruang aksi

kontinu. Algoritma ini menggabungkan *deterministic policy gradient* dengan jaringan saraf dalam, sehingga sangat cocok untuk permasalahan kontrol robotika dan sistem dinamis yang kompleks. DDPG memungkinkan agen untuk menghasilkan aksi presisi tinggi tanpa harus mendiskretkan ruang aksi yang besar.

Formula

Kebijakan dan nilai

Kebijakan parameterik: $\pi_{\theta}(a | s)$

Fungsi nilai: $V_{\phi}(s)$

Return n-langkah (target nilai)

Untuk horizon n dan diskonto $\gamma \in [0,1)$:

$$G_t^{(n)} = \sum_{k=0}^{n-1} \gamma^k r_{t+k} + \gamma^n V_{\phi}(s_{t+n}).$$

Advantage (estimasi)

$$\hat{A}_t = G_t^{(n)} - V_{\phi}(s_t).$$

(Opsional lain: normalisasi $\tilde{A}_t = (\hat{A}_t - \mu_A)/\sigma_A$).

Gradien aktor (policy gradient)

Bentuk ekspektasi:

$$\nabla_{\theta} J(\theta) = \mathbb{E}[\nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \hat{A}_t].$$

Pembaruan stokastik (satu sampel atau mini-batch):

$$\theta \leftarrow \theta + \alpha_{\pi} \nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \hat{A}_t.$$

Entropy bonus (eksplorasi)

$$\mathcal{H}(\pi_{\theta}(\cdot | s_t)) = - \sum_a \pi_{\theta}(a | s_t) \log \pi_{\theta}(a | s_t)$$

(diskret; kontinu gunakan entropi diferensial).

Loss kritikus (mean squared value error)

$$\mathcal{L}_V(\phi) = \mathbb{E}[(G_t^{(n)} - V_\phi(s_t))^2].$$

Objektif total A2C (untuk dimaksimalkan)

$$\begin{aligned} \mathcal{L}_{\text{A2C}}(\theta, \phi) = & \mathbb{E}[\log \pi_{\theta}(a_t | s_t) \hat{A}_t - c_v (G_t^{(n)} - V_\phi(s_t))^2 \\ & \text{aktor} \\ & + c_H \mathcal{H}(\pi_{\theta}(\cdot | s_t))]. \end{aligned}$$

Pada implementasi berbasis minimisasi, gunakan Loss = $-\mathcal{L}_{\text{A2C}}$

Bentuk kontinu (Gaussian policy; opsional)

Dengan $\pi_{\theta}(a | s) = \mathcal{N}(a; \mu_{\theta}(s), \Sigma_{\theta}(s))$:

$$\begin{aligned} \log \pi_{\theta}(a | s) = & -\frac{1}{2}[(a - \mu_{\theta}(s))^{\top} \Sigma_{\theta}(s)^{-1} (a - \mu_{\theta}(s)) + \log |\Sigma_{\theta}(s)| \\ & + d \log (2\pi)]. \end{aligned}$$

Gradien aktor tetap $\nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \hat{A}_t$.

Skema sinkron (A2C klasik)

Kumpulkan rollout dari beberapa *worker* secara **sinkron**, lalu lakukan langkah optimisasi bersama:

$$\{\theta, \phi\} \leftarrow \arg \max_{\theta, \phi} \hat{\mathbb{E}}[\mathcal{L}_{\text{A2C}}(\theta, \phi)].$$

Tabel 12. Perbandingan Algoritma

Pendekatan	Contoh Algoritma	Kelebihan	Kekurangan
<i>Value-Based</i>	Q-Learning, DQN	Konsep sederhana, efektif untuk masalah diskrit	Sulit menangani <i>continuous action space</i>
<i>Policy-Based</i>	REINFORCE, PPO	Fleksibel, baik untuk <i>continuous control</i>	Membutuhkan data banyak, risiko variansi tinggi

Pendekatan	Contoh Algoritma	Kelebihan	Kekurangan
<i>Actor-Critic</i>	A2C, DDPG	Kombinasi kelebihan dua pendekatan	Lebih kompleks, memerlukan sumber daya komputasi besar

Tahapan dalam Reinforcement Learning

Tahapan implementasi RL bersifat siklus dan berulang, meliputi:

A. Definisi Lingkungan (*Environment*)

Tahap awal implementasi **Reinforcement Learning (RL)** adalah mendefinisikan **lingkungan** dengan jelas dan terstruktur. Lingkungan mencakup himpunan **keadaan (states)** serta **tindakan (actions)** yang dapat dipilih agen. Lingkungan dapat berbentuk simulasi komputer sederhana, misalnya permainan papan, hingga konteks dunia nyata seperti **robotika, jaringan komunikasi**, atau sistem **manajemen energi**. Definisi yang tepat sangat penting karena akan menentukan ruang lingkup eksplorasi agen sekaligus memengaruhi kualitas hasil pembelajaran.

B. Interaksi Agen dengan Lingkungan

Setelah lingkungan didefinisikan, **agen** mulai melakukan eksplorasi dengan mengambil tindakan berdasarkan **kebijakan (policy)**. Pada awal proses, tindakan sering dipilih secara acak (*exploration*) agar agen memperoleh gambaran luas tentang lingkungan. Seiring berjalannya waktu, agen melakukan **eksploitasi** pengetahuan yang diperoleh untuk

memilih tindakan yang lebih **optimal**. Proses ini mencerminkan keseimbangan antara *exploration* dan *exploitation* dalam RL

C. Feedback dari Lingkungan

Setiap tindakan menghasilkan **umpan balik** berupa *reward* positif atau negatif. *Reward* berfungsi sebagai indikator keberhasilan atau kegagalan dari suatu tindakan. Jika tindakan mendekatkan agen pada tujuan, *reward* bernilai positif, sedangkan jika menjauh, bernilai negatif. Mekanisme ini penting karena menjadi dasar evaluasi efektivitas kebijakan yang sedang dijalankan.

D. Update Policy

Tahap ini adalah inti dari pembelajaran. **Kebijakan diperbarui** dengan mempertimbangkan *reward* yang diterima serta prediksi nilai ke depan. Pendekatan yang umum digunakan mencakup **value function**, **policy gradient**, atau kombinasi keduanya. Tujuannya adalah membentuk kebijakan yang semakin efektif sehingga agen mampu menyesuaikan diri dengan dinamika lingkungan yang kompleks.

E. Evaluasi Agen

Evaluasi dilakukan untuk menilai **kinerja agen** dalam memaksimalkan *reward* jangka panjang. Proses ini biasanya mencakup pengukuran performa melalui metrik tertentu, misalnya akurasi strategi, efisiensi sumber daya, atau kecepatan konvergensi. Evaluasi berkala memastikan bahwa

agen tidak hanya belajar, tetapi juga menerapkan pembelajaran dengan konsisten dan adaptif.

Contoh Tahapan RL

A. Pengamatan Awal (*State Observation*)

Tahap pertama dimulai ketika **agen** mengamati kondisi lingkungan atau **state (S)**. Pengamatan ini berfungsi sebagai titik awal pembelajaran, karena tanpa pemahaman terhadap kondisi saat ini, agen tidak dapat mengambil langkah strategis. Misalnya, dalam permainan catur, *state* mencerminkan posisi semua bidak di papan. Pada konteks lain seperti robotika, *state* dapat berupa posisi sensor atau data lingkungan. Pemahaman yang akurat terhadap *state* menjadi dasar pengambilan keputusan yang rasional dan efektif.

B. Pemilihan Tindakan (*Action Selection*)

Setelah memahami kondisi, agen harus menentukan **tindakan (action/A)**. Pada fase awal, tindakan bisa bersifat acak sebagai bagian dari *exploration*, sementara pada tahap lebih lanjut agen memanfaatkan pengalaman sebelumnya untuk *exploitation*. Keseimbangan antara *exploration* dan *exploitation* menjadi faktor kunci agar agen tidak hanya mengulang strategi lama, tetapi juga menemukan pola baru yang potensial. Hal ini mencerminkan prinsip penting dalam *reinforcement learning*.

C. Respon Lingkungan (*Reward and New State*)

Setiap tindakan memicu respon dari lingkungan berupa **reward (R)** dan menghasilkan **state baru (S')**. Reward berfungsi sebagai sinyal evaluatif: positif jika tindakan

mendekatkan agen pada tujuan, negatif jika menjauh. State baru yang muncul menandai kelanjutan proses pembelajaran, menciptakan siklus interaksi yang berkesinambungan antara agen dan lingkungan.

D. Pembaruan Kebijakan (*Policy Update*)

Berdasarkan pengalaman, agen melakukan **pembaruan kebijakan (*policy update*)**. Proses ini dapat ditempuh melalui pendekatan **value-based**, **policy gradient**, atau model hibrida. Intinya adalah menyaring pengalaman historis menjadi aturan keputusan yang lebih optimal. Semakin sering siklus berjalan, semakin matang pula kebijakan yang dihasilkan.

E. Proses Iteratif Menuju Optimal

Langkah-langkah ini berlangsung secara **iteratif** hingga tercapai **kebijakan optimal**. Melalui iterasi, agen dapat meningkatkan kualitas keputusan, memperbaiki efisiensi, serta memaksimalkan *reward* jangka panjang. Sifat siklus ini menjadikan *reinforcement learning* fleksibel dan mampu beradaptasi dengan berbagai situasi yang dinamis.

Tahapan ini memperlihatkan bahwa **Reinforcement Learning** bukan sekadar proses linier, melainkan suatu siklus adaptif. Melalui observasi, pemilihan tindakan, respon lingkungan, pembaruan kebijakan, dan iterasi berulang, agen secara bertahap membangun strategi yang matang. Proses ini relevan baik untuk simulasi sederhana maupun aplikasi nyata yang kompleks.

Teknik Evaluasi dalam *Reinforcement Learning*

Evaluasi dalam **Reinforcement Learning (RL)** merupakan langkah penting untuk memastikan bahwa **agen** benar-benar belajar secara efektif. Tanpa evaluasi yang tepat, sulit menilai apakah kebijakan (*policy*) yang dihasilkan telah optimal atau masih memerlukan perbaikan. Beberapa teknik evaluasi yang umum digunakan dapat dijelaskan sebagai berikut:

A. **Cumulative reward**

Cumulative reward mengacu pada jumlah total *reward* yang diperoleh agen selama periode tertentu. Indikator ini memberikan gambaran langsung mengenai sejauh mana agen berhasil mencapai tujuan yang ditetapkan. Semakin tinggi cumulative reward, semakin baik performa agen dalam jangka panjang.

Formula

Cumulative reward mengacu pada jumlah total *reward* yang diperoleh agen selama periode tertentu. Formula matematisnya dapat dituliskan sebagai berikut:

$$G_T = \sum_{t=0}^T r_t$$

Dimana:

G_T : Nilai *cumulative reward* hingga horizon waktu T .

r_t : *Reward* yang diterima agen pada waktu ke- t .

T : Jumlah langkah atau periode waktu evaluasi.

Persamaan ini menjumlahkan semua *reward* yang diterima agen dari awal interaksi hingga periode ke- T . Semakin besar nilai G_T ,

semakin baik performa agen dalam mencapai tujuan yang telah ditetapkan.

B. Average reward

Average reward menghitung rata-rata *reward* per episode. Indikator ini menekankan aspek **konsistensi** dari performa agen. Jika nilai rata-rata stabil dalam berbagai episode, maka dapat disimpulkan bahwa agen memiliki strategi yang dapat diandalkan.

Formula

Average reward menghitung rata-rata *reward* per episode. Indikator ini digunakan untuk menilai **konsistensi** dari performa agen sepanjang proses pembelajaran. Formula matematisnya dituliskan sebagai berikut:

$$\bar{R} = \frac{1}{N} \sum_{i=1}^N R_i$$

Dimana:

$\bar{R}$: Nilai *average reward*.

N : Jumlah episode yang diamati.

R_i : Total *reward* yang diperoleh pada episode ke- i .

Persamaan ini membagi jumlah total *reward* dari seluruh episode dengan jumlah episode N . Nilai yang konsisten pada $\bar{R}$ menunjukkan bahwa agen memiliki kebijakan yang stabil dan dapat diandalkan dalam berbagai kondisi pelatihan.

C. Convergence rate

Convergence rate mengukur seberapa cepat agen mencapai **kebijakan optimal**. Agen yang mampu beradaptasi dan menemukan solusi optimal dengan cepat menunjukkan efisiensi algoritma yang mendasari proses pembelajaran.

Formula

Convergence rate digunakan untuk mengukur seberapa cepat **agen** mencapai **kebijakan optimal** dalam *reinforcement learning*. Formula matematis yang umum digunakan adalah:

$$C = \frac{1}{T_{conv}}$$

Dimana:

C : *Convergence rate*.

T_{conv} : Jumlah langkah (atau episode) yang diperlukan hingga agen mencapai konvergensi, yaitu ketika kebijakan relatif stabil dan mendekati optimal.

Persamaan ini menunjukkan bahwa semakin sedikit waktu atau episode yang dibutuhkan untuk mencapai konvergensi (T_{conv}), maka nilai C semakin besar. Hal ini menandakan bahwa agen mampu belajar secara lebih **efisien** dan cepat dalam menemukan kebijakan optimal.

1. Stability

Stability menggambarkan konsistensi agen selama pelatihan. Agen yang stabil akan menunjukkan performa yang tidak mudah terpengaruh oleh variasi kecil dalam lingkungan. Hal ini penting

agar kebijakan yang terbentuk bersifat **robust** dan dapat dipertahankan.

Formula

Stability digunakan untuk menggambarkan **konsistensi agen** selama proses pelatihan dalam *reinforcement learning*. Salah satu pendekatan matematis untuk mengukur stabilitas adalah dengan menggunakan **varian dari reward**:

$$S = 1 - \frac{Var(R)}{E[R]}$$

Dimana:

S : Indeks *stability*.

$Var(R)$: Variansi dari *reward* selama beberapa episode.

$E[R]$: Nilai harapan (*expected value*) dari *reward*.

Jika variansi *reward* kecil, maka nilai **stability (S)** mendekati 1, yang menunjukkan bahwa agen memiliki performa yang konsisten. Sebaliknya, variansi yang besar akan menurunkan stabilitas, menandakan bahwa agen masih fluktuatif dalam proses pembelajaran.

2. Generalization

Generalization menilai kemampuan agen menerapkan kebijakan pada **lingkungan baru** yang belum pernah dihadapi sebelumnya. Evaluasi ini memastikan bahwa agen tidak hanya unggul di lingkungan pelatihan, tetapi juga adaptif terhadap situasi berbeda.

Formula

Generalization menilai kemampuan agen untuk menerapkan **kebijakan** yang telah dipelajari pada **lingkungan baru** yang belum pernah dihadapi sebelumnya. Secara matematis, kemampuan generalisasi dapat diekspresikan melalui perbandingan performa antara lingkungan pelatihan (*training environment*) dan lingkungan uji (*test environment*):

$$G = \frac{E[R_{test}]}{E[R_{train}]}$$

Dimana:

G : Indeks *generalization*.

$E[R_{test}]$: Nilai harapan (*expected reward*) agen pada lingkungan uji.

$E[R_{train}]$: Nilai harapan (*expected reward*) agen pada lingkungan pelatihan.

Jika nilai $G \approx 1$, maka performa agen pada lingkungan uji hampir sama dengan lingkungan pelatihan, menandakan kemampuan generalisasi yang baik. Sebaliknya, nilai $G < 1$ menunjukkan adanya penurunan performa, yang berarti agen masih mengalami kesulitan dalam beradaptasi dengan kondisi baru.

Contoh, pada simulasi mobil otonom, cumulative reward mencerminkan kemampuan mobil menyelesaikan perjalanan tanpa kecelakaan, sementara convergence rate menunjukkan seberapa cepat mobil belajar mengemudi dengan baik.

Algoritma *Reinforcement Learning* untuk AI Masa Depan

A. Deep Reinforcement Learning (DRL)

Deep Reinforcement Learning (DRL) adalah pendekatan yang menggabungkan kekuatan **deep learning** dengan prinsip *reinforcement learning*. Salah satu contoh populer adalah **Deep Q-Network (DQN)** yang sukses digunakan dalam menyelesaikan permainan Atari. Kelebihan DRL terletak pada kemampuannya mengatasi **ruang keadaan yang besar dan kompleks**, sehingga dapat diaplikasikan pada berbagai bidang mulai dari permainan digital, navigasi otonom, hingga analisis data berskala besar.

B. Proximal Policy Optimization (PPO)

Proximal Policy Optimization (PPO) merupakan algoritma modern yang menekankan keseimbangan antara **stabilitas** dan **efisiensi**. Dengan pendekatan berbasis *policy gradient*, PPO mampu memperbarui kebijakan tanpa menimbulkan perubahan ekstrem yang dapat mengganggu proses pembelajaran. Algoritma ini sering digunakan dalam **simulasi robotika** maupun permainan strategis yang kompleks, karena menawarkan performa tinggi dengan stabilitas yang lebih baik dibandingkan metode sebelumnya.

C. Deep Deterministic Policy Gradient (DDPG)

Untuk masalah yang melibatkan **continuous action space**, **Deep Deterministic Policy Gradient (DDPG)** menjadi pilihan

yang tepat. Algoritma ini sering diaplikasikan pada **robotika industri** yang membutuhkan kontrol presisi, seperti pengaturan lengan robot atau sistem otomasi manufaktur. Dengan memadukan *actor-critic architecture*, DDPG memungkinkan agen belajar strategi kontrol yang halus dan efisien.

D. AlphaZero-like Algorithms

Pendekatan **AlphaZero** dan turunannya menunjukkan kekuatan *reinforcement learning* dalam mengembangkan **Artificial General Intelligence (AGI)**. Melalui mekanisme *self-play* dan integrasi *deep reinforcement learning*, AlphaZero mampu mengalahkan pemain terbaik dunia dalam permainan catur dan Go. Algoritma ini membuktikan potensi RL untuk menghasilkan sistem cerdas yang dapat belajar tanpa supervisi langsung manusia.

Soal Uraian

1. Jelaskan pengertian *Reinforcement Learning* dan bedakan dengan *supervised learning* serta *unsupervised learning*.
2. Sebutkan dan jelaskan jenis-jenis algoritma utama pada *Reinforcement Learning*.
3. Mengapa konsep reward sangat penting dalam *Reinforcement Learning*? Berikan contoh penerapannya.
4. Jelaskan tahapan proses pembelajaran dalam model *Reinforcement Learning*.

5. Apa perbedaan antara *value-based method*, *policy-based method*, dan *actor-critic method*?
6. Bagaimana cara mengukur keberhasilan agen dalam *Reinforcement Learning*? Sebutkan teknik evaluasi yang relevan.
7. Berikan contoh nyata penerapan *Reinforcement Learning* dalam kehidupan sehari-hari.
8. Mengapa *Deep Reinforcement Learning* dianggap potensial dalam pengembangan Artificial Intelligence di masa depan?
9. Jelaskan peran algoritma *Proximal Policy Optimization (PPO)* dan keunggulannya dibandingkan algoritma lain.
10. Menurut Anda, bagaimana algoritma *AlphaZero* dapat menginspirasi pengembangan kecerdasan buatan umum (*Artificial General Intelligence / AGI*)?

Tugas Kelompok

Tujuan:

Menerapkan konsep RL dari Materi D pada studi kasus sederhana yang mudah dipahami.

Membandingkan hasil, menganalisis parameter, dan menyusun laporan ringkas.

Format Kelompok: 3-4 mahasiswa per kelompok.

Pilihan Studi Kasus (pilih salah satu):

1. Maze Solving

Agen belajar keluar dari labirin.

2. CartPole-v1 (Gymnasium)

Agen menyeimbangkan tiang di atas kereta.

3. **Tic-Tac-Toe**

Agen belajar strategi bermain.

4. **FrozenLake-v1 (Gymnasium)**

Dunia grid es licin; agen belajar mencapai goal tanpa jatuh ke lubang.

5. **Taxi-v3 (Gymnasium)**

Agen taksi menjemput & mengantar penumpang ke tujuan secara optimal.

Catatan: pilih lingkungan *discrete state-action* agar implementasi lebih mudah.

Perangkat & Paket yang Disarankan:

Python 3.10+, gymnasium (atau gym), numpy, matplotlib.

(Opsional untuk eksplorasi lanjutan) stable-baselines3 untuk PPO.

Langkah Pengerjaan (minimal)

1. **Definisikan MDP**

Jelaskan *state*, *action*, *reward*, *transition*.

2. **Inisialisasi**

Buat tabel-Q dengan nol, set *hyperparameter* (α , γ , ϵ , episode).

3. **Pelatihan**

Implementasi Q-Learning atau SARSA dengan ϵ -greedy.

4. **Evaluasi**

Jalankan beberapa episode *tanpa eksplorasi* ($\epsilon=0$) untuk mengukur: rata-rata *return*, tingkat keberhasilan, panjang langkah.

5. **Analisis Sensitivitas**

Ubah salah satu parameter (mis. E atau α) dan bandingkan metrik.

6. **Visualisasi**

Grafik konvergensi (return per episode / moving average).

7. **Laporan & Presentasi**

Rangkum temuan, tantangan, dan rekomendasi.

Panduan Reward Shaping (opsional)

FrozenLake +1 di *goal*, 0 di lainnya (baseline). Coba tambah *penalti kecil* (mis. -0.01) per langkah.

Taxi +20 saat sukses, -1 per langkah, -10 *illegal pickup/dropoff*.

(Opsional) Ekstensi PPO dengan Stable-Baselines3

Latih **PPO** pada **CartPole-v1** selama 100–200k langkah.

Bandingkan kurva *reward* dengan tabular Q-Learning.

Tools yang Dikumpulkan

1. **Kode sumber** (.ipynb atau .py) yang dapat dijalankan.
2. **Grafik** konvergensi & tabel metrik evaluasi.
3. **Laporan PDF (4–6 halaman)** berisi: ringkasan, metodologi, hasil, analisis, keterbatasan.
4. **Slide** presentasi (≤ 10 slide).

Imbalance Data

A. CPL

Mahasiswa memahami konsep, masalah imbalance data, serta dampaknya.

B. CPMK

Mahasiswa menganalisis dan membandingkan teknik penanganan imbalance data.

C. Sub-CPMK

Mahasiswa menentukan metode penanganan imbalance data sesuai Supervised Learning, Unsupervised Learning dan Reinforcement Learning

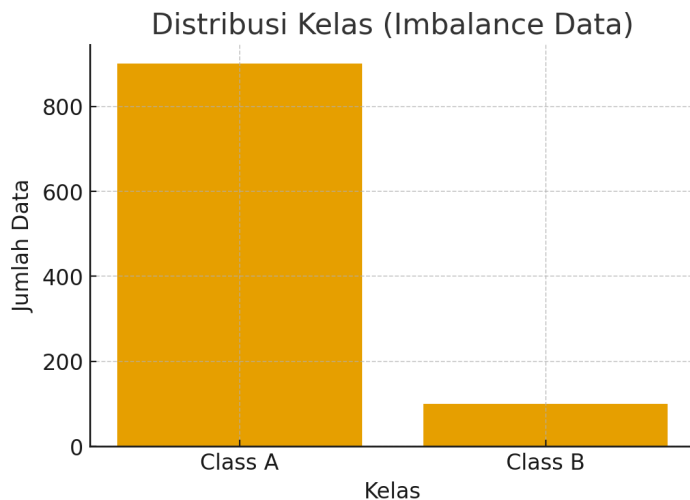
Fenomena **imbalance data** merupakan tantangan fundamental dalam pengembangan model *machine learning*. Ketidakseimbangan distribusi kelas dalam dataset menyebabkan bias prediksi dan berisiko mengabaikan pola penting dari kelas minoritas. Contoh yang sering muncul meliputi deteksi penipuan kartu kredit, diagnosis penyakit langka, dan sistem keamanan siber. Jika masalah ini diabaikan, akurasi model terlihat tinggi tetapi sejatinya tidak mewakili performa sebenarnya.

Bab ini menjelaskan secara menyeluruh pengertian *imbalance data*, variasi masalah yang muncul, dampaknya terhadap kinerja model, teknik penanganan yang dapat diterapkan, metrik evaluasi yang relevan, serta relevansi penerapannya di berbagai sektor.

Pengertian Imbalance Data

Imbalance data adalah kondisi ketika distribusi antar kelas pada sebuah dataset tidak seimbang atau tidak merata. Misalnya, dalam suatu dataset terdapat **90% data kelas mayoritas** dan hanya **10% data kelas minoritas**. Ketidakseimbangan ini menyebabkan model cenderung lebih sering memprediksi kelas dominan. Akibatnya, meskipun nilai **akurasi** terlihat tinggi, kemampuan model dalam mendeteksi kelas minoritas menjadi sangat rendah. Hal ini berpotensi menurunkan kualitas pengambilan keputusan, terutama dalam konteks yang sensitif.

Kondisi ini dapat menghasilkan **bias prediksi**, di mana model belajar lebih banyak dari kelas dominan dan mengabaikan informasi penting dari kelas minoritas. Dampaknya, deteksi terhadap kejadian langka atau kasus penting menjadi sangat terbatas. Dengan demikian, akurasi yang tinggi tidak selalu mencerminkan performa model yang sebenarnya. Oleh karena itu, evaluasi harus melibatkan metrik lain seperti **precision**, **recall**, dan **F1-score**.



Gambar 47. Distribusi Kelas Tidak Seimbang

Visualisasi ini menampilkan perbedaan distribusi data antara **Class A** dan **Class B**. Terlihat jelas bahwa **Class A** memiliki jumlah data yang jauh lebih besar dibandingkan dengan **Class B**. Kondisi ini dikenal sebagai *imbalanced data*, yaitu situasi ketika salah satu kelas memiliki dominasi jumlah yang sangat tinggi.

Fenomena ini sering menjadi masalah dalam pembangunan model *machine learning*, karena model cenderung **terlalu fokus** pada kelas mayoritas dan **mengabaikan kelas minoritas**. Dampaknya, performa model dapat terlihat baik secara keseluruhan, tetapi buruk dalam mendeteksi kelas dengan jumlah data sedikit.

Untuk mengatasi hal ini, beberapa teknik dapat digunakan seperti **oversampling**, **undersampling**, atau menggunakan algoritma yang **sensitif terhadap ketidakseimbangan data**. Dengan memahami visualisasi ini, kita mendapatkan **insight penting** bahwa kualitas data sama pentingnya dengan jumlah data dalam membangun model yang adil dan akurat.

Contoh Kasus

A. Kesehatan

Dataset medis, khususnya pada kasus **penyakit kanker**, jumlah data pasien sehat biasanya jauh lebih banyak dibandingkan pasien dengan diagnosa kanker. Kondisi ini dapat menimbulkan bias model dalam mendeteksi kanker. Meskipun model menunjukkan tingkat **akurasi** tinggi, kemampuannya mendeteksi pasien kanker bisa sangat rendah. Hal ini berbahaya karena pasien dengan kanker memerlukan

perhatian khusus, sehingga deteksi yang salah dapat berimplikasi serius pada keselamatan.

B. Keuangan

Industri keuangan, ketidakseimbangan data terlihat pada kasus **transaksi normal** yang jauh lebih mendominasi dibandingkan kasus **penipuan (fraud)**. Model yang dilatih tanpa memperhatikan ketidakseimbangan ini dapat cenderung mengabaikan pola penipuan. Akibatnya, potensi kerugian finansial bisa meningkat karena transaksi mencurigakan tidak terdeteksi dengan baik. Oleh karena itu, metrik tambahan seperti **recall** sangat penting untuk mengevaluasi kinerja model.

C. Keamanan Siber

Pada analisis **lalu lintas jaringan komputer**, data aktivitas normal mendominasi, sedangkan data **serangan (attack)** jumlahnya sangat kecil. Jika tidak diatasi, model keamanan siber dapat gagal mendeteksi ancaman penting. Hal ini berpotensi membuka celah terhadap serangan berbahaya yang dapat merusak sistem.

Karakteristik

Imbalance data memiliki sejumlah karakteristik penting yang membedakannya dari kondisi distribusi data yang normal. Memahami karakteristik ini sangat krusial agar peneliti maupun praktisi data tidak

terjebak dalam kesalahan interpretasi hasil analisis. Berikut adalah pemaparan komprehensif terkait ciri-ciri utama dari fenomena ini.

A. Distribusi Tidak Merata Antar Kelas

Ciri utama **imbalance data** adalah **ketidakseimbangan distribusi** antara kelas mayoritas dan kelas minoritas. Misalnya, dalam sebuah dataset, kelas mayoritas bisa mencapai **90%** dari total data, sementara kelas minoritas hanya sekitar **10%** atau bahkan lebih kecil. Kondisi ini membuat proses pembelajaran model cenderung terpusat pada kelas dominan dan mengabaikan kelas yang jumlahnya lebih sedikit.

B. Bias Model ke Kelas Mayoritas

Ketidakseimbangan distribusi menyebabkan **bias prediksi**. Model cenderung memprediksi kelas mayoritas karena frekuensinya jauh lebih tinggi. Akibatnya, meskipun model tampak bekerja baik pada data mayoritas, ia gagal mendeteksi pola penting dari kelas minoritas. Bias ini dapat berimplikasi serius pada bidang sensitif seperti kesehatan, keuangan, maupun keamanan siber.

C. Keterbatasan Metrik Akurasi

Imbalance data, metrik akurasi sering menyesatkan. Nilai akurasi yang tinggi tidak selalu mencerminkan performa model yang baik, karena prediksi kelas minoritas cenderung diabaikan. Oleh karena itu, evaluasi perlu menggunakan metrik

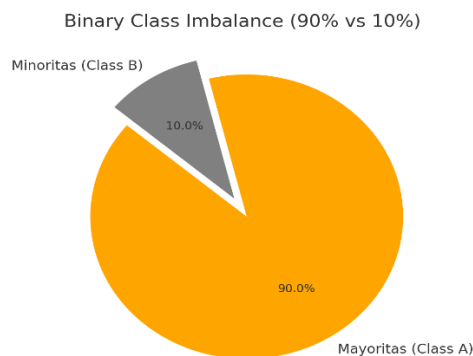
alternatif seperti **precision**, **recall**, dan **f1-score** yang lebih mampu menggambarkan keseimbangan kinerja antar kelas.

Jenis-Jenis Masalah

Fenomena **imbalance data** dapat muncul dalam berbagai bentuk tergantung pada struktur dataset dan karakteristik distribusinya. Mengenali jenis-jenis masalah ini sangat penting agar peneliti maupun praktisi dapat memilih metode penanganan yang sesuai. Berikut adalah penjelasan komprehensif mengenai beberapa jenis utama masalah *imbalance data*.

A. Binary Class Imbalance

Jenis ini terjadi ketika terdapat hanya **dua kelas** dengan distribusi yang timpang. Misalnya, dalam dataset medis, jumlah data pasien **sehat** bisa jauh lebih banyak dibandingkan pasien **sakit**. Ketidakseimbangan ini membuat model lebih cenderung memprediksi pasien sehat, sehingga menurunkan sensitivitas terhadap kasus penyakit.



Gambar 48. Binary Class Imbalance

Visualisasi ini menggambarkan fenomena **binary class imbalance**, di mana terdapat dua kelas dengan distribusi yang sangat timpang. Contohnya dapat ditemukan dalam dataset medis, ketika jumlah data pasien **sehat** jauh lebih banyak dibandingkan pasien **sakit**. Kondisi seperti ini menimbulkan tantangan besar dalam *machine learning* karena model cenderung **memihak kelas mayoritas**.

Akibatnya, prediksi sering mengarah ke kelas dengan jumlah data lebih besar (misalnya pasien sehat), sehingga **sensitivitas terhadap kasus penyakit menurun drastis**. Hal ini berbahaya, terutama dalam konteks medis, karena kasus penyakit yang jumlahnya sedikit justru memiliki dampak besar.

Insight penting dari ilustrasi ini adalah bahwa distribusi data yang tidak seimbang bukan sekadar masalah jumlah, tetapi juga menyangkut **keadilan prediksi**. Untuk mengatasinya, teknik seperti *oversampling*, *undersampling*, atau metode khusus yang peka terhadap distribusi dapat digunakan agar model lebih adil dan akurat.

B. Multi-class Imbalance

Masalah ini muncul ketika dataset memiliki **lebih dari dua kelas** dengan distribusi yang tidak merata. Contohnya, dalam klasifikasi **jenis tumor**, beberapa tipe tumor mungkin hanya memiliki sedikit data dibandingkan tipe lainnya. Hal ini menyebabkan model kurang mampu mengenali pola pada kelas yang jarang muncul.



Gambar 49. Multi-Class Imbalance

Multi-Class Imbalance dapat divisualisasikan melalui diagram batang dengan perbedaan tinggi yang signifikan pada setiap kelas. Misalnya, *Class A* memiliki jumlah data jauh lebih banyak dibandingkan *Class C, D, atau E*. Hal ini memperlihatkan bahwa distribusi data antar kelas tidak merata.

Masalah ini muncul ketika dataset memiliki lebih dari dua kelas dengan distribusi yang tidak seimbang. Contoh nyata dapat ditemukan pada klasifikasi jenis tumor, di mana beberapa tipe tumor memiliki ribuan data, sementara tipe langka hanya memiliki puluhan data. Kondisi ini menimbulkan tantangan besar dalam proses pembelajaran mesin.

Model cenderung **fokus pada pola dari kelas mayoritas**, karena jumlah data yang melimpah memberikan kontribusi dominan dalam pelatihan. Sebaliknya, **kelas minoritas terabaikan** sehingga model gagal mengenali karakteristik penting dari data yang jarang muncul. Hal ini bisa mengakibatkan prediksi yang bias dan menurunkan performa model, terutama pada kasus nyata di mana justru kelas minoritas lebih penting untuk diidentifikasi.

Sebagai *insight*, **imbalance data** dapat menurunkan sensitivitas sistem dalam mendeteksi kejadian langka, seperti penyakit jarang atau kasus penipuan. Oleh karena itu, penting untuk mempertimbangkan metode penyeimbangan seperti *oversampling*, *undersampling*, atau teknik *synthetic data generation* (contohnya *SMOTE*). Dengan strategi ini, distribusi antar kelas bisa diperbaiki sehingga model lebih adil dalam mengenali seluruh kategori.

C. Within-class Imbalance

Jenis ini terjadi di dalam **satu kelas utama** yang sebenarnya memiliki beberapa sub-kelompok, namun tidak semuanya terwakili dengan baik. Misalnya, pada data pasien kanker, terdapat tipe-tipe kanker tertentu yang jumlah datanya sangat terbatas. Akibatnya, model berisiko hanya mengenali tipe kanker yang dominan.

D. Imbalance Dinamis

Masalah ini bersifat **waktu dinamis**, yaitu distribusi kelas berubah seiring berjalannya waktu. Contohnya pada data transaksi keuangan harian, di mana pola transaksi normal dan penipuan (*fraud*) bisa berbeda dari satu periode ke periode lainnya. Perubahan ini menuntut model untuk selalu diperbarui agar tetap relevan..

Dampak terhadap Model

A. Penurunan Akurasi Kelas Minoritas

Model cenderung lebih fokus pada **kelas mayoritas**, sehingga akurasi untuk **kelas minoritas** menurun drastis. Akibatnya, kasus-

kasus penting yang jumlah datanya kecil sering kali gagal terdeteksi. Hal ini dapat menurunkan efektivitas model dalam konteks nyata, terutama pada bidang sensitif.

B. Peningkatan *False Negative* dalam Medis

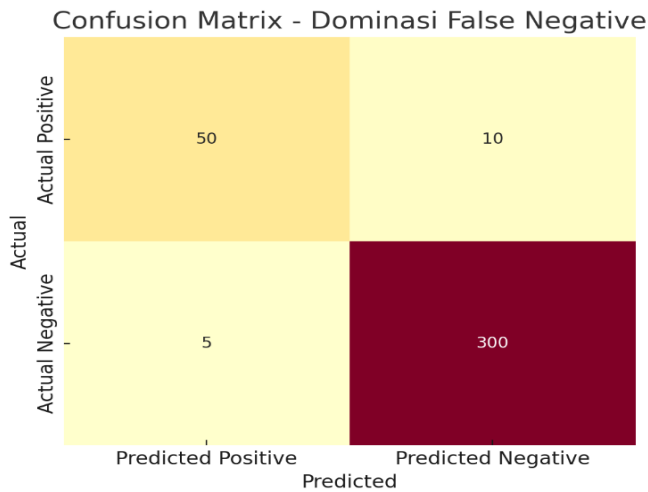
Dalam bidang kesehatan, **false negative** merupakan dampak yang sangat berbahaya. Misalnya, pasien dengan penyakit serius dapat diklasifikasikan sebagai sehat karena jumlah data sakit terlalu sedikit. Kesalahan ini berpotensi mengancam keselamatan pasien dan melemahkan kepercayaan terhadap sistem berbasis kecerdasan buatan.

C. Menurunnya Kemampuan Generalisasi

Model yang dilatih pada data tidak seimbang berisiko gagal mempelajari pola langka dari kelas minoritas. Akibatnya, **kemampuan generalisasi** menurun, sehingga performa model buruk ketika dihadapkan pada data baru yang lebih beragam. Kondisi ini memperlihatkan keterbatasan model dalam menghadapi variasi nyata.

D. Risiko *Overfitting* pada Mayoritas

Ketergantungan pada kelas mayoritas juga meningkatkan risiko **overfitting**. Model terlalu terikat pada pola dominan sehingga mengabaikan informasi dari kelas minoritas. Dampaknya, hasil prediksi menjadi kurang seimbang dan berpotensi tidak adaptif terhadap data baru.



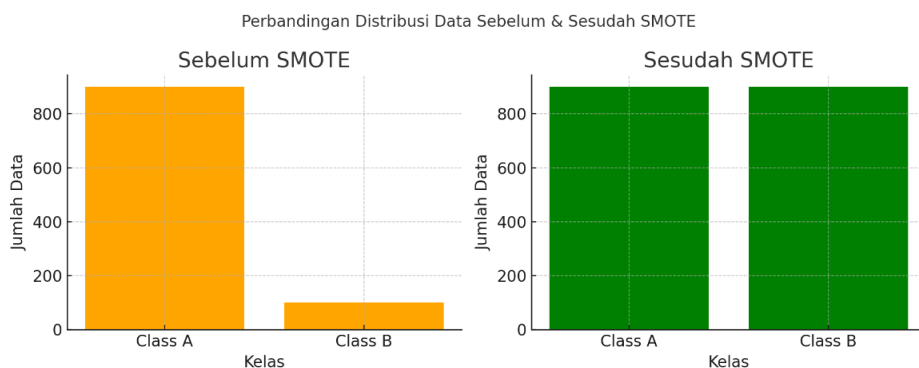
Gambar 50. *confusion matrix* dengan dominasi *False Negative*

confusion matrix dengan dominasi *False Negative* menegaskan konsekuensi serius dari distribusi data yang tidak seimbang. Pada kondisi ini, jumlah kesalahan prediksi terutama muncul ketika kasus positif diprediksi sebagai negatif. Model cenderung lebih fokus pada kelas mayoritas, sehingga akurasi pada kelas minoritas menurun drastis. Kasus-kasus penting dengan jumlah data kecil sering gagal terdeteksi, yang dapat mengurangi efektivitas model dalam konteks nyata, terutama di bidang sensitif seperti kesehatan atau keamanan. Dalam dunia medis, *false negative* merupakan kesalahan yang sangat berbahaya. Misalnya, pasien dengan penyakit serius bisa saja diklasifikasikan sebagai sehat hanya karena jumlah data pasien sakit terlalu sedikit. Kesalahan semacam ini tidak hanya mengancam keselamatan pasien, tetapi juga melemahkan kepercayaan terhadap sistem berbasis *artificial intelligence*. Selain itu, model yang dilatih dengan data tidak seimbang berisiko gagal mempelajari pola langka dari kelas minoritas. Akibatnya, kemampuan generalisasi menurun dan performa model menjadi buruk ketika dihadapkan dengan data

baru yang lebih beragam. Hal ini menegaskan keterbatasan model dalam mengenali variasi nyata yang terjadi di lapangan. Lebih jauh lagi, ketergantungan pada kelas mayoritas meningkatkan risiko *overfitting*, di mana model terlalu terikat pada pola dominan dan mengabaikan informasi penting dari kelas minoritas. Dampaknya, prediksi menjadi tidak seimbang dan kurang adaptif terhadap data baru. Sebagai *insight*, **imbalance data** tidak hanya menurunkan akurasi, tetapi juga memengaruhi keadilan, keandalan, serta keamanan sistem prediksi yang diterapkan di dunia nyata.

Teknik Penanganan

Untuk mengatasi fenomena **imbalance data**, berbagai teknik telah dikembangkan agar distribusi antar kelas lebih seimbang. Salah satu pendekatan utama yang banyak digunakan adalah metode **resampling**, baik dengan cara menambah data minoritas maupun mengurangi data mayoritas. Pemahaman yang tepat mengenai metode ini penting agar penerapannya sesuai dengan kebutuhan.



Gambar 51. Perbandingan Sebelum dan Sesudah SMOTE

Untuk mengatasi fenomena *imbalance data*, berbagai teknik telah dikembangkan agar distribusi antar kelas menjadi lebih

seimbang. Salah satu pendekatan utama yang banyak digunakan adalah metode *resampling*. Teknik ini dapat dilakukan dengan dua cara: menambah jumlah data pada kelas minoritas (*oversampling*) atau mengurangi data pada kelas mayoritas (*undersampling*). Pemahaman yang tepat mengenai metode ini sangat penting agar penerapannya sesuai dengan kebutuhan spesifik dari dataset dan tujuan analisis.

Ilustrasi perbandingan bar chart sebelum dan sesudah *SMOTE* (*Synthetic Minority Over-sampling Technique*) menggambarkan dampak nyata dari pendekatan ini. Pada kondisi awal, *Class A* memiliki dominasi jumlah data yang signifikan dibandingkan *Class B*. Setelah penerapan *SMOTE*, jumlah data pada kelas minoritas ditingkatkan dengan menghasilkan data sintetis yang menyerupai pola aslinya. Hasilnya, distribusi antar kelas menjadi lebih seimbang, memungkinkan model belajar pola dengan lebih adil.

Sebagai *insight*, **SMOTE memberikan solusi efektif untuk mengurangi bias model** yang cenderung fokus pada kelas mayoritas. Dengan distribusi yang lebih setara, model dapat meningkatkan sensitivitas terhadap kelas minoritas dan mengurangi risiko kesalahan prediksi kritis seperti *false negative*. Strategi ini menjadikan model lebih adaptif, adil, dan relevan ketika dihadapkan pada data baru yang kompleks serta beragam.

A. Resampling Methods

1. Oversampling

Metode ini dilakukan dengan **menambah jumlah data pada kelas minoritas**. Salah satu teknik populer adalah *Synthetic Minority Oversampling Technique (SMOTE)* dan *Adaptive Synthetic Sampling (ADASYN)*. Kedua teknik ini bekerja dengan menghasilkan data sintetis baru yang mirip dengan data minoritas asli.

a. **Kelebihan**

Distribusi data menjadi lebih seimbang, sehingga model dapat belajar pola dari kelas minoritas secara lebih baik.

b. **Kekurangan**

Risiko **overfitting** meningkat karena model dapat terlalu terikat pada data sintetis yang dihasilkan.

Formula

Synthetic Minority Oversampling Technique (SMOTE). SMOTE bekerja dengan membuat sampel sintetis baru berdasarkan perbedaan antara data minoritas yang ada dengan tetangganya. Formula matematis SMOTE dapat ditulis sebagai berikut:

$$X_{new} = X_i + \delta \times (X_{nn} - X_i)$$

Dimana:

X_i = sampel minoritas yang dipilih

X_{nn} = tetangga terdekat dari X_i

δ = nilai acak dalam interval $[0,1]$

Selain SMOTE, terdapat juga *Adaptive Synthetic Sampling (ADASYN)*.

ADASYN berfokus pada generasi sampel sintetis pada area di mana data minoritas lebih sulit dipelajari. Formula bobot kesulitan ADASYN dapat dirumuskan sebagai berikut:

$$G_i = \frac{d_i}{\sum_{j=1}^n d_j} \times G$$

Dimana:

G_i = jumlah sampel sintetis yang akan dibuat untuk data minoritas ke- i

d_i = tingkat kesulitan klasifikasi untuk data minoritas ke- i

G = total sampel sintetis yang ingin dihasilkan

Dengan demikian, metode oversampling tidak hanya menambah data secara acak, tetapi juga berupaya menciptakan sampel baru yang lebih representatif terhadap distribusi data sebenarnya.

2. Undersampling

Metode **undersampling** mengurangi jumlah data pada kelas mayoritas untuk mencapai rasio kelas yang lebih seimbang. Pendekatan ini cepat karena ukuran data berkurang, namun berisiko **kehilangan informasi** akibat pembuangan sebagian contoh.

Formula

Notasi Dasar

Misalkan:

N_{maj} : jumlah sampel kelas mayoritas,

N_{min} : jumlah sampel kelas minoritas,

$\rho \in (0, \infty)$: rasio target setelah penyamplingan, didefinisikan sebagai $\rho = \frac{N'_{\text{maj}}}{N_{\text{min}}}$.

Jumlah sampel mayoritas yang dipertahankan ditetapkan:

$$N'_{\text{maj}} = \min(N_{\text{maj}}, \lfloor \rho N_{\text{min}} \rfloor).$$

Fraksi pembuangan (kehilangan informasi) dari kelas mayoritas adalah:

$$\eta = 1 - \frac{N'_{\text{maj}}}{N_{\text{maj}}}.$$

Jika ditargetkan **keseimbangan sempurna** ($\rho = 1$):

$$N'_{\text{maj}} = \min(N_{\text{maj}}, N_{\text{min}}), \eta = 1 - \frac{\min(N_{\text{maj}}, N_{\text{min}})}{N_{\text{maj}}}.$$

Seleksi Acak Tanpa Pengembalian (Random Undersampling)

Anggap sampel mayoritas $\{x_{\text{maj}}^{(i)}\}_{i=1}^{N_{\text{maj}}}$. Definiskan vektor indikator $\mathbf{z} \in \{0,1\}^{N_{\text{maj}}}$ dengan kendala:

$$\sum_{i=1}^{N_{\text{maj}}} z_i = N'_{\text{maj}}$$

Himpunan hasil undersampling:

$$\mathcal{S}'_{\text{maj}} = \{x_{\text{maj}}^{(i)} : z_i = 1\},$$

dengan $\mathbf{z}$ dipilih **seragam** dari semua kombinasi tanpa pengembalian:

$$\mathbf{z} \sim \text{Unif}(\{\mathbf{z} \in \{0,1\}^{N_{\text{maj}}} : \sum_i z_i = N'_{\text{maj}}\}).$$

B. Algorithmic Methods

Selain metode resampling, pendekatan berbasis **algoritma** juga banyak digunakan untuk mengatasi permasalahan *imbalance*

data. Metode ini berfokus pada adaptasi model agar lebih peka terhadap kelas minoritas tanpa harus mengubah struktur dataset secara signifikan. Berikut dua teknik utama yang umum dipakai.

1. **Cost-Sensitive Learning**

Metode ini melibatkan pemberian **bobot kesalahan (cost)** yang lebih tinggi pada prediksi kelas minoritas. Dengan kata lain, model akan lebih "dihukum" ketika salah dalam memprediksi data minoritas dibandingkan mayoritas.

a. **Kelebihan**

Model dapat lebih memperhatikan kelas minoritas karena kesalahan prediksinya dianggap lebih serius.

b. **Kekurangan**

Menentukan bobot yang tepat sering kali menjadi tantangan, karena bobot yang terlalu besar bisa mengganggu stabilitas prediksi kelas mayoritas.

Contoh penerapan banyak digunakan pada bidang medis, di mana salah mendeteksi pasien dengan penyakit serius dianggap jauh lebih berbahaya daripada salah memprediksi pasien sehat.

Formula Dasar

Misalkan:

$y_i \in \{0,1\}$: label kelas sebenarnya (0 = mayoritas, 1 = minoritas),

$\hat{y}_i$: prediksi model,

C_{FN} : cost untuk kesalahan false negative (gagal mengenali minoritas),

C_{FP} : cost untuk kesalahan false positive,

$\ell(y_i, \hat{y}_i)$: fungsi kerugian standar (misalnya, loss 0-1).

Fungsi kerugian berbobot dapat ditulis sebagai:

$$L = \sum_{i=1}^n w_{y_i} \cdot \ell(y_i, \hat{y}_i)$$

dengan bobot kelas didefinisikan:

$$w_{y_i} = \begin{cases} C_{FN}, & \text{jika } y_i = 1 \wedge \hat{y}_i = 0 \\ C_{FP}, & \text{jika } y_i = 0 \wedge \hat{y}_i = 1 \\ 1, & \text{lainnya} \end{cases}$$

Contoh: Logistic Regression

Pada regresi logistik berbobot, fungsi kerugian logistik dimodifikasi menjadi:

$$L(\theta) = - \sum_{i=1}^n w_{y_i} [y_i \log(p_i) + (1 - y_i) \log(1 - p_i)]$$

dengan:

$$p_i = \sigma(\theta^T x_i) = \frac{1}{1 + e^{-\theta^T x_i}}.$$

Bobot w_{y_i} diatur agar kesalahan pada kelas minoritas lebih mahal dibandingkan mayoritas, sehingga model cenderung lebih sensitif terhadap kelas minoritas.

2. Ensemble Methods

Metode ini menggunakan kombinasi beberapa model untuk meningkatkan performa, khususnya dalam mendeteksi kelas minoritas. Pendekatan populer adalah **Random Forest** dan teknik *boosting* seperti *XGBoost* dan *LightGBM*. Algoritma ini mampu meningkatkan sensitivitas deteksi kelas minoritas dengan cara

menggabungkan kekuatan banyak pohon keputusan atau model lemah lainnya.

Metode **ensemble** memanfaatkan kombinasi beberapa model lemah (*weak learners*) untuk meningkatkan performa klasifikasi, terutama dalam mendeteksi kelas minoritas. Salah satu pendekatan populer adalah **boosting**, yang menekankan pembobotan lebih besar pada sampel yang sulit diklasifikasikan (biasanya kelas minoritas).

a. **Kelebihan**

Lebih stabil, akurat, dan sering kali unggul dalam mendeteksi pola kompleks.

b. **Kekurangan**

Membutuhkan sumber daya komputasi lebih besar dan interpretasi model bisa menjadi lebih sulit.

Formula

Misalkan:

Data latih $\{(x_i, y_i)\}_{i=1}^n$, dengan label $y_i \in \{-1, +1\}$.

$h_t(x)$: model lemah ke- t .

α_t : bobot model lemah ke- t .

Fungsi keputusan ensemble adalah:

$$H(x) = \text{sign}\left(\sum_{t=1}^T \alpha_t h_t(x)\right).$$

Update Bobot pada AdaBoost

Pada algoritma **AdaBoost**, bobot tiap sampel $D_t(i)$ diperbarui untuk memberi penekanan pada data yang salah klasifikasi. Rumus pembaruan bobot adalah:

$$D_{t+1}(i) = \frac{D_t(i) \cdot e^{-\alpha_t y_i h_t(x_i)}}{Z_t},$$

dengan:

Z_t : faktor normalisasi agar $\sum_i D_{t+1}(i) = 1$,

$$\alpha_t = \frac{1}{2} \ln \left(\frac{1 - \varepsilon_t}{\varepsilon_t} \right),$$

$\varepsilon_t = \sum_i D_t(i) \cdot 1_{[y_i \neq h_t(x_i)]}$ adalah tingkat kesalahan model lemah.

Penyesuaian untuk Kelas Minoritas

Dalam konteks data tidak seimbang, distribusi awal bobot dapat dimodifikasi:

$$D_1(i) = \begin{cases} \frac{1}{2N_{\min}}, & y_i = \text{minoritas} \\ \frac{1}{2N_{\max}}, & y_i = \text{mayoritas} \end{cases}$$

Dengan demikian, sampel kelas minoritas memperoleh bobot awal lebih tinggi sehingga boosting lebih fokus mendeteksi pola dari kelas tersebut.

C. Hybrid Methods

Dalam konteks *imbalanced data*, **Hybrid Methods** muncul sebagai strategi komprehensif yang menggabungkan pendekatan *resampling* dengan algoritma adaptif. Tujuannya adalah memanfaatkan keunggulan dari kedua metode agar kelemahan masing-masing dapat diminimalisasi. Contoh penerapan populer

adalah kombinasi *Synthetic Minority Oversampling Technique* (**SMOTE**) dengan **Cost-Sensitive Learning**.

Konsep Dasar

Metode **SMOTE** bekerja dengan menambahkan sampel sintetis pada kelas minoritas sehingga distribusi data menjadi lebih seimbang. Sementara itu, **Cost-Sensitive Learning** menekankan pemberian bobot kesalahan (*cost*) yang lebih tinggi pada prediksi yang salah terhadap kelas minoritas. Dengan menggabungkan kedua teknik ini, sistem tidak hanya memiliki distribusi data yang lebih proporsional, tetapi juga mendorong algoritma agar lebih fokus terhadap konsekuensi kesalahan pada kelas yang jarang muncul.

Keunggulan Hybrid Methods

1. Keseimbangan data

Melalui *oversampling*, kelas minoritas memperoleh representasi yang memadai dalam proses pelatihan.

2. Sensitivitas model

Penggunaan bobot kesalahan meningkatkan perhatian model terhadap klasifikasi minoritas.

3. Generalization power

Kombinasi dua pendekatan ini membantu mengurangi bias terhadap mayoritas tanpa terlalu mengorbankan akurasi keseluruhan.

Tantangan dan Pertimbangan

Meski menjanjikan, **Hybrid Methods** juga memiliki tantangan. Penambahan data sintesis berlebihan dapat meningkatkan risiko *overfitting*, sementara penetapan bobot kesalahan yang tidak tepat dapat menimbulkan ketidakseimbangan baru. Oleh karena itu, diperlukan proses validasi menyeluruh untuk menyesuaikan parameter, seperti jumlah sampel sintesis yang ditambahkan dan nilai *cost* yang diberikan.

Secara keseluruhan, **Hybrid Methods** menawarkan solusi efektif bagi permasalahan *imbalanced data*. Dengan mengombinasikan *SMOTE* dan *cost-sensitive learning*, metode ini mampu meningkatkan performa klasifikasi, menjaga integritas data, dan relevan diterapkan dalam berbagai domain dunia nyata, seperti *fraud detection*, *medical diagnosis*, maupun *cybersecurity*. Panjang pendekatan ini menunjukkan bahwa solusi komprehensif seringkali lebih unggul dibandingkan strategi tunggal.

D. Advanced Approaches

Dalam perkembangan terbaru, teknologi **machine learning** dan **deep learning** menghadirkan pendekatan canggih untuk mengatasi masalah *imbalance data*. Pendekatan ini tidak hanya berfokus pada penyeimbangan distribusi data, tetapi juga pada peningkatan kemampuan model dalam memahami pola minoritas yang kompleks. Berikut dua metode utama yang sering dibahas.

1. Deep Learning dengan Attention.

Metode **deep learning** yang dilengkapi dengan mekanisme *attention* dirancang untuk **menyoroti pola penting pada kelas**

minoritas. Mekanisme *attention* bekerja dengan memberi fokus lebih besar pada fitur-fitur tertentu yang relevan, sehingga model tidak hanya mengikuti pola dominan dari kelas mayoritas.

a. **Kelebihan**

Mampu menangkap detail halus dan meningkatkan sensitivitas model terhadap kelas minoritas.

b. **Kekurangan**

Membutuhkan data dan sumber daya komputasi yang besar untuk dilatih secara efektif.

Pendekatan ini banyak digunakan pada aplikasi **natural language processing (NLP)** dan analisis citra medis, di mana informasi minoritas sering kali menentukan kualitas prediksi.

2. **Generative Adversarial Networks (GANs)**

Metode **GANs** digunakan untuk **membuat data sintetis yang realistis** pada kelas minoritas. GAN terdiri dari dua jaringan, *generator* dan *discriminator*, yang saling berkompetisi untuk menghasilkan data baru yang menyerupai data asli.

a. **Kelebihan**

Data sintetis yang dihasilkan lebih alami dibandingkan teknik *oversampling* tradisional.

b. **Kekurangan**

Proses pelatihan GAN cukup kompleks dan rentan terhadap ketidakstabilan.

Evaluasi pada Data Tidak Seimbang

Menghadapi **imbalance data**, penggunaan metrik evaluasi yang tepat sangat penting untuk menilai kinerja model secara adil.

Mengandalkan **akurasi** saja sering kali menyesatkan karena tidak mencerminkan kemampuan model dalam mendeteksi kelas minoritas. Oleh karena itu, diperlukan pendekatan evaluasi yang lebih komprehensif. Berikut adalah metrik-metrik utama yang relevan.

a. **Confusion Matrix**

Confusion matrix merupakan dasar evaluasi yang menampilkan hasil prediksi model dalam bentuk **true positive**, **true negative**, **false positive**, dan **false negative**. Fokus utama pada data tidak seimbang adalah **precision**, **recall**, dan **F1-score**, karena ketiga metrik ini mampu memberikan gambaran lebih jelas terhadap performa kelas minoritas.

Bentuk Matriks

Matriks konfusi dapat direpresentasikan sebagai:

$$\begin{bmatrix} TP & FN \\ FP & TN \end{bmatrix}$$

Dimana:

TP (True Positive): jumlah data positif yang diprediksi positif.

TN (True Negative): jumlah data negatif yang diprediksi negatif.

FP (False Positive): jumlah data negatif yang diprediksi positif.

FN (False Negative): jumlah data positif yang diprediksi negatif.

Metrik Turunan

Berdasarkan *confusion matrix*, beberapa metrik evaluasi dapat diturunkan:

1. **Accuracy**

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

2. **Precision**

$$Precision = \frac{TP}{TP + FP}$$

3. **Recall (Sensitivity)**

$$Recall = \frac{TP}{TP + FN}$$

F1-Score

$$F1 = 2 \cdot \frac{Precision \times Recall}{Precision + Recall}$$

Dengan menggunakan **confusion matrix**, peneliti dapat menilai performa model tidak hanya dari akurasi global, tetapi juga dari kemampuan dalam mendeteksi kelas minoritas maupun mayoritas secara seimbang.

b. **ROC-AUC**

Metrik **Receiver Operating Characteristic - Area Under Curve (ROC-AUC)** digunakan untuk menilai kemampuan model dalam **membedakan antara kelas mayoritas dan minoritas**. Semakin tinggi nilai AUC, semakin baik model dalam membedakan kedua kelas tersebut.

Kurva ROC

Kurva ROC dibangun dengan memplot **True Positive Rate (TPR)** terhadap **False Positive Rate (FPR)** pada berbagai ambang keputusan (*threshold*).

True Positive Rate (TPR) / Recall:

$$TPR = \frac{TP}{TP + FN}$$

False Positive Rate (FPR):

$$FPR = \frac{FP}{FP + TN}$$

Area Under Curve (AUC)

Nilai **AUC** didefinisikan sebagai luas area di bawah kurva ROC:

$$AUC = \int_0^1 TPR(FPR) d(FPR)$$

Interpretasi:

$AUC = 0.5$: performa model setara dengan tebakan acak.

$AUC = 1$: model memiliki kemampuan klasifikasi sempurna.

ROC-AUC memberikan gambaran menyeluruh tentang performa model klasifikasi di berbagai ambang batas. Metode ini sangat relevan pada *imbalanced data* karena tidak hanya mengandalkan akurasi, tetapi juga memperhatikan keseimbangan antara deteksi kelas minoritas dan kesalahan prediksi kelas mayoritas.

c. **Precision-Recall AUC**

Pada kasus di mana kelas minoritas sangat kecil, **Precision-Recall AUC** lebih informatif dibandingkan ROC-AUC. Metrik ini berfokus pada kualitas deteksi kelas minoritas dengan menilai hubungan antara **precision** dan **recall**.

Rumus Dasar

Precision:

$$Precision = \frac{TP}{TP + FP}$$

Recall (Sensitivity / TPR):

$$Recall = \frac{TP}{TP + FN}$$

Kurva Precision-Recall dibentuk dengan memplot **Precision** terhadap **Recall** pada berbagai ambang keputusan (*threshold*).

Area Under Precision-Recall Curve (PR-AUC)

Nilai PR-AUC didefinisikan sebagai luas area di bawah kurva Precision-Recall:

$$PR - AUC = \int_0^1 Precision(Recall) d(Recall)$$

Interpretasi:

- a. Nilai PR-AUC mendekati 1 menunjukkan model mampu mendeteksi kelas minoritas dengan baik tanpa banyak kesalahan positif.
- b. Nilai PR-AUC yang rendah menunjukkan model kurang efektif dalam menjaga keseimbangan antara **Precision** dan **Recall**.

Precision-Recall AUC lebih sensitif terhadap distribusi tidak seimbang dibandingkan ROC-AUC. Oleh karena itu, metrik ini sangat direkomendasikan pada skenario nyata seperti *fraud detection*, *rare disease diagnosis*, atau *anomaly detection* di mana kelas minoritas sangat jarang namun kritis untuk dideteksi.

d. **G-Mean**

Geometric Mean (G-Mean) menilai keseimbangan antara **sensitivitas (recall)** dan **spesifisitas**. Metrik ini penting agar performa pada kedua kelas dapat dipertahankan secara seimbang.

Rumus Dasar

Recall (Sensitivitas / True Positive Rate):

$$Recall = \frac{TP}{TP + FN}$$

Specificity (True Negative Rate):

$$Specificity = \frac{TN}{TN + FP}$$

Geometric Mean (G-Mean):

$$G-Mean = \sqrt{Recall \times Specificity}$$

Interpretasi

- a. Nilai **G-Mean tinggi** menunjukkan bahwa model mampu menjaga keseimbangan antara deteksi kelas minoritas (Recall) dan prediksi benar pada kelas mayoritas (Specificity).
- b. Nilai **G-Mean rendah** menandakan ketidakseimbangan performa, misalnya model hanya unggul dalam mengenali kelas mayoritas tetapi lemah dalam mendeteksi kelas minoritas.

Metrik **G-Mean** memberikan gambaran yang lebih adil pada situasi *imbalanced data* dibandingkan hanya mengandalkan akurasi. Oleh karena itu, G-Mean sering digunakan pada bidang seperti *medical diagnosis*, *fraud detection*, dan *fault detection* di

mana kegagalan mendeteksi kelas minoritas bisa berakibat signifikan.

e. **Matthews Correlation Coefficient (MCC)**

MCC dianggap sebagai metrik yang lebih seimbang untuk distribusi timpang karena mempertimbangkan semua komponen pada *confusion matrix*. Nilai MCC mendekati 1 menunjukkan kinerja model yang baik pada semua kelas.

Rumus Dasar

MCC didefinisikan sebagai:

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

Interpretasi

MCC = 1 : prediksi sempurna (model membedakan kelas dengan sangat baik).

MCC = 0 : performa setara dengan tebakan acak.

MCC = -1 : prediksi sepenuhnya salah arah.

Sebagai metrik yang mempertimbangkan semua elemen pada *confusion matrix*, **MCC** sangat bermanfaat dalam evaluasi model pada data yang tidak seimbang. Oleh karena itu, MCC sering digunakan pada domain seperti *bioinformatics*, *medical diagnosis*, dan *fraud detection* untuk memastikan model tidak bias hanya pada kelas mayoritas.

Contoh Penerapan

A. Kesehatan

Bidang **kesehatan**, peran data sangat penting untuk mendukung proses diagnosis dan pengambilan keputusan klinis. Salah satu tantangan besar yang dihadapi adalah fenomena **imbalance data**, terutama pada kasus **penyakit langka**. Ketidakseimbangan distribusi data dapat menimbulkan konsekuensi serius, seperti meningkatnya risiko *false negative*, yaitu ketika pasien yang sebenarnya sakit justru diklasifikasikan sebagai sehat. Kesalahan ini sangat berbahaya karena dapat menunda penanganan medis yang seharusnya segera dilakukan.

Jika dataset didominasi oleh data pasien sehat, model pembelajaran mesin cenderung mengabaikan pola pada pasien sakit. Hal ini menurunkan **sensitivitas** model terhadap deteksi penyakit langka. Padahal, identifikasi dini terhadap kondisi ini sangat krusial karena sering kali menentukan tingkat keberhasilan pengobatan. Untuk mengurangi risiko kesalahan, metode penanganan **imbalance data** dapat diterapkan pada sistem *screening otomatis*. Sistem ini membantu dokter dengan memberikan peringatan dini terhadap kemungkinan penyakit langka.

Dengan demikian, dokter dapat mengambil keputusan lebih akurat, sekaligus meningkatkan peluang pasien mendapatkan terapi tepat waktu. Penerapan teknik seperti *oversampling*, *cost-sensitive learning*, atau bahkan pendekatan *deep learning dengan attention* mampu meningkatkan kemampuan model dalam mengenali pola minoritas. Hal ini tidak hanya mendukung diagnosis, tetapi juga

membangun kepercayaan terhadap teknologi medis berbasis kecerdasan buatan.

Oleh karena itu, fenomena **imbalance data dalam kesehatan** harus dikelola dengan serius. Dengan strategi yang tepat, sistem berbasis data dapat berfungsi sebagai alat bantu penting untuk mengurangi risiko *false negative* dan mendukung kualitas layanan medis yang lebih aman dan efektif.

B. Keamanan Siber

Era digital modern, **keamanan siber** menjadi salah satu aspek paling krusial untuk menjaga integritas data dan melindungi sistem dari ancaman eksternal. Permasalahan **imbalance data** juga muncul secara signifikan dalam bidang ini, karena data aktivitas normal jauh lebih mendominasi dibandingkan data serangan. Kondisi ini menciptakan tantangan bagi model pembelajaran mesin untuk mendeteksi ancaman langka namun berbahaya.

Salah satu aplikasi utama dalam keamanan siber adalah **Intrusion Detection System (IDS)**, yang berfungsi untuk **mengidentifikasi serangan langka** dalam lalu lintas jaringan. Karena data aktivitas normal jumlahnya sangat besar, model IDS sering kali bias terhadap kelas mayoritas dan gagal mengenali pola serangan minoritas. Dengan menerapkan metode penanganan *imbalance data* seperti *oversampling* atau *cost-sensitive learning*, sensitivitas IDS terhadap ancaman meningkat, sehingga potensi serangan dapat terdeteksi lebih dini. Selain IDS, **spam filtering** juga sangat dipengaruhi oleh fenomena ketidakseimbangan data.

Dalam praktiknya, jumlah email normal jauh lebih besar dibandingkan spam. Jika model hanya mengandalkan akurasi, sistem bisa saja terlihat efektif namun gagal memblokir spam berbahaya. Oleh karena itu, metrik evaluasi seperti **precision**, **recall**, dan **F1-score** lebih relevan untuk menilai kinerja sistem ini. Dengan pendekatan yang tepat, spam filtering dapat lebih andal dalam memisahkan pesan berbahaya dari komunikasi normal.

Secara keseluruhan, fenomena **imbalance data dalam keamanan siber** menegaskan pentingnya teknik penanganan dan evaluasi yang tepat. Baik pada **IDS** maupun **spam filtering**, kemampuan model untuk mendeteksi pola minoritas menentukan seberapa kuat perlindungan terhadap ancaman digital. Dengan strategi yang sesuai, sistem keamanan siber dapat menjadi lebih efektif, adaptif, dan dapat dipercaya.

C. Keuangan

Sektor **keuangan**, fenomena *imbalance data* menjadi salah satu tantangan terbesar dalam penerapan sistem berbasis kecerdasan buatan. Meskipun data transaksi normal mendominasi, kasus **penipuan (*fraud detection*)** maupun prediksi risiko kredit memerlukan perhatian lebih besar karena menyangkut stabilitas finansial dan kepercayaan masyarakat terhadap lembaga keuangan. Kasus penipuan dalam transaksi keuangan biasanya sangat jarang terjadi, namun risiko yang ditimbulkan sangat tinggi.

Ketika model hanya berfokus pada data mayoritas, yaitu transaksi normal, sistem sering gagal mendeteksi penipuan. Hal ini

dapat mengakibatkan kerugian finansial yang besar serta merusak reputasi lembaga keuangan. Dengan menerapkan teknik penanganan *imbalance data* seperti *oversampling*, *ensemble methods*, atau *cost-sensitive learning*, kemampuan model dalam mengenali pola penipuan dapat ditingkatkan secara signifikan. Selain deteksi penipuan, **risk assessment** juga memegang peranan penting dalam memprediksi risiko kredit.

Dalam praktiknya, jumlah peminjam yang melunasi pinjaman tepat waktu jauh lebih banyak dibandingkan dengan yang gagal bayar. Ketidakseimbangan data ini dapat membuat model kurang sensitif dalam memprediksi kelompok berisiko tinggi. Akibatnya, lembaga keuangan bisa salah menilai profil kredit dan menanggung potensi kerugian. Dengan penanganan yang tepat, model mampu mengidentifikasi kelompok minoritas dengan lebih akurat sehingga keputusan pemberian kredit menjadi lebih bijaksana.

Secara keseluruhan, fenomena **imbalance data dalam keuangan** menunjukkan bahwa data minoritas meski jarang, memiliki dampak yang sangat besar. Baik dalam **fraud detection** maupun **risk assessment**, penanganan yang tepat akan mendukung stabilitas sistem keuangan, meningkatkan akurasi prediksi, serta menjaga kepercayaan publik terhadap lembaga keuangan.

Penerapan Imbalance Di Masa Depan

1. AutoML

Automated Machine Learning (AutoML) berkembang sebagai solusi untuk mengotomatisasi proses pemodelan data. Integrasi

teknik khusus penanganan *imbalance data* ke dalam AutoML akan memungkinkan sistem secara otomatis memilih metode terbaik, seperti *oversampling*, *cost-sensitive learning*, atau *ensemble methods*. Dengan demikian, proses pengembangan model menjadi lebih efisien sekaligus tetap menjaga akurasi pada kelas minoritas.

2. **Explainable AI (XAI)**

Seiring meningkatnya kompleksitas model, transparansi menjadi kebutuhan penting. **Explainable AI (XAI)** hadir untuk memberikan penjelasan yang jelas mengenai bagaimana keputusan model dihasilkan. Dalam konteks *imbalance data*, XAI membantu memastikan bahwa prediksi pada kelas minoritas dapat dipahami dan dipercaya oleh pengguna, khususnya pada bidang kritis seperti kesehatan atau keuangan.

3. **Federated Learning**

Federated Learning memungkinkan kolaborasi antar institusi untuk melatih model tanpa harus berbagi data mentah. Hal ini sangat bermanfaat dalam kasus **data langka**, misalnya data medis antar rumah sakit. Dengan pendekatan ini, model dapat belajar dari berbagai sumber tanpa mengorbankan privasi, sekaligus meningkatkan representasi kelas minoritas

Imbalance data adalah tantangan penting dalam praktik *machine learning*. Dengan memahami definisi, bentuk masalah, dampak, penanganan, evaluasi, dan relevansinya, mahasiswa dapat membangun model yang **lebih adil dan andal**. Pemahaman ini vital

bagi perkembangan sistem AI yang mampu beradaptasi dengan kondisi nyata di bidang kesehatan, keuangan, maupun keamanan siber.

Soal Uraian

1. Jelaskan apa yang dimaksud dengan *Imbalance Data* dalam machine learning dan berikan contohnya dalam kehidupan nyata.
2. Mengapa *Imbalance Data* dapat menyebabkan model klasifikasi bias terhadap kelas mayoritas?
3. Sebutkan dan jelaskan minimal dua jenis masalah pada *Imbalance Data*.
4. Bagaimana pengaruh *Imbalance Data* terhadap kinerja algoritma supervised learning seperti *Decision Tree* atau *Logistic Regression*?
5. Bandingkan teknik *oversampling* dengan *undersampling*. Apa kelebihan dan kelemahan masing-masing pendekatan?
6. Jelaskan bagaimana prinsip kerja *SMOTE (Synthetic Minority Oversampling Technique)*.
7. Apa yang dimaksud dengan *cost-sensitive learning* dalam konteks *Imbalance Data*? Berikan contoh penerapannya.
8. Sebutkan dan jelaskan tiga metrik evaluasi yang lebih relevan digunakan dibanding akurasi dalam kasus *Imbalance Data*.
9. Bagaimana *Precision-Recall Curve* dapat memberikan gambaran yang lebih akurat dibanding *ROC Curve* pada dataset yang tidak seimbang?

10. Berikan contoh kasus nyata di bidang kesehatan atau keuangan di mana penanganan *Imbalance Data* sangat penting. Jelaskan teknik apa yang paling tepat digunakan dan mengapa.

Tugas Kelompok

Deteksi Transaksi Fraud Sederhana

Tujuan: membangun model klasifikasi untuk mendeteksi transaksi “fraud” (kelas minoritas) pada data sintetis yang tidak seimbang.

Tujuan Pembelajaran

1. Mempraktikkan identifikasi dan diagnosa *imbalance data*.
2. Menerapkan teknik penanganan: *class weight*, *resampling* (SMOTE), dan evaluasi yang tepat.
3. Membuat rekomendasi berbasis metrik, bukan sekadar akurasi.

Dataset

Gunakan `sklearn.datasets.make_classification` dengan rasio 95% : 5%.

Tugas Teknis

1. Eksplorasi & Diagnosa distribusi kelas.
2. Baseline model (tanpa penanganan imbalance).
3. Terapkan dua teknik penanganan imbalance (misalnya *class weight* & SMOTE).
4. Bandingkan metrik evaluasi (Recall, Precision, F1, PR-AUC).
5. Buat rekomendasi model terbaik untuk deteksi fraud.

Deliverables

- a) Notebook / script.
- b) Laporan ringkas (2-4 halaman).
- c) Slide presentasi (opsional).

.

Ujian Akhir Semester

Mata Kuliah: Metodologi Penelitian

Bentuk Soal: Uraian (10 Soal)

Cakupan Materi: Pertemuan 9-16

Contoh Soal Uraian

A. Soal Uraian

1. Jelaskan **siklus data sains (data science lifecycle)** dan bandingkan dengan siklus pengembangan sistem informasi. Sebutkan minimal tiga perbedaan penting dari keduanya.
2. Data sains bekerja dengan **data terstruktur** dan **data tidak terstruktur**. Uraikan perbedaan keduanya serta contoh pemanfaatannya dalam organisasi pendidikan atau kesehatan.
3. Mengapa **data cleaning** (pembersihan data) dianggap tahapan paling krusial dalam proyek data sains? Jelaskan dampaknya jika tahap ini diabaikan.
4. Tahap **eksplorasi data** sering dilakukan dengan visualisasi sederhana. Jelaskan bagaimana visualisasi membantu dalam menemukan pola awal, sertakan contohnya.
5. Uraikan hubungan antara **feature engineering** dengan akurasi model machine learning. Mengapa pemilihan fitur yang tepat bisa lebih penting daripada algoritma yang digunakan?

6. Evaluasi model merupakan bagian penting dalam data sains. Sebutkan minimal tiga **metrik evaluasi** yang umum digunakan untuk menilai performa model prediksi atau klasifikasi.
7. Data sains sangat bergantung pada **kualitas data**. Jelaskan empat dimensi kualitas data (misalnya akurasi, konsistensi, kelengkapan, dan ketepatan waktu) serta implikasinya terhadap pengambilan keputusan.

Contoh Soal Studi Kasus

B. Soal Studi Kasus (AI, ML, AI sederhana)

8. **Studi Kasus AI:** Sebuah rumah sakit ingin menggunakan **kecerdasan buatan** untuk membantu diagnosis penyakit berbasis citra rontgen paru-paru. Jelaskan bagaimana penerapan AI dapat meningkatkan akurasi diagnosis dan sebutkan dua tantangan etika yang mungkin muncul.
9. **Studi Kasus ML:** Sebuah perusahaan e-commerce ingin memprediksi apakah pelanggan akan melakukan pembelian ulang (repeat purchase) berdasarkan riwayat transaksi. Uraikan bagaimana proses machine learning sederhana (mulai dari data hingga model) dapat diterapkan pada kasus ini.
10. **Studi Kasus AI sederhana:** Sebuah kampus ingin menggunakan sistem chatbot berbasis AI sederhana untuk menjawab pertanyaan mahasiswa tentang jadwal kuliah. Jelaskan alur kerja data sains pada kasus ini mulai dari pengumpulan data pertanyaan, pelatihan model sederhana, hingga implementasi chatbot.

DAFTAR PUSTAKA

- Aliyu, Y., Sarlan, A., Danyaro, K. U., Rahman, A. S. A., Muazu, A. A., & Abubakar, M. Y. (2025). Deep learning techniques for sentiment analysis in code-switched Hausa-English tweets. *International Journal of Information Management Data Insights*, 5, 100330. <https://doi.org/10.1016/j.jjime.2025.100330>
- Boehm, B. (1988). A spiral model of software development and enhancement. *Computer*, 21(5), 61-72. <https://doi.org/10.1109/2.59>
- Brown, T. (2009). *Change by design: How design thinking creates new alternatives for business and society*. Harper Business. <https://books.google.co.id/books?id=x7PjWyVUoVAC>
- Dennis, A., Wixom, B. H., & Roth, R. M. (2015). *Systems analysis and design* (6th ed.). John Wiley & Sons.
- Halford, E., & Gibson, I. (2025). Using machine learning to conduct crime linking of residential burglary. *International Journal of Law, Crime and Justice*, 80, 100716. <https://doi.org/10.1016/j.ijlcj.2024.100716>
- Hamidi, H., & Moradi, M. (2025). Analysis of disease severity and mortality prediction using machine learning during COVID-19. *Acta Psychologica*, 258, 105136. <https://doi.org/10.1016/j.actpsy.2025.105136>
- Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, 28(1), 75-105. <https://doi.org/10.2307/25148625>
- Kendall, K. E., & Kendall, J. E. (2013). *Systems analysis and design* (9th ed.). Pearson Education. <https://books.google.co.id/books?id=PdAvAAAAQBAJ>
- Kendall, K. E., & Kendall, J. E. (2019). *Systems analysis and design* (10th ed.). Pearson.
- Kitchenham, B., & Charters, S. (2007). *Guidelines for performing systematic literature reviews in software engineering* (EBSE-2007-01). Evidence-Based Software Engineering (EBSE).
- Kim, G., Humble, J., Debois, P., & Willis, J. (2016). *The DevOps handbook: How to create world-class agility, reliability,*

- and security in technology organizations. *IT Revolution*.
<https://books.google.co.id/books?id=ui8hDgAAQBAJ>
- Lins, H. A., de Freitas Souza, M., Batista, L. P., da Silva Rodrigues, L. L., da Silva, F. D., Fernandes, B. C. C., de Melo, S. B., das Chagas, P. S. F., & Silva, D. V. (2024). Artificial intelligence for herbicide recommendation: Case study for the use of clomazone in Brazilian soils. *Smart Agricultural Technology*, 9, 100699.
<https://doi.org/10.1016/j.atech.2024.100699>
- Loureiro, A. L. D., Miguéis, V. L., Costa, Á., & Ferreira, M. (2025). Improving customer retention in taxi industry using travel data analytics: A churn prediction study. *Journal of Retailing and Consumer Services*, 85, 104288.
<https://doi.org/10.1016/j.jretconser.2025.104288>
- Obaid, W., Hussain, A., Rabie, T., Abd, D. H., & Mansoor, W. (2025). Multi-model deep learning approach for the classification of kidney diseases using medical images. *Informatics in Medicine Unlocked*, 57, 101663.
<https://doi.org/10.1016/j.imu.2025.101663>
- Ozdemir, S. (2016). *Principles of data science*. Packt Publishing.
<https://books.google.co.id/books?id=9NDcDgAAQBAJ>
- Ozmen, B. B., Singh, N., Shah, K., Berber, I., Singh, D., Pinsky, E., Rampazzo, A., & Schwarz, G. S. (2025). Development of a novel artificial intelligence clinical decision support tool for hand surgery: HandRAG. *Journal of Hand and Microsurgery*, 17, 100293.
<https://doi.org/10.1016/j.jham.2025.100293>
- Pressman, R. S. (2005). *Software engineering: A practitioner's approach* (6th ed.). McGraw-Hill.
<https://books.google.co.id/books?id=bL7QZHtWvaUC>
- Provost, F., & Fawcett, T. (2013). Data science and its relationship to big data and data-driven decision making. *Big Data*, 1(1), 51–59. <https://doi.org/10.1089/big.2013.1508>
- Ramesh, B., & Jarke, M. (2001). Toward reference models of requirements traceability. *IEEE Transactions on Software Engineering*, 27(1), 58–93.
<https://doi.org/10.1109/32.895989>
- Sariman, C., Hallawa, A., & Schmeink, A. (2025). Concurrent shape and controller optimization of autonomous underwater

- vehicles for path following tasks using evolutionary algorithm-driven reinforcement learning. *Ocean Engineering*, 336, 121841. <https://doi.org/10.1016/j.oceaneng.2025.121841>
- Soltani, M., Sharif, S., & Dara, R. (2025). Analyzing social media discourse of avian influenza outbreaks. *Natural Language Processing Journal*, 12, 100176. <https://doi.org/10.1016/j.nlp.2025.100176>
- Sommerville, I. (2016). *Software engineering* (10th ed.). Pearson. <https://books.google.co.id/books?id=tW4VngEACAAJ>
- Tahir, M. F., Tzes, A., El-Fouly, T. H. M., El Moursi, M. S., Xiao, D., & Larik, N. A. (2025). Optimizing solar photovoltaic power forecasting via multi-architecture machine learning framework with multiple hyperparameter optimization techniques. *Energy Strategy Reviews*, 61, 101859. <https://doi.org/10.1016/j.esr.2025.101859>
- Whitten, J. L., Bentley, L. D., & Dittman, K. C. (2004). *Systems analysis and design methods* (7th ed.). McGraw-Hill Irwin. <https://books.google.co.id/books?id=y63MyeySOu8C>
- Wysocki, O., Mak, S., Frost, H., Graham, D. M., Landers, D., & Aslam, T. (2025). Translating the machine; An assessment of clinician understanding of ophthalmological artificial intelligence outputs. *International Journal of Medical Informatics*, 201, 105958. <https://doi.org/10.1016/j.ijmedinf.2025.105958>
- Yu, B., & Kumbier, K. (2020). Veridical data science: The PCS framework. *Proceedings of the National Academy of Sciences*, 117(8), 3920–3929. <https://doi.org/10.1073/pnas.1901326117>

BIOGRAFI PENULIS



Ibnu Rasyid Munthe, S.T., M.Kom, lahir di Rantau Prapat, Kabupaten Labuhanbatu, tahun 1987. Ia menempuh pendidikan Sarjana Teknik di Universitas Nurtanio Bandung dan meraih gelar Magister Komputer dari Universitas Putra Indonesia (YPTK) Padang. Saat ini, beliau merupakan dosen aktif di Universitas Labuhanbatu dengan fokus pada bidang teknologi informasi. Di samping kegiatan akademik, beliau menulis berbagai artikel ilmiah dan buku, serta mengelola kanal *YouTube* @manjaddawajada2022 sebagai media edukasi teknologi. Terdaftar resmi pada SINTA ID 5975973, Google Scholar ID glPcbagAAAAJ, dan Scopus ID 57207882178, beliau konsisten berkontribusi bagi pengembangan riset dan literasi digital di Indonesia.



Marnis Nasution, S.Kom., M.Kom, lahir di Bengkulu pada 30 Maret 1990. Pendidikan dasar hingga menengah ia jalani di kota kelahirannya. Semangat menimba ilmu membawanya melanjutkan studi Strata I dan Strata II di Universitas Putra Indonesia "YPTK" Padang pada jurusan Sistem Informasi, yang ditempuh sejak tahun 2008. Saat ini, Marnis aktif mengabdikan diri sebagai dosen tetap di Universitas Labuhanbatu, Sumatera Utara. Selain mengajar, ia juga produktif menulis karya ilmiah dan buku, dengan tujuan mengembangkan ilmu pengetahuan

sekaligus memberikan kontribusi nyata bagi dunia pendidikan di Indonesia.



Muhammad Irwansyah Hasibuan, S.E., M.Si, lahir di Rantauprapat pada 6 Januari 1979. Ia memperoleh gelar Sarjana Ekonomi dari Program Studi Manajemen Sekolah Tinggi Ilmu Ekonomi Labuhanbatu pada tahun 2003, kemudian melanjutkan pendidikan Magister Sains di Program Studi Magister Ilmu Manajemen Universitas Sumatera Utara dan lulus tahun 2012. Saat ini, ia berprofesi sebagai dosen tetap sekaligus menjabat sebagai Wakil Dekan Fakultas Ekonomi dan Bisnis Universitas Labuhanbatu. Selain kegiatan akademik, ia aktif sebagai Ketua Asosiasi Dosen Labuhanbatu, Presidium KAHMI Labuhanbatu, anggota Lembaga Penelitian dan Kajian MUI Labuhanbatu, DPD Lembaga Kajian Ekonomi Al-Washliyah Labuhanbatu, serta Pengurus Badan Komunikasi Pemuda Remaja Masjid Indonesia Labuhanbatu



Indra Gunawan, M.Kom, adalah anak pertama dari 3 bersaudara. Lahir pada 29 November 1988 di Kota Banda Aceh, Provinsi Nangroe Aceh Darusalam. Besar dan bersekolah di Sumatera Utara. Penulis menyelesaikan Pendidikan Strata 2 di Universitas Sumatera Utara Jurusan Ilmu Komputer pada tahun 2016. Menetap di Kota Pematangsiantar dan menjadi Dosen Tetap di STIKOM Tunas Bangsa Pematangsiantar Program Studi Teknik Informatika. Telah mendapatkan Sertifikasi Dosen pada

Tahun 2018. Aktif menulis naskah kedalam bentuk jurnal penelitian dan jurnal pengabdian yang terakreditasi oleh Sinta dibidang keamanan komputer, kecerdasan buatan dan pengolahan data. Penulis juga dipercaya dan aktif untuk menjadi reviewer pada jurnal penelitian dan jurnal pengabdian terakreditasi oleh Sinta. Terakhir penulis menjabat sebagai Kepala Lembaga Sumber Daya Manusia di STIKOM Tunas Bangsa Pematangsiantar.



Eka Irawan, M. Kom, adalah dosen di Program Studi Sistem Informasi, STIKOM Tunas Bangsa Pematangsiantar. Lahir di Pondok Lama 18 januari 1989 Menyelesaikan pendidikan formal Sarjana di STT Polipropesi Medan pada Program Studi Teknik Informatika (S1) tahun 2010, Universitas Sumatera Utara di Medan pada Program Studi Magister Komputer (S2) tahun 2016 .

Sejak tahun 2010, mulai aktif sebagai dosen tetap di STIKOM Tunas Bangsa Kota Pematangsiantar .

Penulis mengajar mahasiswa D3 Program Studi Manajemen Informatika dan S1 Program Studi Sistem Informasi dan Teknik Informatika Penulis mengajar beberapa mata kuliah D3 seperti , Rekayasa Perangkat Lunak serta mata kuliah S1 seperti Algoritma Perograman, Bahasa Pemrograman, Metode Penelitian dan Analisa Perancangan Sistem Informasi.

Penulis juga aktif melakukan penelitian dan publikasi karya ilmiah di jurnal lokal maupun nasional.



Mila Nirmala Sari Hasibuan, S.Pd., M.Pd, Lahir di Solok, 04 September 1980, Beralamat Jl. Sempurna Lingk. Aek Tapa A Kel. Bakaran Batu Kec. Rantau Selatan Kab. Labuhanbatu, Sumatera Utara. Penulis menyelesaikan Pendidikan Strata 1 Universitas Muhammadiyah Tapanuli Selatan Prodi. Pendidikan Pancasila dan Kewarganegaraan pada tahun 2006. Strata 2 di Universitas Muslim Nusantara (UMN) Al – Washliyah Jurusan Pendidikan Bahasa Indonesia pada tahun 2013. Menetap di Kota Rantauprapat dan menjadi Dosen Tetap pada tahun 2010 s/d sekarang, di Universitas Labuhanbatu Prodi. Pendidikan pancasila dan kewarganegaraan Fakultas keguruan dan ilmu Pendidikan.



Yusmaidar Sepriani, S.Pd., M.Si, Lahir di Medan, 08 september 1987, Beralamat Jl. Tualang by pass, Rantau Prapat, Sumatera Utara. Penulis menyelesaikan Pendidikan Strata 1 Universitas Islam Sumatera Utara Prodi. Pendidikan dan keguruan pada tahun 2009. Strata 2 di Universitas Medan Area Jurusan Manajemen Agribisnis pada tahun 2013. Menetap di Kota Rantauprapat dan menjadi Dosen Tetap pada tahun 2014 s/d sekarang, di Universitas Labuhanbatu Prodi. Agroteknologi Fakultas Sains dan Teknologi.



Badrul Ainy Dalimunthe, S.Pd., M.Si, lahir di Medan pada 18 Januari 1976. merupakan lulusan Fakultas Pertanian Universitas Islam Sumatera Utara (UISU) dan Menamatkan Magister Manajemen Agribisnis dari Universitas Medan Area. Wanita yang kerap dipanggil Adoen ini adalah anak pertama dari pasangan Drs. Hasnan Daud Dalimunthe (ayah) dan Siti Zahara Harahap (ibu). Saat ini berprofesi sebagai dosen tetap di Fakultas Sains Dan Teknologi, Program Studi Agroteknologi di Universitas Labuhanbatu. Sebelumnya berkecimpung di dunia properti dan beberapa bank Swasta Nasional. Selama menjalani profesi dosen, pernah mendapat tugas tambahan di Unit Pusat Tenaga Kerja Dan Alumni (UPTKA) di Sekolah Tinggi Ilmu Pertanian Y-ULB dan memenangkan Hibah Program Bantuan Pengembangan Layanan Pusat Karir (BPLPK) Tahun 2018, diberikan oleh Direktorat Kemahasiswaan Ditjen Pembelajaran dan Kemahasiswaan, Kemristekdikti.



Sahat Parulian Sitorus, S.T., M.Kom, adalah Strata Satu (S1) lulusan dari Universitas Asahan Program Studi Teknik Informatika dan menyelesaikannya pada Tahun 2014 dan mendapat gelar akademik Sarjana Teknik (S.T). Kemudian penulis melanjutkan pendidikan Strata Dua (S2) di Universitas Sumatera Utara Program Studi Teknik Informatika dan memperoleh gelar akademik Magister Komputer (M.Kom). Penulis saat ini berprofesi sebagai Dosen Tetap di

Universitas Labuhanbatu Fakultas Sains dan Teknologi pada Program Studi Teknologi Informasi mulai tahun 2021 s/d sekarang, Kemudian aktif juga menulis artikel ilmiah, buku dan penulis juga beberapa kali memenangkan Program MBKM dari Kemdikbud serta program skema Penelitian Dosen Pemula pada tahun 2022. Telah mendapatkan Sertifikasi Dosen pada Tahun 2024 dan pada tahun 2025 memenangkan program skema Penelitian Dasar Fundamental - Reguler. Penulis juga, aktif menulis pada media elektronik yang berjudul "BON : Berita Online Nasional" pada halaman website <https://berita.jndi.my.id>. Penulis juga sebagai Pendamping PPH Universitas Islam Negeri Walisongo Semarang dan sekaligus pendiri "PT. Jasa Niaga Digital Indonesia", di halaman website <https://jndi.my.id>. Penulis dapat dihubungi melalui email : sahatparuliansitorus4@gmail.com

Buku Ajar Metodologi Penelitian Perancangan Sistem Informasi dan Data Sains hadir sebagai panduan bagi mahasiswa, peneliti, dan praktisi yang ingin menguasai langkah-langkah penelitian ilmiah di bidang sistem informasi dan data sains. Disusun secara sistematis, buku ini tidak hanya membahas teori metodologi, tetapi juga mengaitkannya dengan praktik nyata dalam merancang sistem informasi yang efektif serta mengolah data menjadi pengetahuan bernilai. Pembaca akan dipandu mulai dari perumusan masalah, penyusunan kerangka penelitian, pemilihan metode, hingga analisis hasil yang aplikatif. Dengan bahasa jelas dan contoh relevan, buku ini mempermudah proses belajar sekaligus memperluas wawasan tentang pentingnya metodologi yang tepat dalam menghasilkan inovasi berbasis data di era transformasi digital.



BUKU PT.JNDI

ISBN 978-634-04-5439-0 (PDF)



9

786340

454390