

MODUL PEMBELAJARAN

UML (Unified Modeling Language)

Dasar Pengembangan Perangkat Lunak dan Gim (PPLG) — Kelas X

IDENTITAS MODUL

Komponen	Keterangan
Mata Pelajaran	Dasar-Dasar Pengembangan Perangkat Lunak dan Gim
Jurusan	Pengembangan Perangkat Lunak dan Gim (PPLG)
Kelas	X
Materi	Unified Modeling Language (UML)
Alokasi Waktu	6–8 Pertemuan
Model Pembelajaran	Project Based Learning
Tingkat	Dasar
Prasyarat	Memahami dasar algoritma dan pemrograman

KATA PENGANTAR

Modul ini disusun sebagai bahan pembelajaran bagi peserta didik kelas X PPLG untuk memahami **Unified Modeling Language (UML)** sebagai salah satu dasar penting dalam proses pengembangan perangkat lunak.

Dalam pengembangan aplikasi, seorang programmer tidak hanya dituntut mampu menulis kode program. Sebelum kode dibuat, diperlukan proses analisis dan perancangan agar aplikasi yang dibangun sesuai dengan kebutuhan pengguna. UML dapat digunakan untuk menggambarkan kebutuhan, proses, interaksi, struktur, dan perilaku sebuah sistem secara visual.

Melalui modul ini, peserta didik akan mempelajari konsep UML secara bertahap, mulai dari pengertian UML, fungsi dan manfaatnya, jenis-jenis diagram UML, simbol-simbol yang digunakan, hingga membuat rancangan UML untuk sebuah proyek aplikasi sederhana.

Modul ini menggunakan bahasa yang sederhana dan contoh yang dekat dengan kehidupan peserta didik sehingga dapat digunakan sebagai **modul teori sekaligus panduan praktikum**.

DAFTAR ISI

1. Pendahuluan
2. Tujuan Pembelajaran
3. Pengertian UML
4. Mengapa UML Dibutuhkan?
5. Konsep Dasar UML
6. Aktor dan Sistem
7. Use Case Diagram
8. Activity Diagram
9. Sequence Diagram
10. Class Diagram
11. Object Diagram
12. State Machine Diagram
13. Component Diagram
14. Deployment Diagram
15. Package Diagram
16. Communication Diagram
17. Composite Structure Diagram
18. Timing Diagram
19. Interaction Overview Diagram
20. Perbandingan Diagram UML
21. Studi Kasus
22. Praktikum Proyek UML
23. Latihan
24. Evaluasi
25. Proyek Akhir
26. Glosarium

BAB 1 — PENDAHULUAN

A. Apa Itu UML?

UML (Unified Modeling Language) adalah bahasa pemodelan visual yang digunakan untuk menggambarkan, merancang, dan mendokumentasikan sebuah sistem perangkat lunak.

UML membantu developer menggambarkan sistem dalam bentuk diagram sebelum sistem tersebut dibuat menggunakan bahasa pemrograman.

Sederhananya:

UML adalah "gambar rancangan" sebelum kita membuat aplikasi.

Contohnya, sebelum membangun aplikasi perpustakaan, kita perlu mengetahui:

- siapa yang menggunakan aplikasi;
- apa saja yang dapat dilakukan pengguna;
- bagaimana proses peminjaman buku;
- data apa saja yang diperlukan;
- bagaimana hubungan antar-data;
- bagaimana objek dalam sistem saling berinteraksi.

Semua hal tersebut dapat dimodelkan menggunakan UML.

BAB 2 — TUJUAN PEMBELAJARAN

Setelah mempelajari modul ini, peserta didik diharapkan mampu:

1. Menjelaskan pengertian UML.
 2. Menjelaskan fungsi UML dalam pengembangan perangkat lunak.
 3. Menjelaskan manfaat UML.
 4. Mengenali berbagai jenis diagram UML.
 5. Menentukan diagram UML yang sesuai dengan kebutuhan.
 6. Mengidentifikasi aktor dalam sebuah sistem.
 7. Membuat Use Case Diagram.
 8. Membuat Activity Diagram.
 9. Membuat Sequence Diagram.
 10. Membuat Class Diagram.
 11. Mengenali Object Diagram.
 12. Mengenali State Machine Diagram.
 13. Mengenali Component Diagram.
 14. Mengenali Deployment Diagram.
 15. Menjelaskan diagram UML lainnya.
 16. Menganalisis kebutuhan sebuah sistem sederhana.
 17. Membuat rancangan UML berdasarkan studi kasus.
 18. Mempresentasikan rancangan UML yang telah dibuat.
-

BAB 3 — MENGAPA UML DIBUTUHKAN?

Bayangkan seorang programmer mendapatkan permintaan:

"Buatkan aplikasi perpustakaan sekolah."

Apakah programmer langsung menulis kode?

Sebaiknya tidak.

Pertama-tama harus ditentukan:

- siapa pengguna aplikasi;
- apa yang dapat dilakukan pengguna;
- bagaimana proses peminjaman;
- bagaimana proses pengembalian;
- data buku;
- data anggota;
- transaksi peminjaman;
- aturan denda;
- dan sebagainya.

Jika programmer langsung membuat aplikasi tanpa perancangan, dapat terjadi:

- kebutuhan tidak sesuai;
- kode sulit dikembangkan;
- terjadi banyak perubahan;
- programmer salah memahami permintaan pengguna;
- dokumentasi sistem tidak jelas.

UML membantu mengurangi permasalahan tersebut.

Alur sederhana pengembangan

```
graph TD; A[Kebutuhan] --> B[Analisis]; B --> C[Perancangan UML]; C --> D[Implementasi / Coding]; D --> E[Testing]; E --> F[Deployment]; F --> G[Maintenance];
```

The diagram illustrates a linear development process. It begins with 'Kebutuhan' (Requirements), followed by 'Analisis' (Analysis), 'Perancangan UML' (UML Design), 'Implementasi / Coding' (Implementation / Coding), 'Testing', 'Deployment', and finally 'Maintenance'. Each step is connected to the next by a downward-pointing arrow, indicating a sequential flow.

BAB 4 — MANFAAT UML

UML memiliki beberapa manfaat utama.

1. Mempermudah Analisis

UML membantu developer memahami kebutuhan sistem.

2. Mempermudah Komunikasi

Diagram dapat digunakan untuk berkomunikasi antara:

- programmer;
- analis sistem;
- designer;
- guru;
- client;
- pengguna.

3. Mengurangi Kesalahan

Perancangan dilakukan sebelum coding sehingga kesalahan dapat ditemukan lebih awal.

4. Menjadi Dokumentasi

Diagram UML dapat digunakan sebagai dokumentasi sistem.

5. Membantu Coding

Class Diagram, misalnya, dapat membantu programmer menentukan:

- class;
- atribut;
- method;
- relasi.

BAB 5 — KONSEP DASAR UML

Sebelum membuat diagram UML, terdapat beberapa konsep penting yang harus dipahami.

A. Sistem

Sistem adalah sesuatu yang sedang dikembangkan atau dianalisis.

Contoh:

- Sistem Perpustakaan

- Sistem Kasir
- Sistem Akademik
- Sistem Absensi
- Sistem Penjualan
- Sistem Rental

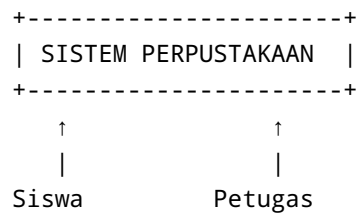
B. Aktor

Aktor adalah pihak yang berinteraksi dengan sistem.

Aktor dapat berupa:

- manusia;
- organisasi;
- perangkat;
- sistem lain.

Contoh sistem perpustakaan:



C. Use Case

Use Case adalah fungsi atau layanan yang disediakan sistem untuk aktor.

Contoh:

- Login
 - Melihat buku
 - Meminjam buku
 - Mengembalikan buku
 - Mengelola buku
 - Mengelola anggota
-

BAB 6 — JENIS-JENIS DIAGRAM UML

UML memiliki berbagai jenis diagram.

Secara umum dapat dikelompokkan menjadi:

1. Structure Diagram

Diagram yang menggambarkan struktur sistem.

Contohnya:

1. Class Diagram
2. Object Diagram
3. Component Diagram
4. Deployment Diagram
5. Package Diagram
6. Composite Structure Diagram

2. Behavior Diagram

Diagram yang menggambarkan perilaku sistem.

Contohnya:

1. Use Case Diagram
2. Activity Diagram
3. State Machine Diagram

3. Interaction Diagram

Diagram yang menggambarkan interaksi antarobjek.

Contohnya:

1. Sequence Diagram
 2. Communication Diagram
 3. Interaction Overview Diagram
 4. Timing Diagram
-

BAB 7 — USE CASE DIAGRAM

A. Pengertian

Use Case Diagram digunakan untuk menggambarkan:

siapa yang menggunakan sistem dan fungsi apa yang dapat dilakukan oleh pengguna tersebut.

Use Case Diagram merupakan salah satu diagram yang paling penting untuk tahap awal analisis sistem.

B. Komponen Use Case Diagram

1. Actor

Digambarkan dengan simbol manusia.

Contoh:

```
0
/|
/
Siswa
```

2. Use Case

Digambarkan dengan bentuk oval.

```
+-----+
| ( Login ) |
+-----+
```

3. System Boundary

Digambarkan dengan kotak yang membatasi sistem.

Contoh:

```
+-----+
| SISTEM PERPUSTAKAAN |
|                       |
```




C. Relasi Use Case

1. Association

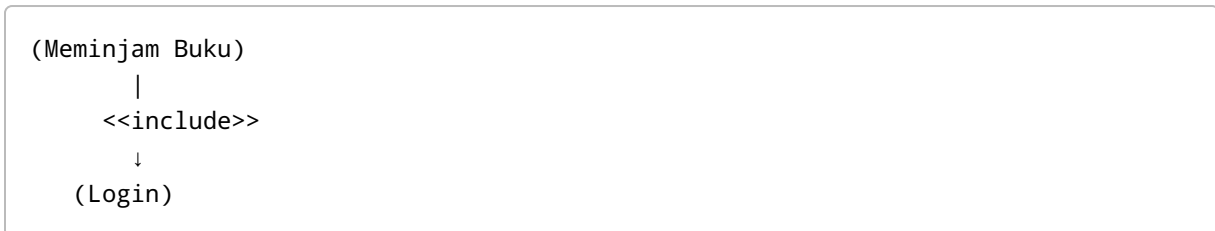
Menunjukkan hubungan aktor dengan use case.



2. Include

`include` menunjukkan bahwa sebuah use case selalu membutuhkan use case lainnya.

Contoh:



Artinya, untuk melakukan peminjaman buku pengguna harus login.

3. Extend

`extend` menunjukkan perilaku tambahan yang hanya terjadi pada kondisi tertentu.

Contoh:



```
<<extend>>
|
(Bayar Denda)
```

Jika buku terlambat dikembalikan, maka sistem menjalankan proses pembayaran denda.

D. Contoh Use Case Sistem Perpustakaan

Aktor:

- Siswa
- Petugas

Use Case:

- Login
 - Melihat buku
 - Meminjam buku
 - Mengembalikan buku
 - Mengelola buku
 - Mengelola anggota
 - Melihat laporan
-

BAB 8 — ACTIVITY DIAGRAM

A. Pengertian

Activity Diagram digunakan untuk menggambarkan alur aktivitas atau proses dalam sistem.

Jika Use Case menjawab:

"Apa yang dapat dilakukan pengguna?"

Activity Diagram menjawab:

"Bagaimana proses tersebut berlangsung?"

B. Simbol Activity Diagram

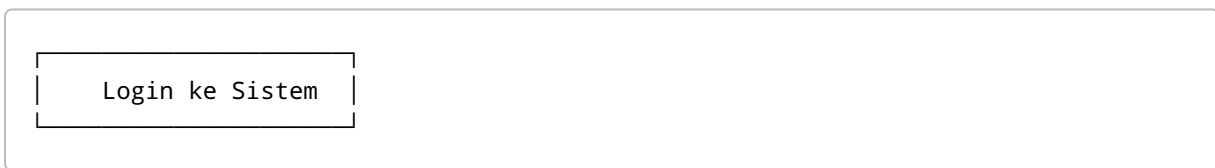
Initial Node

Menunjukkan awal proses.



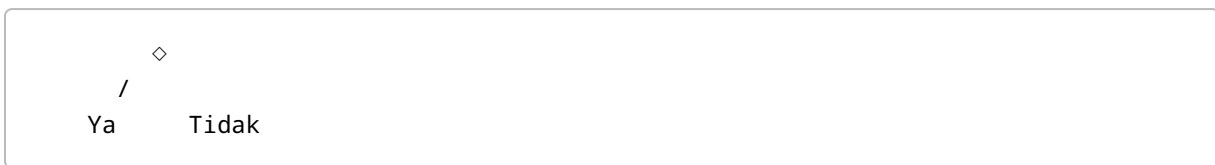
Activity

Menunjukkan aktivitas.



Decision

Digunakan untuk percabangan.

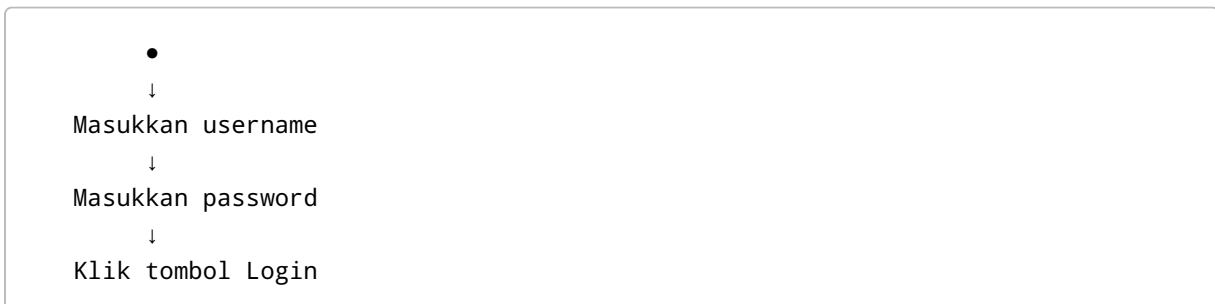


Final Node

Menunjukkan akhir proses.



C. Contoh Activity Diagram Login





BAB 9 — SEQUENCE DIAGRAM

A. Pengertian

Sequence Diagram digunakan untuk menggambarkan urutan komunikasi antara aktor, objek, dan sistem berdasarkan waktu.

Sequence Diagram menjawab pertanyaan:

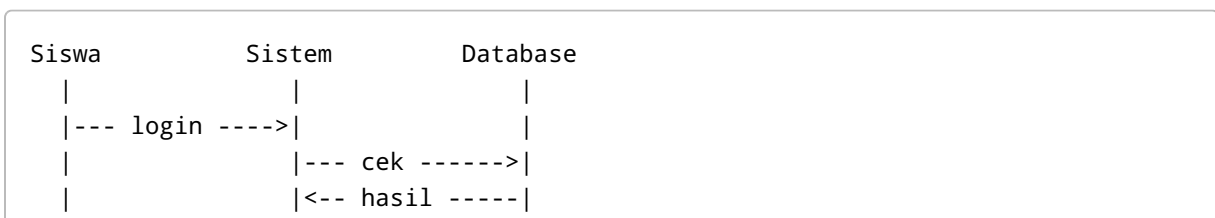
"Siapa berkomunikasi dengan siapa dan dalam urutan seperti apa?"

B. Komponen

Beberapa komponen Sequence Diagram:

- Actor
- Object
- Lifeline
- Message
- Activation
- Return message

C. Contoh Login



```
|<-- berhasil --|  
|                |  
|                |
```

Urutan:

1. Siswa memasukkan username dan password.
2. Sistem menerima data.
3. Sistem memeriksa database.
4. Database mengembalikan hasil.
5. Sistem memberikan respon kepada siswa.

BAB 10 — CLASS DIAGRAM

A. Pengertian

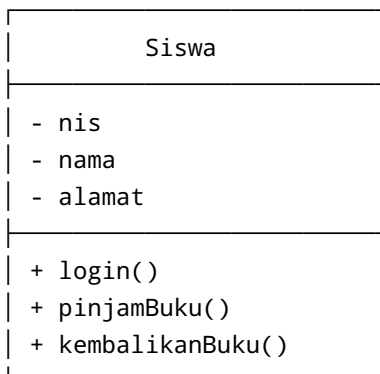
Class Diagram digunakan untuk menggambarkan struktur class dalam sebuah sistem.

Class Diagram sangat penting dalam pemrograman berbasis objek.

Class biasanya memiliki:

1. Nama class
2. Attribute
3. Method

B. Struktur Class



C. Visibility

Simbol yang umum digunakan:

Simbol	Arti
+	Public
-	Private
#	Protected
~	Package

Contoh:

```
- nama : String  
+ login() : Boolean
```

D. Relasi Class

1. Association

Hubungan antar-class.

```
Siswa ----- Buku
```

2. Generalization / Inheritance

Menunjukkan pewarisan.

```
      Pengguna  
      △  
     /  \  
    /    \  
  Siswa  Guru
```

Artinya Siswa dan Guru mewarisi karakteristik dari Pengguna.

3. Aggregation

Hubungan "memiliki", tetapi objek dapat berdiri sendiri.

Contoh:

Kelas ◇ — Siswa

4. Composition

Hubungan memiliki yang lebih kuat.

Pesanan ◆ — DetailPesanan

Jika Pesanan dihapus, DetailPesanan juga ikut dihapus.

BAB 11 — OBJECT DIAGRAM

Pengertian

Object Diagram menggambarkan objek nyata dari sebuah class pada waktu tertentu.

Contoh Class:

Siswa

Contoh Object:

```
siswa01 : Siswa  
nama = "Andi"  
nis = "1001"
```

Perbedaannya:

Class Diagram	Object Diagram
Menjelaskan struktur class	Menjelaskan instance/object

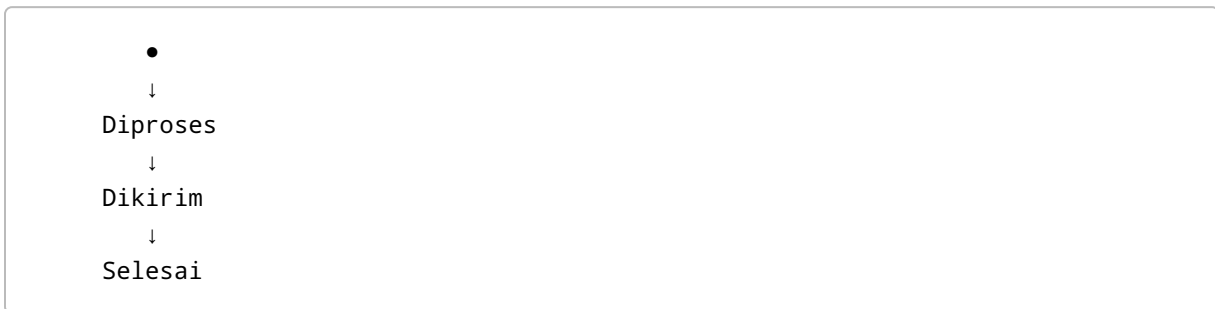
Class Diagram	Object Diagram
Bersifat umum	Contoh nyata
Siswa	andi : Siswa

BAB 12 — STATE MACHINE DIAGRAM

Pengertian

State Machine Diagram menggambarkan perubahan keadaan suatu objek berdasarkan event.

Contoh status pesanan:



Atau:

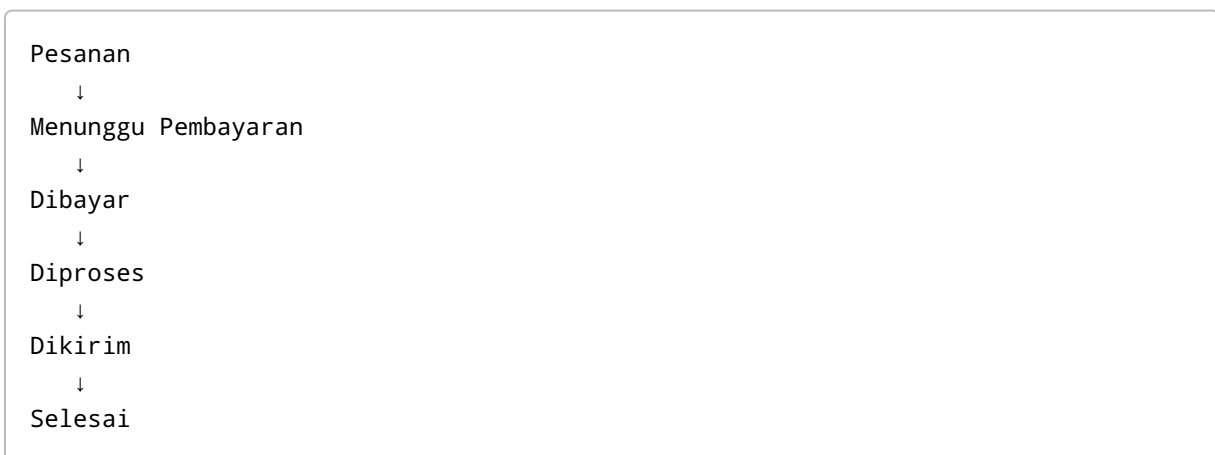


Diagram ini cocok digunakan untuk sistem yang memiliki banyak status.

Contoh:

- status pesanan;
- status tiket;

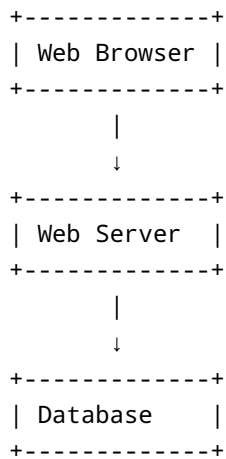
- status pengajuan;
- status pembayaran;
- status peminjaman.

BAB 13 — COMPONENT DIAGRAM

Pengertian

Component Diagram menggambarkan komponen perangkat lunak dalam sebuah sistem dan hubungan antar-komponen.

Contoh:



Dalam aplikasi modern, komponen dapat berupa:

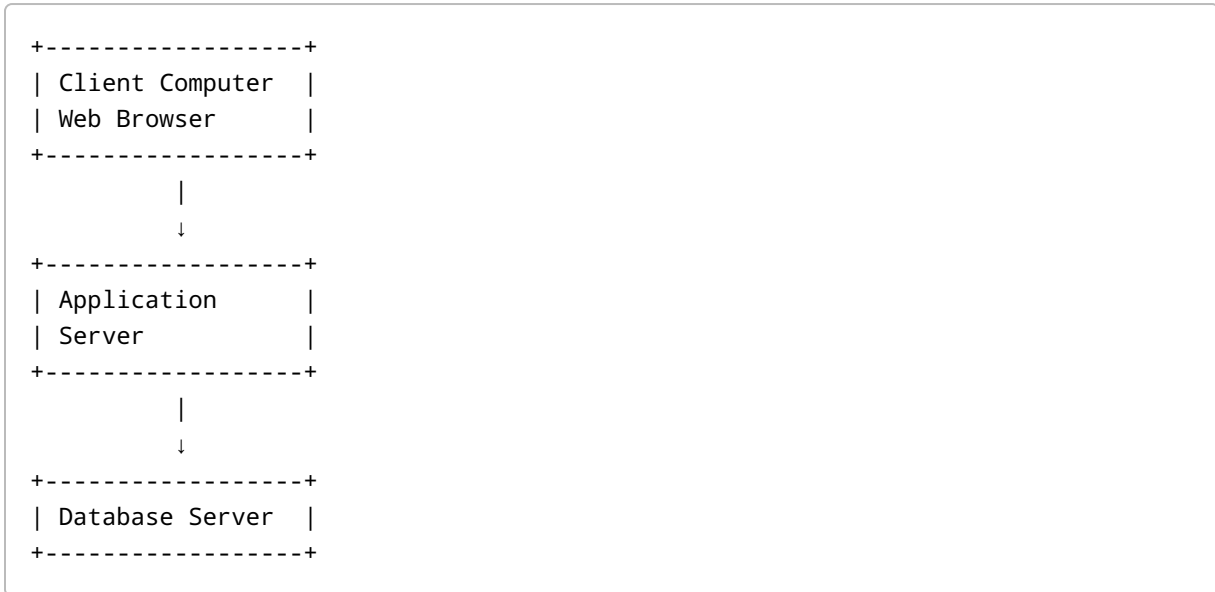
- frontend;
- backend;
- API;
- authentication service;
- database;
- file storage.

BAB 14 — DEPLOYMENT DIAGRAM

Pengertian

Deployment Diagram menggambarkan bagaimana sistem ditempatkan pada perangkat keras atau lingkungan server.

Contoh:



Deployment Diagram membantu memahami:

- server;
- client;
- database server;
- jaringan;
- perangkat yang digunakan sistem.

BAB 15 — PACKAGE DIAGRAM

Pengertian

Package Diagram digunakan untuk mengelompokkan elemen-elemen sistem ke dalam package.

Contoh:

```

+-----+
| Authentication |
| - Login       |
| - Register    |
+-----+

+-----+
| Management    |
| - User        |
| - Product     |
+-----+

+-----+
| Transaction   |
| - Order       |
| - Payment     |
+-----+

```

Package membantu membuat sistem besar menjadi lebih terorganisasi.

BAB 16 — COMMUNICATION DIAGRAM

Pengertian

Communication Diagram menggambarkan komunikasi antarobjek.

Fokus utamanya adalah:

objek mana yang berkomunikasi dan pesan apa yang dikirim.

Contoh:

```

Siswa
|
| 1: login()
↓
Sistem
|
| 2: cekUser()
↓
Database

```

Berbeda dengan Sequence Diagram, Communication Diagram lebih menekankan hubungan antarobjek.

BAB 17 — COMPOSITE STRUCTURE DIAGRAM

Pengertian

Composite Structure Diagram menggambarkan struktur internal sebuah class atau component.

Diagram ini dapat menunjukkan:

- bagian internal;
- port;
- connector;
- hubungan antarbagian.

Diagram ini biasanya digunakan pada sistem yang lebih kompleks.

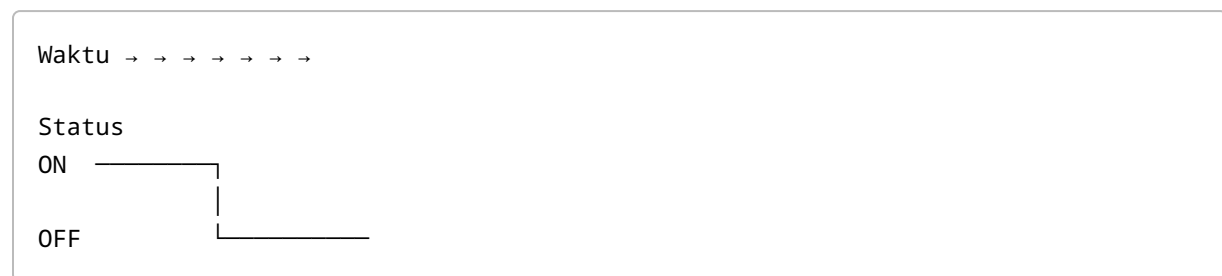
Untuk siswa kelas X, cukup memahami bahwa diagram ini digunakan untuk melihat **struktur internal sebuah komponen atau class**.

BAB 18 — TIMING DIAGRAM

Pengertian

Timing Diagram digunakan untuk menggambarkan perubahan keadaan objek berdasarkan waktu.

Contoh sederhana:



Timing Diagram banyak digunakan pada:

- sistem real-time;
- sistem perangkat keras;
- komunikasi perangkat;
- IoT;

- embedded system.

BAB 19 — INTERACTION OVERVIEW DIAGRAM

Pengertian

Interaction Overview Diagram menggambarkan gambaran umum dari berbagai interaksi atau proses dalam sistem.

Diagram ini dapat menggabungkan konsep:

- Activity Diagram;
- Sequence Diagram;
- interaction.

Contohnya proses:

```
graph TD; Login --> Pilih_Produk[Pilih Produk]; Pilih_Produk --> Checkout; Checkout --> Pembayaran; Pembayaran --> Pengiriman;
```

Pada sistem yang kompleks, diagram ini membantu melihat gambaran besar proses interaksi.

BAB 20 — PERBANDINGAN DIAGRAM UML

Diagram	Fokus Utama
Use Case	Fungsi sistem dan aktor
Activity	Alur aktivitas
Sequence	Urutan interaksi
Class	Struktur class
Object	Object/instance

Diagram	Fokus Utama
State Machine	Perubahan status
Component	Komponen software
Deployment	Penempatan sistem
Package	Pengelompokan elemen
Communication	Komunikasi antarobjek
Composite Structure	Struktur internal
Timing	Perubahan berdasarkan waktu
Interaction Overview	Gambaran interaksi

BAB 21 — STUDI KASUS

Sistem Perpustakaan Sekolah

Kita akan membuat rancangan UML untuk aplikasi:

Sistem Informasi Perpustakaan Sekolah

Pengguna

1. Siswa
2. Petugas

Fitur Siswa

- Login
- Melihat daftar buku
- Mencari buku
- Meminjam buku
- Mengembalikan buku
- Melihat riwayat peminjaman

Fitur Petugas

- Login
- Mengelola buku
- Mengelola anggota
- Memproses peminjaman
- Memproses pengembalian
- Mengelola denda

- Melihat laporan

BAB 22 — ANALISIS AKTOR

Aktor 1: Siswa

Siswa dapat:

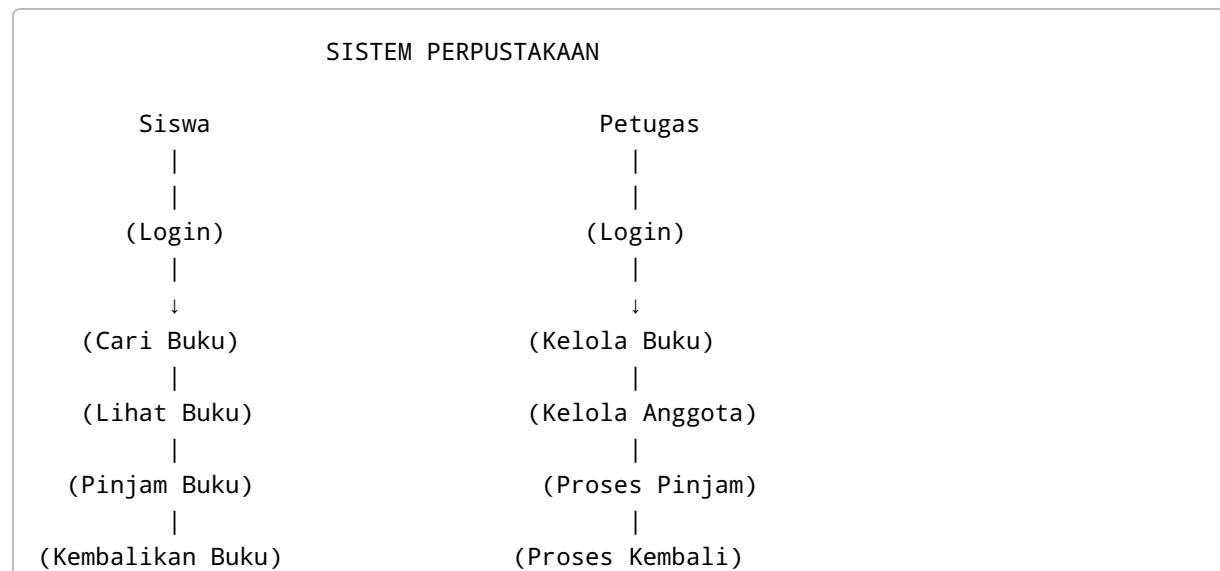
- login;
- mencari buku;
- melihat buku;
- meminjam buku;
- mengembalikan buku;
- melihat riwayat.

Aktor 2: Petugas

Petugas dapat:

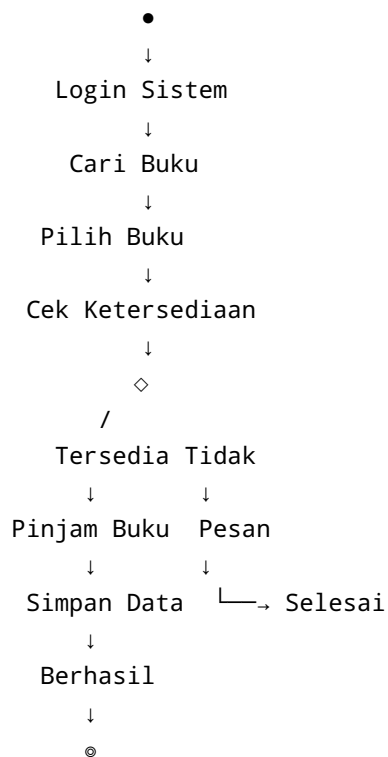
- login;
- mengelola buku;
- mengelola anggota;
- memproses transaksi;
- melihat laporan.

BAB 23 — RANCANGAN USE CASE



BAB 24 — RANCANGAN ACTIVITY

Proses Peminjaman Buku



BAB 25 — RANCANGAN SEQUENCE

Proses peminjaman:

Siswa	Sistem	Database
-- login --->		
	-- cek ----->	
	<-- valid ---	
<-- sukses --		


```
|
|-- pilih ---->|
|    buku      |
|              |
|              |-- cek ----->|
|              |-- tersedia-|
|              |
|-- pinjam -->|
|              |-- simpan -->|
|              |-- berhasil-|
|-- sukses --|
```

BAB 26 — RANCANGAN CLASS DIAGRAM

Class yang dapat ditemukan:

Siswa

```
Siswa
-----
nis
nama
kelas
alamat
-----
login()
pinjamBuku()
kembalikanBuku()
```

Buku

```
Buku
-----
kodeBuku
judul
penulis
penerbit
stok
-----
tambahBuku()
ubahBuku()
hapusBuku()
```

Peminjaman

```
Peminjaman
-----
id
tanggalPinjam
tanggalKembali
status
-----
simpan()
ubahStatus()
```

Relasi:

```
Siswa
|
| melakukan
↓
Peminjaman
|
| terhadap
↓
Buku
```

BAB 27 — LANGKAH MEMBUAT UML

Urutan yang disarankan untuk pemula:

Langkah 1 — Tentukan Sistem

Contoh:

Sistem Informasi Perpustakaan

Langkah 2 — Tentukan Aktor

Contoh:

- Siswa
- Petugas

Langkah 3 — Tentukan Use Case

Contoh:

- Login
- Cari Buku
- Pinjam Buku
- Kembalikan Buku

Langkah 4 — Buat Use Case Diagram

Gambarkan aktor dan fungsi sistem.

Langkah 5 — Buat Activity Diagram

Gambarkan alur proses.

Langkah 6 — Buat Sequence Diagram

Gambarkan komunikasi berdasarkan urutan waktu.

Langkah 7 — Buat Class Diagram

Tentukan:

- class;
- attribute;
- method;
- relasi.

Langkah 8 — Lengkapi Diagram Pendukung

Jika dibutuhkan:

- State Machine;
- Component;
- Deployment;
- Package;
- dan lainnya.

BAB 28 — TOOLS UNTUK MEMBUAT UML

Peserta didik dapat menggunakan berbagai tools pemodelan UML.

Contoh:

- Draw.io / diagrams.net
- StarUML
- Visual Paradigm
- Lucidchart
- PlantUML
- Microsoft Visio

Untuk pembelajaran kelas X, **diagrams.net (Draw.io)** sangat cocok digunakan karena antarmukanya relatif mudah dipahami.

BAB 29 — PRAKTIKUM 1

Membuat Use Case Diagram

Tujuan

Peserta didik mampu mengidentifikasi aktor dan use case.

Studi Kasus

Buat Use Case Diagram untuk:

Sistem Kantin Sekolah

Aktor

- Siswa
- Penjual

Minimal Use Case

Siswa:

- melihat menu;
- melakukan pemesanan;
- melakukan pembayaran.

Penjual:

- mengelola menu;
- menerima pesanan;
- mengonfirmasi pembayaran.

Tugas

Buat diagram menggunakan aplikasi pemodelan UML.

Output

File:

UseCase_Kantin_NamaSiswa

BAB 30 — PRAKTIKUM 2

Membuat Activity Diagram

Buat Activity Diagram untuk proses:

Pemesanan makanan di kantin

Minimal memiliki:

- initial;
- activity;
- decision;
- final.

Contoh alur:

```
Mulai
↓
Lihat Menu
↓
Pilih Makanan
↓
Pesan
↓
Bayar
↓
Pembayaran berhasil?
↙      ↘
Ya        Tidak
↓        ↓
Pesanan diproses  Bayar ulang
```

↓
Selesai

BAB 31 — PRAKTIKUM 3

Membuat Sequence Diagram

Buat Sequence Diagram proses:

Login ke Sistem Akademik

Objek minimal:

- Siswa;
- Sistem;
- Database.

Alur:

1. Siswa membuka halaman login.
2. Siswa memasukkan username.
3. Siswa memasukkan password.
4. Sistem mengirim data ke database.
5. Database memvalidasi data.
6. Sistem menampilkan dashboard atau pesan kesalahan.

BAB 32 — PRAKTIKUM 4

Membuat Class Diagram

Buat Class Diagram untuk:

Sistem Toko Online Sederhana

Minimal class:

- User
- Produk
- Pesanan
- DetailPesanan
- Pembayaran

Setiap class minimal memiliki:

- 3 attribute;
- 2 method.

Kemudian tentukan hubungan antar-class.

BAB 33 — PRAKTIKUM 5

Proyek Mini UML

Tema

Peserta didik memilih salah satu sistem:

1. Sistem Perpustakaan
2. Sistem Kasir
3. Sistem Kantin
4. Sistem Rental
5. Sistem Absensi
6. Sistem Penjualan Online
7. Sistem Akademik
8. Sistem Laundry
9. Sistem Klinik
10. Sistem Pemesanan Tiket

Diagram Minimal

Setiap kelompok wajib membuat:

- Use Case Diagram;
- Activity Diagram;
- Sequence Diagram;
- Class Diagram.

Anggota

1 kelompok terdiri dari 3–4 siswa.

BAB 34 — FORMAT LAPORAN PROYEK

Laporan minimal terdiri dari:

1. Cover

Berisi:

- nama sekolah;
- mata pelajaran;
- judul proyek;
- nama anggota;
- kelas;
- tahun pelajaran.

2. Deskripsi Sistem

Jelaskan sistem yang dibuat.

3. Identifikasi Aktor

Buat tabel:

No	Aktor	Deskripsi
1	Admin	Mengelola sistem
2	Pengguna	Menggunakan layanan

4. Use Case Diagram

Masukkan diagram.

5. Activity Diagram

Masukkan diagram.

6. Sequence Diagram

Masukkan diagram.

7. Class Diagram

Masukkan diagram.

8. Kesimpulan

Jelaskan manfaat UML terhadap proyek.

BAB 35 — LATIHAN PEMAHAMAN

Pilihan Ganda

1. UML merupakan singkatan dari...

- A. Universal Modeling Language
- B. Unified Modeling Language
- C. User Modeling Language
- D. Universal Management Language

Jawaban: B

2. Diagram yang digunakan untuk menggambarkan aktor dan fungsi sistem adalah...

- A. Class Diagram
- B. Sequence Diagram
- C. Use Case Diagram
- D. Deployment Diagram

Jawaban: C

3. Diagram yang menggambarkan alur aktivitas adalah...

- A. Activity Diagram
- B. Class Diagram
- C. Object Diagram
- D. Package Diagram

Jawaban: A

4. Diagram yang menggambarkan struktur class adalah...

- A. Use Case
- B. Activity
- C. Class
- D. Timing

Jawaban: C

5. Dalam UML, pihak yang berinteraksi dengan sistem disebut...

- A. Object
- B. Actor
- C. Class
- D. Package

Jawaban: B

6. Simbol  pada attribute atau method class umumnya menunjukkan...

- A. Private
- B. Protected
- C. Public
- D. Package

Jawaban: C

7. Sequence Diagram berfokus pada...

- A. Struktur database
- B. Urutan interaksi
- C. Tampilan aplikasi
- D. Warna aplikasi

Jawaban: B

8. Diagram yang digunakan untuk menggambarkan perubahan status objek adalah...

- A. State Machine Diagram
- B. Package Diagram
- C. Component Diagram
- D. Object Diagram

Jawaban: A

9. Diagram yang menggambarkan komponen software adalah...

- A. Component Diagram
- B. Activity Diagram
- C. Use Case Diagram
- D. Timing Diagram

Jawaban: A

10. Deployment Diagram menggambarkan...

- A. Alur program
- B. Aktor
- C. Penempatan sistem pada perangkat/server
- D. Attribute class

Jawaban: C

BAB 36 — SOAL ESSAY

Jawablah pertanyaan berikut.

1. Jelaskan pengertian UML dengan bahasa sendiri.
 2. Mengapa UML diperlukan sebelum coding?
 3. Jelaskan perbedaan Use Case Diagram dan Activity Diagram.
 4. Jelaskan fungsi Sequence Diagram.
 5. Apa yang dimaksud dengan Actor?
 6. Apa yang dimaksud dengan Use Case?
 7. Jelaskan fungsi Class Diagram.
 8. Sebutkan minimal 5 jenis diagram UML.
 9. Jelaskan perbedaan Association, Aggregation, dan Composition.
 10. Buat rancangan sederhana sistem yang ingin kamu bangun menggunakan UML.
-

BAB 37 — TUGAS ANALISIS

Perhatikan kasus berikut.

Sebuah sekolah ingin membuat aplikasi absensi siswa. Siswa dapat login dan melihat riwayat absensi. Guru dapat melakukan absensi dan melihat daftar kehadiran siswa. Admin dapat mengelola data siswa, guru, dan kelas.

Pertanyaan

1. Tentukan aktor.
2. Tentukan minimal 8 Use Case.
3. Buat Use Case Diagram.
4. Buat Activity Diagram proses absensi.
5. Buat Sequence Diagram proses absensi.
6. Tentukan minimal 5 class.

BAB 38 — RUBRIK PENILAIAN PROYEK

Aspek	Bobot
Analisis kebutuhan	15%
Use Case Diagram	20%
Activity Diagram	15%
Sequence Diagram	15%
Class Diagram	20%
Kerapian diagram	5%
Presentasi	10%
Total	100%

BAB 39 — KESALAHAN UMUM DALAM MEMBUAT UML

1. Langsung Membuat Diagram Tanpa Analisis

Kesalahan:

langsung menggambar tanpa menentukan kebutuhan.

Solusi:

identifikasi sistem dan aktor terlebih dahulu.

2. Aktor Tidak Tepat

Aktor bukan sekadar nama orang.

Contoh:

"Andi"

biasanya kurang tepat.

Lebih baik:

"Siswa"

karena menggambarkan peran.

3. Use Case Berupa Data

Contoh kurang tepat:

(Nama Siswa)
(Alamat)

Use Case sebaiknya berupa fungsi:

(Mengelola Siswa)

4. Diagram Terlalu Rumit

Untuk sistem sederhana, tidak perlu memasukkan semua diagram UML.

Gunakan diagram sesuai kebutuhan.

5. Class Tidak Memiliki Hubungan

Class Diagram harus menunjukkan hubungan antar-class jika memang terdapat hubungan dalam sistem.

BAB 40 — TIPS MEMBUAT UML YANG BAIK

Gunakan prinsip:

Sederhana

Jangan membuat diagram terlalu penuh.

Konsisten

Gunakan nama yang konsisten.

Misalnya:

Siswa

jangan berganti menjadi:

Murid
PesertaDidik
Student

jika semuanya merujuk pada objek yang sama.

Mudah Dibaca

Atur posisi elemen agar garis tidak saling bertabrakan.

Sesuai Kebutuhan

Tidak semua proyek membutuhkan semua jenis UML.

Berdasarkan Analisis

Jangan membuat diagram hanya agar terlihat lengkap.

BAB 41 — RANGKUMAN

Setelah mempelajari modul ini, dapat disimpulkan bahwa:

1. UML merupakan bahasa pemodelan visual untuk sistem perangkat lunak.
2. UML membantu proses analisis dan perancangan sistem.
3. UML dapat digunakan sebelum proses coding.
4. Actor adalah pihak yang berinteraksi dengan sistem.
5. Use Case menggambarkan fungsi sistem.
6. Use Case Diagram menggambarkan aktor dan fungsi sistem.
7. Activity Diagram menggambarkan alur aktivitas.
8. Sequence Diagram menggambarkan urutan interaksi.
9. Class Diagram menggambarkan struktur class.
10. Object Diagram menggambarkan object atau instance.
11. State Machine Diagram menggambarkan perubahan status.

12. Component Diagram menggambarkan komponen software.
13. Deployment Diagram menggambarkan penempatan sistem.
14. Package Diagram mengelompokkan elemen sistem.
15. Diagram UML harus dibuat berdasarkan kebutuhan sistem.

BAB 42 — GLOSARIUM

Istilah	Pengertian
UML	Bahasa pemodelan visual untuk sistem
Actor	Pihak yang berinteraksi dengan sistem
Use Case	Fungsi yang disediakan sistem
Class	Cetak biru objek
Object	Instance dari class
Attribute	Data/properti sebuah class
Method	Perilaku/fungsi sebuah class
Association	Hubungan antar-elemen
Inheritance	Pewarisan class
Aggregation	Hubungan memiliki yang lemah
Composition	Hubungan memiliki yang kuat
Activity	Aktivitas dalam sistem
Sequence	Urutan interaksi
State	Kondisi/status objek
Component	Bagian perangkat lunak
Deployment	Penempatan sistem
Package	Pengelompokan elemen

PROYEK AKHIR

"Rancang Sistem Impianmu dengan UML"

Setiap peserta didik atau kelompok diminta merancang sebuah aplikasi yang dekat dengan kehidupan sehari-hari.

Contoh:

Aplikasi Manajemen Kantin Sekolah

Tahapan:

```
graph TD; A[Identifikasi Masalah] --> B[Identifikasi Pengguna]; B --> C[Identifikasi Kebutuhan]; C --> D[Use Case Diagram]; D --> E[Activity Diagram]; E --> F[Sequence Diagram]; F --> G[Class Diagram]; G --> H[Presentasi];
```

Identifikasi Masalah
↓
Identifikasi Pengguna
↓
Identifikasi Kebutuhan
↓
Use Case Diagram
↓
Activity Diagram
↓
Sequence Diagram
↓
Class Diagram
↓
Presentasi

Ketentuan

Proyek harus memiliki:

- minimal 2 aktor;
- minimal 6 use case;
- minimal 1 Activity Diagram;
- minimal 2 Sequence Diagram;
- minimal 5 class;
- minimal 3 relasi antar-class.

Presentasi

Setiap kelompok mempresentasikan:

1. masalah yang ingin diselesaikan;
2. tujuan aplikasi;
3. aktor;
4. Use Case Diagram;
5. Activity Diagram;
6. Sequence Diagram;
7. Class Diagram;
8. kesimpulan.

PENUTUP

UML merupakan salah satu keterampilan dasar yang penting bagi siswa PPLG. Kemampuan membuat UML tidak hanya membantu siswa memahami teori pengembangan perangkat lunak, tetapi juga membentuk pola pikir **analitis, sistematis, dan terstruktur**.

Seorang programmer yang baik tidak hanya mampu membuat kode, tetapi juga mampu memahami **masalah yang akan diselesaikan, kebutuhan pengguna, alur sistem, serta struktur aplikasi**.

Karena itu, biasakan:

"Pahami masalah → Analisis → Rancang → Baru Coding."

Dengan memahami UML sejak kelas X, peserta didik akan memiliki dasar yang lebih kuat ketika memasuki materi pemrograman, database, web development, mobile development, maupun pengembangan proyek perangkat lunak pada kelas berikutnya.