BAB II
TINJAUAN PUSTAKA

1.1 Sistem Informasi

Sistem adalah rangkaian dari dua atau lebih komponen-komponen yang saling berhubungan, yang berinteraksi untuk mencapai suatu tujuan.Sebagian besar sistem terdiridari subsistem yang lebih kecil yang mendukung sistem yang lebih besar[1].

Informasi (*information*) adalah data yang telah dikelola dan diproses untuk memberikan arti dan memperbaiki proses pengambilan keputusan. Sebagaimana perannya, pengguna membuat keputusan yang lebih baik sebagai kuantitas dan kualitas dari peningkatan informasi. Sistem informasi adalah sebuah rangkaian prosedur formal dimana data dikelompokkan, diproses menjadi informasi, dan didistribusikan kepada pemakai[2].

1.1.1 Sistem Informasi Terpadu

Sistem Informasi Terpadu adalah konfigurasi yang membantu mengkoordinasikan Memekanisasi operasi dalam suatu organisasi. Alat ini dapat disebut sebagai salah satu langkah maju dari sistem informasi manajemen. Membantu dalam menyediakan informasi yang diperlukan yang dapat memutuskan atas dalam sebuah organisasi. Hal ini dikatakan penting karena satu integrasi pemahaman melibatkan banyak hal-hal teknis[3].

1.1.2 Kelebihan Sistem Informasi Terpadu

Beberapa kelebihan pada sistem informasi terpadu yaitu integrasi yang baik antar bagian pada sistem atau organisasi yang memungkinkan untuk memperoleh hasil keismpulan yang lebih baik. Serta adanya proses bisnis yang digambarkan menggunakan diagram atau bagan.

1.2 Haji dan Umroh

Haji adalah berkunjung ketanah suci (ka’bah) untuk melaksanakan amal ibadah tertentu sesuai dengan syarat, rukun, dan ketentuan yang telah ditetapkan oleh syara’. Haji diwajibkan bagi orang-orang islam yang sudah mampu atau

mempunyai biaya untuk melaksanakannya. Haji dilaksanakanibadah pada bulan zulhijjah.

Umrah adalah berkunjung ke tanah suci atau Baitullah dengan tujuan mendekatkan diri kepada Allah SWT dengan memenuhi syarat tertentu yang telah ditetapkan oleh syara', dan waktunya boleh kapan saja tidak ditentukan seperti halnya haji[3].

1.2.1 Rukun Haji

Rukun dan wajib adalah dua istilah yang digunakan oleh semua ulama fiqh hanya dalam ibadah haji. Keduanya sama-sama mesti dikerjakan. Namun ada perbedaan diantara keduanya, meskipun dalam banyak hal keduanya adalah sama. Rukun dalam haji adalah sesuatu yang sama sekali tidak boleh tertinggal dalam arti bila salah satu rukun yang ditentukan tertinggal, hajinya batal dan oleh karenanya harus diulang kembali tahun berikutnya. Wajib adalah perbuatan yang mesti dilakukan, namum bila satu diantaranya tertinggal tidak membawa kepada batalnya haji itu, hanya diwajibkan melakukan perbuatan lain sebagai penggantinya. Yang menjadi dasar hukum itu adalah dalil yang kuat dari al-Qur'an atau hadits Mutawatir, sedangkan dasar hukum dari yang wajib itu hanyalah dalil yang tidak kuat seperti hadits ahad.

1.2.2 Sayarat Haji

Syarat-syarat yang diwajibkan dalam melaksanakan haji adalah memenuhi kriteria sebagai berikut: Beragama Islam, Bahgh, berakal sehat, merdeka, mampu atau mempunyai kesanggupan. Adapun cara melaksanakan ihramhaji adatiga yaitu:

1. Haji Tamattu' Mengerjakan ibadah umrah dibulan bulan haji, kemudian setelah berumrah ialah mengerjakanhaji pada tahun itu pula. Cara ini mula-mula ihram untuk umrah dari miqat negerinya, setelah selesai dikerjakan semua urusan umrah, kemudian ia ihram lagi dari mekah untuk melaksanakanHaji.
2. Haji Tfrad Melaksanakan ibadah haji saja pada bulan haji. Caranya ialah ihram untuk haji saja dulu dari miqat yang telah ditentukan, setelah selesai semua urusan haji ia kemudian ihram lagi untuk.

3. Haji Qiran Mengerjakan haji dan umrah secara serentak. Caranya ialah seseorang melakukan ihram untuk keduanya pada waktu ihram haji dan mengerjakan semua amalan haji. Dengan sendirinya urusan umrah sudah termasuk di dalamnya yakni melakukantawafsatu kah dan sa'i satu kah pula.

1.3 *Mobile*

Aplikasi *mobile* yaitu program siap pakai yang direkap untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain dan dapat digunakan oleh sasaran yang dituju sedangkan *mobile* dapat di artikan sebagai perpindahan dari suatu tempat ketempat yang lain[4]. Maka aplikasi *mobile* dapat di artikan sebuah program aplikasi yang dapat dijalankan atau digunakan walaupun pengguna berpindah-pindah dari satu tempat ketempat yang lain serta mempunyai ukuran yang kecil. HTML 5 adalah HTML5 merupakan bahasa markah untuk menstrukturkan dan menampikan isi dari jejagat jembar, sebuah teknologi yang dipakai sebagai standar dari internet, HTML5 adalah revisi ke lima dari HTML dengan tujuan untuk mempermudah penerapan pada media teknologi terbaru[5].

1.3.1 Kelebihan *Mobile*

Mobile merupakan perangkat atau teknologi yang memiliki beberapa keunggulan seperti :

1. Akes cepat dan mudah
2. Fleksibel dapat dibawa atau digunakan dimanasaja
3. Kompitable dengan beberapa fungsi seperti akse internet dan media informasi lainnya.
4. Banyaknya peminat atau pengguna *mobile.*

1.3.2 Kekurangan *Mobile*

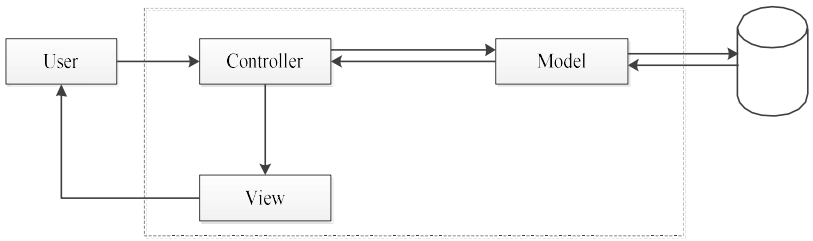
Berdasarkan kelebihan tentunya masih memiliki kekurangan seperti :

1. Keterbatasan platform yang digunakan.
2. Biaya untuk pengembangan aplikasi cukup tinggi
3. Upgrade pengguna harus mendownload untuk memiliki aplikasi terbaru.

1.4 *CodeIgniter*

CodeIgniter adalah *Framework* untuk bahasa pemprograman PHP, yang dibuat Rick Ellis pada tahun 2006. *CodeIgniter* memiliki banyak fitur yang membantu para pengembang PHP untuk dapat membuat aplikasi secara mudah dan cepat serta memiliki sifat yang fleksibel dapat mengembangkan dalam perangkat *web,* dekstop maupun *mobile*[6].

CodeIgniter memilki konsep atau pola *Model-View-Controller* (MVC) sehingga kode-kode dapat di sederhanakan.



Gambar 2.1 Arsitektur MVC

1.4.1 *JQuery Mobile*

JavaScript menjadi bahasa pemrograman yang paling banyak digunakan dan salah satu yang paling sering digunakan adalah *jQuery*, *library JavaScript* dirancang untuk menyederhanakan scripting HTML. Kalau dilansir dari situs *jQuery* sendiri, *jQuery* disebut memiliki karakteristik yang kecil, cepat, dan punya banyak fitur. *jQuery* yaitu memberikan kemampuan pada *jQuery* untuk bisa melakukan lintas *platform*. Sehingga *jQuery* tetap akan memiliki fungsi yang sama sekaligus memperbaiki *error* yang terjadi ketika dijalankan di berbagai jenis *browser* seperti *Safari, Google Chrome, Firefox, Android, dan IOS*[7].

1.4.2 PHP

PHP adalah bahasa *server-side-scripting* yang menyatudengan HTML untuk membuat halaman *web* yang dinamis. *Hypertext Preprocessor* adalah bahasa pemograman *web server-side* yang bersifat *open source*. PHP merupakan *script* yang terintegrasi dengan HTML dan berada pada *server* (*server side* HTML *embedded scripting*). PHP adalah *script* yang digunakan untuk membuat halaman *website* yang dinamis. Dinamis berarti halaman yang akan ditampilkan dibuat saat

halaman itu diminta oleh *client*. Mekanisme ini menyebabkan informasi yang diterima *client* selalu yang terbaru/*up to date*. Semua script PHP dieksekusi pada *server* dimana script tersebut dijalankan. Dengan menggunakan program PHP, sebuah *website* akan lebih interaktif dan dinamis[8].

Sehingga PHP merupakan bahsa pemprograman yang digunakan oleh pengembang untuk membuat sistem *website* dengan kumpulan bahasa HTML dan *script* lainya.

1.4.3 MySql

MySQL adalah singkatan dari *Structue Query Language* yang digunakan untuk mendefinisikan structure data, memodifikasi data pada basis data, menspesifikasi batasan keamanan (*security*), hingga pemeliharaan data.

mendefinisikan *mysql* adalah RDBMS yang cepat dan mudah digunakan, serta sudah banyak digunakan untuk berbagai kebutuhan. *MySQL* merupakan bahasa standar yang paling banyak digunakan untuk mengakses *database* relasional dan merupakan aplikasi yang dapat dipergunakan secara bebas[9].

1.5 Metode Pengembang Sistem

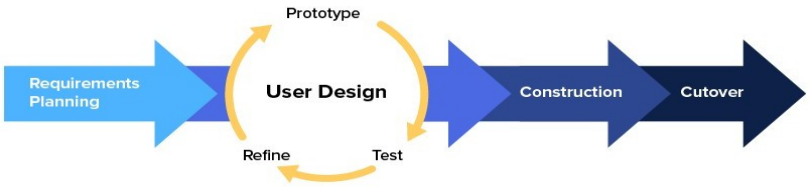
Metode pengembang sistem merupakan metode yang digunakan sebagai alur proses dalam pengembangan, sehingga penelitian dapat di kembangkan sesuai tahapan dari metode pengembang sistem.

2.3.1. *Rapid Application Development*(RAD)

Metode yang berfokus pada pengembangan aplikasi secara cepat, melalui pengulangan dan *feedback* berulang-ulang. RAD diajukan oleh IBM pada tahun 1980 sampai 1990-an, ketika permintaan terhadap aplikasi semakin meningkat. Dengan banyaknya *demand*, orang-orang di dunia IT harus mencari solusi untuk memenuhi permintaan tersebut[10]. Metode ini merupakan semacam cikal bakal *agile project management*, karena bisa mengikuti *pace* bisnis yang terus berkembang dan juga kebutuhan pasar yang terus meningkat. Pengembangan software pada umumnya seperti waterfall model membutuhkan perencanaan yang

terbilang cukup kaku. Klien atau pelanggan seakan ‘dipaksa’ untuk menyetujui banyak hal di awal, tetapi mereka tidak bisa melihat proses pembuatannya.

Keuntungan utama menjalankan *rapid application development* adalah jangka waktu pengembangan lebih cepat.Hal ini dikarenakan *feedback* dari pelanggan cepat didapatkan dan semua perubahan yang dilakukan akan sesuai hasil tersebut. Akan tetapi, salah satu kekurangan RAD adalah kamu membutuhkan tim berisikan *developer* yang benar-benar memiliki *skill* tinggi dan juga metode ini hanya bisa digunakan untuk proyek yang bisa termodulasi.



Gambar 2.2*Rapid Application Development*(RAD)
Sumber: [11]

1. Kelebihan Model RAD

   Kelebihan metodologi RAD menurut Marakas (2006):

   a. Penghematan waktu dalam keseluruhan fase projek dapat dicapai.
   b. RAD mengurangi seluruh kebutuhan yang berkaitan dengan biaya projek dan sumberdaya manusia.
   c. RAD sangat membantu pengembangan aplikasi yang berfokus pada waktu penyelesaian projek.
   d. Perubahan desain sistem dapat lebih berpengaruh dengan cepat dibandingkan dengan pendekatan SDLC tradisional.
   e. Sudut pandang user disajikan dalam sistem akhir baik melalui fungsi-fungsi sistem atau antarmuka pengguna.
   f. RAD menciptakan rasa kepemilikan yang kuat di antara seluruh pemangku kebijakan projek.

2. Kelemahan Model RAD

Kelemahan pada pengembangan tersebut dapat dilihat berdasarkan kesesuaian pengembangan yang dilakukan, berikut adalah kelemahan metode pengembang sistem RAD :

a. Dengan metode RAD, penganalisis berusaha mepercepat projek dengan terburu-buru.
b. Kelemahan yang berkaitan dengan waktu dan perhatian terhadap detail. Aplikasi dapat diselesaikan secara lebih cepat, tetapi tidak mampu mengarahkan penekanan terhadap permasalahan-permasalahan perusahaan yang seharusnya diarahkan.
c. RAD menyulitkan *programmer* yang tidak berpengalaman menggunakan prangkat ini di mana *programmer* dan *analyst* dituntut untuk menguasai kemampuan-kemampuan baru sementara pada saat yang sama mereka harus bekerja mengembangkan sistem

2.3.2 Tahapan Penelitian

Tahapan dalam penelitian sebagai langkah-langkah penelitian yang harus dikerjakan, berikut adalah tahapan penelitian Rapid Application Development.

1. Tahap *Requirements Project*

   RAD dimulai dengan menentukan kebutuhan sebuah proyek (*project requirements*). Pada tahap ini, tim perlu menentukan kebutuhan yang ingin dipenuhi dari sebuah proyek. Kebutuhan ini tidak perlu spesifik. Tapi, sifatnya benar-benar umum dan jumlahnya bisa banyak. Baru dari situ, tim akan menentukan mana kebutuhan yang perlu diprioritaskan. Setelah mendapatkan kebutuhan yang jelas, barulah tim menentukan hal-hal yang lebih detail. Misalkan seperti tujuan, timeline, dan budget yang diperlukan.
2. Tahap Membuat *Prototype*

   Hal yang selanjutnya dilakukan adalah membuat prototype. Developer secepat mungkin akan membuat prototype dari aplikasi yang diinginkan. Lengkap dengan fitur dan fungsi yang berbeda-beda. Tujuannya, sekadar untuk mengecek apakah *prototype* yang dibuat sudah sesuai dengan kebutuhan klien. Meski begitu, tahap ini bisa saja dilakukan berulang-ulang. Kadang juga melibatkan user untuk testing dan memberikan feedback. Proses ini memungkinkan tim mempelajari error yang mungkin muncul ke depannya.

Ini berguna untuk mengurangi error dan debugging. Lewat tahapan ini, tim developer memiliki modal untuk membuat aplikasi yang mudah dipakai, stabil, tidak sering error, dan desainnya pun oke.

3. Proses Pengembangan dan Penngumpulan *Feedback*

   Setelah tahu aplikasi seperti apa yang ingin dibuat, developer mengubah *prototype* ke bentuk aplikasi versi beta sampai dengan final. Jadi, bisa dibilang tahap RAD inilah yang cukup intens. Developer terus-menerus melakukan coding aplikasi, melakukan testing sistem, dan integrasi dengan bagian-bagian lainnya. Karena itulah, developer menggunakan tools dan framework yang mendukung RAD agar cepat. Apalagi proses ini terus diulang sambil terus mempertimbangkan feedback dari klien. Baik itu soal fitur, fungsi, interface, sampai keseluruhan aspek dari produk yang dibuat. Nah, kalau prosesnya berjalan lancar, developer akan melanjutkan ke langkah berikutnya. Yaitu, finalisasi produk atau implementasi. Kalau pun tidak, proses ini kemungkinan akan terus diulang. Pun, kalau apes-apesnya aplikasi tidak tidak menjawab kebutuhan, developer akan kembali ke proses *prototyping*.

4. Tahap *Implementation* (implementasi).

   merupakan tahap pengujian terhadap aplikasi yang dikembangkan. Tahap ini programmer mengembangkan desain menjadi suatu program kemudian dilakukan proses pengujian untuk memeriksa kesalahan sebelum diaplikasikan.

1.6 Alat Pengembang Sistem

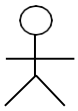
Unified Modelling Language adalah bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung. Berikut ini merupakan penjelasan tentang masing-masing diagram yang ada pada *Unified Modelling Language* .

1.6.1 *Use Case*Diagram

Use case mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. *Use case* digunakan untuk mengetahui

fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Berikut simbol-simbol yang akan digunakan dalam menggambarkan *Use Case Diagram* dapat dilihat pada Tabel
2.1 :

Tabel 2.1 Simbol-simbol Use Case Diagram

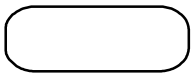
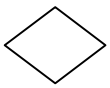
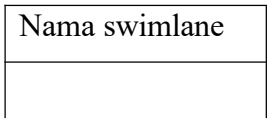
No	Simbol	Keterangan Fungsi
1.	Aktor 	Aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu.
2.	*Use Case* 	*Use Case* adalah deskripsi dari urutan aksi- aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
3.	Asosiasi	Asosiasi adalah apa yang menghubungkan antara objek satu dengan objek yang lainnya.
4.	Generalisasi 	Generalisasi adalah hubungan dimana objek anak (*descendent*) berbagi prilaku dan struktur data dari objek yang ada diatasnya atau sebaliknya dari bawah keatas.
5.	*Defendency*	*Defendency* (ketergantungan) adalah hubungan dimana perubahan yang terjadi pada suatu elemen defenden (mandiri) akan mempengaruhi elemen yang bergantung padanya (independen).

Sumber: (Rosa dan Salahuddin, 2019)

1.6.2 *Activity Diagram*

Activity diagram menggambarkan aliran kerja atau aktivitas dari sebuah sistem atau proses bisnis atau menggambarkan aktivitas sistem bukan apa yang dilakukan aktor. Berikut simbol-simbol yang akan digunakan dalam menggambarkan *activitydiagram* dapat dilihat pada Tabel 2.2:

Tabel 2.2 Simbol *Activity Diagram*

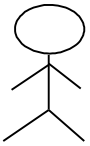
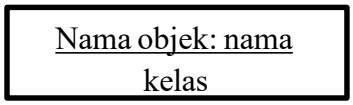
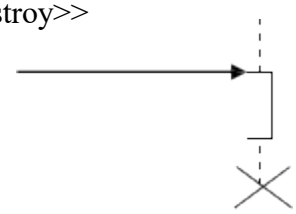
No.	Simbol	Keterangan
1.		Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
2.		Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
3.		Percabangan *(Decision)* merupakan asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
4.		Penggabungan *(Join)* merupakan asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
5.		Swimlane Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas.
6.		Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.

Sumber:[12]

1.6.3 *Squence* Diagram

*Diagramsquence*menggambarkan kelakuan objek pada *use case* dengan mendeskripsikanwaktuhidupobjekdengan*massage*yangdikirimkandanditerima 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. Membuat diagramsekuen juga dibutuhkan untuk melihat skenario yang ada pada *use case*. Banyaknya diagram sekuen 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 dalam diagram sekuen sehingga semakin banyak *use case* yang didefinisikan maka diagram sekuen yang harus dibuat juga semakin banyak.Berikut adalah simbol-simbol yang ada pada *squence* diagrampada Tabel 2.3.

Tabel 2.3 Simbol-simbol *Squence* Diagram

No.	Simbol	Deskripsi
1	Aktor  Atau Nama 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 akor adalah gambar orang, tapi akor belum tentu merupakan orang, biasanya dinyatakan dalam menggunakan kata benda di awal frase nama aktor
2.	Garis hidup /*lifeline*	Menyataan kehidupan suatu objek
3.	Objek 	Menyetakan objek yang berinteraksi peran
4	Pesan tipe *destroy* <<destroy>> 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaliknya jika ada *create* maka ada D*estroy*

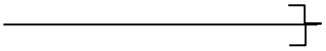
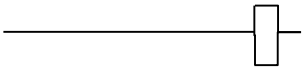
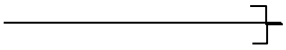
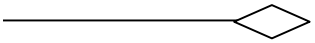
Sumber:[12]

1.6.4 *Class* Diagram

Class diagram mengembangkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Berikut simbol-simbol yang akan digunakan dalam menggambarkan *Class Diagram* dapat dilihat pada Tabel 2.4:

Tabel 2.4 Simbol *Class Diagram*

No.	Simbol	Deskripsi

1.	Nama_kelas +Atribute +Operasi	Kelas pada struktur sistem**.**
2.	Antar Muka/Interface  Nama_Interface	Sama dengan konsep interface dalam pemrograman berorientasi objek.
3.	Asosiasi / Asociation 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan symbol
4.	Asosiasi Berarah / Directed Association 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan symbol.
5.	Generalisasi 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum khusus)
6.	Ketergantungan / dependency 	Relasi antar kelas dengan makna ketergantungan antar kelas.
7.	Agregasi / aggregation 	Relasi antar kelas dengan maksna semua bagian *(whole-part)*

Sumber:[12]