**BAB II**

**LANDASAN TEORI**

* 1. **Tinjauan Pustaka**

**2.1.1 Kajian Penelitian Menurut Gunawan (2016)**

Dalam penelitian yang dilakukan oleh Gunawan (2016) dengan judul “Sistem Informasi Sewa Rumah Kost dan Rumah Kontrakan Berbasis Web di Surakarta”, bahwa diperlukan adanya website yang mencakup informasi mengenai rumah kos dan rumah kontrakan yang ada di Surakarta sehingga terdapat sistem yang terintegrasi dengan baik. Dalam hal ini digunakan PHP (*Personal Home Page*) sebagai bahasa pemrograman berbasis web yang bersifat *open source*. Serta menggunakan MySQL (*My Structured Query Language*) sebagai database yang bersifat open source DBMS (*Database Management System*). Sistem informasi yang dikembangkan oleh Gunawan terdiri dari pengolahan data pemilik, pengolahan data rumah kost dan rumah kontrakan, alamat rumah, keterangan ukuran bangunan dan kamar, dan fasilitas yang ditawarkan. Tetapi sistem informasi tersebut hanya didukung dengan gambar berupa foto rumah saja dan tidak didukung dengan peta navigasi.

**2.1.2 Kajian Penelitian Menurut Abidin (2011)**

Berdasarkan penelitian yang dilakukan oleh Abidin (2011) dengan judul “Aplikasi Pencarian Lokasi Kos di Sekitar Institut Teknologi Surabaya Berbasis Web”, dibutuhkan sistem informasi berbasis web mengenai rumah kost di wilayah sekitar Institut Teknologi Surabaya. Sistem informasi tersebut diperlukan karena adanya kebutuhan yang tinggi oleh mahasiswa terutama mahasiswa baru yang membutuhkan informasi mengenai kost di sekitar kampus sehingga membantu untuk sebagian orang yang belum mengerti mengenai daerah tersebut.

Selain itu, sistem informasi tersebut melayani pemesanan secara online untuk membantu mempermudah pengguna. Metode penelitian yang digunakan untuk menghasilkan sistem informasi terbagi menjadi 3, yaitu : metode observasi, metode studi pustaka, dan metode pengembangan perangkat lunak. Metode studi pustaka digunakan untuk referensi dalam perancangan perangkat lunak, dengan cara pembuatan UML (*Unified Modeling Language*) menggunakan pemrograman berorientasi obyek. Sistem informasi ini dibuat mennggunakan bahasa pemrograman OOP (*Object Oriented Programming*) pada PHP yang dapat membantu visitor dalam pencarian dan pemesanan kos secara online.

# **Sistem Informasi**

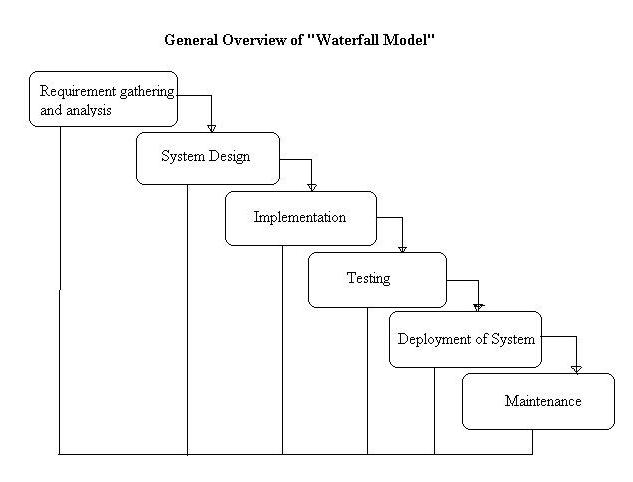
Sistem adalah sekumpulan komponen yang saling berkerja sama untuk mencapai tujuan guna memperbaiki organisasi ke arah yang lebih baik (McLeod, 2002). Sedangkan informasi merupakan salah satu jenis sumber daya yang tersedia bagi manajer, yang dapat dikelolah seperti halnya sumber daya yang lain. Informasi dari komputer dapat digunakan oleh para manajer, non manajer, serta orang-orang dan organisasi-organisasi dalam lingkungan perusahaan (McLoed, 2002). Sistem informasi merupakan suatu sis.tem didalam suatu organisasi yang mempertemukan kebutuhan-kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan (Imbar dan Haryani, 2008).

# **SDLC (*System Development Life Cycle*)**

SDLC (*Systems Development Life Cycle*) dalam rekayasa sistem dan rekayasa perangkat lunak adalah proses pembuatan dan pengubahan sistem serta model dan metodologi yang digunakan untuk mengembangkan sistem-sistem tersebut. Konsep SDLC mendasari berbagai jenis metodologi pengembangan perangkat lunak. Metodologi-metodologi ini membentuk suatu kerangka kerja untuk perencanaan dan pengendalian pembuatan sistem informasi, yaitu proses pengembangan sistem.

* + 1. **Waterfall Model**

Salah satu metodologi pengembangan perangkat lunak adalah *waterfall model. Waterfall model* merupakan model pengembangan perangkat lunak yang melakukan pendekatan secara sistematis dan urut mulai dari level kebutuhan sistem menuju ke tahap analisis, desain, *coding*, *testing*, dan *maintenance*. Disebut dengan waterfall karena tahap demi tahap yang dilalui harus menunggu selesainya tahap sebelumnya dan berjalan berurutan. Sebagai contoh tahap *coding* harus menunggu tahap desain selesai. Secara umum tahapan model *waterfall* dapat dilihat pada gambar berikut:



**Gambar 2.1** Waterfall Model - SDLC

**2.4 Google Maps**

*Google maps* adalah jasa peta gratis dan online disediakan oleh google yang dapat ditemukan di <http://maps.google.com>. Pada situs tersebut kita dapat melihat informasi geografis pada hampir semua wilayah di muka bumi. Layanan ini interaktif, karena di dalamnya peta dapat digeser sesuai keinginan pengguna, mengubah tingkat *zoom,* serta mengubah tampilan peta. *Google maps* juga menawarkan peta yang dapat diseret dan gambar setelit untuk seluruh dunia, serta menawarkan rute perjalanan.

**2.4.1 Google Maps API (Application Programing Interface)**

*Google Maps Application Programming Interface* (API) merupakan suatu fitur aplikasi yang dikeluarkan oleh google untuk memfasilitasi pengguna yang ingin mengintegrasikan *Google Maps* ke dalam *website* masing-masing dengan menampilkan *data point* milik sendiri. Dengan menggunakan *Google Maps* API, *Google Maps* dapat di-*embed* pada *website* eksternal. Agar aplikasi *Google Maps* dapat muncul di *website* tertentu, diperlukan adanya API *key*. API *key* merupakan kode unik yang digenerasikan oleh google untuk suatu *website* tertentu, agar *server Google Maps* dapat mengenali.

* 1. **PHP *(Hypertext Preprocessor)***

Menurut Welling (2008), PHP adalah kepanjangan dari “*Hypertext Preprocessor*”. PHP adalah bahasa scripting web HTML-embedded yang dapat disisipkan ke dalam HTML halaman Web. Tujuan dari bahasa pemrograman PHP adalah untuk memungkinkan pengembang web untuk menulis halaman yang dihasilkan secara dinamis dengan cepat dan mudah. PHP juga bagus untuk menciptakan situs *Web* *database-driven*.

Kelebihan PHP dari bahasa pemrograman web lain yaitu PHP adalah sebuah bahasa script yang tidak melakukan sebuah kompilasi dalam penggunaannya. *Web Server* yang mendukung PHP dapat ditemukan dimana – mana dari mulai apache, IIS, Lighttpd, hingga Xitami dengan konfigurasi yang relatif mudah, dan dalam sisi pengembangan lebih mudah.

**2.6 CodeIgniter**

Menurut Welling (2008), CodeIgniter adalah sebuah *web application framework* yang bersifat open source digunakan untuk membangun aplikasi php dinamis. CI atau CodeIgniter diperkenalkan kepada publik pada tanggal 28 Feburari 2006. Tujuan utama pengembangan codeigniter adalah untuk membantu developer mengerjakan aplikasi yang lebih cepat dibandingkan menulis semua code dari awal. Codeigniter menyediakan berbagai macam library yang dapat mempermudah dalam pengembangan.

Menurut pembuat/penemu PHP **Rasmus Lerdorf**codeigniter lebih ringan dan cepat dibandingkan framework lainnya. Codeigniter sendiri dibangun menggunakan konsep MVC (Model-View-Controller) design pattern, merupakan salah satu framework tercepat dibandingkan framework lainnya.

**2.6.1 Kelebihan CodeIgniter**

CodeIgniter sangat ringan, terstruktur, mudah dipelajari, dokumentasi lengkap dan dukungan yang banyak dari forum CodeIngiter. CodeIgniter/CI memiliki fitur-fitur yang sangat bermanfaat diantaranya:

* **Menggunakan Pattern MVC**.

Dengan menggunakan pattern MVC, struktur kode yang dihasilkan menjadi lebih terstruktur dan memiliki standar yang jelas.

* **URL Freindly**.

URL yang dihasilkan sangat url freindly. Pada CI diminimalisasi penggunaan $\_GET dan digantikan dengan URL.

Dibandingkan framework lainnya, CodeIgniter memiliki beberapa keunggulan, seperti:

* **Kemudahan**

Kemudahan dalam mempelajari, membuat library dan helper, memodifikasi serta meng-integrasikan library dan helper.

* **Kecepatan**

Berdasarkan hasil benchmark CodeIgniter merupakan salah satu framework PHP tercepat yang ada saat ini.

* **Mudah dimodifikasi dan beradaptasi**

Sangat mudah memodifikasi behavior framework ini. Tidak membutuhkan server requirement yang macam-macam serta mudah mengadopsi library lainnya.

* **Dokumentasi lengkap dan jelas**

Codeigniter menyediakan sebuah panduan yang lengkap. Semua informasi yang dibutuhkan tentang CI ada di sana.

* **Learning Curve Rendah**

Codeigniter sangat mudah dipelajari. Dalam pemilihan framework hal ini sangat penting diperhatikan karena kita juga harus memperhatikan skill dari seluruh anggota team. Jika sebuah framework sangat sulit dipelajari maka akan beresiko memperlambat team development.

**2.7 Konsep Basis Data**

Basis data (*database*) adalah kumpulan data (informasi) yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan software untuk memperoleh informasi dari basis data tersebut. Software untuk mengelola basis data disebut Sistem Manajemen Basis Data (Akhmad Dahlan, 2009).

**2.7.1 DDL (Data definition Language)**

DDL (*Data Definition Language*) merupakan perintah SQL yang digunakan untuk mendefinisikan atau mendeklarasikan objek database, menciptakan objek database atau bahkan menghapus objek database. Objek database dapat berupa tabel atau database itu sendiri. DDL juga dapat digunakan untuk membuat koneksi antartabel database beserta batasannya dengan menentukan indek sebagai kuncinya (Ema Utami, 2005: 44).

**2.7.2 DML (Data Manipulation Language)**

DML (*Data Manipulation Language*) merupakan query yang digunakan untuk memanipulasi data, seperti untuk menampilkan data, mengubah data, menghapus data, atau mengisi data (Ema Utami, 2005: 48-58).

**2.7.3 Teknik Perancangan Database**

* **Database**

Database adalah kumpulan dari data yang terhubung secara logika, dan merupakan deskripsi dari data tersebut, yang dirancang untuk memenuhi kebutuhan informasi dari suatu organisasi (Connolly & Begg, 2010). Basis data adalah suatu koleksi data komputer terintegrasi, diorganisasikan dan disimpan dengan suatu cara yang memudahkan pengambilan kembali. Tujuan utama dari konsep basis data adalah meminimumkan pengulangan data. Pengulangan data (*data redudancy*) adalah duplikasi data yang berarti data yang sama disimpan dalam beberapa file (McLeod dan Raymond, 2007:258).

Relational database adalah sebuah kumpulan dari relasi yang telah dinormalisasi dengan nama relasi yang jelas (Connolly & Begg, 2010:146). Relational database merupakan suatu tipe database yang berdasarkan model relational, dimana semua data dilhat oleh user, disusun dalam bentuk tabel-tabel dan semua operasi pada database bekerja pada tabel-tabel tersebut. Relasi antar- tabel pada relational database sudah melalui tahap normalisasi dengan nama relasi yang berbeda-beda. Menurut Connolly & Begg (2010:385) ada 3 jenis relasi dalam tabel yaitu:

1. Relasi one-to-one adalah relasi antara satu record dengan satu record dalam tabel lain yang saling berhubungan. Contohnya: relasi perwalian antara tabel Dosen dan tabel siswa. Satu baris dosen atau satu dosen bisa berhubungan dengan satu baris atau lebih mahasiswa
2. Relasi one-to-many adalah relasi antara satu record dengan lebih dari satu record dalam tabel lain sehingga saling berhubungan. Contohnya: relasi perwalian antara tabel dosen dan tabel siswa. Satu baris dosen atau satu dosen bisa berhubungan dengan satu baris atau lebih mahasiswa.
3. Relasi many-to-many adalah relasi antara banyak record dengan lebih dari satu record dalam tabel lain yang saling berhubungan. Artinya ada banyak baris di tabel satu dan tabel dua yang saling berhubungan satu sama lain. Contohnya: relasi antar tabel mahasiswa dan tabel mata kuliah. Satu baris siswa bisa berhubungan dengan banyak baris mata pelajaran begitu juga sebaliknya.

**2.7.4 Use Case Diagram**

Menurut Booch (2005), suatu *use case diagram* menampilkan sekumpulan *use case* dan aktor (pelaku) dan hubungan diantara *use case* dan aktor tersebut. *Use case diagram* digunakan untuk penggambaran *use case* statik dari suatu sistem. *Use case diagram* penting dalam mengatur dan memodelkan kelakuan dari suatu sistem.

*Use case* menjelaskan apa yang dilakukan sistem (atau subsistem) tetapi tidak tidak menspesifikasikan cara kerjanya. *Flow of event* digunakan untuk menspesifikasikan kelakuan dari *use case*. *Flow of event* menjelaskan *use case* dalam bentuk tulisan dengan sejelas-jelasnya, diantaranya bagaimana, kapan *use case* dimulai dan berakhir, ketika *use case* berinteraksi dengan aktor, obyek apa yang digunakan, alur dasar dan alur alternatif.

  
**Gambar 2.3** Contoh Model *Use Case Diagram*

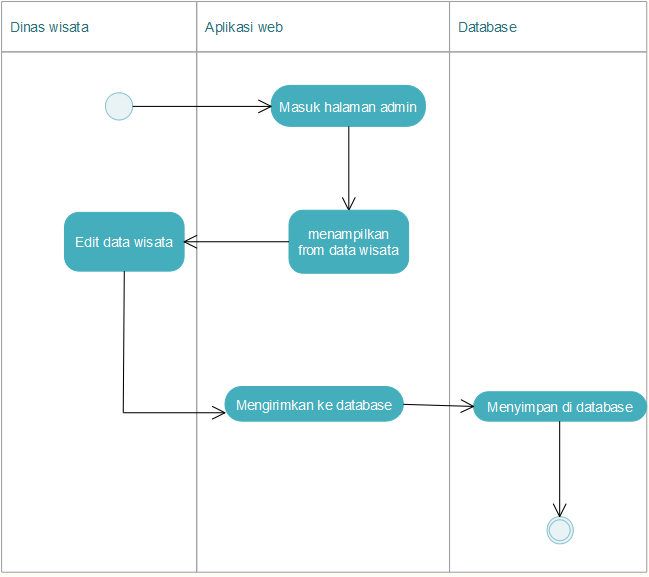
Simbol-simbol *use case* dijelaskan dalam gambar berikut:

|  |  |  |  |
| --- | --- | --- | --- |
| No | Simbol | Nama simbol | keterangan |
| 1 | Hasil gambar untuk aktor use case | Aktor | Siapa saja yang berinteraksi dengan *use case* dalam sistem |
| 2 |  | Use case | Aksi yang dilakukan oleh *aktor* dalam sistem |
| 3 |  | relationship | Hubungan antara *actor* dengan *use case* dan di mana  Mereka berada |

**Gambar 2.4** Simbol-simbol *Use Case*

**2.7.5 Activity Diagram**

Activity diagram adalah sebuah grafik dari titik dan alurnya yang menggambarkan aliran dari aktivitas ke aktivitas dalam suatu sistem. Tujuan pembuatan diagram ini adalah untuk memodelkan aliran informasi secara prosedural dari aktivitas yang beragam secara lebih efektif. Sebuah activity digambarkan dengan menggunakan activity diagram.



**Gambar** 2.5 Contoh Activity Diagram

**2.7.6. Sequence Diagram**

Menurut Sukamto dan Shalahuddin (2013:165), sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan message yang dikirim dan diterima antar objek. Banyaknya diagram *Sequence* yang harus digambar adalah minimal 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.

Berikut ini simbol simbol Sequence diagram menurut Sukamto dan Shalahuddin (2013:165)

|  |  |  |
| --- | --- | --- |
| NO | SIMBOL | KETERANGAN |
| 1 | Atau  aktor | 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  gambar orang, tapi aktor belum tentu  merupakan orang; biasanya dinyatakan  menggunakan kata benda di awal frase  nama actor |
| 2 | http://3.bp.blogspot.com/-BYY-nKRIipM/VR5SagDPd3I/AAAAAAAAAVM/kVkSUJZb6aQ/s1600/Snap%2B2015-04-03%2Bat%2B15.39.48.png | Menyatakan kehidupan suatu objek |
| 3 | objek | Menyatakan objek yang berinteraksi  pesan |
| 4 |  | Menyatakan objek dalam keadaan aktif  dan berinteraksi, semua yang terhubung  dengan waktu aktif ini adalah sebuah  tahapan yang dilakukan di dalamnya. |
| 5 | Create | Menyatakan suatu objek membuat objek  yang lain, arah panah mengarah pada  objek yang dibuat |
| 6 |  | Menyatakan suatu objek memanggil  operasi/metode yang ada pada objek lain  atau dirinya sendiri |
| 7 |  | Menyatakan bahwa suatu objek  mengirimkan data/masukan/ informasi  ke objek lainnya, arah panah mengarah  pada objek yang dikirimi |
| 8 |  | 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 |
| 9 |  | Menyatakan suatu objek mengakhiri  hidup objek yang lain, arah panah  mengarah pada objek yang diakhiri,  sebaiknya jika ada *create* maka ada  *destroy* |

**Tabel 2.7.6** Simbol-simbol diagram sequence