# BAB II

# LANDASAN TEORI



## Kursus Mengemudi

Lembaga Kursus adalah salah satu penyelenggara pendidikan diluar sekolah resmi (non-formal) untuk mengembangkan kemampuan dan keterampilan diri (Mahdy et al., 2021). Kursus Mengemudi secara spesifik dapat diartikan suatu pendidikan untuk mengembangkan kemampuan dan keterampilan diri dalam mengemudikan kendaraan khususnya mobil. KBBI sendiri mendeskripsikan kursus sebagai pelajaran tentang suatu pengetahuan atau keterampilan, yang diberikan dalam waktu singkat. Atau bisa juga diartikan sebagai lembaga di luar sekolah yang memberikan pelajaran serta pengetahuan atau keterampilan yang diberikan dalam waktu singkat.

## Sistem Informasi Berbasis Web

Sistem Informasi menurut Richard Vidgen adalah sebuah kumpulan komponen-komponen yang berinteraksi, komponen yang dimaksud adalah manusia, prosedur-prosedur, dan teknologi-teknologi yang ada, dimana komponen-komponen tersebut secara bersamaan mengumpulkan, memproses, menyimpan, dan mendistribusikan informasi untuk mendukung pengontrolan, pembuatan keputusan dan pengelolaan organisasi/perusahaan. Sistem informasi sendiri berisi informasi tentang organisasi/perusahaan terkait, contohnya, kondisi tentang operasional internal mereka, dan tentang lingkungan di dalam perusahaan tersebut, sebagai contoh informasi tentang para pelanggan, para supplier, dan kompetitor-kompetitor yang ada. Tanpa adanya sistem informasi, sebuah organisasi sulit untuk bertahan. Namun, bukan berarti bahwa sistem informasi harus menggunakan teknologi-teknologi informasi yang berbentuk komputer-komputer atau jaringan internet dan komunikasi, karena banyak sekali bentuk sistem informasi. Organisasi atau perusahaan sangat bergantung pada sistem informasi, meskipun aspek-aspek formal dari sistem informasi ini masih menggunakan sistem pengarsipan berupa kertas pada era sebelum adanya teknologi informasi itu sendiri.

Menurut (Manullang, A. H., et al., 2021), *website* pada dasarnya adalah kumpulan halaman yang saling terhubung melalui jaringan internet yang biasanya berisi bermacam-macam media, media yang dimaksud diantaranya adalah teks, gambar, audio, video, dan lain lain. Secara analogi, *website* dapat diibaratkan sebagai sebuah perpustakaan digital tanpa batas halaman. Setiap halaman dalam *website* berpotensi berisi informasi yang beragam, dan pengguna dapat dengan mudah berpindah antar halaman hanya dengan mengklik tautan.

### *Hyper Text Markup Language* (HTML)

HTML adalah salah satu istilah pemrograman yang paling dikenal oleh masyarakat umum, namun, banyak perdebatan yang mengatakan bahwa sebenarnya HTML bukanlah sebuah bahasa pemrograman. Ada juga beberapa orang yang beranggapan bahwa karena dalam menulis HTML diperlukan setidaknya pemahaman dasar tentang pemrograman, maka HTML dianggap sebagai bahasa pemrograman. Mengacu dari jurnal yang ditulis oleh (Sari et al., 2022) HTML merupakan salah satu bahasa pemrograman yang digunakan untuk membuat website. HTML biasa ditulis untuk membantu perancangan struktur dasar halaman website atau bisa juga dianggap sebagai pondasi awal untuk menyusun kerangka halaman website secara lebih terstruktur sebelum masuk ke tahap desain dan fungsionalitas.

### *Cascading Style Sheet* (CSS)

Pada era modern saat ini, hampir tidak bisa kita temui rangkaian kode HTML tanpa dilengkapi CSS. Menurut (Sari et al., 2022) CSS adalah bahasa pemrograman yang ditujukan untuk memberikan modifikasi tampilan elemen-elemen web seperti *font, outline, background,* menyesuaikan tampilan website dengan ukuran layar, dan sebagainya. Jika HTML digunakan untuk menempatkan konten-konten apa saja yang ingin ditampilkan pada sebuah halaman web, CSS digunakan untuk memberikan pemahaman kepada mesin untuk melakukan modifikasi terhadap tampilan elemen dan penataan tata letak lebih lanjut.

Karena sejatinya, HTML tidak dirancang untuk menentukan aspek visual pada sebuah desain website. Sebab, fokus utama dari HTML adalah membagi struktur sebuah halaman website. Oleh karena itu, dikenalkan skrip “pendamping” untuk memperindah tag-tag HTML yaitu CSS. Selain itu, tujuan penggunaan CSS adalah untuk memberikan kesan konsisten di seluruh website.

### *Tailwind* CSS *Framework* (CSS)

(Somi, M., 2023) dalam jurnalnya menjelaskan bahwa *Framework* *Tailwind* CSS adalah *framework* CSS yang mengutamakan penggunaan kelas utilitas yang paling populer dan bertujuan untuk membangun tampilan antarmuka khusus dengan cepat dan mudah. Maksudnya, berbeda dengan *Bootstrap* yang tergolong *framework* UI *kits*, *tailwind* tidak menyediakan komponen-komponen siap pakai. *Tailwind* tidak mempunyai tema bawaan. Dengan *tailwind*, kita memberikan *style* dengan mengetikkan kelas-kelas yang sudah ditentukan sebelumnya ke kodingan HTML yang kita kerjakan.

Lebih lanjut lagi, (Somi, M., 2023) menjelaskan bahwa dengan menggunakan *tailwind*, memberikan kita kemampuan untuk mempercepat proses pemrograman tanpa kita harus menulis kode CSS di *file* lain melainkan, menulisnya secara bersamaan di kode markup HTML. Selain itu, *tailwind* memberikan kita kemampuan untuk melakukan kustomisasi secara penuh sesuai keinginan kita. Efek samping dari menggunakan *tailwind* adalah kode HTML yang dihasilkan akan jauh lebih panjang

### Javascript

Halaman website yang dihasilkan dari hanya menggunakan bahasa HTML & CSS cenderung statis dan kurang menarik. Untuk membuat tampilan yang lebih dinamis diciptakan sebuah bahasa pemrograman baru demi mengatasi kekurangan ini, yakni *Javascript*. Sebagai referensi, (Noviantoro et al., 2022) menjelaskan bahwa *Javascript* adalah salah satu bahasa pemrograman yang digunakan untuk dapat berjalan di web browser. Pada awal pengembangan bahasa pemrograman ini sempat disebut dengan nama *Mocha*, kemudian berubah penamaannya menjadi *Live-Script*, dimana ketika masa rilis, diubah lagi menjadi *Javascript*. Lebih jauh lagi dijelaskan bahwa *Javascript* adalah *script* program berbasis *client* yang dieksekusi oleh browser sehingga membuat halaman web melakukan tugas-tugas tambahan yang tidak bisa dilakukan hanya dengan memanfaatkan HTML biasa. Selain alasan yang kami sebut diatas, beberapa interaksi yang ingin kami munculkan pada aplikasi ini tidak dapat diselesaikan hanya dengan menggunakan HTML dan CSS.

### jQuery

(Sahrudin A. et al., 2023) menyebutkan pada jurnalnya menjelaskan jQuery sebagai berikut. Pada tahun 2006, John Resig memperkenalkan jQuery, sebuah *library* Javascript yang revolusioner. Sebelum hadirnya jQuery, para pengembang perangkat lunak harus menulis kode Javascript yang kompatibel dengan berbagai macam *browser*. jQuery hadir sebagai terobosan, memberikan para pengembang perangkat lunak kemampuan untuk menulis kode Javascript yang ringkas, mudah dipahami, serta dapat dieksekusi di berbagai *browser*.

### MySQL

MySQL merupakan sistem manajemen basis data relasional (RDBMS) yang populer dan banyak digunakan di berbagai platform (Hermiati R. et al., 2021). Dikenal dengan keunggulannya dalam mendukung skrip PHP, MySQL menawarkan kemudahan integrasi dengan bahasa pemrograman web yang populer ini. Salah satu keunggulan utama MySQL adalah sintaks kuerinya yang sederhana dan mudah dipahami. Bahasa Structured Query Language (SQL) yang digunakan MySQL dirancang dengan intuitif, memungkinkan pengguna untuk mengelola data dengan efektif dan efisien. Selain itu, MySQL terkenal dengan kecepatannya yang luar biasa dalam memproses dan mengakses data. Kecepatan ini menjadikannya pilihan ideal untuk aplikasi web yang membutuhkan performa tinggi dan skalabilitas yang baik.

### *Hypertext Preprocessor* (PHP)

PHP atau *Hypertext Preprocessor* sebagaimana yang dijelaskan (Adrianto S., 2021) adalah sebuah bahasa pemrograman untuk membuat web yang bersifat *server-side* *scripting*, PHP memungkinkan untuk membuat halaman web bersifat dinamis. Selain itu, PHP membutuhkan *Database Management System* (DBMS) untuk dijalankan secara bersamaan. DBMS yang paling populer di kalangan pelajar pemrograman salah satunya adalah MySQL, namun, PHP juga mendukung DBMS lain seperti Oracle, Microsoft Access, Interbase, D-Base, PostgreSQL, dan DBMS yang lainnya.

Satu hal yang mungkin menjadi pertanyaan adalah apabila PHP dikenal dengan *Hypertext Preprocessor*, lalu apa kepanjangan PHP? Menurut tulisan yang diterbitkan melalui web sekawanstudio.com yang ditulis oleh (Miranda R.A., 2023) Pada tahun 1994, ketika Rasmus Lerdorf pertama kali menemukan *hypertext preprocessor*, beliau menggunakannya untuk memantau jumlah pengunjung atau yang sering kita sebut dengan *traffic website* dari halaman web pribadi nya atau dalam bahasa inggris dikenal dengan *Personal Home Page*. Alasan tersebutlah yang menyebabkan bahasa pemrograman ini dijuluki sebagai PHP.

### *Framework* PHP Laravel

Menurut penjelasan dari (Subecs, 2021) Laravel merupakan *framework* PHP yang paling sering digunakan untuk *programmer* pemula dan berpengalaman. Laravel dianggap mampu mengurangi durasi pengembangan sistem perangkat lunak serta mempersiapkan pasar dengan metode PHP berorientasi objek yang lebih modern. *Syntax-syntax* ekspresif dan *function-function* modern yang dimiliki Laravel disukai oleh para *programmer* yang ingin mengembangkan web atau aplikasi yang lebih kompleks. Dengan menggunakan *framework* ini diyakini dapat mempermudah proses pengembangan karena Laravel menggunakan sistem paket modular dimana modul-modul yang disediakan saling terkait satu sama lain, dimana kita bisa mengembangkan sistem perangkat lunak yang lebih luas lagi. *Framework* ini memberikan kita jalan pintas yang memungkinkan *programmer* berkonsentrasi terhadap masalah-masalah yang lebih penting.

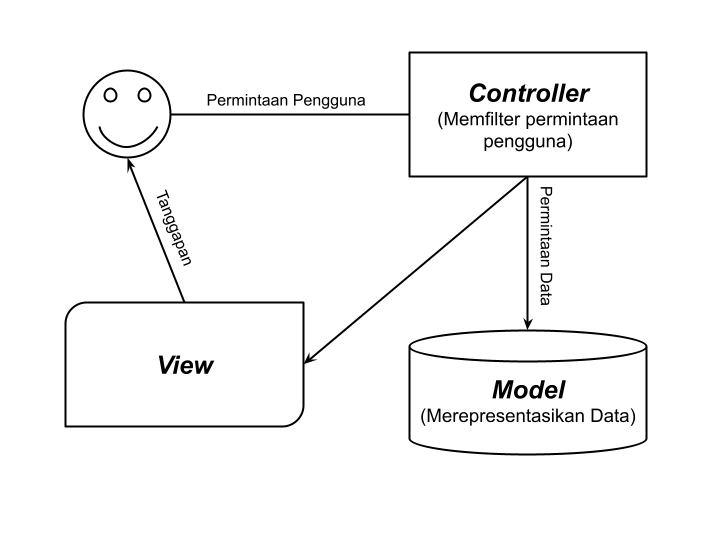
Lebih lanjut (Subecs, 2021) menyebutkan fitur-fitur atau kelebihan-kelebihan yang dimiliki Laravel yang menjadikan sebab mengapa banyak *programmer* menyukai *framework* ini. Kelebihan-kelebihan tersebut diantaranya adalah:

#### Routing

Fungsi utama dari sebuah *framework* sistem perangkat lunak adalah untuk menerima permintaan dari pengguna dan mengirim tanggapan mereka biasanya melalui HTTP(S). Artinya bahwa tugas pertama selama proses penerapan ini ialah menentukan rute-rute yang dibutuhkan. Tanpa rute, sebuah sistem tidak akan mampu mencapai pengguna. Rute itu sendiri pada dasarnya merupakan sebuah URI (*Uniform Resource Identifier*) yang memungkinkan komunikasi dengan dunia luar dengan alamat URL yang diketahui. Pada dasarnya *client* mengirimkan sebuah permintaan untuk mendapatkan rute yang ditentukan, sesuai dengan kebutuhan, meneruskannya ke sebuah *controller* yang memproses permintaan tersebut. Folder */routes* yang berada di dalam folder *root* projek yang berisi *file-file* rute tersebut.

#### Arsitektur MVC

Laravel dibangun diatas pondasi arsitektur MVC. *Model-View-Controller (MVC)* merupakan sebuah pola perancangan yang mencakup tiga bagian utama: *model, view,* dan *controller.* Komponen-komponen ini sudah dikonfigurasi untuk menangani aspek-aspek pengembangan khusus dari aplikasi. Komponen *View* ditujukan untuk menangani logika tampilan pengguna. Komponen *Controller* menerima input data dan memproses input tersebut. Komponen *Controller* ini bekerja sebagai antarmuka antara komponen *model* dan *view.* Sedangkan komponen *Model* adalah logika yang memiliki koneksi dengan data-data yang terkait user. Komponen ini merupakan komponen utama dari pola arsitektur MVC dan merepresentasikan data-data yang ditransaksikan antara *view* dan *controller*. Gambar 2.1 dibawah ini merupakan Skema untuk menjelaskan alur implementasi MVC.



Gambar 2. 1 Skema MVC

(Sumber: Subecz, Z., 2021)

#### Views dan Templates

*Views* bertanggung jawab untuk menampilkan tanggapan dari *controller* sesuai dengan format yang tepat, biasanya sebagai halaman website. *Views* dan relasinya dengan *Controller* ini bisa dikembangkan dengan mudah menggunakan bahasa *Blade template* atau lebih sederhana lagi dengan skrip-skrip PHP biasa. *View* adalah objek yang paling umum kembali ke rute-rute. *View* mendapatkan data dari rute-rute atau *controller* dan menyisipkannya ke sebuah template, sehingga mereka membantu satu sama lain untuk memisahkan logika bisnis dari logika presentasi.

#### Controller

Pada sebuah *framework MVC*, *Controller* sebagaimana namanya, mengontrol alur data antara dataset-dataset dan *views*. Hal ini merupakan mekanisme yang membawa pengguna-pengguna ke lapisan presentasi. *Controller* menerima permintaan, memproses mereka dan mengirimkan tanggapan yang sesuai. *Controller* menangani tugas-tugas seperti menarik data dari *database*, menangani penerimaan data dari form-form dan menyimpan data ke *database*. *Controller* biasanya berada di folder *app/http/controllers*.

#### Berinteraksi dengan database

Ada 3 proses interaksi terbaik dengan *database* dari *framework* Laravel menurut (Subecs, 2021):

1. *Migration*

Migrasi adalah skrip-skrip PHP yang digunakan untuk memanipulasi struktur dan konten *database*. Seorang *programmer* yang bekerja dalam sebuah tim diharapkan untuk mampu dengan baik mensinkronisasi tugas-tugas *database* dengan rekan-rekan *programmer* lain. Dalam hal ini, migrasi beroperasi sebagai sebuah *version control*. Migrasi-migrasi ini dikirimkan bersama dengan detail waktu migrasi, sehingga migrasi ini dieksekusi berdasarkan urutan yang benar. Dengan memanfaatkan migrasi ini memungkinkan *database* memiliki struktur keadaan yang konsisten.

1. *Database Seeding*

Dalam *framework* Laravel, kita bisa mengisi *database* dengan pembantu pengisian *database*, daripada melakukannya dengan manual. Dengan melakukan pengisian ini, kita bisa meng-*upload* data pengujian ke *database* menggunakan sebuah cara sederhana. Kita mampu mencari *file-file* pengisian ini di folder *database/seeders*.

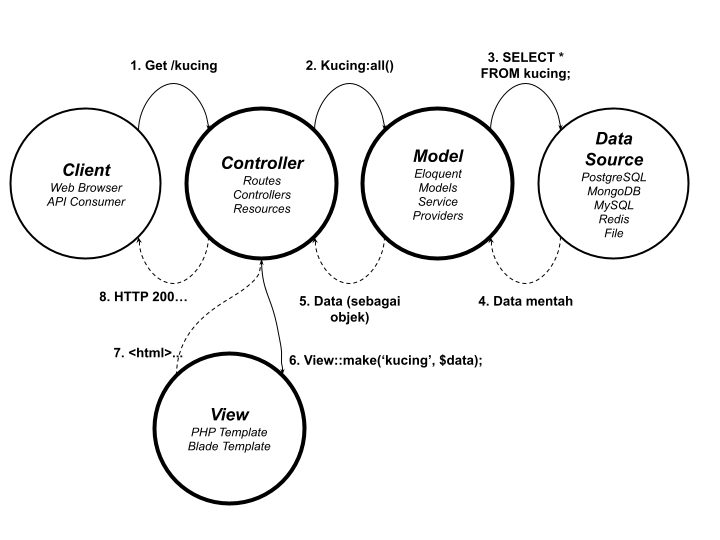
1. *Query Builder*

Laravel memberikan kita *query builder* yang mudah digunakan yang ditujukan untuk membuat dan mengeksekusi *query-query database*. *Programmer* dapat menggunakannya untuk bermacam-macam operasi *database* dan bekerja dengan sempurna dengan sistem-sistem *database* yang didukung Laravel.

#### Model dan Eloquent

*Model* seringkali dihubungkan dengan sumber-sumber informasi aplikasi dan seringkali digunakan bersama dengan rekaman-rekaman *database*. *Model* ini merupakan kelas-kelas yang merepresentasikan entitas-entitas di dalam aplikasi, seperti pengguna, artikel berita, atau *event-event*. Laravel meluncurkan sebuah fitur yang disebut dengan *Eloquent*, yang merupakan pemetaan hubungan antar objek yang efisien dimana para *programmer* dapat menentukan entitas, kemudian memasukkannya ke tabel-tabel *database* terkait dan memanfaatkan metode-metode PHP untuk mengeksekusi *entry* entitas-entitas tersebut, daripada melakukan *statement* SQL secara mentah. Dengan fitur ini, kita bisa menjalankan *query* database ORM dengan efisien, tanpa menuliskan instruksi-instruksi SQL apapun. *Eloquent* berjalan sebagai sebuah lapisan-model pada aplikasi yang dikembangkan.

Gambar 3.2 berikut adalah ilustrasi bagaimana interaksi yang terjadi dari masing-masing komponen ketika diberikan perintah untuk menampilkan semua data yang ada pada tabel kucing.



Gambar 2. 2 Ilustrasi *Model* dan *Eloquent*

(Sumber: Subecz, Z., 2021)

Kita ketahui *client* (melalui *web browser* atau *API Consumer*) ingin menampilkan semua data yang tersimpan pada tabel kucing. Perintah ini diteruskan menuju *Controller* terlebih dahulu untuk diproses permintaan data *client* kemudian menentukan *routes* sumber pengambilan data. Selanjutnya *Model* menangkap perintah pengambilan data dari tabel kucing dan mengirimkannya ke sumber data. Selanjutnya, data yang diambil dari sumber data ini dikembalikan lagi ke *Model* untuk diperiksa apakah data yang diambil hasilnya sesuai dengan perintah yang diberikan, selanjutnya karena Laravel mengusung konsep OOP, data ini ditangkap sebagai objek oleh *Controller*, yang kemudian dibawa ke *View* untuk diberikan skrip PHP yang selanjutnya disisipkan kedalam kode HTML untuk dikembalikan lagi ke *Controller* agar ditampilkan ke *Client*.

#### Authentication dan Authorization

Proses pendaftaran dan *login* pengguna pada sistem perangkat lunak adalah suatu fitur yang umum dan penting. Laravel memberikan *tools-tools* berbeda untuk memanfaatkan fungsi-fungsi yang dibutuhkan pada proses tersebut menjadi lebih aman dan mudah. Disamping sistem autentikasi dari Laravel, kita bisa menggunakan metode sederhana untuk proses otorisasi operasi pengguna ketika mereka akan mengakses sebuah sumber data. Meskipun pengguna sudah sah secara autentikasi, tetapi belum tentu mereka berhak membaca atau mengubah isi beberapa model-model *Eloquent* yang ditangani aplikasi atau rekaman rekaman *database*.

#### Middleware

*Middleware* dimanfaatkan untuk mem-*filter* permintaan HTTP pada saat pengguna / *client* memasuki aplikasi. *Programmer* dapat menggunakan *middleware* untuk lakukan tugas-tugas berikut ini: pemeriksaan autentikasi dan otorisasi, validasi sesi dan modifikasi variabel sesi, membaca, mengatur atau memodifikasi *header* permintaan dan tanggapan; merekam transaksi data; pemanggilan API, dan lain-lain.

#### Validation

Kelas validatorLaravel membantu memvalidasi fungsionalitas elemen-elemen seperti form, model *database*. Validator ini mendukung penerimaan input data; mendeklarasikan peran-peran validasi khusus dan mendeklarasikan pesan-pesan validasi khusus.

#### Pengamanan aplikasi

Sebelum meluncurkan sistem perangkat lunak kita ke lingkungan publik, *programmer* harus mempertimbangkan beberapa pertanyaan-pertanyaan tentang keamanan. Laravel melindungi sistem perangkat lunak terhadap serangan-serangan yang sering terjadi. Beberapa diantaranya adalah: *Cross-site request forgery* (CSRF), *Cross-site scripting* (XSS), SQL *injection*, *mass assignment vulnerability.*

## Rekayasa Perangkat Lunak

Secara bahasa, *Software Engineering* atau yang sering disebut dengan Rekayasa Perangkat Lunak dalam bahasa indonesia (RPL) tersusun dari 2 kata, *Software* dan *Engineering*. *Software* atau perangkat lunak bukan hanya sebuah program seperti yang diasumsikan banyak orang. Program adalah kode komputer yang dapat dieksekusi, yang tujuannya untuk melakukan satu atau lebih komputasi tertentu. Sedangkan *software* adalah kumpulan dari kode pemrograman komputer yang dapat dieksekusi, terorganisir dan terdokumentasi. Lalu, *Engineering* sendiri adalah semua hal yang meliputi pengembangan produk (baik fisik maupun digital) dengan menerapkan dan memanfaatkan prinsip-prinsip dan metodologi ilmiah yang tersusun dengan baik.

Sehingga, *Software Engineering* merupakan salah satu bagian dari *Engineering* yang berurusan dengan rekayasa perangkat lunak yang disusun secara baik dan menerapkan prinsip-prinsip, prosedur dan metodologi ilmiah. Hasil akhir dari RPL tentunya adalah perangkat lunak yang efisien, stabil dan bebas dari *bug*. Namun, apabila kita mengacu kepada pengertian menurut organisasi IEEE, RPL adalah (1) Penerapan dari sebuah pendekatan yang sistematik, disiplin, dan terukur untuk mengembangkan, mengoperasikan dan memelihara sebuah perangkat lunak; dalam hal ini penerapan rekayasa perangkat lunak. (2) Studi tentang pendekatan-pendekatan dari pernyataan diatas.

Selain IEEE, Fritz Bauer, ilmuwan komputer asal Jerman, mendefinisikan RPL sebagai “…pembentukan dan pemanfaatan prinsip teknik suara agar tercipta perangkat lunak yang ekonomis, yang stabil dan bekerja secara efisien di mesin sesungguhnya.”. Mengutip dari modul pembelajaran Rekayasa Perangkat Lunak – Pendekatan Terstruktur & Berorientasi Objek karya (Bahar et al., n.d.) Rekayasa perangkat lunak (*software engineering*) merupakan suatu disiplin ilmu yang membahas semua aspek produksi perangkat lunak (*software*), mulai dari tahap awal kajian spesifikasi / kebutuhan sistem sampai pemeliharaan sistem setelah digunakan (Sommerville, 2016). Pada definisi ini, ada dua istilah kunci: yang Pertama ‘Disiplin rekayasa’, yang berarti bahwa teknisi RPL membuat suatu alat bekerja. Mereka menerapkan teori, metode, dan alat bantu yang sesuai, selain itu mereka menggunakannya dengan selektif dan selalu mencoba mencari solusi terhadap permasalahan, walaupun tidak ada teori atau metode yang mendukung.

### *Incremental Model*

Model pengembangan perangkat lunak *incremental model* dilakukan dengan membagi projek perangkat lunak menjadi bagian-bagian yang lebih kecil atau dalam istilah *incremental model* disebut dengan *iterative build.* Gambar 2.3 dibawah ini adalah siklus atau skema pengembangan perangkat lunak untuk *incremental model*.



Gambar 2. 3 Siklus Pengembangan Perangkat Lunak *Incremental Model*

(**Sumber**: Sachan D., 2024)

Fitur utama dari model ini adalah proses iteratifnya, di mana setiap iterasi yang dilakukan selanjutnya dikembangkan, diuji, dan diintegrasikan satu per satu. Dengan membagi projek ke bagian-bagian kecil, tim pengembang dapat memprioritaskan fitur dan mengadaptasi perubahan-perubahan kecil dengan lebih efisien. Untuk lebih jelasnya kami akan menjelaskan setiap tahapan-tahapan di dalam *incremental model* :

1. **Analisa Kebutuhan** merupakan tahap awal yang krusial dalam pengembangan perangkat lunak. Pada tahap ini, tim pengembang berfokus pada pengumpulan dan analisis kebutuhan perangkat lunak secara menyeluruh. Kebutuhan yang didapatkan berfungsi sebagai fondasi yang kokoh untuk memandu fase-fase pengembangan selanjutnya. Hasil akhir dari Requirements Gathering adalah dokumen yang berisi uraian rinci mengenai kebutuhan perangkat lunak. Dokumen ini harus ditulis dengan jelas, terstruktur, dan mudah dipahami oleh semua pihak yang terlibat dalam proyek.

## *Business Process Model and Notation* (BPMN)

*Business Process Model and Notation* adalah standar khusus untuk digunakan sebagai benchmark untuk pemodelan proses bisnis yang menghasilkan notasi grafis untuk memvisualisasikan proses bisnis (Firdaus et al., 2022). *Business Process Model and Notation* (BPMN) merupakan standar global yang diakui untuk pemodelan proses bisnis. Dirancang untuk memvisualisasikan dan menganalisis proses bisnis secara jelas dan mudah dipahami, BPMN telah menjadi alat penting bagi para pemangku kepentingan dalam berbagai organisasi. BPMN menyediakan notasi grafis yang terstruktur dan konsisten untuk merepresentasikan langkah-langkah, keputusan, dan aliran data dalam suatu proses bisnis. Notasi ini memungkinkan para pelaku bisnis, analis, dan pengembang perangkat lunak untuk memahami proses bisnis dengan cara yang sama, sehingga meningkatkan komunikasi dan kolaborasi di antara mereka.

## *Unified Modeling Language* (UML)

Menurut buku dengan judul “*The Unified Modeling Language Reference Manual”* UML adalah tujuan umum dari bahasa pemodelan visual yang digunakan untuk menspesifikasi, memvisualisasikan, menyusun, dan mendokumentasi hal-hal yang terkait sistem perangkat lunak. UML mencatat semua keputusan dan pemahaman mengenai sistem-sistem yang wajib dibangun nantinya. Selain itu, UML digunakan untuk memahami, merancang, mengeksplorasi, mengkonfigurasi, memelihara, dan mengontrol informasi terkait sistem yang dikembangkan. Namun, UML bukan merupakan bahasa pemrograman, melainkan sebuah *tools* yang dapat menginspirasi pembuatan program yang selanjutnya bisa dikembangkan menggunakan bahasa pemrograman lainnya.

Tujuan dari UML sendiri diantaranya adalah UML dibuat sebagai tujuan utama untuk bahasa pemodelan dimana semua pelaku pengembangan perangkat lunak bisa menggunakannya. Bertujuan untuk mencakup konsep-konsep metode ternama yang nantinya mereka gunakan sebagai bahasa pemodelan. UML tidak dimaksudkan untuk dijadikan metode pengembangan yang lengkap, sebab UML tidak memiliki langkah-langkah mendetail tentang proses pengembangan perangkat lunak. Sekali lagi ditegaskan oleh (Jacobson et al., 2021) UML mencakup konsep-konsep yang dianggap penting untuk mendukung sebuah proses iteratif yang modern berdasarkan penerapan dengan arsitektur yang kuat untuk menyelesaikan kebutuhan berdasarkan masalah-masalah dan kasus-kasus yang dialami pengguna.

Yang terakhir, inti dari tujuan UML adalah pemodelan yang sesederhana mungkin selama masih mampu memenuhi syarat-syarat pemodelan sistem praktis secara penuh yang nantinya akan dibuat. Teknik-teknik pemodelan dari UML yang kita kenal diantaranya adalah *Use Case Diagram*, seperti yang kami jelaskan sebelumnya, kemudian terdapat *Activity Diagram*, *Class Diagram*, *Sequence Diagram*,kemudian banyak pemodelan-pemodelan lain yang kurang populer seperti *Statechart Diagram*, *Collaboration Diagram*, *Component Diagram*, *Deployment Diagram*, *Extensibility Construct*, dan pemodelan-pemodelan yang lain.



### *Use Case Diagram*

Menurut (Simanullang et al., 2021) pada jurnalnya dengan judul Sistem Informasi Pemesanan Menu Makanan Pada Rm Sedep Roso Rantauprapat Berbasis Web, peneliti sempat sedikit menjelaskan tentang istilah ini. *Use Case Diagram* adalah suatu pola atau gambaran yang menunjukkan kelakuan atau kebiasaan sistem. Sedangkan (Setiawansyah et al., 2022) menjelaskan bahwa *Use Case Diagram* adalah sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. *Use Case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem.

### *Activity Diagram*

(Jacobson et al., 2021) menjelaskan *activity diagram* sebagai perwujudan khusus dari kondisi mesin yang ditujukan untuk memodel komputasi dan alur kerja sistem. Kondisi yang digambarkan pada *activity graph* mewakili kondisi ekseskusi komputasi yang dilakukan, bukan kondisi suatu objek secara spesifik. Umumnya, *activity graph* berasumsi bahwa komputasi yang terjadi tidak dipengaruhi oleh kejadian eksternal. *Activity diagram* biasanya berisi percabangan, lebih sering lagi percabangan kendali yang terbagi dua, dimana selanjutnya percabangan tersebut berjalan atau diproses bersamaan. Alur yang diproses secara bersamaan ini merepresentasikan aktivitas-aktivitas yang dapat dikerjakan oleh objek-objek atau orang-orang berbeda secara bersamaan pada sebuah organisasi. Seringkali, kejadian yang terjadi bersamaan ini muncul dari adanya agregasi, dimana objek memiliki proses yang harus dieksekusi secara bersamaan sendiri. Aktivitas yang terjadi secara bersamaan ini sebetulnya dapat dieksekusi secara bersamaan atau satu persatu.

### *Class Diagram*

(Oktriwina, 2021) menjelaskan *class diagram* sebagai salah satu diagram struktur statis yang dimiliki UML dimana *class diagram* menggambarkan struktur sistem dengan menunjukkan sistem *class*, atribut-atribut yang dimiliki, metode-metode yang dapat dieksekusi, dan hubungan antar objek.(Jacobson et al., 2021) menjelaskan *class* itu sendiri adalah penjelasan dari konsep wewenang aplikasi atau solusi dari aplikasi. *Class* dapat dideskripsikan dari berbagai tingkatan presisi dan seberapa konkrit kita ingin mendeskripsikannya. Pada awal tahapan pengembangan perangkat lunak, *class diagram* seringkali berisi aspek-aspek logika dari masalah yang ingin diatasi. Sedangkan pada akhir tahapan pengembangan, *class diagram* dapat diisi keputusan-keputusan desain dan detail-detail implementasi di dalamnya.