BAB II

TINJAUAN PUSTAKA

2.1 Teori Umum

2.1.1 Pengertian Perangkat Lunak

Menurut Iskandar (2018:68) “Software adalah bagian sistem komputer yang tidak memiliki wujud, software juga bisa memiliki pengertian sebagai data yang berformat digital dan disimpan secara digital yang hanya bisa dibaca oleh komputer”

Sedangkan Kadir (2017:2) mengatakan bahwa, “Perangkat lunak adalah intruksi-intruksi yang ditujukan kepada komputer agar dapat melaksanakan tugas sesuai kehendak pemakai”.

Berdasarkan pendapat diatas, dapat disimpulkan bahwa perangkat lunak merupakan sistem yang ditujukan kepada komputer sesuai intruksi dari pemakai untuk memproses suatu data atau informasi.

2.1.2 Pengertian Komputer

Hamacher, Vranesic, dan Zaku (dalam Krisbiantoro, 2018:1) mengatakan bahwa, “Komputer didefinisikan sebagai sebuah mesin penghitung elektronik yang cepat dapat menerima informasi input digital, memprosesnya sesuai dengan suatu program yang tersimpan di memorinya (*stored program*) dan menghasilkan output informasi”.

Sedangkan menurut Kadir (2017:2), “Komputer merupakan peralatan elektronik yang bermanfaat, melaksanakan berbagai pekerjaan yang dilakukan oleh manusia ”.

Berdasarkan pendapat di atas, dapat disimpulkan bahwa definisi dari komputer adalah sebuah alat elektronik yang dapat menerima dan memproses data digital sesuai perintah user sehingga menghasilkan informasi yang bermanfaat bagi *user*.

2.1.3 Pengertian Data

Menurut Siregar (2015:16) menyatakan bahwa, “Data merupakan kumpulan fakta atau angka atau segala sesuatu yang dapat dipercaya kebenarannya sehingga dapat digunakan sebagai dasar untuk menarik suatu kesimpulan”

Sama halnya menurut Jayanti dan Sumiari (2018:1), “Data merupakan catatan atas kumpulan fakta yang mewakili suatu objek".

Berdasarkan pendapat di atas, dapat disimpulkan bahwa definisi dari data adalah fakta dari suatu objek yang kemudian akan digunakan dan dikelola agar menghasilkan suatu informasi yang bermanfaat.

2.1.4 Pengertian Basis Data

Jayanti dan Sumiari (2018:2), “Basis data adalah sekumpulan data yang terintegrasi, yang diorganisasi untuk memenuhi kebutuhan para pemakai di dalam suatu organisasi”

Subandi dan Syahidi (2018:3), "Basis data dapat diungkapkan sebagai suatu pengorganisasian data dengan bantuan komputer yang memungkinkan data dapat diakses dengan mudah dan cepat”.

Jadi dapat penulis simpulkan pengertian basis data adalah sekumpulan data yang terkomputerisasi dan saling berhubungan agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.

2.1.5 Metode Pengembangan Sistem

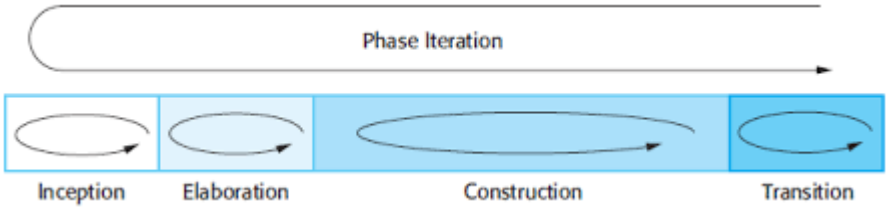
Penelitian ini menggunakan metode pengembangan perangkat lunak dengan RUP (*Rational Unified Process*). Menurut Sukamto dan Shalahuddin (2018:125), “RUP (*Rational Unified Process*) adalah pendekatan pengembangan perangkat lunak yang dilakukan berulang-ulang (*iterative*), fokus pada arsitektur (*architecture-centric*), lebih diarahkan berdasarkan penggunaan kasus (*use case driven*)”.

Sedangkan menurut Suryana dalam Triwahyuni dan Saputra (2015:2), “*Rational Unified Process* (RUP) merupakan suatu metode rekayasa perangkat

lunak yang dikembangkan dengan mengumpulkan berbagai *best practises* yang terdapat dalam industri pengembangan perangkat lunak."

Jadi dapat disimpulkan bahwa RUP adalah metode pengembangan perangkat lunak dengan menggunakan *use case driven* dan pendekatan iteratif. Adapun tahap-tahap (*fase*) dalam metode pengembangan RUP menurut Sukamto dan Shalahuddin (2018:129-131) adalah sebagai berikut:

Gambar 2.1 Alur Hidup RUP



Sumber : Sukamto dan Shaluddin (2018-128)

1. *Inception* (permulaan)

   Tahap ini lebih pada memodelkan proses bisnis yang dibutuhkan (*bussiness modeling*) dan mendefinisikan kebutuhan akan sistem yang akan dibuat (*requirements*).

   Tahap yang dibutuhkan pada tahap ini yakni:

   a. Memahami ruang lingkup dari proyek (termasuk pada biaya, waktu, kebutuhan, resiko dan lain sebagainya)

   b. Membangun kasus bisnis yang dibutuhkan.

   Hasil yang diharapkan dari tahap ini adalah memenuhi Lifecycle Objective Milestone (batas atau tonggak objektif dari siklus) dengan kriteria berikut :

   a. Umpan balik dari pendefinisian ruang lingkup, perkiraan biaya, dan perkiraan jadwal.

   b. Kebutuhan dimengerti dengan pasti (dapat dibuktikan) dan sejalan dengan kasus primer yang dibutuhkan.

   c. Kredibilitas dari perkiraan biaya, perkiraan jadwal, penentuan skala prioritas, resiko, dan proses pengembangan.

   d. Ruang lingkup purwarupa (prototype) yang akan dikembangkan.

e. Membangun garis dasar dengan membandingkan perencanaan aktual dengan perencanaan yang direncanakan

Jadi dari penjelasan diatas dapat disimpulkan bahwa tahap inception yakni tahap menganalisis kebutuhan perangkat lunak, melakukan penyelidikan awal, membangun prosedur sistem yang akan diterapkan, melakukan studi kelayakan, serta memahami kebutuhan fungsional dan non fungsional dari sistem yang akan dibangun.

1. Analisis kebutuhan perangkat lunak

   Analisa kebutuhan merupakan langkah awal untuk menentukan perangkat lunak seperti apa yang akan dihasilkan, ketika kita melaksanakan sebuah proyek pembuatan perangkat lunak. Perangkat lunak yang baik dan sesuai dengan kebutuhan pengguna sangat bergantung kepada keberhasilan dalam melakukan analisa kebutuhan. (Handoyo dan Setiawan, 2009:36).

2. Penyelidikan Awal

   Menurut Yahaya dkk (2006:2-3), Penyelidikan Awal pada hakikatnya adalah mencari jawaban atau menyelesaikan masalah dengan jawaban yang benar atau sekurang-kurangnya dapat mengetahui kebenaan yang logis melalui pemikiran yang mempunyai kesinambungan dengan fakta. Penyelidikan dapat disimpulkan sebagai cara menguraikan masalah yang akan dihadapi.

3. Studi kelayakan

   Kurniawati dkk (2014:2), studi kelayakan merupakan suatu kegiatan penelitian pada suatu proyek atau investasi untuk menentukan layak atau tidak suatu proyek tersebut dilaksanakan. Adapun kelayakan yang dinilai menurut Sutabri (2012:59) yakni sebagai berikut:

   a. Kelayakan Operasional

      Menyangkut apakah secara operasional sistem yang baru dapat dilaksanakan dengan sumber daya manusia yang tersedia dan metode training yang ditawarkan, pelayanan purna jual atau pemeliharaan serta efisien dan efektivitas sistem baru

b. Kelayakan Teknis

Menyangkut apakah hardware/software yang akan dikembangkan tersedia, jadwal pelaksaan serta sistem keamanan data

c. Kelayakan Ekonomis

Menyangkut biaya untuk membuat dan menjalankan sistem baru serta keuntungan yang akan diperoleh dari sistem tersebut.

4. Kebutuhan fungsional dan non fungsional

Kebutuhan fungsional adalah jenis kebutuhan yang berisi proses-proses apa saja yang nantinya dilakukan oleh sistem. Kebutuhan fungsional juga berisi informasi-informasi apa saja yang harus ada dan dihasilkan oleh sistem. (Al Fatta, 2007:63)

Menurut Faslah dkk (2017:38), Kebutuhan fungsional adalah proses yang harus dilakukan oleh sistem yang mendukung apa yang dilakukan pengguna atau menyediakan informasi yang diperlukan pengguna. Sedangkan kebutuhan non fungsional adalah hal-hal yang terkait operasional, kinerja, kemanan serta politik dan budaya yang harus dipenuhi oleh sistem.

2. *Elaboration* (perluasan/perencanaan)

Tahap ini lebih difokuskan pada perencanaan arsitektur sistem. Tahap ini juga dapat mendeteksi apakah arsitektur sistem yang diinginkan dapat dibuat atau tidak. Mendeteksi resiko yang mungkin terjadi dari arsitektur yang dibuat. Tahap ini lebih pada analisis dan desain sistem serta implementasi sistem yang fokus pada purwarupa sistem (*prototype*).

Hasil yang diharapkan dari tahap ini adalah memenuhi Lifecycle Architecture Milestone(batas atau tonggak arsitektur dari siklus) dengan kriteria berikut:

a. Model kasus yang digunakan (use case) dimana kasus dan aktor yang terlibat telah diidentifikasi dan sebagian besar kasus harus dikembangkan. Model use case harus 80 persen lengkap dibuat.

b. Deskripsi dari arsitektur perangkat lunak dari proses pengembangan sistem perangkat lunak telah dibuat.

c. Rancangan arsitektur yang dapat diimplementasikan dan mengimplementasikan use case. d.Kasus bisnis atau proses bisnis dan daftar resiko yang sudah mengalami perbaikan (revisi) telah dibuat.

e. Rencana pengembangan untuk seluruh proyek telah dibuat.

f. Purwarupa (prototype) yang dapat didemonstrasikan untuk mengurangi setiap resiko teknis yang diidentifikasi.

3. *Construction* (kontruksi)

Tahap ini fokus pada pengembangan komponen dan fitur-fitur sistem. Tahap ini lebih pada implementasi dan pengujian sistem yang fokus pada implementasi perangkat lunak pada kode program. Tahap ini menghasilkan produk perangkat lunak dan hasil pengujian sistem yaitu dengan blackbox *validation testing* dan *compatibility testing.* Menurut Cholifah dkk (2018:2017) *Blackbox Testing* merupakan salah satu metode yang mudah digunakan karena hanya memerlukan batas bawah dan batas atas dari data yang di harapkan,Estimasi banyaknya data uji dapat dihitung melalui banyaknya fielddata entri yang akan diuji, aturan entri yang harus dipenuhi serta kasus batas atas dan batas bawah yang memenuhi.

4. *Transition* (transisi)

Tahap ini lebih pada deployment atau instalasi sistem agar dapat dimengerti oleh user. Tahap ini menghasilkan produk perangkat lunak dimana menjadi syarat dari *Initial Operational Capability Milestone* atau batas/tongggak kemampuan operasional awal. Aktifitas pada tahap ini termasuk pada pelatihan user, pemeliharaan dan pengujian sistem apakah sudah memenuhi harapan user.

2.2 Teori Khusus

2.2.1 Pengertian Kamus Data

Supardi (2015:7) mengatakan bahwa, "Kamus data merupakan model yang tidak menggunakan notasi grafis sebagaimana halnya DFD ."

Sedangkan menurut Sukamto dan Shalahuddin (2018:73), "Kamus data adalah kumpulan daftar elemen data yang mengalir pada sistem perangkat lunak sehingga masukan (input) dan keluaran (output) dapat dipahami secara umum (memiliki standar cara penulisan)".

Jadi dapat disimpulkan bahwa kamus data adalah daftar elemen data yang tidak berbentuk grafis dalam penulisannya agar mudah dipahami. Kamus data memiliki beberapa simbol sebagai berikut :

Tabel 2.1 Simbol-simbol Kamus Data

No	Simbol	Keterangan
1.	=	Terdiri dari, mendefinisikan, diuraikan menjadi, artinya, disusun.
2.	+	Dan
3.	**( )**	Opsional (boleh ada atau tidak)
4.	**{ }**	Pengulangan
5.	**[|]**	Seleksi, memilih salah satu dari sejumlah alternatif
6.	***...***	Komentar
7.	**@**	Identifikasi atribut kunci
8.	**!**	Pemisah sejumlah alternatif pilihan simbol []

Sumber : Supardi (2015:7-8)

2.2.2 Pengertian *Unified Modeling Language* (UML)

Sukamto dan Shalahuddin (2018:133), menjelaskan tentang pengertian *Unified Modeling Language* bahwa, *"Unified Modeling Language* (UML) adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan requirement, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek."

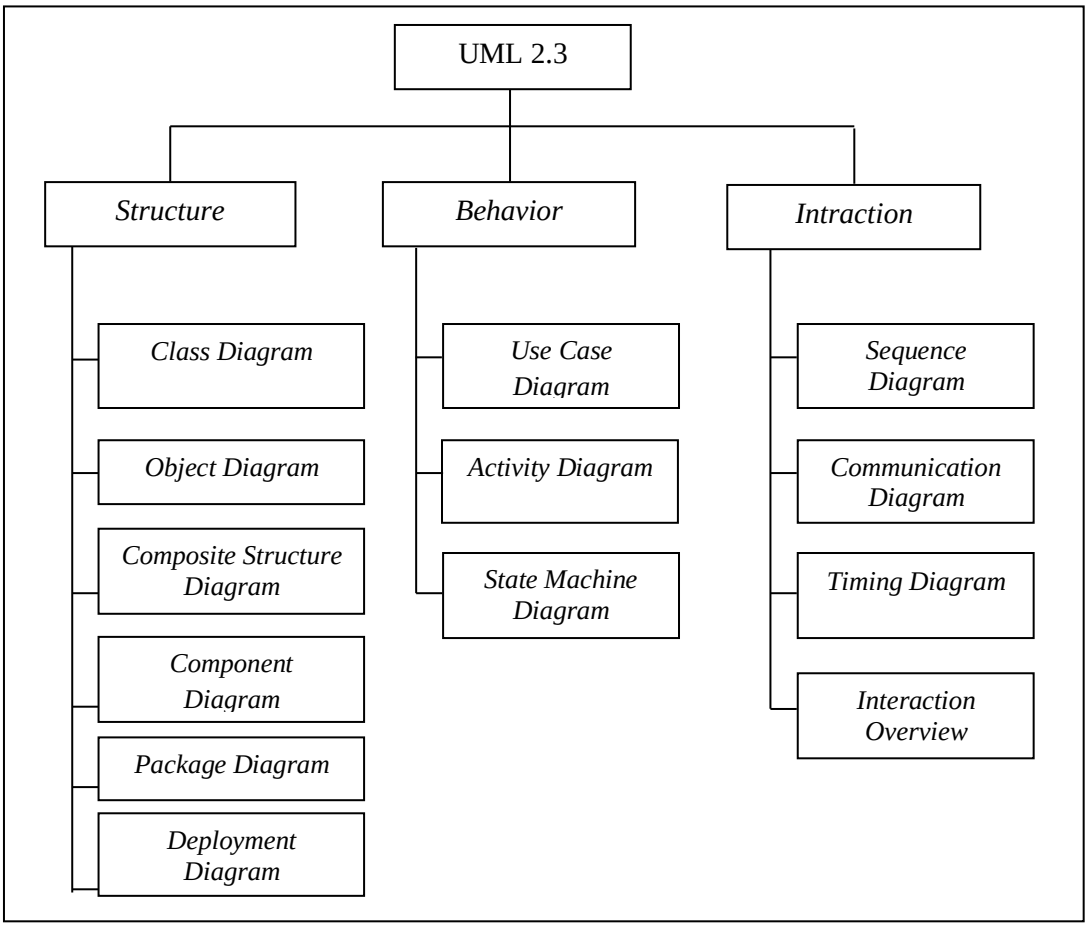
Sedangkan menurut Windu dan Grace dalam Suendri (2018:2), "*Unified Modeling Language* (UML) adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangat lunak".

Jadi dapat disimpulkan bahwa UML adalah bahasa yang dibutuhkan pada pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak dengan diagram dan teks-teks pendukung.

2.2.3 Kategori *Unified Modeling Language* (UML)

Menurut Sukamto dan Shalahuddin (2018:140), "Pada UML 2.3 terdiri dari 13 macam diagram yang dikelompokkan dalam 3 kategori". Pembagian kategori dan macam-macam diagram Menurut Sukamto dan Shalahuddin tersebut dapat dilihat pada gambar dibawah:

Gambar 2.2 Kategori dan Macam-macam Diagram UML



Sumber : Sukamto dan Shalahuddin (2018:141)

Berikut penjelasan singkat dari pembagian kategori pada diagram UML menurut Sukamto dan Shalahuddin (2018:141) :

1) *Structure diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.
2) *Behavior diagrams* yaitu kumpulam diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahana yang terjadi pada sebuah sistem.
3) *Interaction diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.

Gambar 2.3 Tampilan Logo UML

2.2.4 Jenis-Jenis Diagram UML

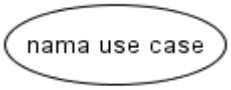
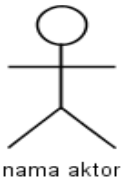
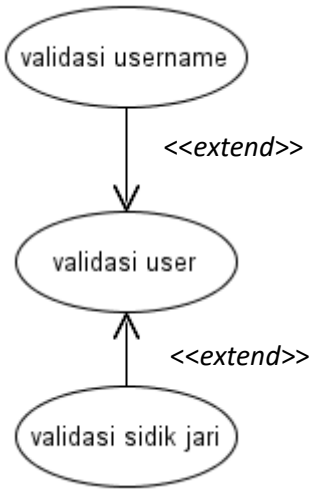
2.2.4.1 Pengertian *Use case* Diagram

Menurut Sukamto dan Shalahuddin (2018:155), *"Use case* atau diagram *use case* merupakan pemodelan untuk kelakuan *(behavior)* sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat."

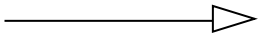
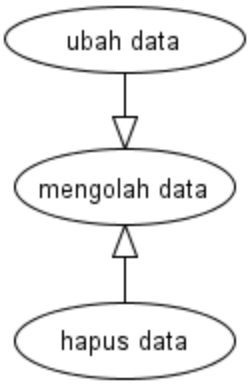
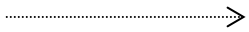
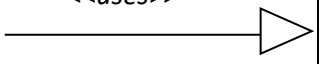
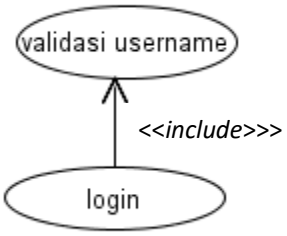
"*Use case* diagram adalah sesuatu atau proses merepresentasikan hal-hal yang dapat dilakukan oleh aktor dalam menyelesaikan sebuah pekerjaan." (Manalu dalam Heriyanto, 2018:67)

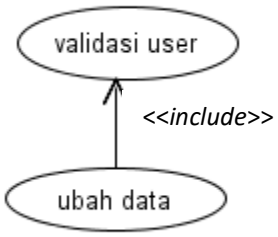
Dari pernyataan diatas dapat ditarik kesimpulan bahwa *Use case* diagram adalah suatu pemodelan sistem informasi yang akan dibuat dengan cara sistem lain melakukan interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Adapun simbol-simbol yang digunakan dalam *use case* adalah sebagai berikut:

Tabel 2.2 Simbol-simbol *Use case* Diagram

No	Simbol	Deskripsi
1.	*Use case* 	fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal-awal frase nama *use case*.
2.	aktor / *actor* 	orang, proses, atau siatem 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.
3.	asosiasi / *association* _____	komunikasi antar aktor dan *use case* yang berpartisioasi pada *use case* atau *use case* memiliki ingeraksi dengan actor.
4.	ekstensi / *extend* 	relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang di tambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu; mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek; biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan, misalnya  arah panah mengarah pada *use case* yang ditambahkan; biasanya *use case* yang menjadi *extend*-nya merupakan jenis yang sama dengan *use case* yang menjadi induknya

Lanjutan Tabel 2.2 Simbol-simbol *Use case* Diagram

No	Simbol	Deskripsi
5.	Generalisasi / *generalization* 	hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya,  misalnya: arah panah mengarah pada *use case* yang menjadi generalisasinya (umum).
6.	menggunakan / include / uses  <<*include*>> <<*uses*>> 	relasi tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini ada dua sudut pandang yang cukup besar mengenai include di *use case*: *a. Include* berarti *use case* yang ditambahkan akan selalu di panggil saat *use case* tambahan dijalankan, misalnya pada kasus berikut:  *b. Include* berarti *use case* yang tambahan akan selalu melakukan pengecekan apakah *use case* yang di tambahkan telah dijalankan sebelum *use case* tambahan dijalankan, misal pada kasus berikut:

		 kedua interpretasi di atas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan.

Sumber: Sukamto dan Shalahuddin (2018:156-158)

Ada dua hal utama pada *use case* menurut Sukamto dan Shalahuddin (2018:155), yaitu:

1. Aktor merupakan 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, tapi aktor belum tentu merupakan orang.
2. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

2.2.4.2 Pengertian *Class Diagram*

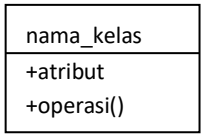
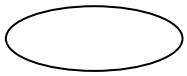
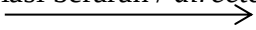
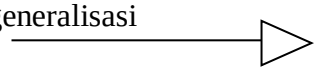
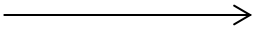
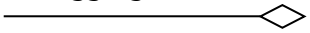
Menurut Sukamto dan Shalahuddin (2018:141-142), *"Class Diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Diagram kelas dibuat agar pembuat program atau *programmer* membuat kelas-kelas sesuai rancangan di dalam diagram kelas agar antara dokumentasi perancangan dan perangkat lunak sinkron."

Menurut Hendini (2016:11), "*Class Diagram* merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggung jawab entitas yang menentukan perilaku sistem."

Jadi dapat disimpulkan bahwa *class diagram* adalah model desain yang menggambarkan struktur sistem hubungan antar kelas dan menjelaskan kelas-

kelas tersebut. Adapun simbol-simbol yang digunakan dalam *class* diagram adalah sebagai berikut:

Tabel 2.3 Simbol-simbol *Class* Diagram

No	Simbol	Deskripsi
1.	kelas 	Kelas pada struktur sistem
2	antarmuka / interface  nama_interface	Sama dengan konsep interface dalam pemrograman berorientasi objek
3.	asosiasi / association	Relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai *multiplicity*
4.	asosiasi berarah / *directed assosiation* 	Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*
5.	generalisasi 	Relasi antarkelas dengan makna generalisasi – spesialisasi (umum khusus)
6.	kebergantungan / *dependency* 	Relasi antarkelas dengan makna kebergantungan antar kelas
7.	agregasi / *aggregation* 	Relasi antarkelas dengan makna semua-bagian (*whole-part*)

Sumber: Sukamto dan Shalahuddin (2016:146-147)

2.2.4.3 Pengertian *Activity Diagram*

Menurut Sukamto dan Shalahuddin (2018:161), “Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem.”

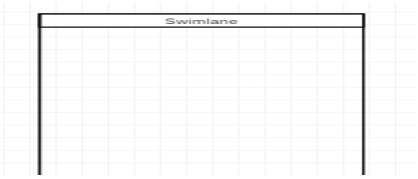
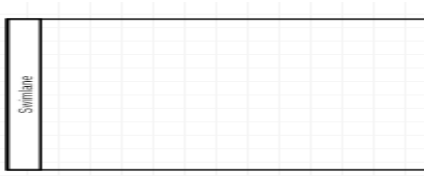
"Diagram *activity* menunjukan aktivitas-aktivitas, objek, state, transisi *state* dan *event*. Dengan kata lain kegiatan diagram alur kerja menggambarkan perilaku sistem untuk aktivitas." (Haviluddin dalam Suendri, 2018:3)

Dari kedua pernyataan diatas, penulis menyimpulkan bahwa diagram *activity* adalah diagram alur kerja yang menggambarkan aktivitas yang dilakukan oleh sistem. Adapun simbol-simbol yang digunakan dalam *activity diagram* adalah sebagai berikut:

Tabel 2.4 Simbol-simbol *Activity Diagram*

No	Simbol	Deskripsi
1.	Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal
2.	Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
3.	Percabangan / *decision* 	Asosiasi percabangan di mana jika ada pilihan aktivitas lebih dari satu
4.	Penggabungan / *join* 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
5.	Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir

Lanjutan Tabel 2.4 Simbol-simbol *Activity Diagram*

No	Simbol	Deskripsi
6.	Swimlane  atau 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

Sumber: Sukamto dan Shalahuddin (2018:162-163)

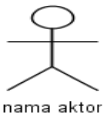
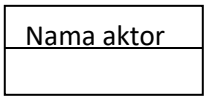
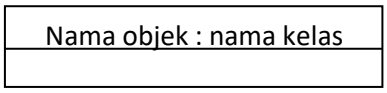
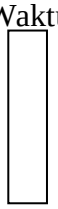
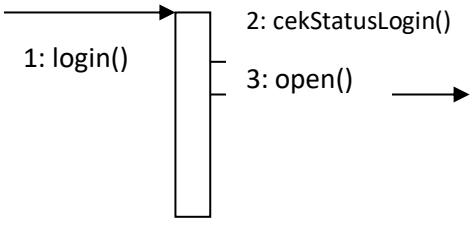
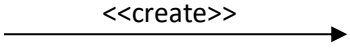
2.2.4.4 Pengertian *Sequence Diagram*

"Diagram sekuen menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan message 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 diinstansikan menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat scenario yang ada pada *use case*." (Sukamto dan Shalahuddin, 2018:165).

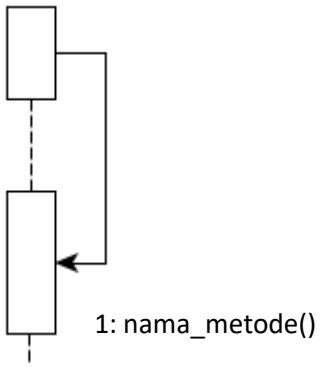
"Secara mudahnya *sequence* diagram adalah gambaran tahap demi tahap, termasuk kronologi (urutan) perubahan secara logis yang seharusnya dilakukan untuk menghasilkan sesuatu sesuai dengan *use case* diagram." (Haviluddin dalam Suendri, 2018:3).

Dari pernyataan diatas dapat penulis simpulkan bahwa *sequence* diagram adalah gambaran urutan kelakuan objek yang terlibat dalam *use case* diagram. Berikut adalah simbol-simbol pada Sequence Diagram :

Tabel 2.5 Simbol-simbol *Sequence* Diagram

No	Simbol	Deskripsi
1.	Actor  Atau  tanpa waktu aktif	orang, proses, atau sistem lain yang berinteraksi dengan 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.	Garis hidup / *lifeline* 	menyatakan kehidupan suatu objek
3.	Objek  	menyatakan objek yang berinteraksi pesan
4.	Waktu aktif 	menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya, misalnya  maka cek Status Login () dan open() dilakukan di dalam metode login() aktor tidak memiliki waktu aktif
5.	Pesan tipe create  	menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat

Lanjutan Tabel 2.5 Simbol-simbol pada *Sequence* Diagram

No	Simbol	Deskripsi
6.	Pesan tipe call 1: nama metode()	menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri,  arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi/metode maka operasi/metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi
7.	Pesan tipe send 1: masukan	menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi
8.	Pesan tipe return 1: keluaran	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.	Pesan tipe destroy <<destroy>>	menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada create maka ada destroy

Sumber: Sukamto dan Shalahuddin (2018:165-167)

2.3 Teori Judul

2.3.1 Pengertian Aplikasi

Menurut Pane, dkk (2020:53), "Aplikasi adalah suatu perangkat lunak (software) atau program komputer yang beroperasi pada sistem tertentu yang diciptakan dan dikembangkan untuk melakukan perintah tertentu".

Jogiyanto dalam Pane, dkk (2020:4), "Aplikasi adalah penggunaan dalam suatu komputer, intruksi (*instruction*) atau pernyataan (*statement*) yang disusun sedemikian rupa sehingga computer dapat memproses input menjadi output"

Jadi dapat disimpulkan bahwa aplikasi adalah intruksi atau pernyataan yang diciptakan dalam program komputer untuk melakukan atau memproses suatu perintah tertentu .

2.3.2 Pengertian Pendaftaran

Menurut Departemen Pendidikan dan Kebudayaan dalam Magdalena dan Rachman (2017:39), **"**Pendaftaran adalah proses, cara, perbuatan mendaftar yaitu pencatatan nama, alamat dsb dalam daftar. Jadi, pendaftaran adalah proses pencatatan identitas pendaftar kedalam sebuah media penyimpanan yang digunakan dalam proses pendaftaran."

Adapun dalam Kamus Besar Bahasa Indonesia (KBBI) menyatakan bahwa, "Pendaftaran adalah pencatatan nama, alamat, dan sebagainya dalam sebuah daftar".

Dari pernyataan diatas, maka penulis menarik kesimpulan bahwa pendaftaran adalah proses pencatatan identitas diri sebagai pendaftar kedalam media penyimpanan yang nantinya akan digunakan dalam segala hal sesuai kepentingan.

2.3.3 Aplikasi Pendaftaran Paket Umrah Pada PT. Sako Utama Wisata Berbasis Android

Aplikasi Pendaftaran Paket Umrah Pada PT. Sako Utama Wisata Berbasis Android digunakan untuk pendaftaran jamaah umrah, melihat informasi mengenai paket- paket umrah pada PT. Sako Utama Wisata.

2.4 Teori Program

2.4.1 Pengertian Android

Yudhanto dan Wijayanto (2017:1), "Android adalah sistem operasi berbasis Linux yang dirancang untuk perangkat bergerak layar sentuh seperti telepon pintar dan computer tablet."

Menurut Firly (2018:2), "Android yaitu dalam Bahasa inggris istilah Android berarti "Robot yang menyerupai manusia", hal tersebut dapat terlihat jelas pada icon android yang menggambarkan sebuah robot berwarna hijau yang memiliki sepasang tangan dan kaki".

Jadi kesimpulannya yaitu Android adalah sebuah sistem operasi perangkat *mobile* berbasis linux yang menggambarkan sebuah robot menyerupai manusia.

2.4.2 Pengertian *Framework Ionic*

Menurut Rofiq dan Putri (2017:172), "Framework Ionic adalah sekumpulan teknologi yang dikembangkan untuk membangun aplikasi mobile hybrid yang powerful, cepat, mudah dan juga memiliki tampilan yang menarik".

Sedangkan menurut Muttaqien dan Roslidar (2019:2), "Ionic adalah sebuah framework untuk membangun aplikasi berbasis mobile dan desktop menggunakan teknologi web (HTML, CSS, dan JavaScript)".

Berdasarkan pendapat tersebut dapat disimpulkan bahwa Framework Ionic adalah sekumpulan teknologi web untuk membangun aplikasi berbasis mobile.

2.4.3 Pengertian HTML

Abdulloh (2018:7), "HTML merupakan singkatan dari *Hypertxt Markup Language* yaitu bahasa standar web yang dikelola penggunaanya oleh W3C (*World Wide Web Consortium*) berupa tag-tag yang menyusun setiap elemen dari website."

Solichin (2016:10), "HTML merupakan bahasa pemrograman web yang memberitahukan peramban web (*web browser*) bagaimana menyusun dan menyajikan konten di halaman web."

Dari beberapa definisi diatas, dapat disimpukan HTML adalah skrip yang berupa tag-tag untuk membuat struktur dan menampilkan konten di halaman web.

2.4.4 Pengertian PHP

Gambar 2.4 Logo PHP

Menurut Abdulloh dalam Sa'ad (2020:22), "PHP merupakan singkatan dari *Hypertext Preprocessor* yang merupakan *server-side programing*, yaitu bahasa pemrograman yang diproses di sisi server."

Sedangkan menurut Solichin (2016:11), "PHP merupakan salah satu bahasa pemrograman berbasis web yang ditulis oleh dan untuk pengembang web."

Jadi, dapat disimpulkan bahwa PHP adalah bahasa pemrograman web berbasis server-side programming (bahasa pemrograman yang diproses disisi server) yang memparsing kode PHP dari kode web dengan ekstensi (.)php, yang kemudian akan dikirim ke browser web.

2.4.5 Pengertian *MySQL*

Gambar 2.5 Logo MySQL

Menurut Harianto, dkk (2019:13-14) menyatakan bahwa, "Mysql adalah salah satu jenis *database server* yang sangat terkenal dan banyak digunakan untuk membangun aplikasi web yang database sebagai sumber dan pengelolaan datanya. Kepopulerannya MySQL antara lain karena MySQL menggunakan SQL sebagai dasar untuk mengakses databasenya sehingga mudah untuk digunakan."

Sedangkan menurut Nugroho dalam Radillah (2018:14), "MySQL adalah sebuah program database server yang mampu menerima dan mengirim datanya dengan sangat cepat, *multi user* serta menggunakan perintah standar SQL (*Structure Quered Language)*."

Dari beberapa pendapat di atas dapat disimpulkan bahwa *MySQL* adalah salah satu jenis *database* yang digunakan untuk mengelola data dan menggunakan perintah standar SQL.

2.4.6 Pengertian Xampp

Gambar 2.6 Logo XAMPP

Laisina, dkk (2018:140) "XAMPP adalah perangkat lunak bebas, yang mendukung banyak sistem operasi, merupakan kompilasi dari beberapa program. Fungsinya adalah sebagai server yang berdiri sendiri (localhost), yang terdiri atas program Apache HTTP Server, MySQL database, dan penerjemah bahasa yang ditulis dengan bahasa pemrograman PHP dan Perl".

Santosa dan Nurmalina (2017:86) menyatakan bahwa, "Dengan menggunakan XAMPP tidak dibingungkan dengan penginstalan program-program lain, karena semua kebutuhan telah tersedia oleh XAMPP. Yang terdapat pada XAMPP diantaranya: Apache, MySQL, PHP, FilZilla FTP Server, PhpmyAdmin dll".

Berdasarkan pendapat tersebut, dapat disimpulkan bahwa XAMPP yaitu web server yang menyediakan beberapa paket perangkat lunak didalamnya.

2.4.7 Pengertian Sublime Text

Gambar 2.7 Logo Sublime Text

Menurut Bos dalam Pahlevi, dkk (2018:29), "Sublime Text merupakan salah satu text editor yang sangat powerful yang dapat meningkatkan produktivitas dan mengembangkan kualitas kode yang tinggi".

Sama halnya menurut Supono dan Putratama (2018:14), " Sublime Text merupakan perangkat lunak *text editor* yang digunakan untuk membuat atau mengedit suatu aplikasi. Sublime Text mempunyai fitur plugin tambahan yang memudahkan programmer".

Berdasarkan kedua pendapat diatas, maka dapat disimpulkan Sublime Text merupakan text editor yang berfungsi untuk membuat dan mengedit aplikasi.

2.4.8 Pengertian CSS

Menurut Azis, dkk (2019:36), "CSS (*Cascading Style Sheet*) adalah sebuah dokumen yang terdiri dari kode program yang digunakan untuk membuat elegan tampilan dari tampilan halaman *website* yang dibuat".

Sedangkan menurut Pahlevi, dkk (2018:28), "CSS kepanjangan dari *Cascading Style Sheet* adalah bahasa-bahasa yang merepresentasikan halaman web. Seperti warna, *layout*, dan *font*".

Dari kedua pendapat diatas dapat disimpulkan bahwa CSS adalah kode program digunakan untuk mempercantik tampilan warna, *layout* dan *font* pada halaman *website.*