9️⃣

Modul 9 - Class Diagram

Durasi Modul = 2-5 menit

Capaian pembelajaran:

1. Mahasiswa dapat menjelaskan definisi dan tujuan class diagram dalam konteks pemodelan perangkat lunak.
2. Mahasiswa memahami berbagai elemen dalam class diagram, seperti kelas, atribut, metode, asosiasi, dan relasi antar kelas (misalnya, inheritance dan aggregation).
3. Mahasiswa mampu merancang class diagram berdasarkan kebutuhan sistem yang telah diidentifikasi, termasuk penggambaran relasi dan hierarki antar kelas.
4. Mahasiswa dapat menggunakan class diagram sebagai alat untuk analisis dan desain sistem, termasuk mengidentifikasi entitas utama dan hubungan antar entitas.
5. Mahasiswa dapat membaca dan menjelaskan class diagram yang telah dibuat oleh orang lain, serta memberikan masukan untuk perbaikan jika diperlukan.

Deskripsi Pembelajaran :

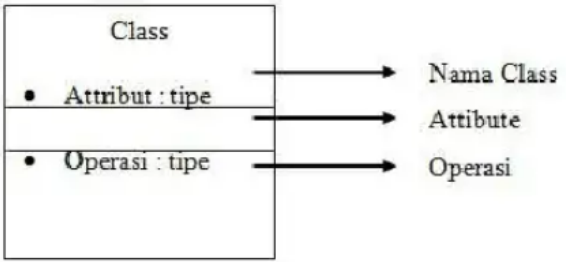
Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. Class menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). Diagram class akan memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi dan relasi-relasi antar objek. Diagram ini dapat digunakan untuk mendeskripsikan tipe objek dalam sistem dan keterhubungan mereka secara luas.

9.1 Class Diagram

Class diagram* atau diagram kelas** adalah salah satu jenis diagram struktur pada ***Unified Modelling Language (UML) yang **menggambarkan dengan jelas struktur serta deskripsi class, atribut, metode, dan hubungan dari setiap objek**. Ia **bersifat statis,** dalam artian diagram kelas bukan menjelaskan apa yang terjadi jika kelas-kelasnya berhubungan, **melainkan** menjelaskan hubungan apa yang terjadi. Diagram kelas ini sesuai jika diimplementasikan ke proyek yang menggunakan konsep *object-oriented* karena gambaran dari class diagram cukup mudah untuk digunakan.

Desain model dari diagram kelas ini sendiri dibagi menjadi dua bagian. Bagian pertama merupakan penjabaran dari database. Bagian kedua merupakan bagian dari modul MVC, yang memiliki *class interface, class control, dan class entity.*

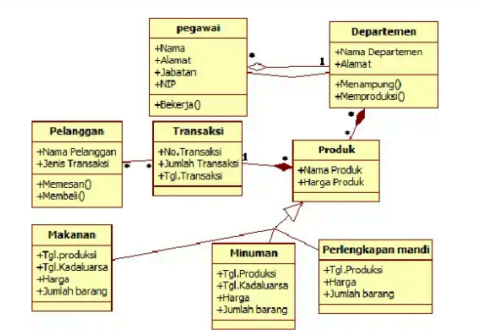
Diagram class memiliki 3 area pokok, yaitu :



Gambar 9.1 Sebuah class dalam UML

- *Class name* (dan stereotype) : Area ini berisikan nama yang akan diberikan untuk kelas tersebut.
- Atribut : Area ini akan diisi oleh elemen-elemen dari kelas yang bersangkutan.
- *Method* atau operasi : Pada area ini akan diisikan tindakan tindakan yang akan dilakukan oleh atribut dari kelas tersebut.

Contoh *Class Diagram* :



Gambar 9.2 Contoh class diagram penjualan

💡 **Keterangan:**

- **Class atau *table* departeme**n mempunya agresi dengan class atau table pegawai karena departemen ini dapat berdiri sendiri. Kemudian banyak pegawai dapat bekerja dalam satu departemen, jadi many to 1.
- **Class atau *table* transaksi** tidak dapat berdiri sendiri, sebab ia harus ada table produk. Hal ini berlaku terhadap table produk, sebab membutuhkan table departemen.
- Banyak pelanggan yang bisa melakukan banyak transaksi. Satu transaksi bisa mencakup banyak produk.

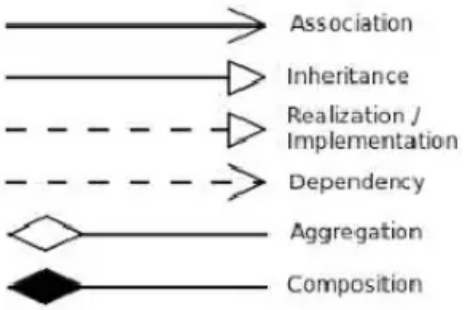
1. **Visibilitas Class Diagram**

   Visibilitas itu sendiri adalah **hak akses terhadap attribute.** Dalam *Object Oriented Programming* (OOP) tentu mengenal yang namanya Enkapsulasi. Pada class diagram pula mengimplementasikan hal tersebut. Terdapat **4 operasi visibilitas yang pelru kamu ketahui :**

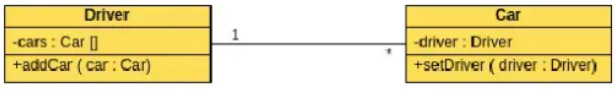
Operasi Visibilitas	**Simbol**	**Penjelasan**
Public	+	(+) Boleh diakses oleh semua kelas
Private	-	(-)Hanya boleh diakses oleh kelas itu sendiri
Protected	#	(#)Bisa diakses oleh kelas itu sendiri dan turunannya
Package	~	

(~) Bisa diakses oleh object lain pda paket yang sama

Secara umum, berikut beberapa hubungan antarkelas pada class diagram :



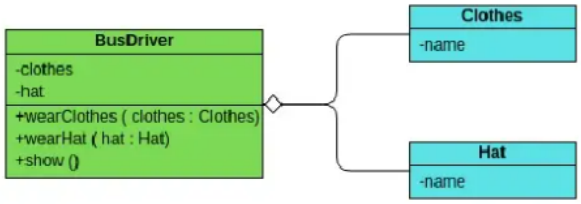
1. **Asosiasi,** yaitu hubungan statis antar kelas , biasanya menggambarkan kelas yang memiliki atribut berupa kelas lain. Terdapat banyak jenis asosiasi, misalnya :
   a.
   Asosiasi sederhana, yaitu bentuk asosiasi sederhana ( ). **Misalnya, mobil dan pengemudi, satu mobil sesuai dengan pengemudi tertentu, dan satu pengemudi dapat mengendarai beberapa mobil.**



Gambar 9.3 Hubungan antar class diagram dengan Asosiasi

b. Agregasi, yaitu hubungan yang menyatakan bagian, biasanya hubungan data master dan detailnya. Misal satu pembelian terdiri dari sejumlah item barang ( ). **Relasi agregasi juga mewakili hubungan antara keseluruhan dan sebagian kelas, objek anggota adalah bagian dari objek keseluruhan, tetapi objek anggota dapat eksis secara independen dari objek keseluruhan.** Misalnya, sopir bus dan pakaian kerja dan topi adalah bagian dari hubungan keseluruhan, tetapi mereka dapat dipisahkan. Pakaian kerja

dan topi dapat dikenakan pada pengemudi lain. Pengemudi bus juga dapat mengenakan pakaian kerja dan topi lainnya.

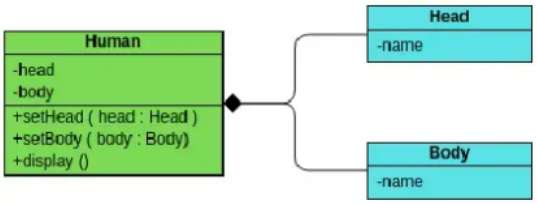


Gambar 9.4 Hubungan antar class diagram dengan Agregasi

c. ***Navigability*** menunjukan arah query/komunikasi antar objek, bisa satu atau dua arah, terlihat pada tanda panahnya ( ).

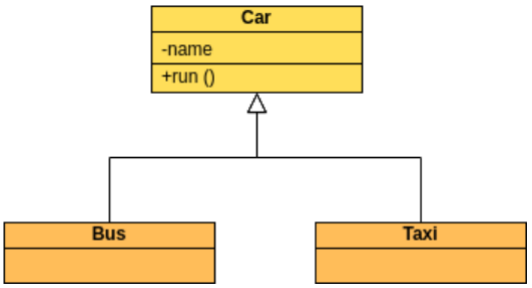
d.

Campuran / Composit / Komposisi sering disebut **campuran asosiasi ( ).** Dikatakan komposisi **karena satu objek kelas induk memiliki objek kelas anak yang lain dan objek kelas anak itu tidak dapat ada secara berarti tanpa objek kelas induk**



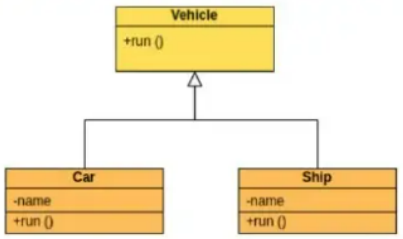
Gambar 9.5 Hubungan antar class diagram dengan Komposisi

2. **Generalization/Inheritance,** yaitu hubungan hierarkis antara anak dan bapak, **karena** kelas dapat **diturunkan dari kelas lain** dan **mewarisi semua atribut dan metode kelas asalnya serta menambahkan fungsionalitas baru ( ).** Dalam hubungan pewarisan, subkelas mewarisi semua fungsi kelas induk, dan kelas induk **memiliki semua atribut, metode, dan subkelas**. Subclass berisi informasi tambahan selain informasi yang sama dengan kelas induk. **Misalnya, bus, taksi, dan mobil adalah mobil, semuanya memiliki nama, dan semuanya dapat berada di jalan**.



Gambar 9.6 Hubungan antar class diagram dengan Generalization/Inheritance

3. **Implementasi *(realization)***, yaitu hubungan antara objek yang menjamin adanya pola khusus dalam perilaku anggota objek lainnya. Ini dapat diwujudkan dengan adanya kelas yang mengimplemtasikan interface tertentu ( ).



Gambar 9.7 Hubungan antar class diagram dengan Realisasi/Implementasi

4. **Ketergantungan *(dependency)*** yaitu sebuah kelas membutuhkan objek lain untuk bisa memfungsikan dirinya sendiri dengan baik (----------). Hubungan ketergantungan adalah hubungan “penggunaan”. Perubahan pada suatu hal tertentu dapat mempengaruhi hal lain yang menggunakannya, dan menggunakan ketergantungan bila perlu untuk menunjukkan bahwa satu hal menggunakan yang lain. **Misalnya, mobil bergantung pada bensin. Jika tidak ada bensin, mobil tidak akan bisa melaju.**

Gambar 9.8 Hubungan antar class diagram dengan ketergantungan

d. Hubungan dinamis, yaitu rangkaian pesan (message) yang di-passing dari satu kelas ke kelas lain.

Dalam diagram *Unified Modelling Language (UML),* asosiasi dua arah dapat memiliki dua panah atau tidak ada panah, dan asosiasi satu arah atau asosiasi diri memiliki panah.

💡

> Dalam hubungan multiplisitas, Anda dapat menambahkan angka secara langsung ke baris terkait untuk menunjukkan jumlah objek di kelas terkait.
>
> > 1..1 : Hanya satu
> >
> > 0..* : Nol atau lebih
> >
> > 1..* : Satu atau lebih
> >
> > 0..1 : Tidak ada atau hanya satu
> >
> > m..n : Paling sedikit m, paling banyak n (m≤n)

📚REFERENSI

Aziefah, & Nurmasari, J. (2013). Pemrograman Berorientasi Objek. Pekanbaru: Politeknik Caltex Riau.

Barclay, K., & Savage, J. (2004). Object Oriented Design with UML and Java. Elsevier.

Deitel, P., & Deitel, H. (2012). Java: How to Program (9 ed.). US: Prentice Hall.