# BAB II
# LANDASAN TEORI

## 2.1. Tinjauan Teori Umum

Dalam penyusunan skripsi ini dicantumkan beberapa teori umum yang dapat dijadikan sebagai landasan teori, penulis juga menggunakan bahan acuan kepustakaan yang bersumber pada penelitian-penelitian sebelumnya. Hal ini berguna sebagai pembanding serta bahan referensi bagi penulis.

### 2.1.1. Definisi Aplikasi

Aplikasi adalah suatu program yang siap untuk digunakan yang dibuat untuk melaksanankan suatu fungsi bagi pengguna jasa aplikasi serta penggunaan aplikasi lain yang dapat digunakan oleh suatu sasaran yang akan dituju (Ilmiah et al. 2015).

### 2.1.2. *Android*

*Android* adalah sistem operasi yang berbasis *Linux* untuk telepon seluler seperti telepon pintar dan komputer tablet. *Android* menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak (Lukman 2016).

*Android* adalah sistem operasi dan *platform* perangkat lunak yang digunakan untuk pengembangan aplikasi. *Android*, yang merupakan game

changer potensial bagi organisasi pengembangan *mobile*. *Android* ditempatkan dengan baik untuk memenuhi kebutuhan pasar yang semakin meningkat. Android adalah yang pertama dalam teknologi *platform* pengembangan *mobile* generasi baru, yang membuat *platform*nya dikembangkan membuka keunggulan yang berbeda bagi para pesaingnya (Dongare et al. 2014).

Perbedaan *Platform Android* :

- *Complete* : Perancang mengambil pendekatan komprehensif saat mengembangkan *Platform Android*.
- *Open* : *Platform Android* menyediakan lisensi *open source*.
- Gratis : Aplikasi *Android* bebas berkembang. Tidak diperlukan biaya pendaftaran atau sertifikasi.

Menurut (Ghosal et al. 2015) Ada beberapa fitur dari aplikasi *android* diantaranya:

a. *Android Open Accessory Development Kit*

Perangkat keras USB diizinkan terhubung dengan android perangkat melalui *Android Open Accessory support* yang mana adalah bagian dari Android 3.1. Perangkat keras USB *eksternal* berkomunikasi dengan perangkat dalam mode khusus yang disebut Mode "*aksesori*".

b. *App Inventor for android applications*

Berdasarkan *Open Blocks Java Library* di MIT berbasis *web* Lingkungan dikembangkan terutama untuk *visualisasi*

c. *Native development kit*

*System load Library* dapat memuat kode *Java* dari *native* kelas di bawah *Virtual Machine* bernama Dalvik VM.

d. *Hyper Next Android Creator*

Pemrogram pemula bisa mendapatkan banyak bantuan untuk membuat *Android* Aplikasi dengan bantuan *Hyper Next Android Creator* (HAC).

### 2.1.3. Defini Pemesanan

Menurut kotler pemesanan barang dan jasa adalah suatu kegiatan transaksi yang menyatakan keinginan atau rencana untuk memiliki atau membeli barang dan jasa tersebut sebelum melakukan transaksi finasial yakni berupa pembayaran finasial terhadap produk atau jasa yang diinginkan (Dwi Puspita 2014).

Tujuan pemesanan yaitu :

1. Memaksimumkan pelayanan bagi konsumen
2. Meminimumkan investasi pada persediaan
3. Perencanaan kapasitas
4. Pengesahan produksi dan pengendalian produksi
5. Persediaan dan kapasitas
6. Penyimpanan dan pergerakan material
7. Peralatan, *routing* dan proses *planning*
8. Dan lain–lain

### 2.1.4. Definisi UKM

Usaha kecil dan menengah (UKM) adalah salah satu bidang yang memberikan kontribusi yang segnifikan dalam memacu pertumbuhan ekonomi Indonesia. Hal ini dikarenakan daya serap UKM terhadap tenaga kerja yang sangat besar dan dekat dengan rakyat kecil (Jaidan Jauhari 2010).

Usaha kecil dan menengah (UKM) adalah pelaku bisnis yang bergerak pada berbagai bidang usaha, yang menyentuh kepentingan masyarakat. Data BPS (2010), jumlah usaha kecil dan menengah (UKM) mencapai 42,5 juta unit atau 99,9 persen dari keseluruhan pelaku bisnis di tanah air. UKM memberikan kontribusi yang signifikan terhadap penyerapan tenaga kerja, yaitu sebesar 99,6 persen. Sementara itu, kontribusi UKM terhadap Produk Domestik Bruto (PDB) sebesar 56,7 persen (Nurhayati 2015).

### 2.1.5. *Software Development Life Cycle (SDLC)*

*SDLC* atau *Software Development Life Cycle* adalah proses mengubah atau mengembangkan suatu sistem perangkat lunak dengan menggunakan model-model dan metodologi yang digunakan orang untuk mengembangkan sistem-sistem perangkat lunak sebelumnya (berdasarkan *best practice* atau cara-cara yang sudah teruji baik) (Rosa A.S dan Shalahuddin 2013).

### 2.1.6. *Database*

Menurut Kristanto, dalam jurnal(Lia Septiningrum, Wawan Laksito 2012) yang tertuang dalam buku yang berjudul “Konsep dan Perancangan *Database*” menjelaskan bahwa *Database* adalah kumpulan file-file yang mempunyai kaitan

antar satu file dengan file yang lain sehingga membentuk satu bangunan data untuk menginformasikan satu perusahaan, instansi dalam batasan tertentu.

Sedangkan menurut Fatansyah, dalam jurnal(Lia Septiningrum, Wawan Laksito 2012) Sistem basis data adalah sistem yang terdiri atas kumpulan file (tabel) yang saling berhubungan (dalam sebuah basis data di sebuah sistem komputer) dan sekumpulan program (DBMS) yang memungkinkan beberapa pemakai dan atau program lain untuk mengakses dan memanipulasi file-file.

*Database* terdiri dari dua penggalan kata yaitu data dan base, yang artinya berbasiskan pada data, tetapi secara konseptual, d*atabase* diartikan sebuah koleksi atau kumpulan data-data yang saling berhubungan (*relation*), disusun menurut aturan tertentu secara logis, sehingga menghasilkan informasi. Secara prinsip, dalam suatu *database* tercakup dua komponen penting, yaitu data dan informasi (Daniel Bunga Paillin 2012).

## 2.2. Tinjauan Teori Khusus

Dalam penyusunan ini di cantumkan beberapa teori khusus yang enjelaskan tentang proses pembuatan aplikasi pemesanan bibit ikan. Berikut beberapa teori dan pendapat para ahli, diantaranya dalah :

### 2.2.1. *Waterfall Model*

Model SDLC air terjun (*waterfall*) sering juga disebut model sekuensial linier (*sequential liniear*) atau alur hidup klasik (*classic life cycle*).

Model air terjun menyediakan pendekatan alur hidup perangkat lunak secara sekuensial atau terurut dimulai dari analisis, desain, pengodean, pengujian dan

tahap pendukung (*support*). Berikut adalah gambar model air terjun : (Rosa A.S dan Shalahuddin 2013).

**Gamabr 2.1** Ilustrasi Model *Waterfall*

1. Analisa kebutuhan perangkat lunak

   Proses penginputan kebutuhan dilakukan secara intensif untuk menspesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh *user*. Spesifikasi kebutuhan perangkat lunak pada tahap ini perlu untuk dilakukan dokumentasi.

2. Desain

   Desain perangkat lunak adalah proses multi langkah yang fokus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antarmuka, dan prosedur pengodean. Tahap ini mentranslasi kebutuhan perangkat lunak dari tahap analisis kebutuhan ke representasi desain agar dapat diimplementasikan menjadi program pada tahap selanjutnya. Desain perangkat lunak yang dihasilkan pada tahap ini juga perlu didokumentasikan.

3. Pembuatan kode program

   Desain harus diimplementasikan kedalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai dengan desain yang telah

dibuat pada tahap desain.

4. Pengujian

Pengujian fokus pada perangkat lunak secara dari segi lojik dan fungsional dan memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk meminimalisir kesalahan (*error*) dan memastikan keluaran yang dihasilkan sesuai dengan yang diinginkan.

5. Pendukung (*support*) atau pemeliharaan (*maintenance*)

Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah dikirimkan ke *user*. Perubahan bisa terjadi karena adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian atau perangkat lunak harus beradaptasi dengan lingkungan baru.Tahap pendukung atau pemeliharaan dapat mengulangi proses pengembangan mulai dari analisis spesifikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru.

### 2.2.2. *Unifield Modeling Language* (UML)

UML (*Unifield Modeling Language*) adalah bahasa yang digunakan untuk memodelkan perangkat lunak yang akan dibuat dan perlu adanya standarisasi agar orang diberbagai negara dapat mengerti pemodelan perangkat lunak" (Ujang Mulayana dan Dian Gustina 2016). UML (*Unifed Modeling Language*) merupakan alat bantu untuk pemodelan sistem dengan menggunakan OOAD. UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembangan sistem untuk membuat cetak biru rancangan sistem dalam bentuk

baku, mudah di mengerti dan dilengkapi dengan mekanisme efektif untuk berbagi serta mengkomunikasikan rancangan sistem (Darlianto et al. 2016).

**2.2.2.1. *Class Diagram***

Class adalah spesifikasi yang akan menghasilkan objek dan merupakan inti dari pengembangan dan desain berorientasi objek. Class menggambarkan keadaan (atribut atau properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (Rosa A.S dan Shalahuddin 2013). Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut :

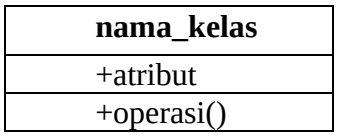
a. Kelas main

b. Kelas yang menangani tampilan sistem (*view*)

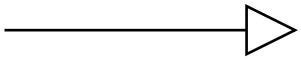
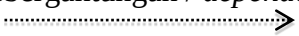
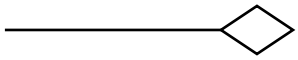
c. Kelas yang diambil dari pendefinisian *use case* (*controller*)

d. Kelas yang diambil dari pendifinisian data (model)

**Tabel 2.1** Simbol *Class Diagram*

| Simbol | Deskripsi |
|---|---|
| Kelas<br>  <br> | Kelas pada struktur system |
| Antarmuka / *Interface*<br> <br>**nama_interface** | Sama dengan konsep *interface* dalam pemograman berorientasiobjek |
| Asosiasi / *association*<br> | Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity* |

Tabel lanjutan

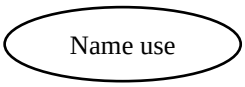
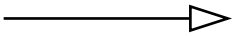
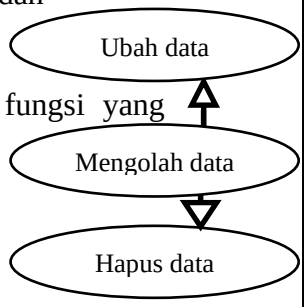
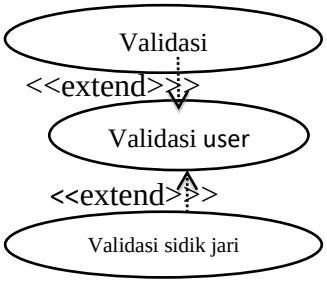
| Simbol | Deskripsi |
|---|---|
| Asosiasi berarah / *ditected association* | Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity* |
| Generalisasi  | Relasi antar kelas dengan makna generalisasi-spesialisasi(umum khusus) |
| Kebergantungan / *dependency*  | Relasi antara kelas dengan makna kebergantungan antar kelas |
| Agregasi / *aggregation*  | Relasi antar kelas dengan makna semua bagian ( *whol –part* ) |

**2.2.2.2. *Use Case Diagram***

*Use case diagram* merupakan permodelan untuk melakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat (Rosa A.S dan Shalahuddin 2013). Ada dua hal utama pada *use case* yaitu :

1. Aktor merupakan orang, proses, atau sistem lain yang akan berinteraksi dengan sistem informasi yang akan dibuat.
2. Use case merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau pesan.

**Tabel 2.2** Simbol *Use case Diagram*

| Simbol | Deskripsi |
|---|---|
| *Use case*<br>  | Fungsional yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antara unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja diawal di awal frase nama *use case.* |
| Aktor / *actor*<br> | Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, biasanya dinyatakan menggunakan kata benda diawal frase nama aktor. |
| Generalisasi/*generalization*<br>  | Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainya, misalnya Arah panah mengarah pada *use case* yang menjadi generalisasi (umum)<br>  |
| Ektensi / *extend*<br><<extend>><br> | Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan yaitu , mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek, biasanya *use case* tambahan memiliki nama depan sama dengan *use case* yang ditambahkan misal Arah panah mengarah pada *use case* yang ditambahkan<br>  |

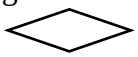
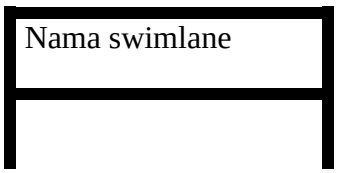
Tabel lanjutan

| Simbol | Deskripsi |
|---|---|
| Menggunakan / include / *uses*<br><br><<include>><br><br> <br><br><<uses>><br> | Relasi *use case* tambah ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case.* Ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini<br><br>Ada dua sudut pandang yang cukup besar mengenai *include* di *use case*: *Include* berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* yang ditambahkan dijalankan. *Include* berarti *use case* yang tambahan akan selalu melakukan pengecekan apakah *use case* yang ditambahkan telah dijalankan sebelum *use case* tambahan dijalankan. Kedua interpretasi di atas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan |
| Asosiasi / *association*<br> | Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor. |

### 2.2.2.3. *Activity Diagram*

*Activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak (Rosa A.S dan Shalahuddin 2013).

**Tabel 2.3** Simbol *Ativity Diagram*

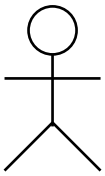
| Simbol | Deskripsi |
|---|---|
| Status awal | Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal |
| Aktivitas | Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja |
| Percabangan / *decision*  | Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu |
| Penggabungan / *joint*  | Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu |
| Status Akhir | Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir |
| Swimlane   | Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi |

### 2.2.2.4. *Sequence Diagram*

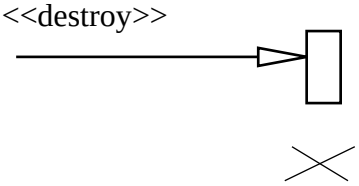
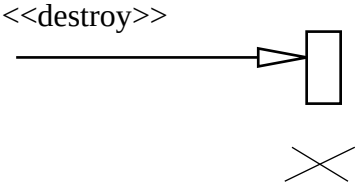
Diagram sekuen atau *sequence diagram* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan mesage yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah use

case beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu (Rosa A.S dan Shalahuddin 2013).

**Tabel 2.4** Simbol *Diagram Sekuen*

| Simbol | Deskripsi |
|---|---|
| Aktor / *actor*<br>  <br>Atau<br>Nama aktor<br>Tanpa waktu aktif | Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, biasanya dinyatakan menggunakan kata benda diawal frase nama aktor. |
| Garis hidup<br> | Menyatakan kehidupan suatu objek |
| Waktu aktif<br> | Menyatakan objek dalam keadaan aktif dan berinteraksi pesan |
| Objek<br>Nama objek : nama kelas | Menyatakan objek berinteraksi pesan |

Tabel Lanjutan

| Simbol | Deskripsi |
|---|---|
| Pesan tipe destroy<br><br><<destroy>><br>  | Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri. |
| Pesan tipe destroy<br><br><<destroy>><br>  | Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri. |

**2.2.2.5. Skenario Diagram**

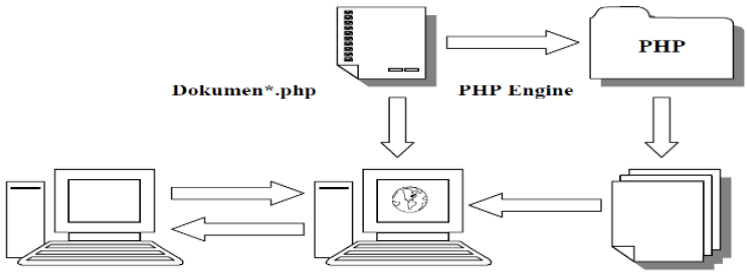
Skenario adalah sebuah cerita atau narasi yang menjelaskan alur dari sistem tersebut. Akan tetapi skenario bukan deskripsi spesifikasi teknis dari suatu produk perangkat lunak. Skenario memiliki beberapa komponen utama, yaitu tujuan, ruang lingkup, sudut pandang pemangku kepentingan, visualisasi, singkat, rekursif, dekomposisi, dan penyempurnaan. Dari sebuah skenario, seorang analis dapat dengan mudah menganalisa apa saja diagram kasus penggunaan yang akan dibuat. Hal ini dikarenakan, dalam skenario tertulis siapa saja aktor yang berperan di dalamnya dan apa saja hal yang dilakukan oleh aktor itu sendiri (Fauzan & Pramono n.d.).

**Tabel 2.5** Skenario Diagram

| | |
|---|---|
| Use Case Name | menghitung honor dosen |
| Actor | BAAK dan Puket 1 |
| Description | *Use case* menggambarkan BAAK menghitung honor dosen dan diserahkan oleh *actor* puket 1 |
| Pre condition | BAAK harus menghitung honor dosen terlebih dahulu |
| Post Condition | Pemberian data honor dosen kepada puket 1 |
| Scenario | Honor dosen dihitung oleh bagian BAAK sesuai dengan honor yang di terima dan absensi dosen mengajar, setelah itu data hasil hitungan honor dosen diserahkan kepada Puket 1 untuk disetujui |

### 2.2.3. PHP (VERSI 5)

Menurut Arief (2011) "PHP (*Hypertext Preprocessor*) adalah bahasa *server-side scripting* yang menyatu dengan HTML untuk membuat halaman *web* yang dinamis. Karena PHP merupakan *server-side scripting* maka sintaks dan perintah-perintah PHP akan dieksekusi di *server* kemudian hasilnya dikirimkan ke *browser* dalam format HTML. Dengan demikian kode program yang ditulis dalam PHP tidak akan terlihat oleh user sehingga keamanan halaman *web* yang dinamis, yaitu halaman *web* yang dapat membentuk suatu tampilan berdasarkan permintaan terkini, seperti menampilkan isi basis data ke halaman *web* (Ujang Mulayana dan Dian Gustina 2016). Simulasi proses eksekusi kode PHP yang disisipkan pada halaman HTML.




**Gamabr 2.2** Proses eksekusi kode PHP

Variabel dapat disebut sebagai sebuah tempat penyimpanan data bagi nilai numeris maupun non numeris, agar dapat digunakan pada bagian lain dari *script* program PHP. PHP mendukung berbagai jenis variabel yaitu:

a. *Integer* (bilangan bulat)

b. Bilangan *floating point* (presisi tunggal, ganda)

c. *Boolean*

d. *Null* (untuk variabel yang belum diset)

e. *String*

f. *Array*

g. *Object*

h. *Resource*

Dalam penulisan scipt PHP ada beberapa aturan yang harus diperhatikan, diantaranya adalah :

1. Penamaan file PHP

   File php harus diakhiri dengan ekstensi .php, selain ekstensi tersebut, web server tidak akan menganggapnya sebagai file PHP sehingga tidak akan memprosesnya.

2. Selalu gunakan open tag <? Php atau <? =

   <? Php atau <? = paling baik dan aman digunakan karena keduanya selalu dapat digunakan, dan tidak bergantung pada setting apapun.

3. Tidak perlu menggunakan *close tag* ? > pada *script full* php

   gunakan close tag ?> hanya jika script PHP yang kita buat bergabung dengan script lain seperti HTML yang jelas closing tag tersebut sebagai akhir dari script PHP.

4. Penulisan statement (perintah) pada PHP

   Statement adalah semua perintah yang akan menginstruksikan PHP untuk melalukan sesuatu, baik perintah sederhana seperti menyimpan data ke *variabel $a* = 1 atau mencetak *output* seperti *echo $a*, hingga perintah rumit seperti logika *if else* atau *for loop* Setiap statement ini harus diakhiri dengan tanda *semicolon* atau ; kecuali perintah yang menggunakan kurung kurawa {}, contoh seperti statement *class, if else, do while*, dll.

**2.2.4. XAMPP (V3.2.2)**

XAMPP merupakan paket php berbasis *open source* yang dikembangkan oleh sebuah komunitas *Open Source*. Dengan menggunakan XAMPP kita tidak perlu lagi melakukan penginstalan program yang lain karena semua kebutuhan telah disediakan oleh XAMPP. Beberapa paket yang telah disediakan adalah *Apache, MySql*, Php, *Filezila*, dan Php *myadmin* (Hendrianto 2014).

XAMPP merupakan “perangkat lunak bebas, yang mendukung banyak sistem operasi, merupakan kompilasi dari beberapa program. Fungsinya adalah

sebagai *server* yang berdiri sendiri (*locahost*), yang terdiri atas program *Apache HTTP Server, MySQL database,* dan penerjemah bahasa yang ditulis dengan bahasa pemrograman PHP dan *Perl*. Program ini tersedia dalam GNU *General Public License* dan Bebas, merupakan *web server* yang mudah digunakan yang dapat melayani tampilan *web* yang dinamis (Ujang Mulayana dan Dian Gustina 2016).

### 2.2.5. *DATABASE MYSQL* (VERSI 5.5.35)

*MySQL* adalah *multiuser database* yang menggunakan bahasa *Structured Query Language (SQL). MySQL* dalam operasi *client server* melibatkan *server daemon MySQL* disisi *server* dan berbagai macam program serta *library* yang berjalan disisi *client. MySQL* mampu menangani data yang cukup besar. Perusahaan yang mengembangkan *MySQL* yaitu TEX, mengaku mampu menyimpan data lebih dari 40 *database*, 10.000 tabel, dan sekitar 7.000.000 baris totalnya kurang lebih 100 *Gigabyte* data (Hendrianto 2014).

MySQL banyak memiliki keistimewaan. Berikut ini diantaranya :

a. *Portability MySQL* dapat berjalan stabil pada berbagai sistem operasi diantaranya seperti : *Linux FreeBSD. Mac OS X server, Solaris, Amiga, HP-UX* dan masih banyak lagi.
b. *Open Source MySQL* didistribusikan secara *open source* dibawah lisensi *GPL* sehingga dapat dipergunakan secara cuma-cuma.
c. *Multiuser MySQL* dapat digunakan oieh beberapa user dalam waktu yang bersamaan tanpa banyak masalah atau konflik. Hal ini memungkinkan sebuah *database server MySQL* dapat diakses *client* secara bersamaan.

*d. Perfomance Tuning MySQL* memiliki kecepatan yang menkjubkan dalam menangani *query* sederhana, dengan kata lain dapat memproses lebih banyak *SQL* per satuan waktu.

*e. Column Types MySQL* memiiiki banyak kolom yang sangat kompleks, seperti : *signed/unsigned integer, float, double, char, varchar, text, blob, date, time, datetime, timestamp, year, set* serta *enum*.

*f. Command dan Functions*

*MySQL* memiliki operator dan fungsi secara penuh yang mendukung perintah *SELECT* dan *WHERE* dalam *query*.

### 2.2.6. *Jquery Mobile*

*JQuery mobile* adalah *web framework* yang dioptimalkan untuk perangkat layar sentuh (*touch-optimized*). Perkembangan perangkat bergerak yang semakin pesat membuat *jQuery mobile* akan menjadi tren karena *library jQuery* telah disesuaikan dengan tampilan handphone, tablet, desktop (Sari & Sholeh 2013) .

*Jqurey mobile* adalah sebuah *framework* (kerangka kerja) yang dibangun diatas *jquery* yang menyediakan elemen-elemen *user interface*(antar muka) dan fitur-fitur untuk aplikasi *web mobile* yang dapat berjalan di berbagai perangkat *mobile* maupun *desktop*(*Cross flatform*) (Imam Mulhim 2014). Penggunaan *Jquery Mobile* memungkinkan pengembang dapat mengembangkan program dengan lebih cepat, karena aksebilitas lebih cepat dan tidak memerlukan konfigurasi *javascript*. Sehingga *library* ini dapat lebih mudah diakses dan memiliki ukuran yang lebih ringkas. Dukungan penggunaan layar sentuh membuat aplikasi yang dibangun lebih optimal pada beberapa piranti yang ada

atau dapat dikatakan pula aplikasi tersebut dapat bersifat *multi platform*. adapun cara untuk memperoleh *jquey mobile* adalah mengunjungi *website jquery mobile Project* (jquerymobile.com). Terdapat dua paket yang tersedia, pertama merupakan *file* yang tidak terkompresi dan untuk keperluan *debugging* saja. Paket lain adalah versi minimalis, didalamnya terdapat Ekstrasi *file* yang dapat dimanfaatkan secara langsung. Sifat dari *JQuery Mobile* :

1. Memudahkan pengembangan *user interface* untuk *mobile web apps*.
2. Konfigurasi antarmukanya bersifat *markup-driven*, yang berarti dapat membuat seluruh aplikasi antarmuka dasar dalam kode *HTML*, tanpa perlu menulis satu baris *JavaScript*.
3. Memastikan bahwa *user interface* bekerja pada *web browser*.
4. Menyediakan *event* khusus untuk mendeteksi *event* dari piranti *mobile* dan piranti *touchscreen seperti tap, tap-and-hold, swipe* dan perubahan orientasi.

### 2.2.7. *Code Igniter* (CI)

*CodeIgniter* adalah *framework* aplikasi *web* yang *open source* untuk bahasa pemrograman PHP. *CodeIgniter* memiliki banyak fitur yang membuatnya berbeda dengan *framework* lainnya. Tidak seperti beberapa *framework* PHP lainnya, dokumentasi untuk *framework* ini sangat lengkap, yang mencakup seluruh aspek dalam *framework*. *CodeIgniter* juga mampu berjalan pada lingkungan *Shared Hosting* karena memiliki ukuran yang sangat kecil, namun memiliki kinerja yang sangat luar biasa(Setiadi 2016).

Dari segi pemrograman, *CodeIgnite*r kompatibel dengan PHP4 dan PHP5, sehingga akan berjalan dengan baik pada *web host* yang banyak dipakai saat ini. *CodeIgniter* menggunakan pola design Model- *View-Controllwe* (MVC), yang merupakan cara untuk mengatur aplikasi *web* ke dalam 3 bagian yang berbeda, yaitu:

1. Model- lapisan abstraksi *database*

2. *Views*- file-file tampilan template depan

3. *Controller*- logika bisnis dari aplikasi

Pada intinya *CodeIgniter* juga membuat penggunaan ekstensif dari pola *design Singleton*. Maksudnya adalah cara untuk me-*load class* sehingga jika *class* itu di panggil dalam beberapa kali, kejadian yang sama pada *class* tersebut akan digunakan kembali. Hal ini sangat berguna dalam koneksi *database,* karena kita hanya ingin menggunakan satu koneksi setiap kali *clas*s itu digunakan. *CodeIgniter* dikembangkan oleh Rick Ellis, dengan versi awal yang dirilis pada tanggal 28 februari 2006.

**2.2.8. HTML (VERSI 5)**

HTML (*Hypertext Markup Language*) adalah sebuah bahasa pemrograman berbentuk skrip-skrip yang berguna untuk membuat sebuah halaman web. HTML dapat dibaca oleh berbagai platform seperti : Windows, Linux, Macintosh. Kata ”*Markup Language*” pada HTML menunjukkan fasilitas yang berupa tanda tertentu dalam skrip HTML dimana kita bisa mengatur judul, garis, tabel, gambar, dan lain lain dengan perintah yang telah ditentukan pada elemen HTML. HTML

sendiri dikeluarkan oleh W3C (*Word Wide Web Consortin*), setiap terjadi perkembangan level HTML harus dievakuasi ketat dan disetujui oleh W3C (Uswatun Hasanah dan Sukadi 2013).

### 2.2.9. CSS (VERSI 3)

CSS (*Cascading Style-Sheet*), yaitu sebuah pengembangan atas kode HTML yang sudah ada sebelumnya. Dengan CSS, bisa menentukan sebuah struktur dasar halaman *web* secara lebih mudah dan cepat, serta irit size (Uswatun Hasanah dan Sukadi 2013).