**BAB III**

**LANDASAN TEORI**

* 1. **Website**

*World Wide Web (WWW)* atau biasa disebut dengan *web,* merupakan salah satu sumber daya internet yang berkembang pesat. Informasi *web* didistribusikan melalui pendekatan *hypertext,* yang memungkinkan suatu teks pendek menjadi acuan untuk membuka dokumen yang lain. Dengan pendekatan *hypertext* ini seseoranng dapat memperoleh informasi dengan meloncat dari suatu dokumen ke dokumen yang lain. Dokumen-dokumen yang diakses pun dapat tersebar di berbagai mesin, dan bahkan di berbagai negara. Bagai jarring laba-laba, jejaring *web* telah membentang ke seluruh penjuru dunia. Tidak hanya sebatas pada lembaga-lembaga penelitian yang ingin mempublikasikan hasil riset. *Web* juga banyak digunakan oleh perusahaan bisnis yang ingin mengiklankan produk atau melakukan transaksi bisnisnya, bahkan digunakan sebagai *web* penyedia/layanan informasi yang dibutuhkan oleh khalayak ramai.[1]

Sejarang *web* dimulai pada bulan Maret 1989. Ketika itu, tim Berner-Lee yang bekerja di Laboratorium Fisika Partikel Eropa atau yang dikenal dengan nama *CERN* (*Consei European pour la Recherece Nuclaire*) yang berada di Geneva, Swiss, mengajukan protocol suatu tata cara untuk komunikasi sistem distribusi infomasi internet yang digunakan untuk berbagai informasi di antara para fisikawan. Protokol inilah yang selanjutnya dikenal sebagai *protocol World Wide Web* dan dikembangkan oleh *World Wide Web Consortium* (*W3C*). *W3C* sendiri merupakan konsorsium dari sejumlah organisasi yang berkepentingan dalam pengembangan berbagai standar yang berkaitan dengan *web*.[1]

Pada awalnya aplikasi *web* dibangun hanya dengan menggunakan bahasa yang disebut *HTML* (*HyperText Markup Language*), dan protocol yang digunakan dinamakan *HTTP* (*HyperText Transfer Protocol*). Pada perkembangan berikutnya, sejumlah skrip dan objek dikembangkan untuk memperluas kemampuan *HTML*. Pada saat ini, banyak skrip seperti itu antara lain *PHP*, dan *ASP*, sedangkan contoh yang berupa objek antara lain adalah *applet* (*Java*). Aplikasi *web* sendiri dapat dibagi menjadi dua bagian yaitu *web* statis dan *web* dinamis. [1]

**3.2. PHP (Perl Hypertext Preprosessor)**

*PHP* adalah *Bahasa Server Side-Scriping* yang menyatu dengan HTML untuk membuat halaman *web* yang dinamis. Karena *PHP* merupakan *server side-scriipting*, maka *sintaks* dan perintah-perintah *PHP* akan diesksekusi di*server*, kemudian hasilnya akan dikirimkan ke *browser* dengan format *HTML*. Maka dari itu kode program yang akan ditulis dalam *PHP* tidak akan terlihat oleh pengguna atau *user* sehingga keamanan halaman *website* akan terjamin. Selain itu *PHP* juga didesain untuk membuat halaman *website* yang dinamis, yaitu sebuah halaman *website* yang dapat membuat suatu tampilan berdasarkan perintah terbaru, seperti menampilkan isi basis data ke halaman *website*. [6]

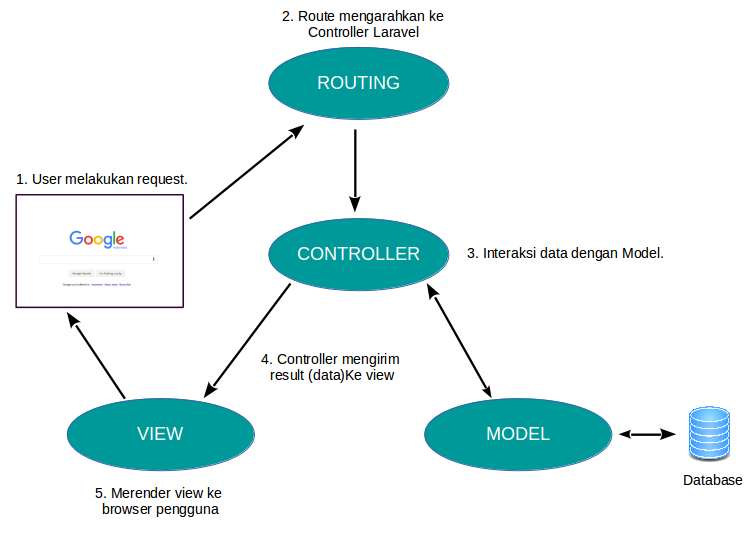
Bahasa pemrograman *PHP* atau singkatan dari *Personal Home Page* merupakan sebuah bahasa skrip yang tertanam dalam *HTML* untuk dieksekusi bersifat *server side*. *PHP* merupakan salah satu bahasa pemrograman yang bersifat *open source*, sehingga *source code PHP* dapat disebarluaskan dan diubah secara bebas. Jika anda ingin mendownload versi terbaru *PHP* anda dapat mendownloadnya di situs resminya. *PHP* juga mampu berjalan di berbegai *web server* seperti *PWS (Personal Web Server), Apache, IIS (Internet Information Server)* dan *Xitami*. Selain itu, *PHP* juga dapat berjalan di banyak sistem operasi yang sudah banyak beredar saat ini seperti *Microsoft Windows* (Semua Versi), *Mac OS*, *Linux* dan *Solaris*. *PHP* dapat dibangun untuk modul *web server* *Apache* dan *binary* yang dapat berjalan sebagai *CGI (Common Gateway Interface).* *PHP* dapat mengatur *cookies*, mengirim *HTTP header*, mengatur *authentication* dan *redirect user*. Keunggulan yang dimiliki oleh bahasa pemrograman *PHP* yaitu mempunyai kemampuan untuk melakukan koneksi ke berbagai macam *software* sistem management berbasis data atau *DBMS* *(Database Management System)* sehingga dapat membuat suatu halaman *website* dinamis. Selain itu, *PHP* juga memiliki konektivitas yang baik dengan beberapa *DBMS tertentu seperti MySQL, Oracle, mSQL, Sybase*, *Microsoft SQL Server*, *Adabas*, *Solid, PostgreSQL, Unix dbm, Velocis, FilePro, dBase,* dan juga semua *database* yang ber-*interface ODBC.*[7]

**3.3. Laravel**

*Laravel*adalah sebuah *framework* *PHP* yang dirilis dibawah lisensi *MIT*, dibangun dengan konsep *MVC (model view controller)*. *Laravel* merupakan *framework open source* yang cukup popular beberapa tahun belakangan ini. Sejak diluncurkan pada tahun 2011, membuat *website* dengan *Laravel* banyak digemari oleh berbagai komunitas programmer di *Github* sebelum menyebar ke seluruh dunia. Laravel sangat cocok digunakan untuk pengembangan sistem yang cukup kompleks. Jika hanya sistem sederhana dirasa tidak sebanding dengan proses pengembangnya. Sama seperti *framework PHP* lainnya. *Laravel* dibangun dengan basis *MVC (Model-View-Controller)*. Yang disebut dengan *Model* biasanya bagian yang berurusan dengan *database*. *View* adalah bagian antarmuka atau bagian depan aplikasi, segala sesuatu yang diproses dalam sistem akan ditampilkan pada *view*. sedangkan *Controller* adalah bagian yang menangani atau menengahi antar *model* dan *view* jadi *controller* yang berperan sebagai pengendali dari sebuah sistem. *Laravel* dilengkapi *command line tool* yang bernama “Artisan” yang bisa digunakan untuk *packaging bundle* dan *instalasi bundle*. [3]

**3.3.1. Sistem Kerja Laravel**

Setiap aplikasi memiliki beberapa tahapan untuk mengerjakan begitu pula dengan *framework Laravel.* Alur kerja dari *Laravel* dapat dilihat pada gambar 3.1.

Gambar 3.1 Alur Kerja Framework Laravel

1. User melakukan inputan *request* melalui *browser* atau *form.*
2. *Request* tersebut kemudian akan di terima *controller* untuk diproses *request*nya.
3. Dianggap *request* tersebut melibatkan *request* ke *database,* maka diteruskanlah *request* tersebut ke *Model. Model* kemudian mengambil data yang diminta *controller.*
4. Setelah mendapatkan data yang diminta *user*, maka akan dikembalikan ke *controller*.
5. Kemudian *Controller* akan meneruskan data yang diberikan *Model* ke *View*.
6. *View* kemudian memberikan data kepada *user* sesuai *request.*
   1. **MySQL**

*MySQL* merupakan *RDBMS* *(server database)* yang mengelola *database* dengan cepat menampung dalam jumlah sangat besar dan dapat di akses oleh banyak *user*. *MySQL* adalah *database server open source* yang cukup popular keberadaanya. Dengan berbagai keunggulan yang dimiliki, membuat *software database* ini banyak digunakan oleh praktisi untuk membangun suatu *project*. Adanya fasilitas *API (Application Programming Interface)* yang dimiliki oleh *Mysql*, memungkinkan bermacam-macam aplikasi komputer yang ditulis dengan berbagai bahasa pemograman dapat mengakses basis data *MySQL. MYSQL* termasuk jenis *RDBMS (Relational Database Management System)*. Sedangkan *RDMS* sendiri akan lebih banyak mengenal istilah seperti *table*, baris dan kolom digunakan dalam perintah-perintah di *MYSQL*.[9]

* 1. **PHPStrom**

*PhpStorm* adalah *IDE* lintas *platform* komersial untuk *PHP*, yang dibangun oleh perusahaan *JetBrains* yang bermarkas di Republik Ceko. *PhpStorm* menyediakan editor untuk *PHP, HTML, dan JavaScript* dengan analisis *kode on-the-fly*, pencegahan kesalahan, dan *refactoring* otomatis untuk kode *PHP* dan *JavaScript.* Kalau bicara masalah IDE biasanya bakal banyak yang bilang bahwa IDE itu lamban dan berat. Sebenarnya wajar karena memang tugas nya bukan sekedar sebagai text editor, tapi banyak fitur lainnya. Dan dibandingkan dengan IDE lain, untuk pengembangan web PHPStorm masih lebih ringan dan cepet. Memang gak seringan sublime text atau atom ketika starting up, tapi itu worthed dibanding dengan fitur yang ditawarkan. PHPStorm memiliki beberapa fitur *basic* sebuah IDE seperti: FTP client, GIT client, database client, terus ada juga *control* buat vagrant, composer, npmjs, gulp dan grunt juga. Terus ada *compiler* buat sass, less dan typescript. Fitur ini saja sudah cukup membuat saya terbantu melakukan banyak hal cukup dari 1 aplikasi saja. Fitur lainnya seperti auto formatter, ini fitur agak curang sih, membuat kode saya jadi rapi dan sesuai standar gak bakal acak kadut lagi. Lalu ada fitur function extractor yang membuat function saya jadi lebih ramping dan *re-usable* banget. Terus juga ada code refactor, membantu saya ketika merubah nama variable, function, class atau bahkan file sekali jalan tanpa kuatir ada yang lupa diganti atau bentrok. Yang paling sering di pakai juga built in terminal emulator, fitur yang saya selalu cari duluan ketika nyoba texteditor ataupun IDE, Dan masih banyak fitur untuk mempermudah penulisan kode.[5]

**3.6. Unifield Modelling Language (UML)**

*UML* merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung. *UML* bertujuan untuk menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak.[11]

**3.6.1. Use Case Diagram**

Use *case* diagram merupakan pemodelan untuk kelakuan sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan informasi yang akan dibuat. Simbol-simbol yang ada pada *use case* diagram dapat dilihat pada Tabel 1:

Tabel 3.1. Simbol Use Case Diagram.

| **No.** | **Simbol** | **Nama** | **Keterangan** |
| --- | --- | --- | --- |
| 1. |  | *Actor* | Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*. |
| 2. |  | *Dependency* | Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*). |
| 3. |  | *Generalization* | Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*). |
| 4. |  | *Include* | Menspesifikasikan bahwa *use case* sumber secara *eksplisit*. |
| 5. |  | *Extend* | Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan. |
| 6. |  | *Association* | Apa yang menghubungkan antara objek satu dengan objek lainnya. |
| 7. |  | *System* | Menspesifikasikan paket yang menampilkan sistem secara terbatas. |
| 8. |  | *Use Case* | Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor. |
| 9. |  | *Collaboration* | Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi). |
| 10. |  | *Note* | Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi. |

**3.6.2. Activity Diagram**

Activity diagram atau diagram aktivitas menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sisten atau menu yang ada pada perangkat lunak. Simbol-simbol yang ada pada diagram aktivitas dapat dilihat pada Tabel 3.2 berikut:

Tabel 3.2. Simbol Activity Diagram.

| **No.** | **Simbol** | **Nama** | **Keterangan** |
| --- | --- | --- | --- |
| 1. |  | *Actifity* | Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain |
| 2. |  | *Action* | State dari sistem yang mencerminkan eksekusi dari suatu aksi |
| 3. |  | *Initial Node* | Bagaimana objek dibentuk atau diawali. |
| 4. |  | *Actifity Final Node* | Bagaimana objek dibentuk dan dihancurkan |
| 5. |  | *Fork Node* | Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran |

**3.6.3. Sequence Diagram**

*Sequence Diagram* digunakan untuk menggambarkan kelakuan objek pada *usecase* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek.

Tabel 3.3. Simbol Sequence Diagram

| **No.** | **Simbol** | **Nama** | **Keterangan** |
| --- | --- | --- | --- |
| 1. |  | *Life Line* | Objek *entity*, antarmuka yang saling berinteraksi. |
| 2. |  | *Message* | Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi |
| 3. |  | *Message* | Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi |

**3.6.4. Class Diagram**

Diagram kelas adalah inti dari proses pemodelan objek. Baik *fordward engineering* maupun *reverse engineering* memanfaatkan diagram ini. *Forward engineering* adalah proses perubahan model menjadi kode program sedangakan *reverse engineering* sebaliknya, merubah kode program menjadi model.

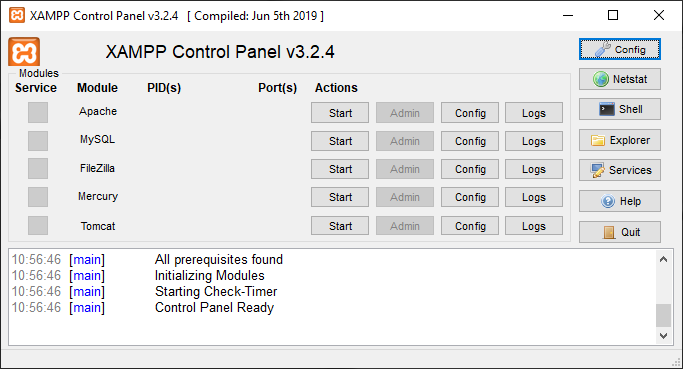
Diagram kelas memiliki fitur-fitur yang menyertainya yaitu atribut dan operasi. Atribut (*attribute*) dan operasi (*operation*) menggambarkan perilaku suatu kelas serta perluasannya seperti *stereotypes, tagged values,* dan batasan (*constraints*) merupakan fitur-fitur sebuah kelas.

Tabel 3.4. Simbol Class Diagram

| **No.** | **Simbol** | **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 aktor. |
| 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. |

* 1. **Xampp**

*XAMPP* adalah paket program web lengkap yang dapat Anda pakai untuk belajar pemrograman *web*, khususnya *PHP* dan *MySQL*. *Xampp* perangkat lunak *open source* yang diunggah secara gratis dan bisa dijalankan di semua semua operasi seperti *windows, linux, solaris*, dan *mac*. Xampp akan menginstalasi dan mengkonfigurasikannya secara otomatis untuk anda, atau konfigurasi. Nantinya, anda dapat memiliki server pribadi dalam komputer (*localhost*).[1]

Tampilan control panel xampp seperti gambar 3.2.

Gambar 3.2. Tampilan Control Panel XAMPP

Di dalam folder utama xampp, terdapat beberapa folder penting yang perlu diketahui. Untuk lebih memahami setiap fungsinya, dapat melihat penjelasannya sebagai berikut:

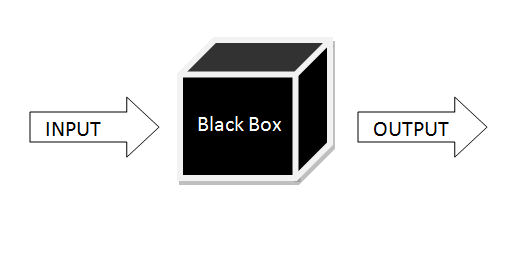
Tabel 3.5. penjelasan folder xampp

| **Folder** | **Keterangan** |
| --- | --- |
| *Apache* | Folder utama *Apache Webserver* |
| *Htdocs* | Folder utama untuk menyimpan data-data latihan *web,* baik *PHP* maupun *HTML* biasa. |
| *Manual* | Berisi *subfolder* yang didalamnya terdapat manual program dan *database* termasuk manual *PHP* dan *MySQL* |
| *MySQL* | Folder utama untuk *database* *MySQL Server.* |
| *PHP* | Folder utama untuk program *PHP* |

* 1. **Blackbox Testing**

*Black box testing* adalah pengujian yang dilakukan hanya mengamati hasil eksekusi melalui data uji dan memeriksa fungsional dari perangkat lunak. Jadi dianalogikan seperti kita melihat suatu kotak hitam, kit hanya bisa melihat penampilan luarnya saja, tanpa tau ada apa dibalik bungkus hitam nya. Sama seperti pengujian *black box*, mengevaluasi hanya dari tampilan luarnya (*interfacenya*), fungsionalitasnya.tanpa mengetahui apa sesungguhnya yang terjadi dalam proses detilnya (hanya mengetahui *input* dan *output*).[7]

*Black Box* pengujian adalah metode pengujian perangkat lunak yang menguji fungsionalitas aplikasi yang bertentangan dengan struktur *internal*. Pengetahuan khusus dari kode aplikasi / struktur *internal* dan pengetahuan pemrograman pada umumnya tidak diperlukan. Uji kasus dibangun di sekitar spesifikasi dan persyaratan, yakni, aplikasi apa yang seharusnya dilakukan. Menggunakan deskripsi *eksternal* perangkat lunak, termasuk spesifikasi, persyaratan, dan desain untuk menurunkan uji kasus. Tes ini dapat menjadi fungsional atau *non-fungsional*, meskipun biasanya *fungsional*. Perancang uji memilih *input* yang *valid* dan tidak *valid* dan menentukan *output* yang benar. Tidak ada pengetahuan tentang struktur internal benda uji itu.[7]

Metode uji dapat diterapkan pada semua tingkat pengujian perangkat lunak: unit, integrasi, fungsional, sistem dan penerimaan. Ini biasanya terdiri dari kebanyakan jika tidak semua pengujian pada tingkat yang lebih tinggi, tetapi juga bisa mendominasi unit testing juga. Metode *blackbox* *testing* seperti gambar 3.3.

Gambar 3.3 Metode Blackbox Testing

*Blackbox* *testing* untuk menemukan hal-hal berikut:

1. Menemukan fungsi-fungsi yang tidak benar atau hilang di dalam suatu *software*.

2. Mencari kesalahan *interface* yang terjadi saat *software* dijalankan.

3. Untuk mengetahui kesalahan dalam struktur data atau akses *database* *eksternal* di dalam suatu aplikasi.

4. Menguji kinerja dari *software*.

5. Menginisialisasikan dan mencari kesalahan dari terminasi *software* itu sendiri.