

BAB 2

TINJAUAN PUSTAKA

2.1 PHP

PHP adalah singkatan dari *PHP Hypertext Preprocessor*, yang merupakan sebuah bahasa scripting yang terpasang pada HTML. Sebagian besar sintaks mirip dengan bahasa C, Java dan Perl, ditambah beberapa fungsi PHP yang spesifik. Tujuan utama pengguna bahasa ini adalah untuk memungkinkan perancangan web menulis halaman web dinamika dengan cepat. Program PHP harus diterjemahkan oleh web-server sehingga menghasilkan kode html yang dikirim ke browser agar dapat ditampilkan. [3]

2.2 JavaScript

JavaScript adalah bahasa yang digunakan untuk membuat program yang digunakan agar dokumen HTML yang ditampilkan dalam browser menjadi lebih interaktif, tidak sekedar indah saja. JavaScript memberikan beberapa fungsionalitas ke dalam halaman *web*, sehingga dapat menjadi sebuah program yang disajikan dengan menggunakan antarmuka *web*. [4]

2.3 HTML

HTML adalah singkatan dari *HyperText Markup Language*. HTML merupakan *file* teks yang ditulis menggunakan aturan-aturan kode tertentu untuk kemudian disajikan ke *user* melalui suatu aplikasi *web browser*. Setiap informasi yang tampil di *web* selalu dibuat menggunakan kode HTML. Oleh karena itu, dokumen HTML sering disebut juga sebagai *web page* (halaman web). [5]

2.4 Framework

Framework merupakan koleksi atau kumpulan potongan-potongan yang disusun atau diorganisasikan sedemikian rupa, sehingga dapat digunakan untuk membantu membuat aplikasi utuh harus membuat semua kodenya dari awal. Macam-macam *framework* PHP diantaranya: *Zend Framework*, *Cake PHP*, *Trax*, *Symfony*, *CodeIgniter* dan lain-lain. [6]

2.5 CodeIgniter

CodeIgniter (CI) adalah *framework* pengembangan aplikasi (*Application Development Framework*) dengan menggunakan PHP, suatu kerangka untuk bekerja atau membuat program dengan menggunakan PHP yang sistematis. *Framework* CodeIgniter merupakan *framework* yang memiliki dokumentasi yang jelas dan lengkap, yang memudahkan pengembang untuk mempelajari dengan mudah.

Fitur-fitur CodeIgniter :

- a. Sistem berbasis *Model-View-Controller*,
- b. Benar-benar *framework* yang ringan,
- c. Memiliki fitur *class database* yang mendukung beberapa platform,
- d. Dukungan *database* dengan *Active Record*,
- e. *Form* dan validasi data,
- f. *Class* untuk *upload file*,
- g. *Class* untuk pengiriman email, dan lain-lain.

Dalam CodeIgniter terdapat teknik pemrograman yaitu dengan teknik pemrograman MVC (*Model View Controller*) yang merupakan teknik pemrograman yang populer saat ini, yang mengharapkan pemrograman secara disiplin untuk membagi program menjadi 3 bagian yaitu :

- a. Model

Model adalah komponen dari program yang menggunakan teknik MVC, yang digunakan untuk melakukan penyediaan atau pemrosesan data.

- b. View

View secara prinsip merupakan dari program yang menggunakan teknik MVC, yang digunakan untuk menampilkan hasil dari proses/ View digunakan untuk menampilkan data hasil pemrosesan kedalam format dokumen HTML.

- c. *Controller*

Controller merupakan komponen utama dari suatu program yang merupakan teknik pemrograman MVC. *Controller* secara sederhana adalah sebuah file *class* yang memiliki nama dapat diasosiasikan dengan sebuah URL. [7]

2.6 CSS

CSS (*Cascading Style Sheet*) adalah suatu teknologi yang digunakan untuk memperindah tampilan halaman website (situs). CSS mempunyai 2 bagian utama yaitu *selectors* dan deklarasi. Yang dimaksud *selectors* biasanya elemen HTML yang ingin diubah, sedangkan deklarasi biasanya terdiri dari properti dan nilai. Properti adalah atribut *style* yang ingin diubah dan setiap properti memiliki nilai. [8]

2.7 Oracle

Oracle merupakan *software* database yang banyak dipakai di perusahaan-perusahaan besar di seluruh dunia saat ini. *Software* ini juga banyak diminati oleh para konsultan pembuat aplikasi yang berkaitan dengan database. Sistem keamanannya yang handal membuat para profesional yang berkecimpung dalam dunia database lebih memilih oracle sebagai perangkat untuk menunjang kegiatan bisnis mereka.

Disamping sistem *security* yang handal, oracle merupakan *software* database yang bisa menampung serta mengelola data dengan kapasitas yang sangat besar serta dapat mengaksesnya dengan sangat cepat pula. Sintak SQL nya yang hampir seluruhnya telah memenuhi standar ANSI-92 lebih memudahkan para *programmer* database dalam membangun aplikasi baik dari sisi '*back end*' maupun dari sisi '*front end*'. Demikian pula bagi seorang administrator yang berkecimpung dalam menangani keamanan database akan merasa diuntungkan serta dimudahkan dengan *software* oracle yang telah '*establish*'. [9]

2.8 MySQL

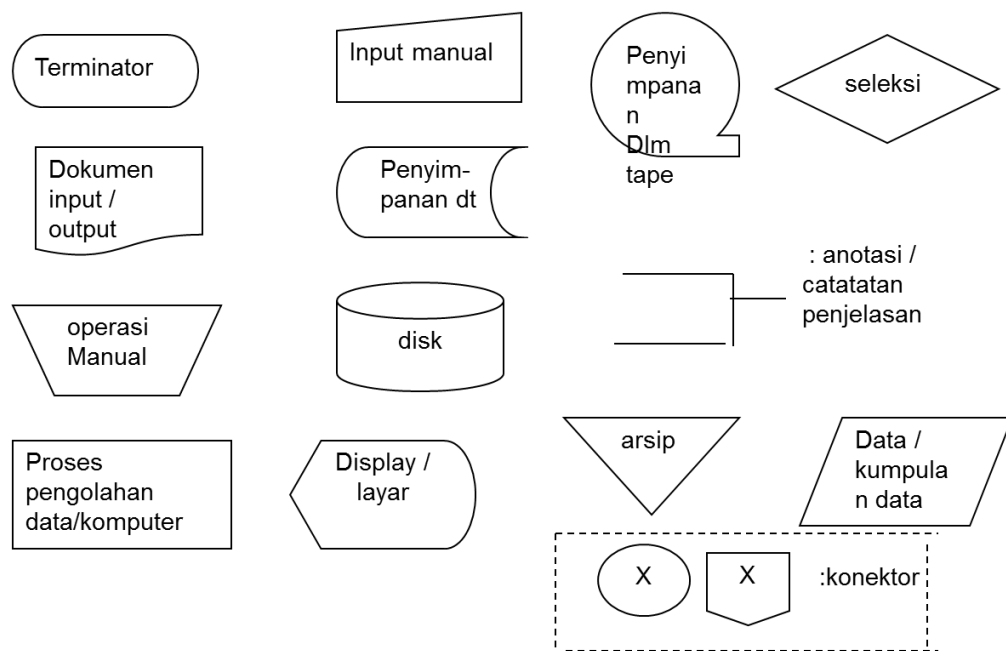
MySQL adalah suatu sistem manajemen basis data relasional (RDBMS-Relational Database Management System) yang merupakan salah satu jenis *database server* terkenal dan banyak digunakan untuk membangun aplikasi *web* yang menggunakan *database* sebagai sumber dan pengelolaan datanya. Kepopuleran MySQL antara lain karena MySQL menggunakan SQL sebagai bahasa dasar untuk mengakses *database*-nya sehingga mudah untuk digunakan, kinerja *query* cepat dan mencukupi untuk kebutuhan *database* perusahaan-perusahaan skala menengah-kecil.

MySQL juga bersifat *open source* dan *free* pada berbagai *platform*. MySQL dan PHP dianggap sebagai pasangan *software* pengembangan aplikasi *web* yang ideal. MySQL lebih sering digunakan untuk membangun aplikasi berbasis *web*, umumnya pengembangan aplikasinya menggunakan bahasa pemrograman PHP. [10]

2.9 Flowmap

Flowmap merupakan diagram yang menggambarkan aliran dokumen pada suatu prosedur kerja di organisasi dan memperlihatkan diagram alir yang menunjukkan arus dari dokumen, aliran data fisis, entitas-entitas sistem informasi dan kegiatan operasi yang berhubungan dengan sistem informasi. Penggambaran biasanya diawali dengan mengamati dokumen apa yang menjadi media data atau informasi. Selanjutnya ditelusuri bagaimana dokumen tersebut terbentuk, ke bagian atau entitas mana dokumen tersebut mengalir, perubahan apa yang terjadi pada dokumen tersebut, proses apa yang terjadi terhadap dokumen tersebut, dan seterusnya. [11]

Berikut simbol-simbol dari *flowmap*:



Gambar 2-1
Simbol-simbol *Flowmap*. [11]

2.10 UML

UML singkatan dari *Unified Modeling Language* yang berarti bahasa pemodelan standart. Ketika membuat model menggunakan konsep *UML* ada aturan-aturan yang harus diikuti. Bagaimana elemen pada model-model yang dibuat berhubungan satu dengan lainnya harus mengikuti standar yang ada. *UML* bukan hanya sekedar diagram, tetapi juga menceritakan konteksnya. [12]

UML diaplikasikan untuk maksud tertentu, antara lain untuk :

- a. Merancang perangkat lunak,
- b. Sarana komunikasi antara perangkat lunak dengan proses bisnis,
- c. Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem,
- d. Mendokumentasikan sistem yang ada, proses-proses dan organisasinya.

Ada beberapa macam jenis diagram *UML* antara lain :




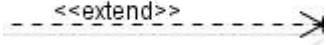
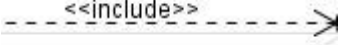
- a. *Use case diagram*
- b. *Class diagram*
- c. *Activity diagram*
- d. *Sequence diagram*
- e. *Conceptual diagram*
- f. *Colaboration diagram*
- g. *Component diagram*
- h. *Package diagram*


2.11 Use Case Diagram

Use Case menggambarkan *external view* dari sistem yang akan kita buat modelnya. Diagram *use case* bersama dengan narasi *use case* dan skenario mendefinisikan tujuan suatu sistem atau pengklasifi lain seperti interprise, subsistem atau komponen. [12]

Berikut adalah simbol-simbol *use case* diantaranya adalah :

Tabel 2-1
Simbol-simbol *use case*. [12]

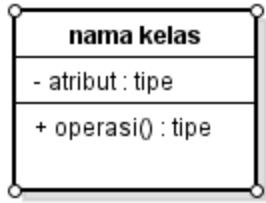


Simbol	Fungsi
	Menggambarkan pihak-pihak yang berperan dalam sistem
	Aktifitas yang disiapkan oleh sistem yang menggambarkan fungsi tertentu dalam suatu sistem berupa komponen, kejadian atau kelas
	Mengindikasikan aktor mana yang berinteraksi dengan <i>use case</i> dalam suatu sistem
	Merupakan relasi <i>use case</i> tambahan ke sebuah <i>use case</i> dimana <i>use case</i> yang ditambahkan dapat berdiri sendiri walau tanpa <i>use case</i> tambahan itu.
	Merupakan relasi <i>use case</i> tambahan ke sebuah <i>use case</i> dimana <i>use case</i> yang ditambahkan memerlukan <i>use case</i> ini untuk menjalankan fungsinya.



Simbol	Fungsi
	Menggambarkan hubungan turunan antara <i>use case</i> atau antar aktor

2.12 Class Diagram

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas. [1]

Tabel 2-2
Simbol-simbol *class diagram*. [1]


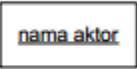



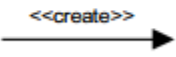
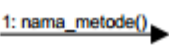
Simbol	Deskripsi
<p>Kelas</p> 	Kelas pada struktur sistem
<p>Asosiasi</p> 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan <i>multiplicity</i> .
<p>Generalisasi</p> 	Relasi anat kelas dengan makna generalisasi-spesialisasi (umum khusus).

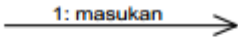
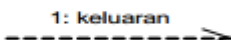

Agregasi 	Relasi antar kelas dengan makna semua-bagian (whole-part)
Asosiasi berarah 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan <i>multiplicity</i> .

2.13 Sequence Diagram

Diagram *sequence* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima anatar objek. Oleh karena itu untuk menggambarkan diagram *sequence* maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Banyaknya diagram *sequence* yang harus digambarkan adalah sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram *sequence* sehingga semakin banyak *use case* yang didefinisikan maka diagram *sequence* yang harus dibuat juga semakin banyak. [1]

Tabel 2-3
Simbol-simbol *sequence diagram*. [1]

Simbol	Deskripsi
<p>Aktor</p>  <p>nama <i>actor</i> atau</p>  <p>tanpa waktu aktif</p>	<p>Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.</p>
<p>Garis hidup</p> 	<p>Menyatakan kehidupan suatu objek.</p>
<p>Objek</p> 	<p>Menyatakan objek yang berinteraksi pesan.</p>
<p>Waktu aktif</p> 	<p>Menyatakan objek dalam keadaan aktif dan berinteraksi pesan.</p>
<p>Pesan tipe <i>create</i></p> 	<p>Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat.</p>
<p>Pesan tipe <i>call</i></p> 	<p>Menyatakan suatu objek memanggil operasi atau metode yang ada pada objek lain atau dirinya sendiri. Arah panah mengarah pada objek yang memiliki</p>

Simbol	Deskripsi
	operasi atau metode, karena memanggil operasi atau metode maka operasi atau metode yang dipanggil harus ada pada diagram kelas sesuai keals objek yang berinteraksi.
Pesan tipe send 	Menyatakan bahwa suatu objek mengirimkan data atau masukan atau informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim.
Pesan tipe return 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.
Pesan tipe destroy 	Menyatakan suatu objek mengakhiri hidup objek yang lain. Arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada create maka ada destroy.

2.14 Entity Relationship Diagram

ERD kepanjangan dari *Entity Relationship Diagram* adalah salah satu pemodelan basis data konseptual yang menggambarkan basis data ke dalam bentuk entitas-entitas dan relasi yang terjadi di antara entitas-entitas yang ada. Entitas diartikan sebagai objek di dunia nyata yang bisa dibedakan dengan objek yang lain. Relasi diartikan sebagai hubungan yang terjadi diantara satu entitas dengan entitas lainnya. [13]

Elemen-elemen ERD yaitu :

a. Entitas

Sesuatu apa saja yang ada dalam sistem, nyata maupun abstrak di mana data tersimpan atau di mana terdapat data.


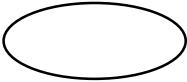
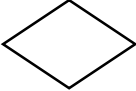

b. Relasi

Hubungan antara entitas satu dengan entitas yang lainnya sesuai dengan proses bisnisnya.

c. Atribut

Deskripsi kelompok data yang mempunyai karakteristik yang sama (data yang mendeskripsikan entitas dan relasi), merupakan *field* yang akan disimpan.

Tabel 2-4
Simbol-simbol ERD. [13]

Simbol	Keterangan
	Entitas
	Atribut
	Relasi
	Garis

2.15 SMS Gateway

Salah satu metode komunikasi yang handal saat ini adalah pesan pendek. Implikasinya, salah satu model komunikasi data yang bisa dipakai adalah SMS. Artinya, SMS tersebut harus bisa melakukan transaksi dengan *database*. Untuk itu perlu dibangun sebuah sistem yang disebut dengan *SMS gateway*. Pada prinsipnya, *SMS Gateway* adalah sebuah perangkat lunak yang menggunakan bantuan komputer dan memanfaatkan teknologi seluler yang diintergrasikan untuk mendistribusikan pesan-pesan yang di-*generate* lewat sistem informasi melalui media SMS yang di-*handel* oleh jaringan seluler. [14]

2.16 Gammu

Gammu merupakan salah satu *tools* untuk mengembangkan aplikasi SMS Gateway yang cukup mudah diimplementasikan dan gratis. Gammu bisa dikatakan sebagai “Sang aktor utama”, karena komponen inilah yang menjembatani pentransferan data-data SMS dari *handphone* atau *mobile* modem ke komputer atau sebaliknya. Kelebihan Gammu dari *tool* SMS Gateway lain adalah:

- a. Gammu dapat dijalankan di Windows maupun Linux.
- b. Banyak *device* atau ponsel yang compatible dengan Gammu.
- c. Gammu dapat membantu menggunakan fitur-fitur yang ada pada ponsel dengan kelebihan efisien.
- d. Baik kabel data USB maupun SERIAL, semuanya kompatibel di Gammu. [15]