**BAB IV**

**PERANCANGAN DAN PERSIAPAN IMPLEMENTASI**

1. **Instalasi Aplikasi yang akan digunakan**

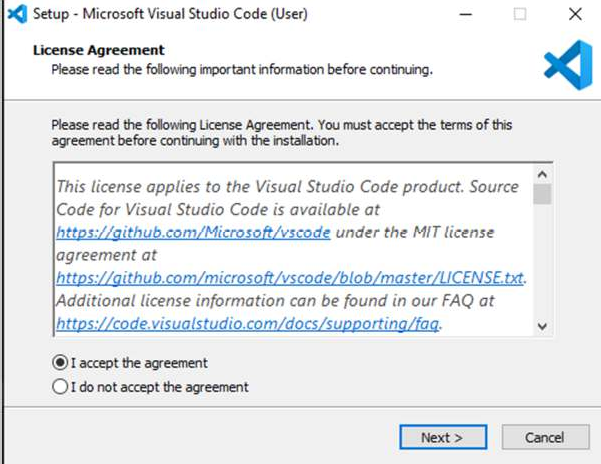
Visual studio code (VS Code) ini adalah sebuah teks editor ringan dan handal yang dibuat oleh Microsoft untuk sistem operasi *multiplatform,* artinya tersedia juga untuk versi Linux, Mac, dan Windows. Teks editor ini secara langsung mendukung bahasa pemrograman Javascript, Typescript, dan Node.js, serta bahasa pemrograman lainnya dengan bantuan plugin yang dapat dipasang via *marketplace* Visual Studio Code (seperti C++, C#, PHP, Python, Go, Java, dst).

Berikut ini merupakan langkah-langkah instalasi visual studio code pada sistem operasi windows.

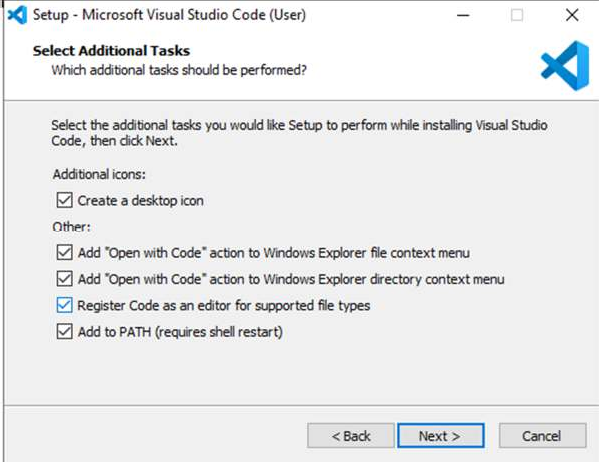
1. Pertama, *download* terlebih dahulu file installer visual studio code melalui situs resminya pada link berikut ini *https(:)//code(dot)visualstudio(dot)com/download.*



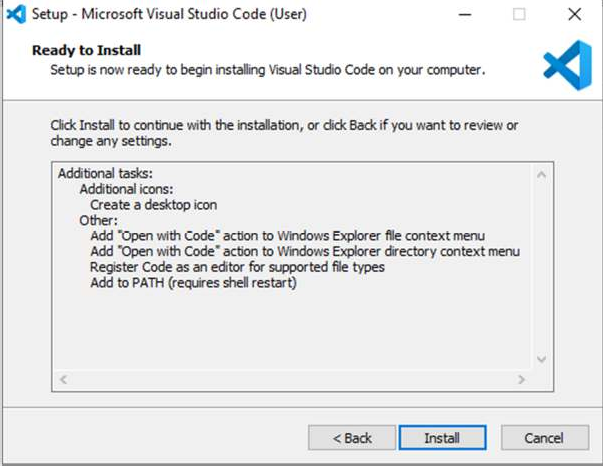
1. Terdapat 2 piliahn untuk sistem operasi windows, diantaranya ada yang 64 bit dan 32 bit. Pilihlah installer sesuai dengan spesifikasi laptop yang digunakan.
2. Jika, sudah berhasil di *download*, lanjut pada proses instalasi, pertama klik pada file installer vs code.
3. Pilih “I accept the aggrement” untuk menyetujui “License agreement”, kemudian klik next.



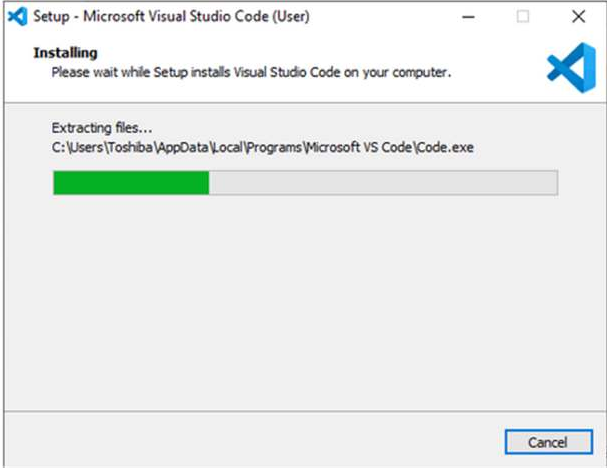
1. Pada bagian select additional tasks, centang semuanya kemudian klik next.



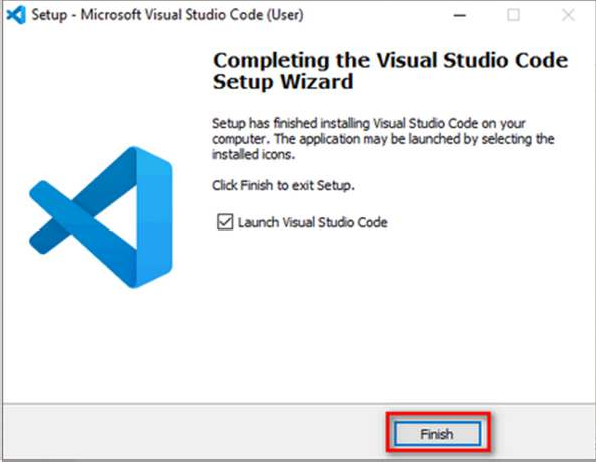
1. Kemudian klik install untuk melanjutkan proses instalasi.



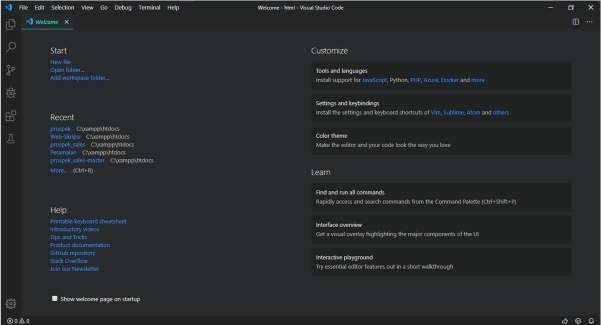
1. Tunggu proses instalasi hingga selesai.



1. Setelah selesai klik Finish.



1. Jika berhasil, maka tampilan utamanya akan seperti berikut.



1. **Xampp**

Xampp adalah sebuah paket perangkat lunak (*software*) komputer yang sistem penamaannya diambil dari akronim kata Apache, MySQL / MariaDB, PHP, dan PERL. Sementara huruf “X” yang terdapat pada awal kata berasal dari istilah *cross platform* sebagai simbol bahwa aplikasi ini bisa dijalankan di empat sistem operasi yang berbeda.

Jika dijabarkan secara gamblang, masing-masing huruf yang ada di dalam nama XAMPP memiliki arti sebagai berikut.

**X = *Cross Platform*,** merupakan kode penanda untuk *software cross platform* atau yang bisa berjalan di banyak sistem operasi.

**A = Apache**, apache adalah aplikasi *web server* yang bersifat gratis dan bisa dikembangkan oleh banyak orang (*open source).*

**M = MySQL / MariaDB**, MySQL atau MariaDB merupakan aplikasi *database server* yang dikembangkan oleh orang yang sama. MySQL berperan dalam mengolah, mengedit, dan menghapus daftar melalui *database*.

**P = PHP**, huruf “P” yang pertama dari akronim kata XAMPP adalah inisial untuk menunjukkan eksistensi bahasa pemrograman PHP. Bahasa pemrograman ini biasanya digunakan untuk membuat *website dinamis*.

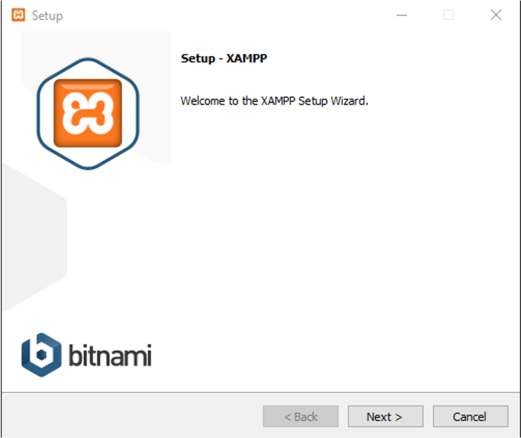
**P = Perl**, untuk huruf “P” selanjutnya merupakan singkatan dari bahasa pemrograman Perl yang kerap digunakan untuk memenuhi berbagai macam kebutuhan. Perl ini bisa berjalan di dalam banyak sistem operasi, sehingga sangat fleksibel dan banyak digunakan.

Berikut merupakan langkah-langkah untuk melakukan instalasi xampp pada sistem operasi windows.

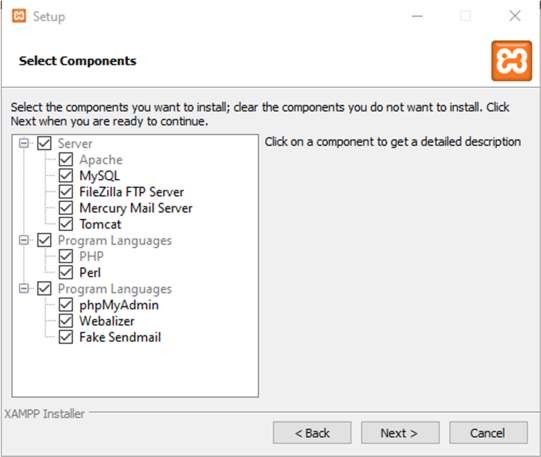
1. Pertama, *download* terlebih dahulu file installer xampp pada link berikut ini. *https(:)//apachefriends(dot)org.*



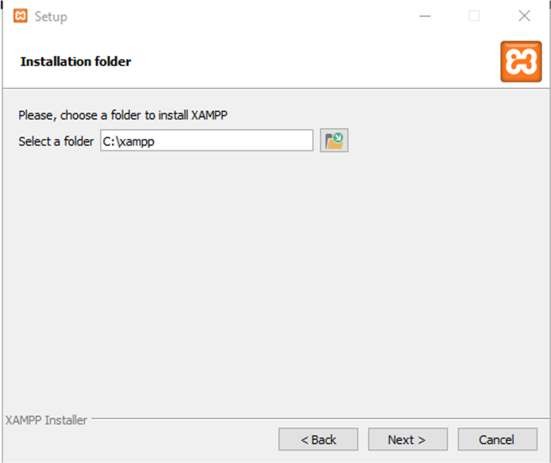
1. Pilih file installer sesuai dengan spesifikasi laptop yang digunakan.
2. Klik file xampp yang sudah berhasil di *download*, maka akan muncul tampilan seperti ini.



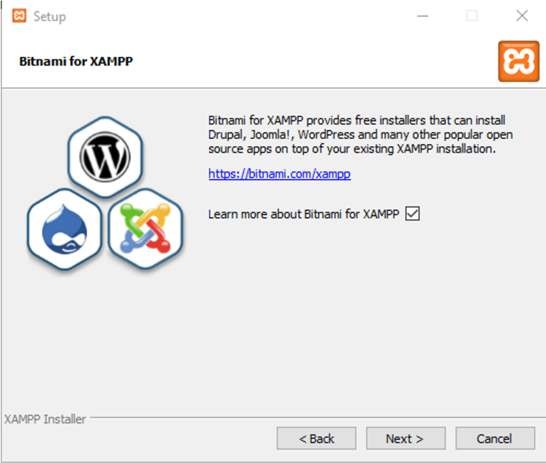
1. Kemudian, pilih next untuk melanjutkan pada proses instalasi.



1. Pada tampilan *select component*, centang semua pilihannya agar dapat menjalankan semua program yang dipilih. Kemudian pilih *next*.



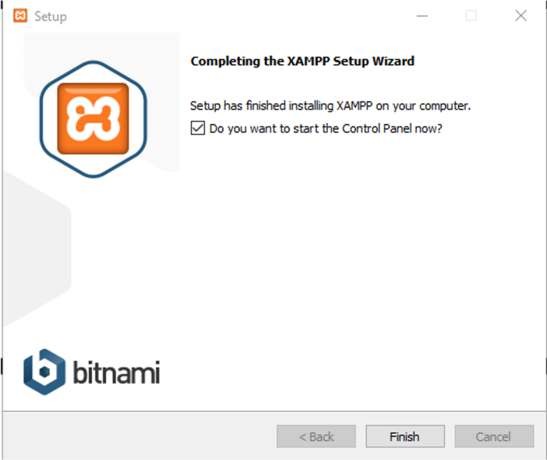
1. Pada, *installation folder* secara *default* sistem akan menyimpannya pada direktori C. Jika sudah klik *next* untuk melanjutkan proses instalasi.



1. Pada tampilan ini langsung saja klik *next* untuk melanjutkan.



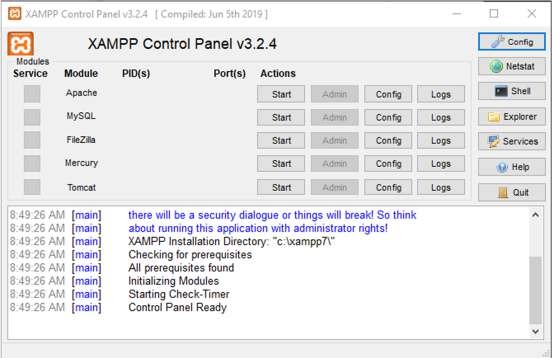
1. Kemudian tunggu hingga proses instalasi selesai.



1. Jika telah selesai, selanjutnya klik *Finish* untuk membuka *control panel*.



1. Pilih bahasa yang akan digunakan pada xampp. Lalu klik *save*.

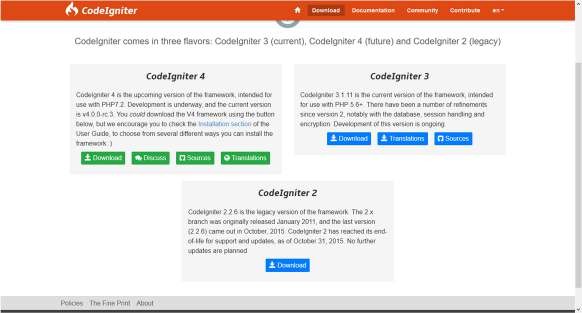


1. Jika berhasil, maka tampilan utama *control panel* dari xampp seperti berikut.
2. **CodeIgniter**

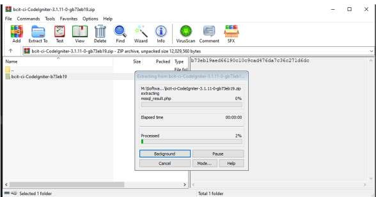
Codeigniter (CI) merupakan sebuah *web application framework* yang bersifat *open source* dimana digunakan untuk membangun aplikasi php dinamis. *Framework* itu sendiri merupakan abstraksi di dalam sebuah perangkat lunak yang menyediakan fungsi yang *generic* sehingga dapat dirubah oleh kode yang dibuat *user*, sehingga dapat menyediakan perangkat lunak untuk aplikasi tertentu.

Berikut ini merupakan tahapan-tahapan untuk instalasi codeigniter pada sistem operasi windows.

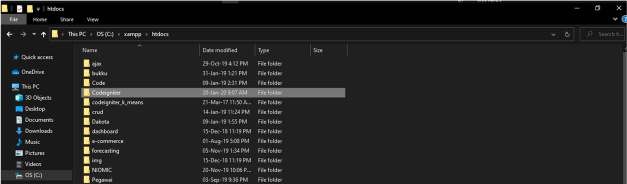
1. Pertama, *download* terlebih dahulu codeigniter pada *website* resminya pada link di bawah ini *https(:)//codeigniter(dot)com/download*.



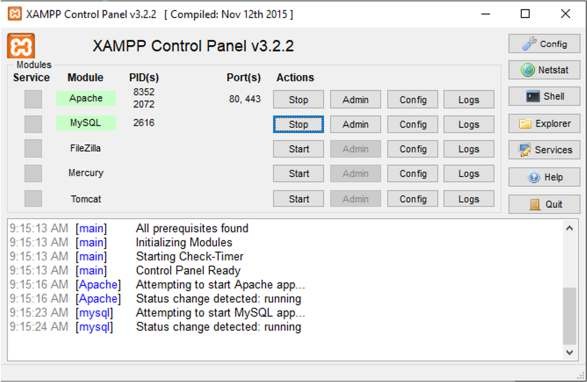
1. Setelah berhasil di *download*, selanjutnya ekstrak file tersebut.



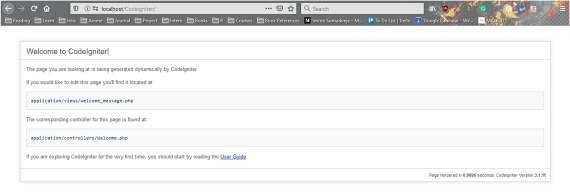
1. Kemudian, *copy* file yang telah di ekstrak ke dalam direktori htdocs, seperti gambar berikut.



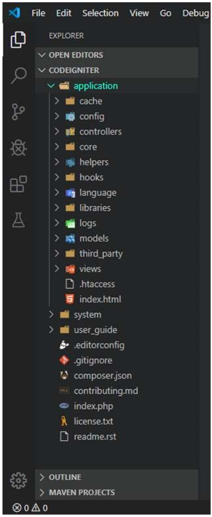
1. Untuk menjalankannya, kita menggunakan xampp, untuk itu silakan jalankan terlebih dahulu aplikasi xampp-nya seperti berikut.
2. Buka aplikasi xamppnya, kemudian klik tombol start pada kolom actions baris pertama untuk menjalankan apache, dan klik tombol start pada kolom actions baris kedua untuk menjalankan MySQL.



1. Setelah xamppnya berhasil dijalankan, proses selanjutnya tinggal memanggil folder yang telah disimpan pada direktori htdocs di browser. Maka tampilan utamanya seperti gambar berikut.



1. Berikut ini merupakan struktur folder yang terdapat pada codeigniter, diantaranya sebagai berikut.



1. *Application*, merupakan folder yang pada dasarnya menyimpan aplikasi yang sedang kita buat.
2. *Cache*, merupakan folder yang menyimpan semua cache yang dibuat oleh cache library.
3. *Config,* merupakan folder yang menyimpan informasi mengenai konfigurasi aplikasi seperti autoload, database, routes, dan lainnya.
4. *Controller,* merupakan folder yang menyimpan controller-controller aplikasi yang dapat digunakan untuk menyusun aktivitas program.
5. *Core,* merupakan folder untuk memperluas *class* inti codeigniter.
6. *Helpers*, merupakan folder untuk menyimpan helpers.
7. *Hooks,* merupakan folder untuk menyimpan hooks untuk mengubah alur fungsi dari core codeigniter.
8. *Language*, merupakan folder untuk menyimpan bahasa-bahasa yang akan digunakan.
9. *Libraries*, merupakan folder untuk menyimpan library.
10. *Logs,* merupakan folder untuk menyimpan semua error log apabila error log diaktifkan.
11. *Models,* merupakan folder untuk menyimpan models yang akan didefinisikan tabel dari database yang dapat kita gunakan oleh controller yang kita buat untuk mengakses database.
12. *Third\_party,* merupakan folder untuk menyimpan fungsi-fungsi tambahan dalam cara kerja codeigniter.
13. *Views,* merupakan folder untuk menyimpan tampilan dari aplikasi yang kita buat.
14. *System*, merupakan folder untuk menyimpan sistem inti dari codeigniter dan menyimpan semua file *system* pada core *framework*.

* Folder core, tempat menyimpan semua class library bawaan Codeigniter.
* Folder database, tempat menyimpan semua driver database drivers dan class yang akan digunakan.
* Folder fonts, tempat menyimpan semua font yang digunakan *image manipulation* library.
* Folder helpers, tempat menyimpan semua helper core Codeigniter.
* Folder language, tempat menyimpan file bahasa.
* Folder libraries, tempat menyimpan semua library.

Bila anda perhatikan pada struktur folder di dalam folder application ada nama folder yang sama yaitu core, helpers, libraries.

* Folder core, helper dan libraries yang ada di folder system tempat semua class library dan helper asli yang terbawa oleh Codeigniter.
* Folder core, helpers dan libraries yang ada di folder application adalah tempat untuk semua custom class library dan custom helper yang akan mungkin akan dibuat oleh programmer. Dengan adanya custom library dan custom helper, maka kita dapat membuat helper atau library sendiri. Hal ini terjadi karena terkadang library atau helper bawaan Codeigniter tidak menyediakan fungsionalitas yang dibutuhkan.

Folder system berisi file-file inti Codeigniter, sebaiknya anda tidak merubah-ubah isi di dalam folder ini. Pada prakteknya, jika kita membuat aplikasi memakai Codeigniter maka sebagian besar waktu akan dihabiskan untuk bekerja dengan folder application dan sub folder yang ada di dalamnya.

1. **Helper**

Helper adalah file-file program pada Codeigniter yang berisi fungsi -fungsi bantuan yang dapat dipakai untuk mengerjakan tugas tertentu. Isi dari helper adalah fungsi-fungsi biasa, bukan class. File-file native helper (helper asli bawaan Codeigniter) tersimpan di system/helpers.

Codeigniter memiliki beberapa helper, namun di dalam buku ini kita tidak membahas secara detail penggunaan dari masing-masing helper tersebut. Kita hanya akan membahas manfaat helper tersebut dan cara memanggilnya. Untuk mengetahui fungsi-fungsi yang ada pada suatu helper secara detail, anda dapat menemukannya pada user\_guide di bagian helper yang dimaksud.

Untuk memanggiil suatu helper, anda dapat menggunakan perintah :

* $this⭢load⭢helper(‘nama\_helper’);

Jika ingin memanggil beberapa helper sekaligus, maka nama helper diberikan dalam suatu array :

* $helpers = [‘helper1’, ‘helper’, ‘helper3’];
* $this⭢load⭢helper($helpers);

Atau dibisa dipanggil secara langsung :

* $this⭢load⭢helper([‘helper1’, ‘helper2’, ‘helper3’]);

Berikut ini adalah beberapa helper yang ada di dalam Codeigniter:

* Array Helper, berisi fungsi-fungsi yang berkaitan dengan array :

$this⭢load⭢helper(‘array’);

* Captcha Helper, berisi fungsi-fungsi untuk pembuatan captcha. Captcha adalah cara untuk membedakan apakah user adalah seseorang atau bot/robot. Biasanya dalam bentuk karakter gabungan angka ataupun huruf ataupun teka-teki soal untuk diisikan ke dalam suatu input *text.* Captcha yang dihasilkan helper ini merupakan captcha dasar berupa gambar berisi campuran antara huruf dan angka :

$this⭢load⭢helper(‘captcha’);

* Cookie Helper, berisi fungsi-fungsi yang berkaitan dengan pembuatan cookie :

$this⭢load⭢helper(‘cookie’);

* Date Helper, berisi fungsi-fungsi yang berkaitan dengan waktu dan tanggal :

$this⭢load⭢helper(‘date’);

* Directory Helper, berisi fungsi-fungsi berkaitan dengan folder :

$this⭢load⭢helper(‘directory’);

* Download Helper, manfaat utama helper ini adalah untuk membuat force download, yaitu suatu fungsi yang akan memaksa user untuk mendownload suatu file ketika link menuju suatu file di-klik oleh user.

$this⭢load⭢helper(‘download’);

* Email Helper, berisi fungsi-fungsi yang berkaitan dengan pengiriman email :

$this⭢load⭢helper(‘email’);

* File Helper, berisi fungsi-fungsi yang berkaitan dengan file, misalnya membaca isi file text, menyimpan file text dan sebagainya :

$this⭢load⭢helper(‘file’);

* HTML Helper, berisi fungsi-fungsi yang berkaitan dengan pembuatan element-element HTML :

$this⭢load⭢helper(‘html’);

* Inflector Helper, berisi fungsi-fungsi yang berkaitan dengan penformatan untuk string , misalnya membuat bentuk plural dari string “book” menjadi “books”, atau sebaliknya. Bisa juga untuk mengubah string ke dalam bentuk camelCase :

$this⭢load⭢helper(‘inflector’);

* Language Helper, berisi fungsi-fungsi yang berkaitan dengan bahasa :

$this⭢load⭢helper(‘language’);

* Number Helper, berisi fungsi-fungsi yang berkaitan dengan angka, misalnya untuk memformat satuan sata dalam bytes :

$this⭢load⭢helper(‘number’);

* Path Helper, berisi fungsi yang berkaitan dengan file path server :

$this⭢load⭢helper(‘path’);

* Security Helper, berisi fungsi yang berkaitan dengan keamanan, misalnya terkait dengan XSS Filtering :

$this⭢load⭢helper(‘security’);

* Smiley Helper, berisi fungsi-fungsi yang membantu memanage emoticons :

$this⭢load⭢helper(‘smiley’);

* String Helper, berisi fungsi yang berkaitan dengan string, misalnya untuk menghapus double slash pada alamat URL, atau untuk menghasilkan random string dan sebagainya :

$this⭢load⭢helper(‘string’);

* Text Helper, berisi fungsi-fungsi yang berkaitan dengan teks. Misalnya untuk membatasi jumlah kata yang akan ditampilkan pada suatu artikel, membatasi jumlah karakter, menjalankan fungsi sensor yang menggantikan daftar kata-kata yang terlarang dengan kata-kata pengganti lainnya, dan sebagainya :

$this⭢load⭢helper(‘text’);

* Typography Helper, berisi fungsi-fungsi untuk memformat teks agar tepat secara semantic, misalnya mengubah / ganti baris menjadi tag <br>, mengubah suatu karakter ke dalam entitas HTML dan sebagainya :

$this⭢load⭢helper(‘typography’);

* URL Helper, berisi fungsi-fungsi yang berkaitan dengan URL. Misalnya untuk membuat link, mendapatkan nila base url, mendapatkan nilai URL, melakukan redirect, dan sebagainya :

$this⭢load⭢helper(‘url’):

* XML Helper, berisi fungsi untuk bekerja dengan XML :

$this⭢load⭢helper(‘xml’);

1. **Library**

Library Codeigniter membantu anda dalam mengerjakan operasi tertentu dengan lebih cepat dan efisien. Tiap library adalah sebuah class PHP dengan berbagai method yang dapat digunakan begitu class tersebut di load oleh Controller. Beberapa class sangat berguna, dan beberapa diantaranya mungkin akan selalu anda butuhkan di berbagai Controller.

Jika demikian, class yang termasuk dalam kategori ini sebaiknya di set autoload (dijalankan secara otomatis) misalnya library database ataupun library session.

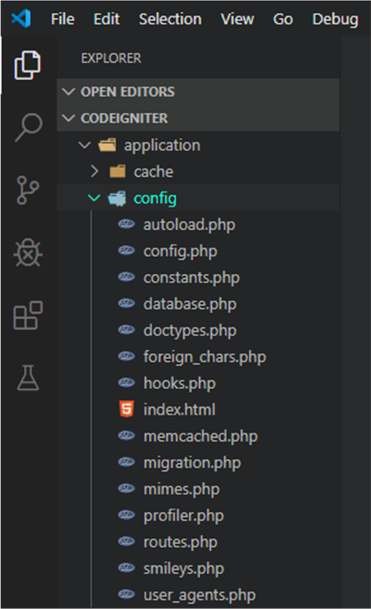
Library-library yang terdapat dalam Codeigniter di antaranya adalah :

* Benchmarking, library benchmarking selalu aktif. Dapat digunakan untuk menghitung penggunaan memory.
* Calendaring, library ini digunakan untuk membuat kalender bulan dan tahu tertentu.
* Config, library config diinisialisasi secara otomatis oleh system. Digunakan untuk menerima informasi mengenali konfigurasi system.
* Database, library database sangat berguna sekali jika aplikasi terhubung dengan database, dilengkapi dengan berbagai method untuk memanipulasi database.
* Email, digunakan untuk mengirim email.
* Encryption, berguna untuk operasi enkripsi.
* File uploading. Library digunakan untuk melakukan upload file. Menyediakan fitur untuk validasi, agar file sesuai dengan *mime type*, ukuran (size dalam kilobyte) atau dimensi untuk gambar.
* FTP, library FTP digunakan untuk mentransfer file ke semua server.
* HTML Table, jika anda membutuhkan cara yang cepat untuk menggenerate table html, maka gunakan library ini. Library html table akan menggenerate table dari array atau hasil dari query database.
* Image Manipulation, digunakan untuk meresize, membuat thumbnail, crop atau rotate gambar, serta watermark. Beberapa fungsi membutuhkan support PHP yang lebih baik, misalnya GD/GD2.
* Input and Security, digunakan untuk pre-process input data dari form dan URL serta untuk menghandle beberapa fungsi security (misalnya untuk menghadapi serangan XSS attacks.
* Language, library ini digunakan untuk meload beberapa bahasa yang berbeda.
* Loader, library ini secara otomatis diload oleh system. Anda akan sering menggunakan library ini untuk meload view di dalam controller, selain itu juga untuk meload libraries.
* Output, otomatis diload oleh system. Library ini mempunyai sebuah fungsi utama yaitu mengirimkan halaman web ke browser yang merequest. Juga digunakan untuk proses caching.
* Pagination, digunakan untuk membuat pagination dari hasil query database. Dapat diatur beberapa record yang ingin ditampilkan dalam satu halaman. Anda juga dapat mengatur tampilan pagination.
* Session, digunakan untuk *maintenance state* seorang user. Library ini tidak menggunkan session milik PHP, tetapi menghasilkan data untuk session sendiri.
* Template Parser, digunakan untuk membuat template yang berisi pseudocode. Jika anda pernah menggunakan Smarty, maka Template Parser Codeigniter tidak selengkap dan sepowerfull Smarty tetapi cukup berguna.
* Trackback, digunakan untuk mengirim dan menerima data Trackback.
* Unit Testing, library ini digunakan untuk unit testing.
* URI Class, library ini diload secara otomatis oleh system. Digunakan untuk memparser URL, membaginya ke dalam segment-segment yang dapat dilewatkan melalui Controller atau disimpan sebagai variabel.
* User Agent, digunakan untuk mengidentifikasi browser, mobile device atau robot (misalnya milik search engine) yang berkunjung ke situs anda.
* Form Validation, sebuah class yang sangat berguna. Digunakan untuk memvalidasi input form.
* XML-RPC, digunakan untuk menseting XML-RPX client dan server.
* Zip Encoding, digunakan untuk membuat arsip Zip untuk text dan data binary.

Untuk lebih jelasnya mengenai library pada Codeigniter, anda dapat melihat user\_guidenya.

1. **Konfigurasi dasa pada CodeIgniter**

Dalam memulai codeigniter, ada beberapa konfigurasi dasar yang perlu kita ketahui, diantaranya autoload.php, config.php dan database.php. Semua konfigurasi pada codeigniter, terletak pada satu tempat yakni di dalam folder *application/config*.



1. Autoload.php, file ini digunakan untuk mengatur fungsi-fungsi yang akan dimuat otomatis di awal ketika program dijalankan. Untuk melakukan konfigurasi pada file autoload.php, silakan buka file- nya seperti gambar berikut.



1. Kemudian temukan kode berikut.



1. Ubah kode tersebut menjadi seperti berikut.



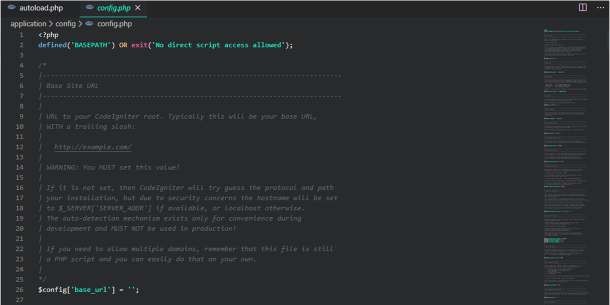
Kode tersebut dapat diartikan kita dapat meload library ‘database’ secara otomatis.

1. Selanjutnya, temukan kode berikut, dan tambahkan ‘url’ didalamnya.



Kode tersebut dapat diartikan kita dapat meload helper “url” secara otomatis.

1. Config.php, pada file ini terdapat beberapa konfigurasi yang secara standar sudah terkonfigurasi, namun terdapat beberapa konfigurasi yang perlu diperhatikan, untuk konfigurasi dasar, cukup mengetahui konfigurasi base\_url.



1. Kemudian temukan kode berikut.



1. Kemudian tambahkan kode tersebut, menjadi nama folder yang disimpan pada htdocs.



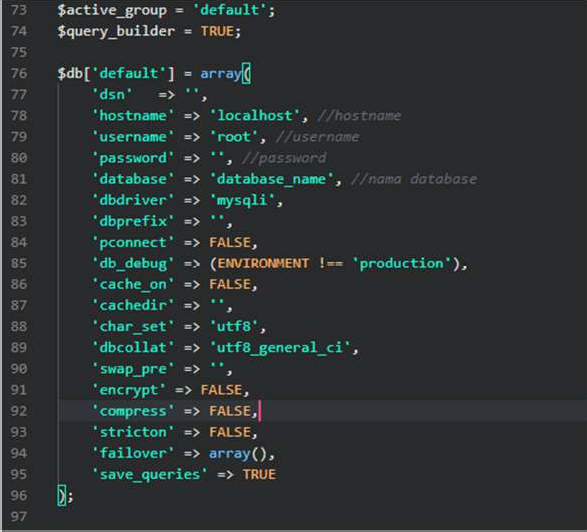
1. Database.php, file ini digunakan untuk melakukan konfigurasi yang berkaitan dengan konfigurasi database dari website yang akan dibuat. Adapun konfigurasi yang perlu diperhatikan, diantaranya: hostname, username, password dan database. Buka file database.php pada teks editor seperti gambar berikut.



1. Kemudian temukan kode seperti berikut pada file database.php.



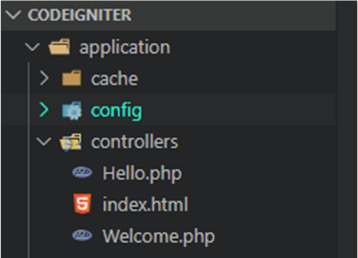
1. Kemudian tambahkan kode tersebut seperti gambar berikut.



1. Hello World Codeigniter

Untuk menguji pemahaman pada codeigniter, kita akan mencoba dengan menampilkan text “Hello World” pada browser menggunakan controller.

* Buat sebuah controller dengan nama Hello.php seperti gambar berikut.



* Setelah itu, ketikan kode seperti gambar berikut.



* Jika berhasil maka hasilnya akan seperti gambar berikut.



1. **Analisis dan Perancangan Sistem**

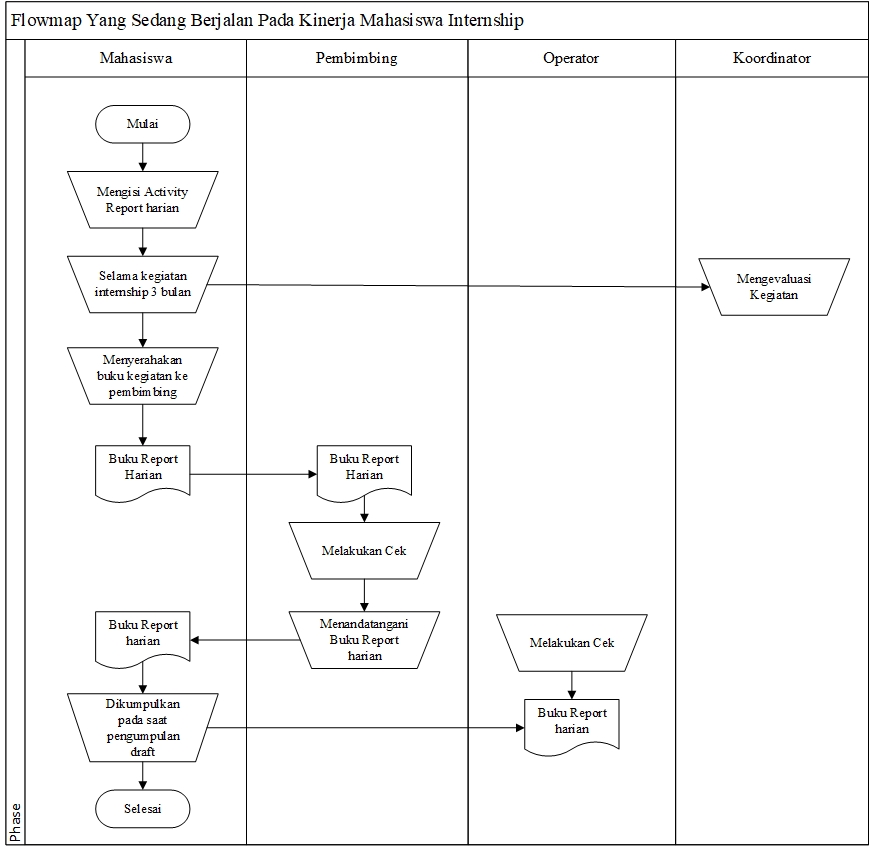
Perencanaan sistem menyangkut estimasi dari kebutuhan-kebutuhan fisik, tenaga kerja dan dana yang dibutuhkan untuk mendukung pengembangan sistem ini serta untuk mendukung operasinya setalah diterapkan sedangkan analisis merupakan proses untuk menentukan bentuk dari kebutuhan sistem yang menunjang kebutuhan pada saat membangun dan implementasi. Secara garis besar disebut juga sebagai proses mempelajari aktifitas system untuk memahami gambaran menyeluruh tentang sehingga perancang telah mengetahui apa saja kebutuhan dari sistem tersebut.

**4.5.1. Analisis Sistem Berjalan (*Current System*)**

Sistem kegiatan mahasiswa *internship* di Politeknik Pos Indonesia saat ini sedang berjalan masih menggunakan cara manual lebih khususnya dibagian pengawasan kinerja atau laporan kegiatan harian yang dilakukan mahasiswa tersebut. Kendala yang terjadi adalah kurang efektif karena data dapat dimanipulasi apabila dilakukan dengan cara manual sehingga data yang dibutuhkan kurang valid.

**4.5.1.1. Analisis Prosedur yang berjalan (*Flowmap*)**

*Flowmap* untuk analisis prosedur yang sedang berjalan dibuat agar alur sistem monitoring kinerja mahasiswa *internship* yang sedang berjalan dapa lebih mudah dipahami. Berikut adalah analisis sistem yang sedang berjalan dalam bentuk *flowmap* atau prosedur :



*Gambar 4.1 Flowmap kegiatan harian internship yang sedang berjalan*

Keterangan :

Pada *flowmap* tersebut dapat dijelaskan bahwa mahasiswa mengisi formulir kehadiran kegiatan *internship.* Di dalam buku tersebut tercantum form catatan yang harus diisi oleh mahasiswa/i selama kegiatan *internship* berlangsung yaitu 3 (tiga) bulan untuk sebagai *report* harian, *activity* apa saja yang dilakukan oleh mahasiswa/i pada tempat *internship*nya. Kemudian mahasiswa/i menyerahkan *buku/logbook* tersebut kepada pembimbing untuk dilakukannya pengecekan serta menandatangani *report* harian dan operator melakukan pengecekan pada saat pengumpulan *draft*.

**4.5.1.2. Anlisis Dokumen Yang Digunakan**

Analisis dokumen yang digunakan merupakan tahap analisis terhadap beberapa dokumen yang digunakan, yaitu :

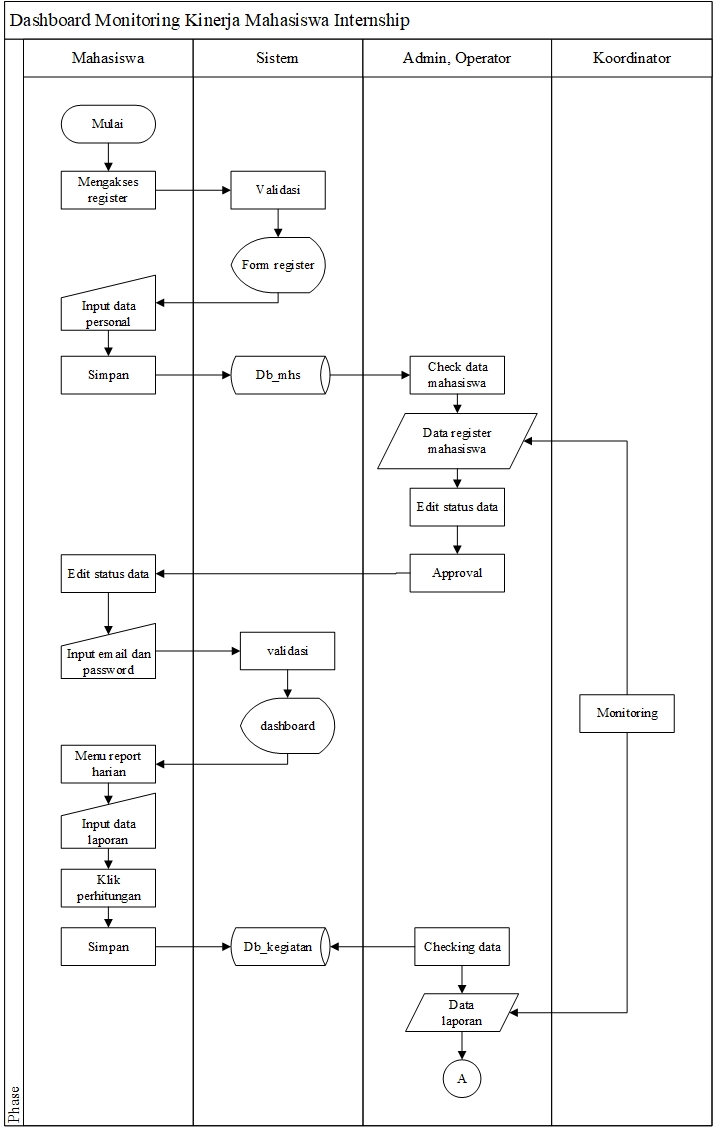
1. Formulir *report* harian , berupa form catatan aktivitas yang dilakukan dan menjadi sebagai buku *logbook*.
2. Dokumen data laporan, menjelaskan mengenai data-data laporan yang dikirimkan oleh mahasiswa berupa laporan atau materi ke pembimbing eksternal/mentor.

**4.5.2. Analisis Sistem yang akan dibangun**

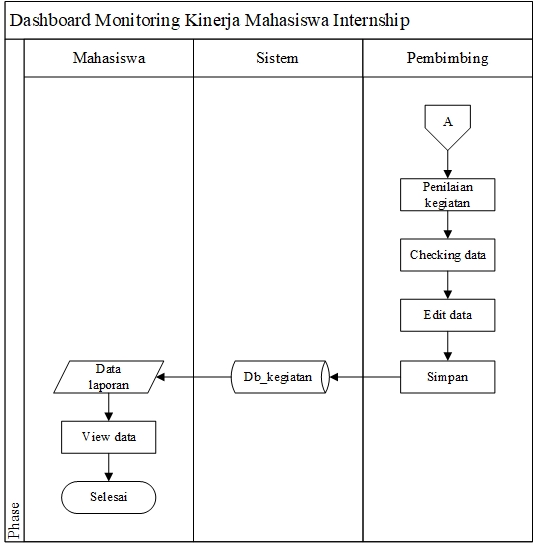
Dari analisis yang telah dilakukan maka akan dibangun sistem yang dapat memenuhi kebutuhan perusahaan/perguruan tinggi dalam pengolahan data kinerja mahasiswa *internship*. Sistem ini sebelumnya dilakukan secara manual dan akan dibuat terkomputerisasi sehingga proses bisnis yang terjadi dapat lebih baik lagi dan meng-efisien-kan kinerja mahasiswa yang *internship* dan memperkecil kesalahan dalam data laporan.

Proses-proses yang sebelumnya dilakukan secara manual akan dijadikan sistem yang mempermudah user. Sistem yang akan dibangun memfokuskan pada pengolahan data laporan kinerja sehingga proses lebih efektif dapat berjalan sesuai kebutuhan user.

**4.5.2.1 Analisis Prosedur yang akan dibangun (*flowmap*)**



*Gambar 4.2 Flowmap monitoring kinerja mahasiswa internship yang akan dibangun*

*Gambar 5.3 Flowmap monitoring kinerja mahasiswa internship yang akan dibangun*

Keterangan :

Pada gambar *flowmap* diatas dapat dijelaskan aktor pertama dimulai dari mahasiswa melakukan tahap registrasi lalu tahap kedua aktor yang akan menerima data registrasi adalah operator untuk mengubah status data. Setelah data di *approval* maka operator memberikan hak akses untuk login kepada mahasiswa lalu sistem akan memvalidasi akun tersebut. Selanjutnya sistem akan menampilkan halaman *dashboard* dan mahasiswa memilih menu *report* harian kemudian meng-input data kegiatan yang dilakukan *user* tersebut lalu menyimpannya. Data kegiatan yang sudah dilakukan aksi maka akan menjadi data laporan yang tersimpan pada *database*, kemudian aktor pembimbing prodi DIV Teknik Informatika akan melakukan tahap *check*-ing menyeluruh data laporan untuk dilakukannya penilaian sebagaimana proses tersebut sama dengan koordinator hanya perbedaan dari segi aksi yaitu hanya me-monitoing ke seluruhan data.

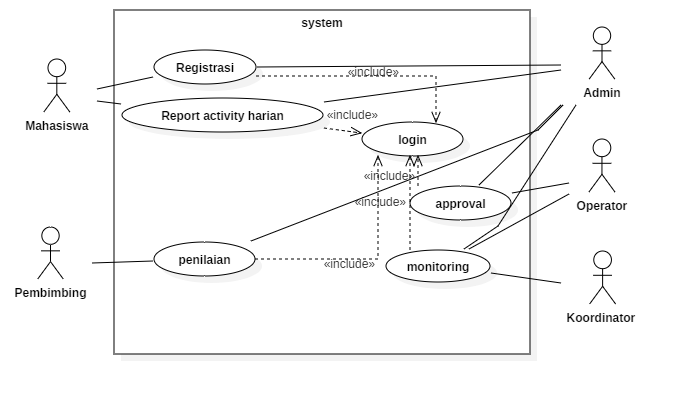
**4.5.2.2. Analisis Dokumen yang dibangun**

Dalam sistem monitoring kinerja mahasiswa *internship* ini terdapat beberapa dokumen yang digunakan, yaitu :

1. Dokumen data laporan, menjelaskan mengenai data-data laporan yang dikirimkan oleh mahasiswa berupa laporan atau materi ke pembimbing untuk dijadikan sebagai penilaian.

**5.2. UML (*Unified Modelling Language*)**

**5.2.1. Usecase Diagram**



*Gambar 5.4 Usecase Diagram Dashboard Monitoring Kinerja Mahasiswa Internship*

**5.2.1.1. Definsi Aktor**

Pada bagian ini akan dijelaskan aktor-aktor yang terlibatkan didalam sistem:

*Tabel 5.1 Definisi Aktor*

|  |  |  |
| --- | --- | --- |
| **No** | **Aktor** | **Deskripsi** |
| 1. | Admin | Admin adalah pihak yang bertanggung jawab atas segala pengolaan data yang ada didalam sistem. |
| 2. | Koordinator | Koordinator adalah pihak yang bertanggung jawab serta melakukan koordinasi kegiatan yang dilakukan. |
| 3. | Mahasiswa | Mahasiswa adalah pihak yang memiliki hak untuk melakukan pengisian *report activity* harian. |
| 4. | Operator | Operator adalah pihak yang bertugas menjaga, melayani, menjalankan suatu peralatan, mesin, telepon dan sebagainya |
| 5. | Pembimbing | Pembimbing adalah pihak yang bertugas untuk melakukan penilaian tugas yang dilakukan oleh mahasiswa. |

**5.2.1.2. Definisi *Usecase***

Pada bagian ini akan dijelaskan usecase yang ada didalam sistem :

*Tabel 5.2 Definis Usecase*

|  |  |  |
| --- | --- | --- |
| **No** | **Usecase** | **Deskripsi** |
| 1. | Registrasi | Merupakan aktivitas untuk melakukan pendaftaran data diri sehingga terhubung dengan sistem. |
| 2. | *Report activity* harian | Merupakan aktivitas untuk mencatat kegiatan yang dilakukan dalam bentuk sebagai laporan. |
| 3. | Approval | Merupakan aktivitas untuk melakukan penerimaan data apabila sesuai dengan kebutuhan. |
| 4. | Penilaian | Merupakan aktivitas untuk melakukan pengolahan informasi dan mengukur terkait pencapaian hasil kinerja yang telah dilakukan. |
| 5. | Monitoring | Merupakan suatu kegiatan mengamati secara seksama suatu keadaan atau kondisi, termasuk juga perilaku atau kegiatan tertentu. |
| 6. | Login | Merupakan pemberian hak akses kepada pihak yang terkait dengan sistem (aktor). |

**5.2.1.3. Skenario Usecase**

Skenario untuk tiap masing-masing usecase diatas adalah :

*Tabel 5.3 Skenario Diagram Usecase Registrasi*

|  |  |
| --- | --- |
| **Identifikasi** | |
| **Nomor** | UC1 |
| **Nama** | *Registrasi* |
| **Tujuan** | Merupakan aktivitas untuk melakukan pendaftaran data diri sehingga terhubung dengan sistem. |
| **Deskripsi** | |
| **Aktor** | Mahasiswa/i |
| **Skenario Utama** | |
| **Kondisi Awal** | Form sudah tersedia |
| **Kondisi Akhir** | Data registrasi mahasiswa telah berhasil ditambahkan. |
| ***Main Flow Event*** | |
| **Aksi aktor** | **Reaksi sistem** |
| 1. Mahasiswa memilih *link register a new membership* | 1. Sistem akan menampilkan form registrasi mahasiswa |
| 1. Mahasiswa akan melakukan *insert* data diri pada form yang tersedia. | 1. Mengambil data dari *database* kemudian terjadi eksekusi terhadap perintah yang diberikan dan nantinya akan disimpan di *database* |
| ***Exceptional Flow of Event*** | Jika aktor salah dalam memasukkan data maka sistem akan menampilkan pesan *error*. Sedangkan jika data valid maka data akan disimpan di *database*. |

*Tabel 5.4 Skenario Diagram Usecase Report activity harian*

|  |  |
| --- | --- |
| **Identifikasi** | |
| **Nomor** | UC2 |
| **Nama** | *Report activity harian* |
| **Tujuan** | Merupakan aktivitas untuk mencatat kegiatan yang dilakukan dalam bentuk sebagai laporan. |
| **Deskripsi** | |
| **Aktor** | Mahasiswa/i |
| **Skenario Utama** | |
| **Kondisi Awal** | Form sudah tersedia |
| **Kondisi Akhir** | Data *report* harian telah berhasil ditambahkan. |
| ***Main Flow Event*** | |
| **Aksi aktor** | **Reaksi sistem** |
| 1. Mahasiswa memilih menu *report* harian | 1. Sistem akan menampilkan form *report activity harian*. |
| 1. Mahasiswa akan melakukan *insert* data laporan form yang tersedia dan melakukan perhitungan. | 1. Mengambil data dari *database* kemudian terjadi eksekusi terhadap perintah yang diberikan dan nantinya akan disimpan di *database* |
| ***Exceptional Flow of Event*** | Jika aktor salah dalam memasukkan data maka sistem akan menampilkan pesan *error*. Sedangkan jika data valid maka data akan disimpan di *database*. |

*Tabel 5.5 Skenario Diagram Usecase approval*

|  |  |
| --- | --- |
| **Identifikasi** | |
| **Nomor** | UC3 |
| **Nama** | *Approval* |
| **Tujuan** | Merupakan aktivitas untuk melakukan penerimaan data apabila sesuai dengan kebutuhan. |
| **Deskripsi** | |
| **Aktor** | Admin, Operator |
| **Skenario Utama** | |
| **Kondisi Awal** | Form sudah tersedia |
| **Kondisi Akhir** | Data mahasiswa telah berhasil dikelola  (*insert, update, delete, view*). |
| ***Main Flow Event*** | |
| **Aksi aktor** | **Reaksi sistem** |
| 1. Aktor memilih menu data mahasiswa | 1. Sistem akan menampilkan form isian data mahasiswa |
| 1. Aktor akan melakukan *insert, update, view* atau *delete* data. | 1. Mengambil data dari *database* kemudian terjadi eksekusi terhadap perintah yang diberikan dan nantinya akan disimpan di *database* |
| ***Exceptional Flow of Event*** | Jika aktor salah dalam memasukkan data maka sistem akan menampilkan pesan *error*. Sedangkan jika data valid maka data akan disimpan di *database*. |

*Tabel 5.6 Skenario Diagram Usecase penilaian*

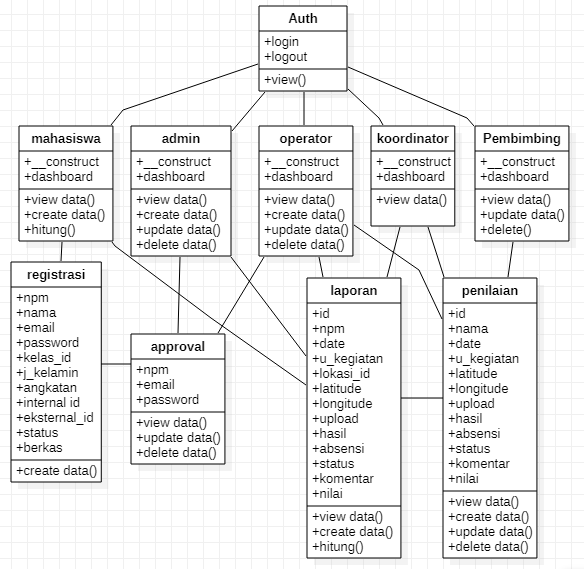
|  |  |
| --- | --- |
| **Identifikasi** | |
| **Nomor** | UC4 |
| **Nama** | *Penilaian* |
| **Tujuan** | Merupakan aktivitas untuk melakukan pengolahan informasi dan mengukur terkait pencapaian hasil kinerja yang telah dilakukan. |
| **Deskripsi** | |
| **Aktor** | Pembimbing |
| **Skenario Utama** | |
| **Kondisi Awal** | Form sudah tersedia |
| **Kondisi Akhir** | Data laporan telah berhasil dikelola  (*insert, update, delete, view*). |
| ***Main Flow Event*** | |
| **Aksi aktor** | **Reaksi sistem** |
| 1. Aktor memilih menu penilaian,selanjutnya meng-klik *detail* data. | 1. Sistem akan menampilkan form isian data laporan mahasiswa per-*progress* |
| 1. Aktor akan melakukan *insert, update, view* atau *delete* data. | 1. Mengambil data dari *database* kemudian terjadi eksekusi terhadap perintah yang diberikan dan nantinya akan disimpan di *database* |
| ***Exceptional Flow of Event*** | Jika aktor salah dalam memasukkan data maka sistem akan menampilkan pesan *error*. Sedangkan jika data valid maka data akan disimpan di *database*. |

*Tabel 5.7 Skenario Diagram Monitoring*

|  |  |
| --- | --- |
| **Identifikasi** | |
| **Nomor** | UC5 |
| **Nama** | *Monitoring* |
| **Tujuan** | Merupakan aktivitas untuk melakukan pengolahan informasi dan mengukur terkait pencapaian hasil kinerja yang telah dilakukan. |
| **Deskripsi** | |
| **Aktor** | Koordinator |
| **Skenario Utama** | |
| **Kondisi Awal** | Form sudah tersedia |
| **Kondisi Akhir** | Semua data yang dikelola.  (*view*). |
| ***Main Flow Event*** | |
| **Aksi aktor** | **Reaksi sistem** |
| 1. koordinator semua menu yang ada pada aplikasi. | 1. Sistem akan menampilkan form isian data dari setiap menu yang dibuka |
| 1. Koordinator akan *view* semua data. | 1. Mengambil data dari *database* kemudian terjadi eksekusi terhadap perintah yang diberikan dan nantinya akan disimpan di *database* |
| ***Exceptional Flow of Event*** | Jika aktor salah dalam memasukkan data maka sistem akan menampilkan pesan *error*. Sedangkan jika data valid maka data akan disimpan di *database*. |

**5.3 Class Diagram**

**5.3.1 Class Diagram**

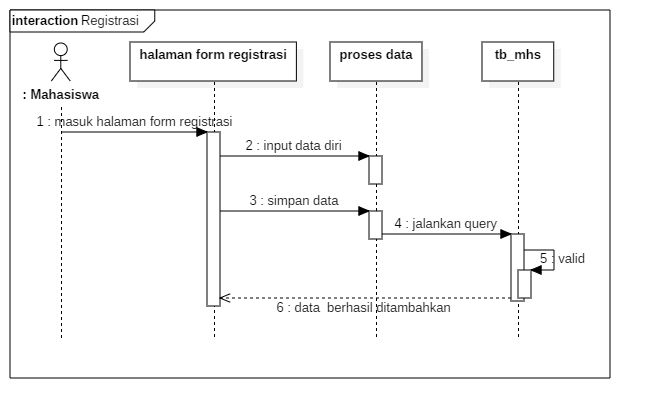


*Gambar 5.5 Class Diagram*

**5.3.2 *Sequence* Diagram**

*Sequence* diagram digunakan untuk memodelkan pengiriman pesan (*message*) antar *objects*. *Sequence* diagram menjelaskan secara detail urutan proses yang dilakukan dalam system untuk mencapai tujuan dari *use case*, intraksi yang terjadi antar *class*, operasi apa saja yang terlibat, urutan antar operasi dan informasi yang diperlukan oleh masing-masing operasi.

1. ***Sequence* Diagram Registrasi**

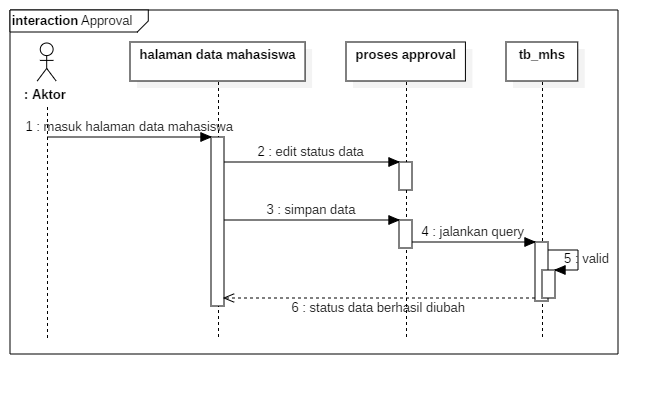


*Gambar 5.6 Sequence Diagram Registrasi*

Keterangan :

Pada *sequence* diagram ini menjelaskan proses registrasi. Aktor yang melakukan proses tersebut adalah mahasiswa. Pertama mahasiswa masuk pada halaman *form* registrasi lalu meng-input-kan data dirinya. Apabila data diri mahasiswa sudah dilengkapi maka proses selanjutnya adalah menyimpan data dan dilanjutkan dengan proses *OpenTable* ke tabel yang dituju yaitu tb\_mahasiswa. Selanjutnya sistem akan memvalidasi hasil dari operasi, apakah berhasil atau gagal yang dimunculkan dengan *reply message*(*alert*).

1. ***Sequence* Diagram *Approval***

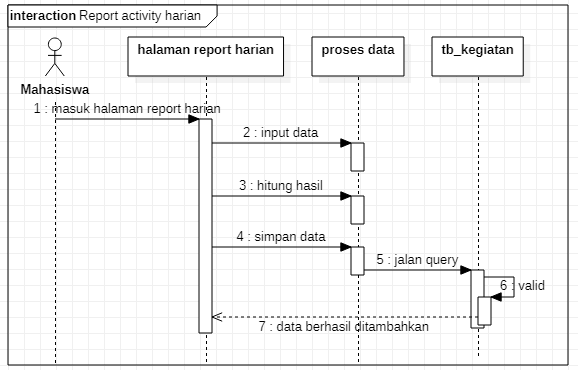


*Gambar 5.7 Sequence Diagram Approval*

Keterangan :

Pada *sequence* diagram ini menjelaskan proses *approval* yang dilakukan oleh admin dan operator. Proses yang dilakukan adalah menerima data registrasi mahasiswa/i untuk memberi hak akses masuk ke aplikasi. Pertama aktor masuk ke halaman data mahasiswa lalu meng-*edit* status datanya, kemudian melakukan perintah simpan data dan dilanjutkan dengan proses *OpenTable* ke tabel yang dituju yaitu tb\_mahasiswa. Selanjutnya sistem akan memvalidasi hasil dari operasi, apakah berhasil atau gagal yang dimunculkan dengan *reply message*(*alert*).

1. ***Sequence* Diagram *Report Activity* Harian**

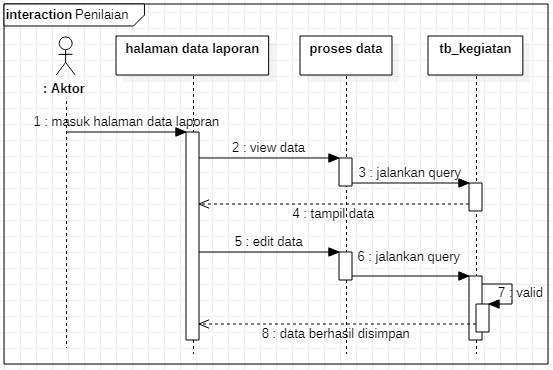


*Gambar 5.8 Sequence Diagram Report Activity Harian*

Keterangan :

Pada *sequence* diagram ini menjelaskan proses untuk meng-*input* data kegiatan laporan harian yang dilakukan ditempat *internship* dan aktor yang melakukannya adalah mahasiswa. Untuk melakukan proses tersebut mahasiswa masuk ke halaman *report* harian yang sudah tersedia *form input*-an, lalu mengisi data sesuai dengan kegiatan yang dilakukan serta melakukan perhitungan jarak dan proses selanjutnya menyimpan data, kemudian dilanjutkan dengan proses *OpenTable* ke tabel yang dituju yaitu tb\_kegiatan. Selanjutnya sistem akan memvalidasi hasil dari operasi, apakah berhasil atau gagal yang dimunculkan dengan *reply message*(*alert*).

1. ***Sequence* Diagram Penilaian**

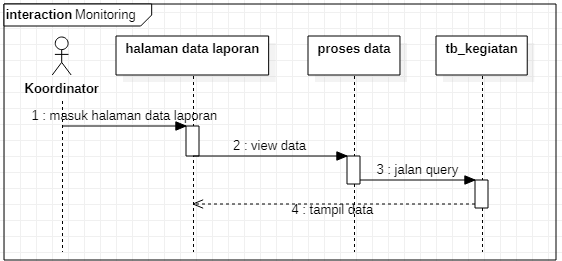


*Gambar 5.9 Sequence Diagram Penilaian*

Keterangan :

Pada *sequence* diagram ini menjelaskan proses penilaian terhadap kinerja mahasiswa dalam kontek me-monitoring dari data laporan masuk atau kegiatan yang dilakukan yang dituangkan dalam laporan. Dibalik proses tersebut aktor yang melakukannya adalah pembimbing, pertama aktor masuk ke halaman data laporan. Kemudian sistem akan menampilkan semua data laporan masuk, lalu aktor akan melakukan *edit* data untuk memberikan keterangan absensi, status, nilai dan dilanjutkan dengan proses *OpenTable* ke tabel yang dituju yaitu tb\_penilaian. Selanjutnya sistem akan memvalidasi hasil dari operasi, apakah berhasil atau gagal yang dimunculkan dengan *reply message*(*alert*).

1. ***Sequence* DiagramMonitoring**



*Gambar 5.10 Sequence Diagram Monitoring*

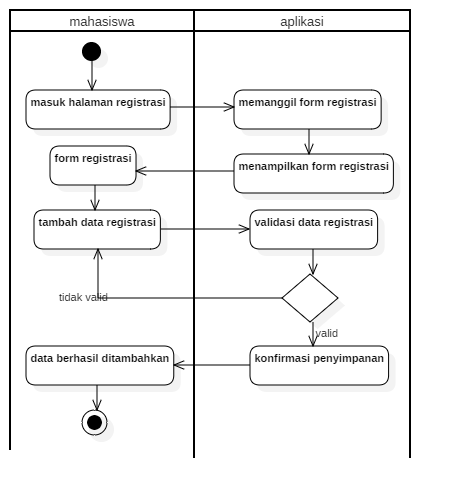
Keterangan :

Pada *sequence* diagram ini menjelaskan proses kelola laporan yang dilakukan oleh aktor koordinator. Pertama aktor masuk ke halaman data laporan dan sistem akan menampilkan semua data laporan serta menampilkan *button* aksi yaitu *view* data. Selanjutnya aktor meng-klik *button* tersebut dan dilanjutkan dengan proses *OpenTable* ke tabel yang dituju yaitu tb\_kegiatan. Selanjutnya sistem akan memvalidasi hasil dari operasi, apakah berhasil atau gagal yang dimunculkan dengan *reply message*(*alert*).

**5.3.3 *Activity* Diagram**

*Activity* diagram menggambarkan berbagi aliran aktivitas dalam system yang sudah dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berhasil. *Activity* diagram juga dapat menggambarkan proses pararel yang mungkin terjadi pada beberapa eksekusi.

1. ***Activity* Diagram Registrasi**



*Gambar 5.11 Activity Diagram Registrasi*

Keterangan :

Pada *activity* diagram ini menjelaskan proses registrasi yang dilakukan oleh aktor mahasiswa. Mahasiswa masuk ke halaman registrasi lalu sistem menampilkan halaman *form* registrasi ke mahasiswa. Setelah itu mahasiswa dapat melakukan pengolahan data registrasi. Pengolahan yang pertama yaitu tambah data registrasi yang dimana sistem akan melakukan pengecekan apakah data sudah ditambahkan atau belum ditambahkan, apabila telah ditambahkan maka sistem dapat mengkonfirmasi penyimpanan dan menampilkan *alert*.

1. ***Activity* Diagram *Approval***

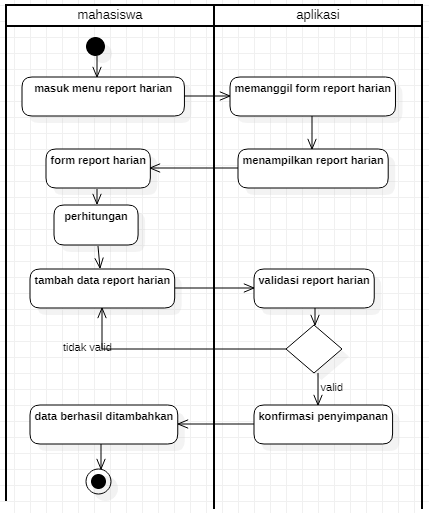


*Gambar 5.12 Activity Diagram Approval*

Keterangan :

Pada activity diagram ini menjelaskan proses approval yang dilakukan oleh aktor admin dan operator. Aktor masuk menu data mahasiswa lalu sistem menampilkan halaman data mahasiswa ke aktor. Setelah itu aktor dapat pengolahan data mahasiswa. Pengolahan yang pertama yaitu edit status data mahasiswa yang dimana sistem akan melakukan pengecekan apakah data sudah diubah atau sebaliknya, apabila telah diubah sistem dapat mengkonfirmasi penyimpanan dan menampilkan a*lert*.

1. ***Activity* Diagram *Report Activity* Harian**

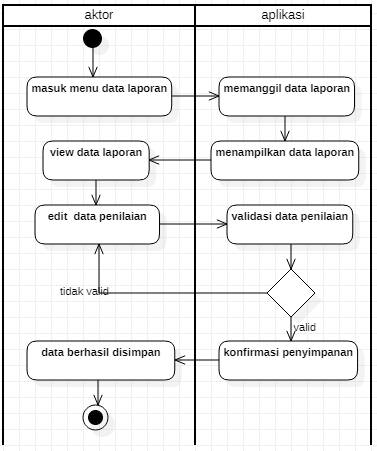


*Gambar 5.13 Activity Diagram Report Activity Harian*

Keterangan :

Pada *activity* diagram ini menjelaskan proses *report* harian yang dilakukan oleh aktor mahasiswa. Mahasiswa masuk menu *report* harian kemudian sistem menampilkan halaman data laporan ke mahasiswa. Setelah itu mahasiswa dapat melakukan pengolahan data laporan. Pengolahan yang pertama yaitu tambah data laporan dan perhitungan yang dimana sistem akan melakukan pengecekan apakah data sudah ditambahkan atau belun ditambahkan,apabila telah ditambahkan maka sistem dapat mengkonfirmasi penyimpanan dan menampilkan *alert*.

1. ***Activity* Diagram Penilaian**

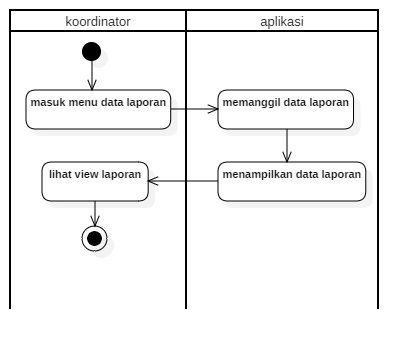


*Gambar 5.14 Activity Diagram Penilaian*

Keterangan :

Pada *activity* diagram ini menjelaskan proses penilaian yang dilakukan oleh aktor pembimbing. Admin masuk menu data laporan lalu sistem menampilkan halaman data laporan ke admin. Setelah itu admin dapat melakukan pengolahan data laporan. Pengolahan yang pertama yaitu detail data laporan yang dimana sistem akan menampilkan data-data keterangan dari laporan itu sendiri dan pengolahan yang kedua yaitu tambah data penilaian yang dimana sistem akan melakukan pengecekan apakah data sudah ditambahkan atau belum ditambahkan, apabila telah ditambahkan maka sistem dapat mengkonfirmasi penyimpanan dan menampilkan *alert*.

1. ***Activity* Diagram Monitoring**



*Gambar 5.15 Activity Diagram Monitoring*

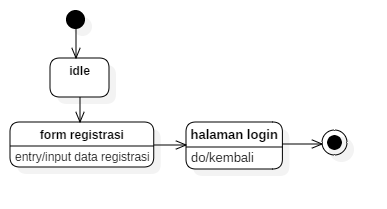
Keterangan :

Pada *activity* diagram ini menjelaskan proses kelola laporan yang dilakukan oleh aktor koordinator. Aktor masuk menu data laporan dan sistem menampilkan halaman data laporan ke aktor.setelah itu aktor dapat melakukan *check*-ing data laporan dengan memberikan aksi pada *button view*, lalu sistem menampilkan keterangan data laporan yang lengkap.

**5.3.4 *Statechart* Diagram**

*Statechart* diagram menggambarkan transisi dan perubahan keadaan dari suatu *state* ke *state* lainnya. Suatu objek pada *system* sebagai akibat dari stimulansi yang diterima *statechart* diagram mendeskripsikan bagaimana suatu objek mengalami perubahan status adanya *trigger* dan *event-event*. Menunjukkan kondisi yang dapat dialami atau terjadi pada sebuah objek.

1. ***Statechart* Diagram Registrasi**

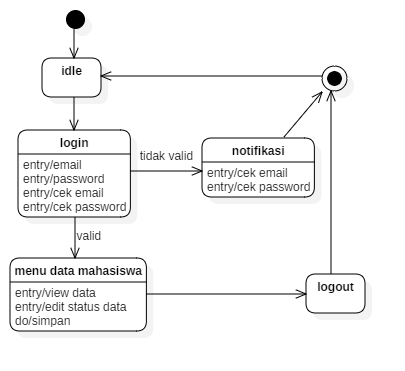


*Gambar 5.16 Statechart Diagram Registrasi*

Keterangan :

Pada *statechart* diagram ini menjelaskan proses program registrasi yang dilakukan oleh aktor mahasiswa. Aplikasi dalam keadaan *idle* dan status aplikasi berubah saat mahasiswa melakukan aksi yaitu meng-*input* data pada form registrasi. Setelah melakukan operasi tersebut, aktor kembali pada halaman *login*.

1. ***Statechart* Diagram *Approval***

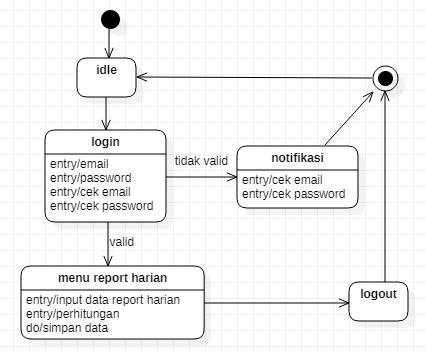


*Gambar 5.17 Statechart Diagram Approval*

Keterangan :

Pada *statechart* diagram ini menjelaskan proses *approval* data mahasiswa yang dilakukan oleh aktor admin dan operator. Aplikasi dalam keadaan *idle*, kemudian status aplikasi berubah saat aktor melakukan *login*. Apabila login valid maka aktor di alihkan ke halaman utama dan jika *login* tidak valid maka aktor kembali ke halaman *login*. Setelah aktor berhasil melakukan *login* dan masuk ke menu data mahasiswa maka aktor dapat melakukan pegolahan berupa *view*, *edit* dan hapus data mahasiswa. Setelah melakukan pengolahan data mahasiswa maka aktor dapat melakukan *logout*.

1. ***Statechart* Diagram *Report Activity* Harian**

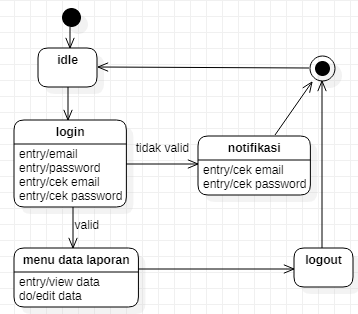


*Gambar 5.18 Statechart Diagram Report Activity Harian*

Keterangan :

Pada *statechart* diagram ini menjelaskan proses meng-*input* data laporan kegiatan harian yang dilakukan oleh aktor mahasiswa. Aplikasi dalam keadaan *idle* lalu status aplikasi berubah saat mahasiswa melakukan *login*. Apabila *login* valid maka mahasiswa dialihkan ke halaman utama dan apabila *login* tidak valid maka mahasiswa kembali ke halaman login. Setalah mahasiswa berhasil melakukan *login* dan masuk ke halaman utama maka mahasiswa memilih menu *report* harian dan melakukan pengelolaan berupa input, perhitungan dan simpan data *report* harian. Setelah melakukan pengelolaan data *report* harian maka mahasiswa dapat melakukan *logout*.

1. ***Statechart* Diagram Penilaian**

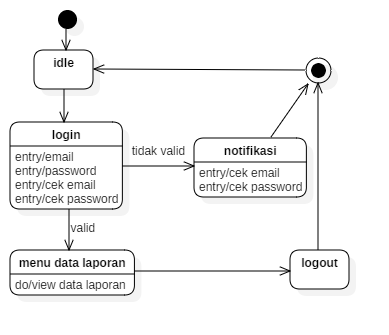


*Gambar 5.19 Statechart Diagram Penilaian*

Keterangan :

Pada *statechart* diagram ini menjelaskan proses penilaian yang dilakukan oleh aktor pembimbing. Aplikasi dalam keadaan *idle* lalu status aplikasi berubah saat aktor melakukan *login*. Apabila *login* valid maka aktor di alihkan ke halaman utama dan apabila *login* tidak valid maka aktor kembali ke halaman *login*. Setelah aktor berhasil melakukan *login* dan masuk ke halaman utama maka aktor masuk ke menu data laporan serta melakukan pengelolaan berupa tambah, *view* dan *edit* data laporan. Setelah melakukan pengelolaan data laporan maka aktor dapat melakukan *logout*.

1. ***Statechart* Diagram Monitoring**

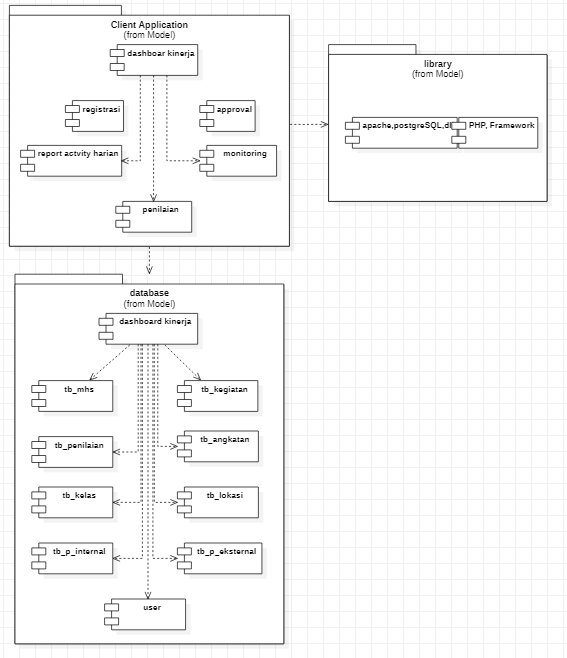


*Gambar 5.20 Statechart Diagram Monitoring*

Keterangan :

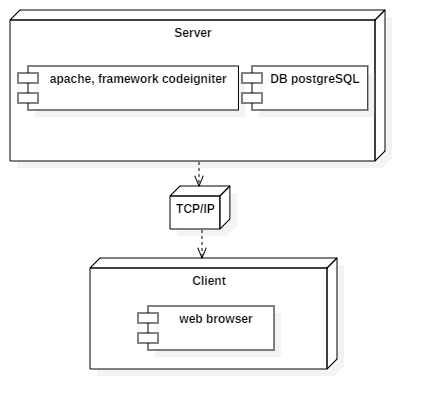
Pada *statechart* diagram ini menjelaskan proses kelola laporan yang dilakukan oleh aktor koordinator. Aplikasi dalam keadaan *idle* lalu status aplikasi berubah saat aktor melakukan *login*. Apabila login valid maka aktor di alihkan ke halaman utama dan apabila *login* tidak valid maka aktor kembali ke halaman *login*. Setelah admin berhasil melakukan *login* dan masuk ke menu data laporan maka aktor dapat melakukan aski seperti *check*-ing data dengan menekan pada *button* detail dan sistem akan menampilkan data yang lengkap. Setelah melakukan *check*-ing data laporan maka aktor dapat melakukan *logout.*

**5.3.5 *Component* Diagram**



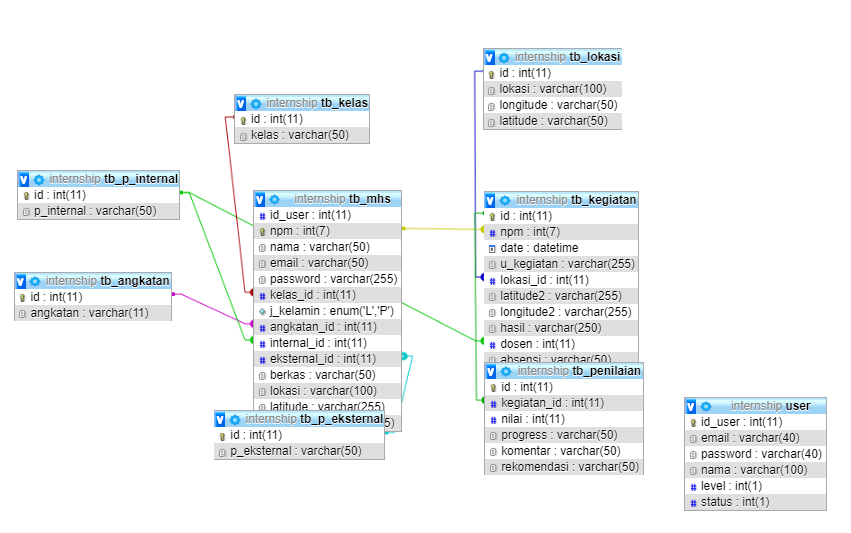
*Gambar 5.21 Component Diagram*

**5.3.6 *Deployment* Diagram**



*Gambar 5.22 Deployment Diagram*

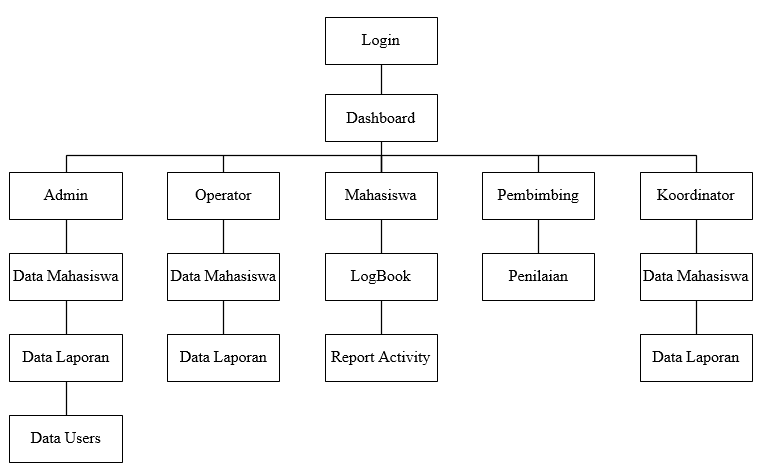
**5.3.7 Perancangan Database**



*Gambar 5.23 Perancangan Database*

**5.3.8 Struktur Menu**

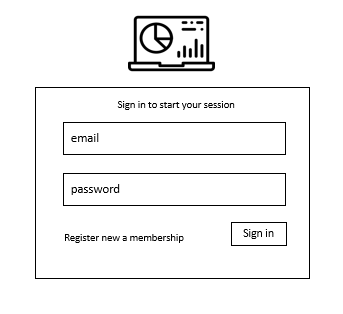
Fungsi-fungsi yang dirancang pada tahap perancangan ini dibagi ke dalam beberapa menu yang bertujuan untuk memudahkan pengoperasian program. Menu yang digunakan pada program saat ini dapat dilihat pada struktur berikut :



*Gambar 5.24 Struktur Menu*

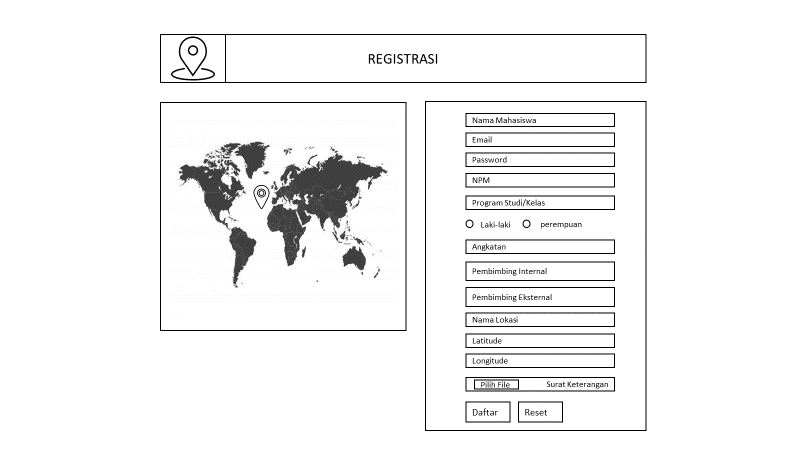
**5.3.9 Perancangan *Interface***

1. **Form *Login***



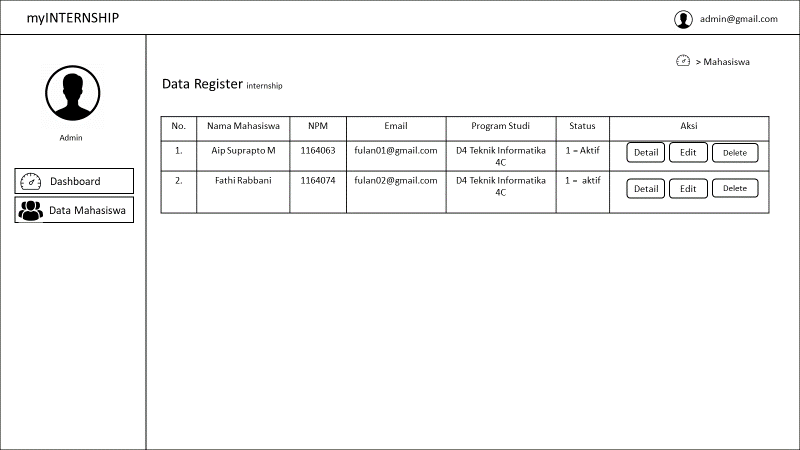
*Gambar 5.25 Login*

1. **Form Registrasi**

****

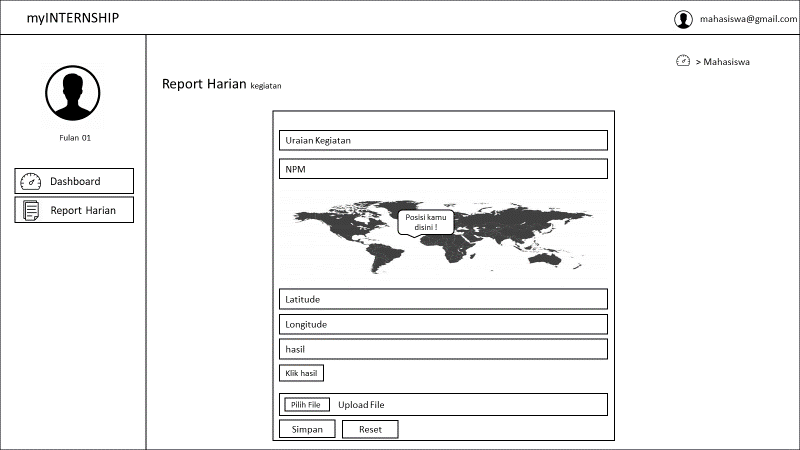
*Gambar 5.26 Form Registrasi*

1. **Halaman *Approval* Data Mahasiswa**

****

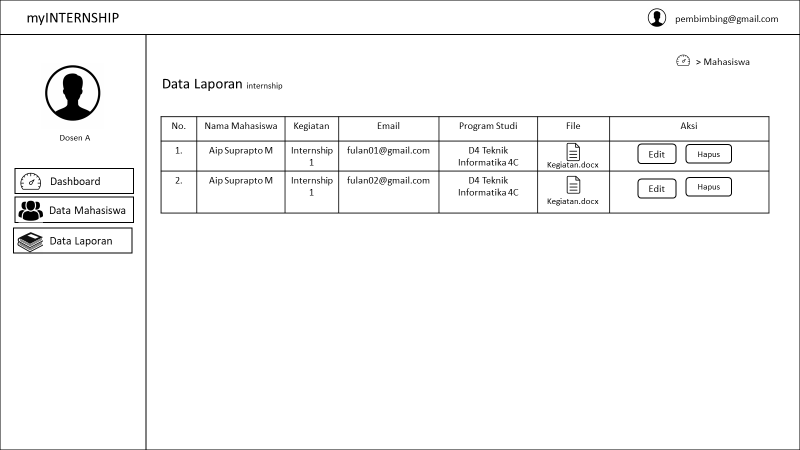
*Gambar 5.27 Halaman Approval Data Mahasiswa*

1. ***Form Report Activity* Harian**

****

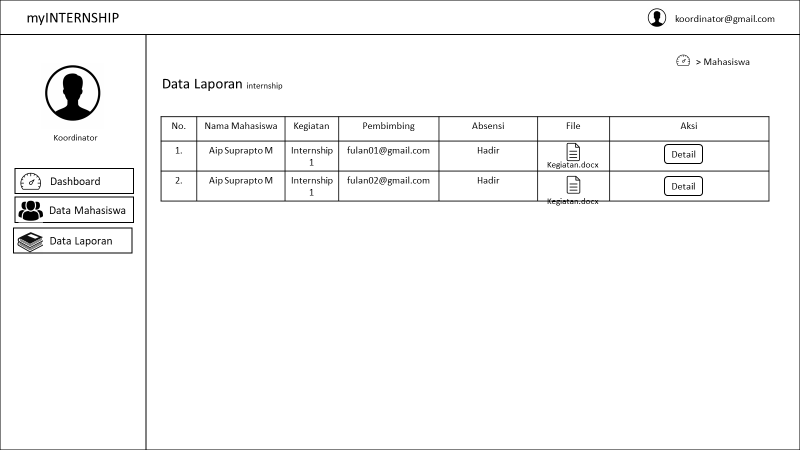
*Gambar 5.28 Form Report Activity Harian*

1. ***Form* Penilaian**

****

*Gambar 5.29 Form Penilaian*

1. ***Form* Monitoring**

****

*Gambar 5.30 Form Monitoring*

**5.4** **Perancangan Arsitektur Perangkat Lunak dan Perangkat Keras Sistem**

Dalam perancangan dashboard monitoing kinerja mahasiswa internship membutuhkan beberapa perangkat lunak yaitu sebagai berikut.

**5.4.1 Perangkat Lunak**

Perangkat lunak pendukung yang digunakan adalah sebagai berikut :

*Tabel 5.8 Perangkat Lunak*

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Jenis |  | Keterangan |
| 1. | Sistem Operasi | : | *Microsoft Windows 10 Profesional 64-bit* |
| 2. | Bahasa Pemrograman | : | *PHP* dengan *Framework CI* (*Codeignite*r) |
| 3. | Database | : | *PostgreSQL* |
| 4. | Perangkat Lunak | : | *Visual Studio Code*  *Microsoft Visio* 2013  *StarUML* 3.1.0 |

*Tabel 5.8 Perangkat Lunak*

**5.4.2 Perangkat Keras**

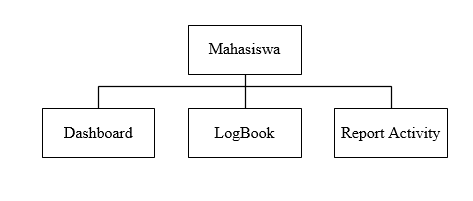
Perangkat keras pendukung yang digunakan adalah sebagi berikut :

*Tabel 5.8 Perangkat Keras*

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Jenis |  | Keterangan |
| 1. | *Processor* | : | Intel® core™i3 |
| 2. | *Memory* | : | 4 GB |
| 3. | *Monitor* | : | LCD 14,1 *Inchi* |
| 4. | *Mouse dan keyboard* | : | *Standard* |

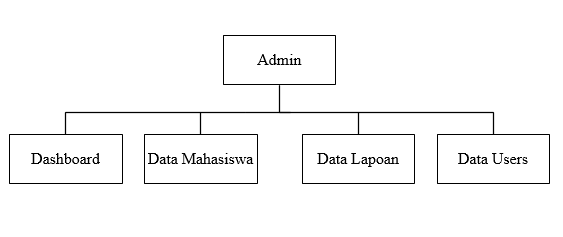
*Tabel 5.9 Perangkat Keras*

**5.5 Pemetaan Struktur Diagram User / Aktor Sistem**

 Dibawah ini merupakan pemetaan struktur diagram menu untuk mahasiswa.

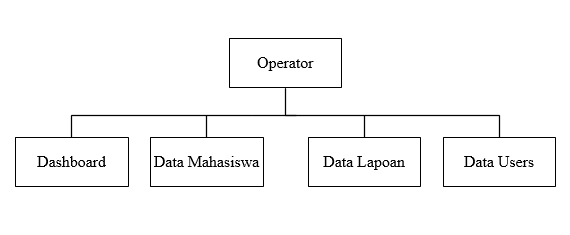
*Gambar 5.31 Pemetaan Struktur Diagram Mahasiswa*

Dibawah ini merupakan pemetaan struktur diagram menu untuk admin.

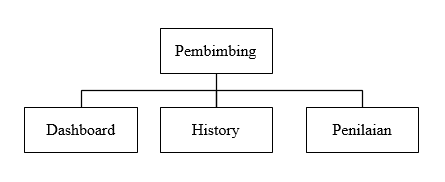


*Gambar 5.32 Pemetaan Struktur Diagram Admin*

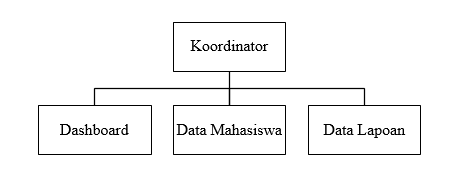
Dibawah ini merupakan pemetaan struktur diagram menu untuk operator.



*Gambar 5.33 Pemetaan Struktur Diagram Operator*

Dibawah ini merupakan pemetaan struktur diagram menu untuk pembimbing.

*Gambar 5.34 Pemetaan Struktur Diagram Pembimbing*

Dibawah ini merupakan pemetaan struktur diagram menu untuk koordinator

*Gambar 5.35 Pemetaan Struktur Diagram Koordinator*