BAB 2

LANDASAN TEORI

2.1 PT. Metanouva Informatika

PT. Metanouva Informatika adalah perusahaan yang berdiri secara resmi sebagai badan usaha pada tahun 1998 dan bergerak pada bidang Teknologi Informasi dan Multimedia. PT. Metanouva Informatika memiliki satu head office dan dua workshop, salah satu workshop PT. Metanouva Informatika adalah Digitak.

Digitak berfokus pada bidang *Web Design & Development, Customized Application, Management Information System* dan *IT-Support / Maintenance* untuk kebutuhan bisnis perusahaan dan *private* instansi yang berdiri sejak 2014. Digitak beralamat Jl. Gn. Batu Dalam Komplek Citra Asri Permai No.C-26, Pasirkaliki, Kec. Cimahi Utara, Kota Cimahi, Jawa Barat 40514.

2.1.1 Logo PT. Metanouva Informatika

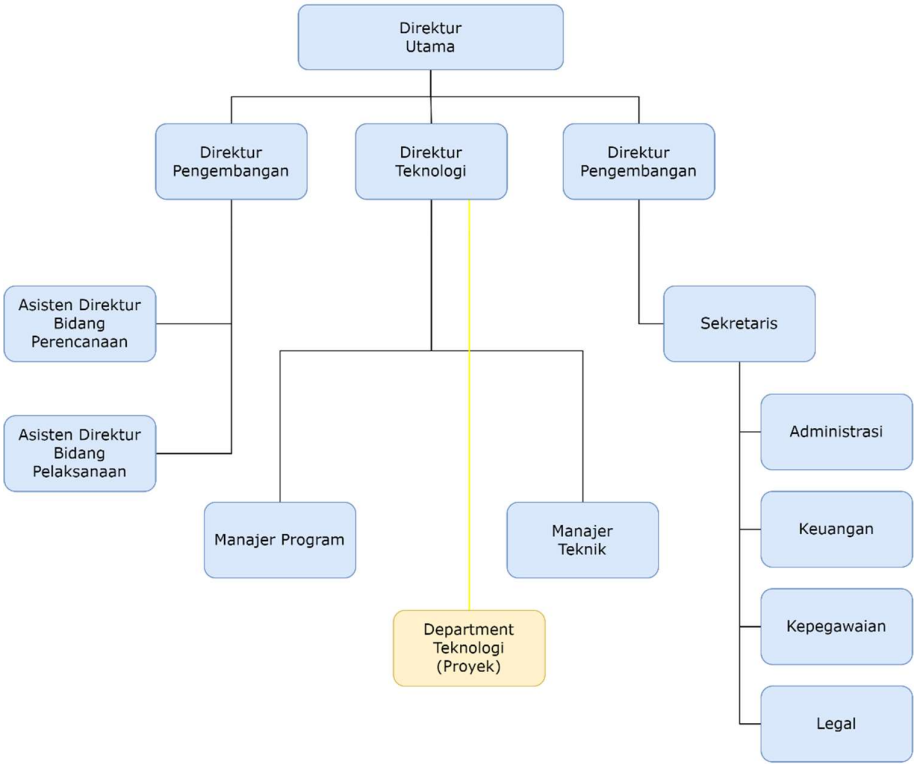
Gambar 2.2 adalah logo dari Digitak yang merupakan salah satu workshop PT. Metanouva informatika.

Gambar 2.1 Logo PT. Metanouva Informatika

Logo Digitak memiliki warna hitam dan jingga. Warna jingga melambangkan kegembiaran dan antusias perusahaan. Digitak memiliki slogan "Unlimited Innovation" yang menjelaskan bahwa Digitak tidak pernah habis untuk melakukan inovasi pada bidang teknologi.

2.1.2 Struktur Organisasi

Pada gambar 2.1 merupakan struktur organisasi yang ada pada PT. Metanouva informatika.



Gambar 2.2 Struktur Organisasi

Pada PT. Metanouva Informatika (Digitak) terdapat jabatan struktural dan jabatan fungsional, jabatan struktural adalah jabatan yang mengacu pada pengelompokan pekerjaan berdasarkan tingkat hierarki dan tanggung jawab dalam organisasi. Dalam struktur jabatan struktural, individu ditempatkan dalam level hierarki yang berbeda dam setiap level memiliki wewenang dan tanggung jawab yang berbeda. Sedangkan untuk jabatan fungsional adalah jabatan yang berfokus pada pengelompokan pekerjaan berdasarkan fungsi atau spesialisasi tertentu. Dalam struktur organisasi fungsional, individu ditempatkan dalam departemen atau unit yang berkonsentrasi pada tugas dan tanggung jawab yang serupa.

Jabatan fungsional pada digitak yaitu pada department teknologi atau proyek yang dimana berfokus untuk melakukan perancangan, pembangunan, dan pengembangan perangkat lunak.

2.2 *Project* Teknologi Informasi

Proyek teknologi informasi adalah suatu usaha kompleks pada satu waktu dengan tidak rutin yang melibatkan banyak sumber daya manusia dengan kompetensi pada bidang teknologi informasi untuk terlibat dalam kebutuhan konsumen sesuai dengan spesifikasi dan design yang telah ditentukan dengan batasan waktu, biaya anggaran, dan sumberdaya [3].

Proyek memiliki karakteristik yaitu [9]:

1. Memiliki Batasan waktu

   Proyek bersifat sementara maka dari itu proyek harus dilakukan dengan rencana yang terstruktur agar tidak terjadi kerugian, proyek memiliki waktu mulai dan waktu akhir yang dimana ditentukan oleh konsumen sesuai dengan kebutuhan.

2. Adanya Batasan atau scope

   Proyek memiliki Batasan atau ruang lingkupnya sendiri, pengerjaan proyek dibatasi dengan Batasan – Batasan kebutuhan konsumen, jika pengerjaan proyek tidak dibatasi maka proyek tersebut tidak memiliki tujuan yang jelas.

3. Penuh dengan Ketidakpastian

   Perkembangan teknologi berkonsekuensi terhadap perminataan kebutuhan konsumen yang terus berkembang. Hal ini yang membuat suatu proyek it memiliki ketidakpastian yang memungkinkan suatu proyek it gagal dilaksanakan.

4. Menghasilkan produk yang unik

   Salah satu tujuan dari proyek adalah menciptakan suatu produk yang unik atau memiliki fungsi – fungsi yang belum ada pada proyek sebelumnya. Kebutuhan konsumen dalam mengerjakan sesuatu berbeda yang dimana ini membuat suatu pekerjaan khas dan membutuhkan fungsi – fungsi yang berbeda, maka dari itu suatu proyek harus memenuhi kebutuhan suatu pekerjaan.

5. Memiliki sumber daya (manusia atau lainnya)

   Suatu proyek membutuhkan kompetensi manusia dan sumber daya baik untuk proses pengerjaan maupun untuk teknologi yang akan digunakan untuk mendukung proses pengerjaan.

6. Memiliki sasaran dan tujuan

   Tujuan dari proyek dapat berupa efesiensi dalam waktu pengerjaan, produk berkualitas, dan biaya anggaran. Kualias dari produk yang dikerjakan harus sesuai dengan perjanjian yang telah ditentukan antara pekerja proyek dan juga konsumen.

7. Adanya *stakeholder*

   Proyek harus memiliki *stakeholder*, *stakeholder* adalah orang atau kelompok yang berkontribusi atau terlibat dalam pengerjaan proyek. Berikut adalah *stakeholder* dari department teknologi (proyek) digitak*:*

- *Project Manager*
- *Project Administrator*
- *System Analyst*
- *Database Administrator*
- *IT Audit*
- *Programmer*
- *Software Tester*
- *Technical Writer*
- *Computer Operator*

2.3 Project Management

Manajemen proyek adalah suatu metode atau cara pengelolaan yang dikembangkan sejak abad ke-20 secara ilmiah dan intensif untuk menghadapi kebutuhan konsumen dan kegiatan khusus yang bersifat proyek.

Manajemen proyek bertujuan untuk efektifitas penggunaan sumber daya dan pelaksanaan kegiatan sesuai dengan tujuannya dengan mengelola biaya, manajemen waktu, control kualitas dan lainnya [9]. Berikut adalah tahapan manajemen proyek TI [9]:

- **Pendefinisian proyek**

  Menentukan tujuan dari suatu proyek dan faktor pertimbangan agar proyek berhasil.

- **Inisialilasi proyek**

  Perencanaan pada awal proyek yang menentukan sumber daya yang digunakan untuk pengerjaan proyek.

- **Perencanaan proyek**

  Penguraian tahap – tahap proyek yang dilakukan, bagaimana proyek tersebut dilaksanakan dengan melihat segitiga manajemen proyek yaitu kualitas, waktu, dan biaya.

- **Pelaksanaan proyek**

  Waktu dimana proyek dilaksanakan sesuai dengan waktu yang ditentukan.

- **Pengendalian dan pemantauan proyek**

  Pengambilan keputusan yang diperlukan berdasarkan fakta lapangan sehingga proyek dapat berjalan dengan baik dan lancar.

- **Penutupan proyek**

  Menerimana hasil dari proyek yang telah dikerjakan dan mengakhiri semua penggunaan sumber daya.

2.4 Website

Website adalah kumpulan halaman yang menampilkan informasi data teks, gambar, objek diam atau bergerak, animasi, suara, video, dan gabungan dari semua komponan tersebut baik dinamis maupun statis yang membentuk suatu rangkaian yang berkaitan satu sama lain yang dimana masing – masing dihubungkan melalui halaman jaringan (*Hyperlink*) yang dapat diakses melalui perangkat lunak peramban [10]. Website umumnya merupakan bagian dari suatu nama domain atau subdomain di *World Wide Web* (WWW) di Internet [11].

Website terbagi menjadi dua sifat yaitu dinamis dan statis. Website statis adalah web yang bersifat tidak mudah berubah isi komponennya atau tidak interaktif, perubahan dapat dilakukan dengan pengkodean [10]. Sementara website dinamis

adalah Bersifat dinamis apabila isi informasi website selalu berubah-ubah, dan isi informasinya interaktif dua arah berasal dari pemilik serta pengguna website [11].

2.5 Aplikasi

Aplikasi menurut Yuhefizar tahun 2012 adalah program yang sengaja dibuat dan dikembangkan sebagai pemenuh kebutuhan penggunanya dalam menjalankan suatu pekerjaan tertentu [12].

Pada tahun 2014 Sri Widianti berpendapat aplikasi merupakan sebuah software (perangkat lunak) yang bertugas sebagai front – end pada sebuah sistem yang dipakai untuk mengelolah berbagai macam data sehingga menjadi sebuah informasi yang bermanfaat untuk penggunanya dan juga sistem yang berkaitan [12].

2.6 *Web Browser* (Peramban)

Web browser adalah aplikasi perangkat lunak yang digunakan untuk mencari, mengambil, dan menampilkan informasi di *World Wide Web*, termasuk halaman web, foto, video, dan file lainnya. Browser juga memiliki kemampuan untuk menampilkan kode semantik seperti HTML, JavaScript, CSS dan bahasa pemrograman website pada halaman yang mudah dipahami semua orang. Ada beberapa jenis browser yang tersedia untuk pengguna Internet [11]. Sumber informasi web diidentifikasikan dengan Uniform Resource Identufuer (URL) [13].

Web browser pertama adalah mosaic yang merupakan suatu *text browser,* dan sekarang *web browser* atau peramban sudah mengalami pengembangan yang dimana berkembang kedalam bentuk multimedia seperti *google chrome*, *mozilla firefox*, *microsoft edge*, dan lainnya [14].

2.7 Use Case Diagram

Use case adalah sebuah cara pemodelan untuk membantu memodelkan sistem yang memiliki satu atau lebih *system requirements* pada UML [15], [16]. Use case menggambarkan setiap fungsionalitas yang dimiliki oleh *system.* Dan use case adalah pusat dari model *system*.

Use case sangat baik digunakan untuk segala aspek seperti *object-oriented system development, design, testing,* dan *documentation. Use case* menggambarkan *system requirement* dari bagian terluar yang menjelaskan secara detail apa yang user butuhkan [16].

Berikut adalah *attribute* dari use case diagrams:

- *Actor*

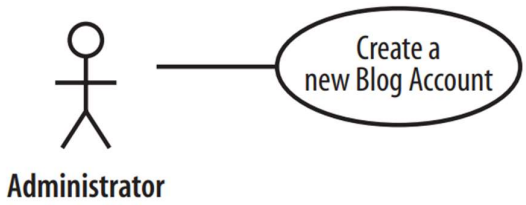
  Actor adalah orang atau sistem yang terlibat dalam fungsionalitas system yang dimodelkan. Umumnya actor digambarkan dengan *stick man* atau bisa juga *stereotyped box* [16]. Gambar 2.3 adalah contoh gambar dari actor.

Gambar 2.3 *Actor use case*
Sumber: Buku Learning UML 2.0 [16]

- *Use cases*

  Use cases merepresentasikan setiap fungsionalitas dari sistem, sebuah komponen, atau bahkan *class.* Setiap use case harus memiliki nama yang mendeskripsikan fungsinya [15]. Use case dapat diidentifikasikan dari *user's requirement*. Pada tahap ini semua *user's requirement* yang telah ditulis dipertegas agar dapat mendeskripsikan *system* dengan jelas [16]. Use case dapat digambarkan dengan bentuk oval yang tertera pada gambar 2.4 dan contoh penggunaannya digambarkan pada gambar 2.5 [16].

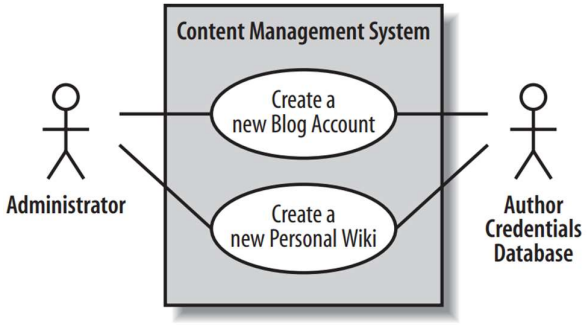
Gambar 2.4 *Use case*
Sumber: Buku Learning UML 2.0 [16]



Gambar 2.5 *Pengunaan use case*
Sumber: Buku Learning UML 2.0 [16]

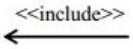
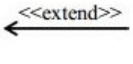
- *System boundaries*

  System boundaries adalah cangkupan dari system yang dibuat. Apabila usecase berada pada *system boundaries*, maka *actor* berada pada luar *system.* Hal ini sangat berguna untuk mendefinisikan atau menjelaskan cangkupan sistem dan menjelaskan tanggung jawab dari masing – masing *actor* saat mendesain sistem, sub-sistem, dan komponen. *System boundaries* digambarkan dengan sebuah kotak yang mengelilingi *use case* tetapi membiarkan *actor* diluar kotak. Gambar *system boundaries* dapat dilihat pada gambar 2.6 [15], [16].



Gambar 2.6 *System boundaries*
Sumber: Buku Learning UML 2.0 [16]

Pada Gambar 2.7 merupakan simbol yang digunakan pada *use case diagram* dan fungsi dari simbol yang ada pada *use case diagram.*

Simbol	Keterangan
	Aktor : Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan *use case*
	*Use case* : Abstraksi dan interaksi antara sistem dan aktor
	Association : Abstraksi dari penghubung antara aktor dengan use case
	Generalisasi : Menunjukkan spesialisasi aktor untuk dapat berpartisipasi dengan use case
	Menunjukkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya
	Menunjukkan bahwa suatu use case merupakan tambahan fungsional dari use case lainnya jika suatu kondisi terpenuhi

Gambar 2.7 Simbol *use case diagram*
Sumber: https://www.jagoanhosting.com/blog/use-case-diagram/

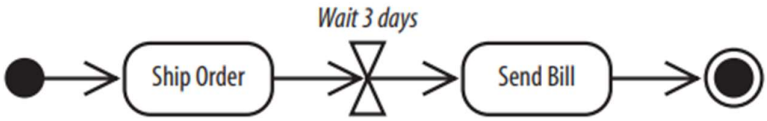
2.8 Activity Diagram

Activity diagram adalah alat pemodelan yang berfokus pada eksekusi dan perilaku sistem daripada fokus pada perakitan sistem. tidak seperti UML lainnya, Activity diagram dapat digunakan pada banyak hal bukan hanya pemodelan sistem contohnya seperti proses bisnis, *software processes*, atau *workflows*. *Activity diagram* menggambarkan aktivitas sampai aktivitas terkecil [15].

Activity diagram memungkinkan pengembang untuk menentukan bagaimana sistem mencapai tujuan dari fungsinya karena activity diagram menggambarkan aktivitas sistem secara keseluruhan [16]. Berikut merupakan komponen dari *activity diagram:*

- *Start and Finish*

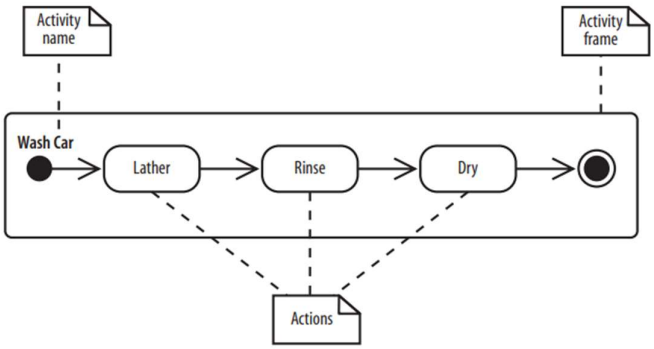
Digunakan untuk mulai dan berakhirnya aktivitas. Mulai aktivitas digambarkan dengan titik bulat dan berakhirnya aktivitas digambarkan dengan titik bulat dengan *border.* Gambar 2.8 menggambarkan *start* dan *finish*.



Gambar 2.8 *Start* dan *Finish*
Sumber: Buku Learning UML 2.0 [16]

- *Activities and Actions*

  Activities merupakan gambaran dari proses yang terjadi pada sistem sedangkan *Actions* merupakan keseluruhan dari activities yang terjadi pada sistem. *Activities* dan *Actions* digambarkan seperti kertas dengan ujung terlipat. Gambar 2.9 adalah gambaran dari *Activities and Actions.*

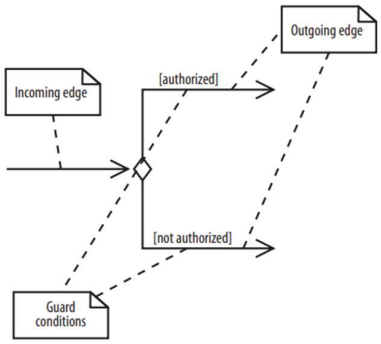


Gambar 2.9 *Activities* dan *Actions*
Sumber: Buku Learning UML 2.0 [16]

- *Decisions and Merges*

  Decisions dan *merges* digunakan untuk mengeksekusi aktivitas yang mempunyai dua kondisi yang berbeda. Pada umunya digambarkan dengan bentuk *Diamond* dengan beberapa *nodes* yang keluar atau masuk

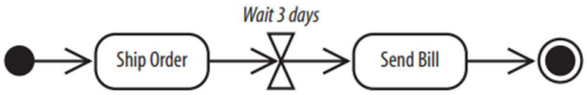
pada *diamond-shape.* Pada gambar 2.10 merupakan contoh dari *Decisions and Merges.*



Gambar 2.10 *Decisions* dan *merges*
Sumber: Buku Learning UML 2.0 [16]

- *Time Events*

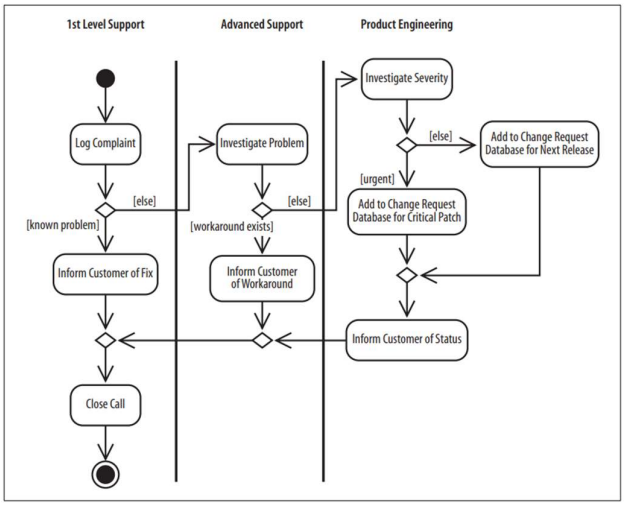
  Time events adalah penggambaran untuk sebuah waktu tunda pada sistem. waktu tunda digambarkan dengan bentuk jam pasir [16]. Gambar 2.11 menunjukan penggambarkan dari *time events.*



Gambar 2.11 *Time events*
Sumber: Buku Learning UML 2.0 [16]

- *Swimlane*

  Swimlane digunakan untuk menunjukan tanggung jawab aktivitas dari setiap partisipan yang terlibat pada sistem [16]. *Swimlane* digambarkan dengan persegi atau batasan yang dikhususkan untuk setiap partisipan. Gambar 2.12 adalah contoh dari *swinlane.*



Gambar 2.12 *Swimlane*
Sumber: Buku Learning UML 2.0 [16]

Pada gambar 2.13 akan menjelaskan simbol – simbol pada *activity diagram* dan fungsi dari simbol tersebut.

Simbol	Nama	Keterangan
	Status awal	Sebuah diagram aktivitas memiliki sebuah status awal.
	Aktivitas	Aktivitas yang dilakukan sistem,aktivitas biasanya diawali dengan kata kerja.
	Percabangan / Decision	Percabangan dimana ada pilihan aktivitas yang lebih dari satu.
	Penggabungan / Join	Penggabungan dimana yang mana lebih dari satu aktivitas lalu digabungkan jadi satu.
	Status Akhir	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
	Swimlane	Swimlane memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

Gambar 2.13 Simbol *activity diagram*
Sumber: https://www.dicoding.com/blog/apa-itu-activity-diagram/

2.9 Class Diagram

Class diagram adalah diagram fundamental pada UML. *Class diagram* digunakan untuk mengambarkan hubungan antar *class* pada sistem perangkat lunak [15]. Class adalah pusat dari semua *object-oriented system. Class diagram* mendeskripsikan perbedaan jenis objek yang dimiliki oleh sistem dan relasinya [16]. Berikut adalah bagian dari *class diagram* [16]*:*

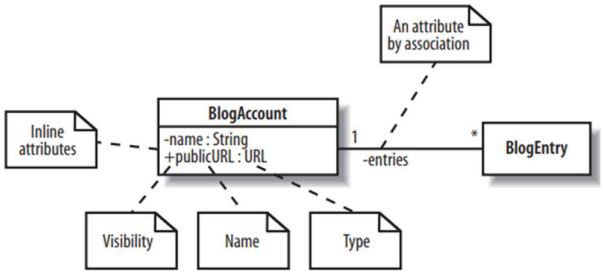
- *Class*

  Class adalah *blueprint* dari sebuah object yang memiliki *attributes*.
- *Behaviour*
- *Behaviour* adalah sifat dari object atau hal yang dapat dilakukan oleh *class (Object).*
- *Relations*

  Relation digunakan untuk menujukan relasi dari setiap class yang ada pada sistem.
- *Visibility*

  Visibility berguna untuk mendeklarasikan status dari *behaviour* atau *class* yang ada pada sistem. Terdapat empat notasi *visibility* yaitu *public (+), private (-), package (~),* dan *protected (#).*

Gambar 2.14 adalah contoh dari *class diagram*:



Gambar 2.14 *Class Diagram*
Sumber: Buku Learning UML 2.0 [16]

Gambar 2.15 merupakan simbol yang ada pada *class diagram* dan fungsi dari simbol *class diagram*:

NO	GAMBAR	NAMA	KETERANGAN
1		*Generalization*	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
2		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4		*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor
5		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
6		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri
7		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya

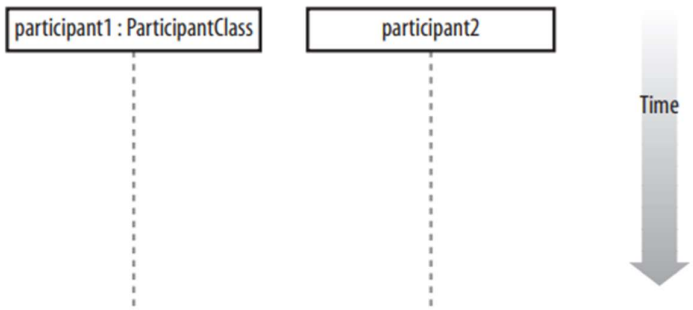
Gambar 2.15 Simbol *class diagram*
Sumber: https://www.pinhome.id/blog/contoh-class-diagram/

2.10 Sequence Diagram

Sequence diagram digunakan untuk memodelkan interaksi *runtime* yang penting antar komponen yang menyusun sistem. Berikut merupakan bagian dari *sequence diagram* [16]*:*

- *Time*

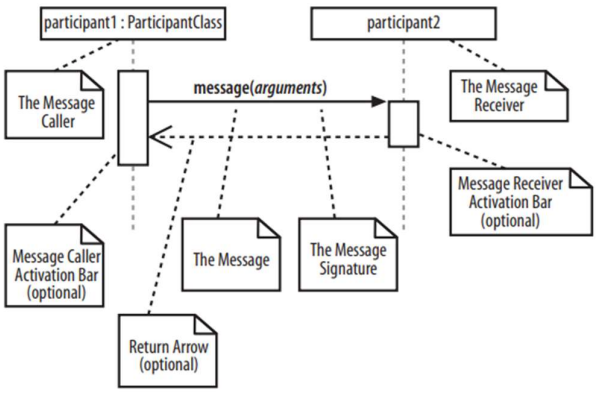
  Sequence diagram mendeskripsikan berjalannya perintah, maka dari itu waktu adalah faktor yang penting, *time* pada *Sequence diagram* merepresentasikan *order* bukan durasi. Gambar 2.16 adalah *time* pada *Sequence diagram*.



Gambar 2.16 *Time Sequence diagram*
Sumber: Buku Learning UML 2.0 [16]

- *Events, Signals,* dan *Messages*

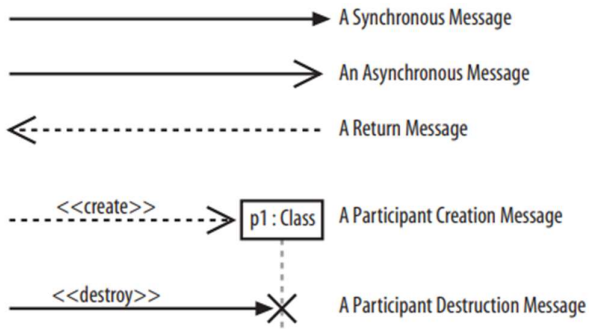
Bagian terkecil dari sebuah interaksi adalah *event. Event* digambarkan garis titik pada setiap interaksi. *Event* adalah *building blocks* untuk *signals* dan *messages. Signals* dan *messages* adalah suatu konsep yang sama dengan berbeda nama. *Signal* adalah istilah yang sering digunakan oleh system designer sedangkan software designer lebih sering menggunakan istilah *message.* Singal atau messages ditulis diantara *event.* Gambar 2.17 merupakan gambar dari *Events, signals,* dan *Messages.*



Gambar 2.17 *Events, Signals, dan Messages*
Sumber: Buku Learning UML 2.0 [16]

- *Message Arrows*

Message Arrows adalah tipe dari *arrowhead* yang setiap bentuknya memiliki arti yang berbeda [16]. Pada gambar 2.18 adalah bentuk - bentuk dari *Message Arrows.*



Gambar 2.18 *Message Arrows*
Sumber: Buku Learning UML 2.0 [16]

2.11 Php Hypertext Preprocessor (PHP)

PHP adalah salah satu bahasa pemograman *open-source* yang ditujukan untuk pengembangan web dan dapat dijalankan pada sebuah file HTML. Bahasa PHP merupakan gambaran dari beberapa bahasa pemograman yaitu C, Java, dan Perl [10]. PHP merupakan bahasa pemrograman *server-side* yang di mana program akan dijalankan di server dan hasilnya akan diintegrasikan ke dalam kode sumber HTML [4].

PHP bahasa yang dibuat untuk pelengkap HTML yang memungkinkan pembuatan aplikasi dinamis dengan fungsi pengolahan data dan pemrosesan data. Semua *syntax* yang diberikan akan sepenuhnya dijalankan pada server dan hasil akhirnya akan dikirim ke *browser* [17].

2.12 Laravel *Framework*

Laravel adalah sebuah *framework* PHP (*Hypertext Preprocessor*) yang dirilis di bawah lisensi MIT, yang dibangun dengan konsep *Model-View-Controller* (MVC). Laravel adalah *framework* PHP yang dibuat oleh Taylor Otwell dan diliris pada tahun 2011 dan sekarang berada diversi 9 (Laravel 9) [18].

Laravel memiliki banyak fitur modern seperti *blade template engine, artisan, database migration, pagination,* dan *eloquent* ORM (*Object Relation Mapping*) yang membantu proses pengembangan website dan juga dapat memudahkan programmer. [18].

2.13 JavaScript

JavaScript adalah suatu bahasa pemograman yang dikembangkan untuk dapat berjalan pada *web browser* [11]. JavaScript merupakan bahasa pemrograman web yang bersifat *Client-Side Programming Language*. Yang merupakan suatu tipe bahasa pemrograman yang dilakukan oleh client. Aplikasi client yang dimaksud mengarah kepada web browser seperti Google Chrome, Mozilla Firefox, dan lain sebagainya sehingga banyak yang dapat melihat dan mencuri isi *Source Code* javaScript tersebut [19].

Javascript berbentuk kumpulan *script* yang berjalan pada suatu dokumen HTML yang dimana memiliki kemampuan untuk menyempurnakan tampilan pada sistem *web-based-application* yang dikembangkan. Adapun karakteristik dari javascript yaitu [20]:

- Berjenis *high-level progamming language.*
- Bersifat *client-side.*
- Berorientasi pada *object.*
- Bersifat *loosely typed.*

2.14 Web Server

Web server adalah sebuah perangkat lunak yang menyediakan layanan berbasis data dengan bantuan protokol HTTP atau HTTPS dari *client-side* menggunakan aplikasi peramban untuk melakukan permintaan data dan server akan mengirimkan data dengan bentuk halaman web dan umumnya berbentuk HTML. Halaman web yang diminta bisa terdiri dari berkas teks, gambar, video, file, dan lainnya [21].

2.15 Web Service

Menurut W3C yang merupakan Lembaga internasional standarisasi, *web service* adalah sistem perangkat lunak yang didesain dapat mengoperasikan mesin ke mesin melalui jaringan. Arsitektur *web service* menurut Chen adalah memodelkan interaksi antara tiga peran yaitu penyedia, konsumen, dan pendaftar layanan tersebut. Pruter menggunakan *web service* untuk mengkoneksikan perangkat – perangkat yang sudah diketahui maupun tidak diketahui di dalam satu jaringan computer [4].

Web service adalah suatu fasilitas yang digunakan penyedia layanan yang berbentuk informasi dan data kepada sistem lain agar dapat terkoneksi untuk melakukan interaksi dengan sistem tersebut melalui layanan – layanan yang tersedia.

Web service menyimpan data dengan format JSON atau XML sehingga data yang ada dapat diakses pada sistem lain walaupun berbeda platform, sistem operasi

dan bahasa pemograman. Pada dasarnya arsitektur *web service* melibatkan tiga komponen didalamnya yaitu [22]:

- *Service Provider*

  service provider berfungsi sebagai penyedia layanan / *service* dan mengolah *registries* agar layanan dapat tersedia.

- *Service Registry*

  Berfungsi sebagai *local center* yang mendeskripsikan semua layanan / *service* yang telah terdaftar atau didaftarkan.

- *Service Requestor*

  Service requester berfungsi meminta layanan yang mencari dan menemukan layanan yang dibutuhkan dan menggunakan layanan tersebut.

2.16 Application Programming Interface (API)

API adalah antarmuka atau interface yang digunakan untuk menggunakan aplikasi atau layanan dari sebuah program tanpa perlu membuat ulang program atau layanan tersebut. API memungkinkan pengembangan pemakaian fungsi yang sudah ada pada aplikasi lain. Tujuan dari penggunaan API ini adalah untuk saling berbagi data antar aplikasi yang berbeda dan juga mempercepat proses pengembangan aplikasi dengan cara menyediakan sebuah function yang terpisah sehingga pengembang tidak perlu membuat fitur yang sama atau serupa [23].

API bekerja pada tingkat sistem operasi yang membantu aplikasi berkomunikasi dengan layer dasar dan yang lain mengikuti satu set protokol dan spesifikasi yang telah disesuaikan [23].

2.17 Representational State Transfer (REST)

REST adalah arsitektur klien – server dimana klien mengirim request pada server dan server memproses *request* dan mengembalikan sebuah respons (transaksi). Setiap transaksi bersifat independen dan tidak terkait dengan transaksi lainnya (*stateless*) [5].

REST mengirimkan perintah yang akan dikerjakan oleh server menggunakan metode-metode HTTP *request method* yang disebut *verb*. Mengacu pada penelitian Lee [10] dan Rahman [11] terdapat delapan HTTP *request method*, yaitu GET, POST, PUT, DELETE, OPTIONS, HEAD, TRACE, dan CONNECT. Dalam penggunaan API REST hanya menggunakan empat dari metode-metode tersebut, yaitu: GET, POST, PUT, dan DELETE [4].

2.18 OAuth 2.0

OAuth 2.0 merupakan sebuah protokol *single sign-on* (sso), OAuth adalah sebuah standar umum untuk autentikasi dan otorisasi yang dapat mengizinkan aplikasi pihak ketiga untuk mengakses sumber daya pada server yang menyediakan sumber daya tanpa perlu membagikan kredensial pengguna [24]. Salah satu contoh OAuth 2.0 adalah Google SSO, yaitu aplikasi pihak ketiga dapat login menggunakan akun goole.

2.19 Google Calendar API

Google calendar adalah sebuah aplikasi yang diluncurkan oleh google pada 13 april 2006 dan secara umum dirilis pada juli 2009. Pengguna harus memiliki akun google untuk menggunakan google calendar dan melakukan pengelolaan waktu. *Google Calendar API* adalah sebuah *API REST* yang dapat diakses melalui panggilan *explicit* HTTP atau melalui *google Client libraries, API* menampilkan Sebagian besar fitur yang tersedia pada *interface web google calendar* [25].

2.20 JavaScript Object Notation (JSON)

JSON adalah sebuah format pesan balikan yang berukuran kecil yang mudah dibaca dan ditulis oleh manusia, dan juga mudah ditulis dan diurai oleh mesin. JSON merupakan salah satu pesan balikan yang dapat digunakan dalam sebuah REST API [4].

JSON terbagi dalam dua struktur, yang pertama adalah gabungan name/value, atau yang biasa dikenal object atau record dalam bahasa pemrograman. dan yang

kedua adalah List value, atau yang biasa dikenal larik dalam bahasa pemrograman [4].

2.21 Phpmyadmin

Phpmyadmin adalah aplikasi *open-source* yang digunakan secara gratis untuk melakukan pemograman ataupun administrasi MySQL melalui jaringan lokal maupun internet yang mendukung fitur mengelola *database*, tabel – tabel, *fields, relations, index, user, permissions* dan lainnya [26], [27].

Perbedaan phpMyAdmin dengan MySQL yaitu terletak pada fungsinya. Phpmyadmin adalah alat untuk memudahkan pengoperasian data MySQL, sedangkan MySQL adalah sistem manajemen basis data relasional (RDBMS) yang digunakan untuk tempat penyimpanan data. Phpmyadmin digunakan sebagai alat untuk pengolahaan dan mengatur data pada MySQL [27].

2.22 Blackbox Testing

Blackbox testing adalah metode pengujian yang berfokus pada spesifikasi fungsionalitas dari perangkat lunak yang dikembangkan. *Tester* mendefinisikan kumpulan kondisi *input* dan melakukan pengetesan pada fungsional program.

Cara kerja metode *Blackbox testing* adalah mencoba program yang dikembangkan dengan mencoba memasukan data pada setiap *form* yang ada pada program yang dikembangkan. Pengujian ini diperlukan untuk mengetahui program berjalan sesuai dengan prosedur dan spesifikasi yang telah ditentukan [28].