BAB 2
TINJAUAN PUSTAKA

2.1 TEORI YANG BERKAITAN DENGAN APLIKASI

2.1.1 Sistem

Menurut Romney dan Steinbart (2015:3), sistem adalah suatu rangkaian yang terdiri dari dua atau lebih komponen yang saling berhubungan dan saling berinteraksi satu sama lain untuk mencapai tujuan dimana sistem biasanya terbagi dalam sub sistem yang lebih kecil yang mendukung sistem yang lebih besar.

Menurut Azhar Susanto (2013:22), SIstem adalah kumpulan atau *group* dari sub sistem/bagian/komponen apapun baik fisik maupun non-fisik yang saling berhubungan satu dengan yang lainnya dan bekerjasama dengan harmonis untuk mencapai satu tujuan tertentu.

Jadi dapat disimpulkan bahwa sistem adalah sebuah rangkaian yang terdiri dari komponen-komponen yang saling berhubungan satu dengan yang lainnya yang bertujuan untuk mencapai sebuah tujuan yang telah ditentukan oleh pengguna.

2.1.2 Informasi

Menurut Keri E. Pearlson and Carol S. Saunders (2014:81), Informasi adalah data yang diberkahi dengan relevansi dan tujuan, orang mengubah data menjadi informasi dengan mengaturnya ke dalam beberapa unit analisis. Menentukan unit analisis yang tepat melibatkan menafsirkan isi data dan meringkasnya menjadi bentuk yang lebih padat.

2.1.3 Sistem Informasi

Menurut Satzinger, Jackson, dan Burd (2012:7), Sistem informasi merupakan kumpulan dari komponen-komponen yang mengumpulkan, memproses, menyimpan, dan menyediakan output dari setiap informasi yang dibutuhkan dalam proses bisnis serta aplikasi yang digunakan melalui perangkat lunak, database dan bahkan proses manual yang terkait.

Menurut Gelinas dan Dull (2012:12), Sistem Informasi adalah sistem yang di buat secara umum berdasarkan seperangkat komputer dan komponen manual yang dapat dikumpulkan, disimpan dan diolah untuk menyediakan output kepada user.

Menurut Stair and Reynolds (2012:415), Sistem Informasi adalah suatu sekumpulan elemen atau komponen berupa orang, prosedur, database dan alat yang saling terkait untuk memproses, menyimpan serta menghasilkan informasi untuk mencapai suatu tujuan *(goal).*

Jadi dapat disimpulkan bahwa Sistem Informasi merupakan kumpulan dari komponen-komponen yang memproses, menyimpan, dan menyediakan hasil *(goal)* yang diperoleh dari keterkaitan perangkat lunak, *database* bahkan proses manual pada sebuah bisnis.

2.1.4 Aplikasi

Pengertian aplikasi menurut para ahli adalah sebagai berikut :

a. Menurut Syamsu Rizal, Eko Retnadi dan Andri Ikhwana (2013:23), Aplikasi adalah penggunaan dalam suatu perangkat komputer, instruksi atau pernyataan yang disusun hingga sedemikian rupa komputer dapat memproses masukan menjadi keluaran.
b. Menurut Joni Supriyono Arif Pramadya (2013:52), Aplikasi adalah perangkat lunak yang digunakan untuk membantu pemakai komputer untukk menyelesaikan pekerjaannya.
c. Menurut Nazarudin Safaat H (2012:9) Aplikasi adalah suatu subkelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan suatu tugas yang diinginkan pengguna.
d. Menurut Hengky W. Pramana (2012:17), Aplikasi adalah suatu unit perangkat lunak yang dibuat unutk melayani kebutuhan akan beberapa aktifitas seperti sistem perniagaan, game, pelayanan masyarakat, periklanan, atau semua proses yang dilakukan manusia.

Jadi sebuah aplikasi adalah suatu unit perangkat lunak yang dibangun untuk melayani kebutuhan dari pengguna.

2.1.4.1 *Mobile Application*

Menurut Cruz (2016:6-8), Aplikasi *mobile* atau *mobile application* terdiri dari perangkat lunak atau *set* program yang berjalan pada perangkat *mobile* dan melakukan tugas-tugas tertentu bagi pengguna. Aplikasi *mobile* merupakan *segmen* yang berkembang cepat di teknologi informasi dan komunikasi global. *Mobile apps* ini juga sangat mudah untuk digunakan (*user friendly*), murah dan mampu berjalan di sebagaian besar macam ponsel termasuk ponsel pada *entry level.* Aplikasi *mobile* memiliki kegunaan yang cukup luas seperti melakukan panggilan, mengirim pesan, *browsing, online chatting,* dan lain sebagainya.

Dari sudut pandang teknis aplikasi *mobile* berbeda di beberapa platform, seperti *Android, IOS, Blackberry, Windows, Symbian* dan lain-lain.

Menurut area cakupan dari suatu aplikasi, ada beberapa kategori yang membedakan aplikasi *mobile* dengan aplikasi lainnya, diantara lain :

1. Komunikasi : *Internet browsing, email* dan jaringan sosial.
2. *Games* : Teka-teki atau strategi, kartu *casino, action* atau petualangan.
3. Multimedia : *Graphics,* dan *image viewer*, presentasi pemirsa, pemain audio, pemain video.
4. Produktivitas : Kalender, kalkulator, *diary*, *notepad.*
5. Perjalanan : Konverter mata uang, penerjemah, *GPS* atau *maps*, rute perjalanan dan cuaca,
6. *Utilities : profile manager, screen server, task manager, call manager dan file manager.*

2.1.4.2 Komponen Aplikasi

Fitur penting *android* adalah bahwa satu aplikasi dapat menggunakan elemen dari aplikasi lain (untuk aplikasi yang memungkinkan). Sebagai contoh, sebuah aplikasi memerlukan fitur *scroller* dan aplikasi lain telah menggunakan fitur *scroller* yang baik dan memungkinkan aplikasi lain menggunakannya. Maka pengembang tidak perlu lagi mengembangkan hal serupa untuk aplikasinya, cukup menggunakan *scroller* yang telah ada. Agar fitur tersebut dapat bekerja, sistem harus dapat menjalankan aplikasi ketika setaiap bagaian aplikasi itu dibutuhkan, dan pemanggilan objek untuk bagian itu. *Android* tidak memiliki satu tampilan utama program seperti fungsi *main()* pada aplikasi lain. Sebaliknya, aplikasi memiliki komponen penting yang memungkinkan sistem untuk memanggil dan menjalankan ketika dibutuhkan.

1. *Activity*

   Activity merupakan bagian paling penting dalam sebuah aplikasi karena *activity* menyajikan tampilan visual program yang sedang digunakan oleh pengguna. Setiap *activity* dideklarasikan dalam sebuah kelas yang bertugas untuk menampilkan antar muka pengguna yang terdiri dari *views* dan respon terhadap *event*. Setiap aplikasi memiliki sebuah *activity* atau lebih. Biasanya pasti akan ada *activity* yang pertama kali tampil ketika aplikasi dijalankan. Perpindahan antara *activity* dengan *activity* lainnya diatur malalui sistem, sengan memanfaatkan *activity stack*. Keadaan suatu *activity* ditentukan oleh posisinya dalam tumpukan *activity*. LOFA (*Last In First Out*) dari semua apliaksi yang sedang berjalan. Bila satu *activity* baru dimulai, *activity* yang sebelumnya digunakan digunakan makan akan dipindahkan ke tumpukan paling atas. Jika pengguna ingin menggunakan *activity* sebelumnya, cukup menekan tombol *Back* atau menutup *activity* yang sedang digunakan, maka *activity* yang berada diatas akan aktif kembali. *Memory manager android* menggunakan tunmpukan ini untuk menentukan prioritas aplikasi berdasarkan *activity*, memutuskan untuk mengakhiri suatu aplikasi dan mengambil sumber daya dari apliaksi tersebut.

2. *Service*

Suatu *service* tidak memiliki tampilan antar muka, melainkan berjalan di *background* untuk waktu yang tidak terbatas. Komponen *service* diproses tidak terlihat,memperbarui sumber data dan menampilak *notifikasi*. *Service* digunakan untuk melakukan pengolahan data yang perlu terus diproses, bahkan ketika *activity* tidak aktif atau tidak tampak.

3. *Intents*

   Intents merupakan sebuah mekanisme untuk menggambarkan tindakan tertentu, sepeti memilih foto, menampilkan halaman *web* dan lain sebagainya. *Intents* tidak selalu dimulai dengan menjalankan aplikasi, namun juga digunakan oleh sistem untuk memberitahukan ke aplikasi bila terjadi suatu hal, misalnya pesan masuk. *Intents* dapat *eksplisit* atau *implisit*, contohnya jika suatu aplikasi ingin menampilan *URL*, sistem akan menentukan komponen apa yang dibutuhkan oleh *intents* tersebut.

4. *Broadcast Receivers*

   Broadcast Receivers merupakan komponen yang sebenarnya tidak melakukan apa-apa keculai menerima atau bereaksi menyampaikan pemberitahuan. Sebagian besar *broadcast* berasal dari sistem misalnya, baterai sudah hampir habis, informasi zona waktu yang berubah, atau pengguna telah merubah bahasa *default* pada perangkat. Sama halnya dengan *service, broadcast receiver* tidak menampilkan antar muka pengguna. Namun, *broadcast receiver* dapat menggunakan *Notification Manager* untuk memberitahukan sesuatu kepada pengguna.

5. *Content Providers*

   Content Providers digunakan untuk mengeloka dan berbagi *database*. Data dapat disimpan dalam file sistem, dalam database *SQLite*, atau dengan cara lain yang pada prinsipnya sama. Dengan adanya *content provider* memungkinkan antar aplikasi untuk saling berbagi data. Komponen ini sangat berguna ketika sebuah aplikasi membutuhkan data dari aplikasi lain, sehingga mudah dalam penerapannya.

2.1.5 Diagram UML (*Unified Modeling Language*)

Menurut Ary Budi Warsito, Muh. Yusup dan Moh Iqbal Awi Makaram (2015:9) *UML* atau *Unified Modeling Language* adalah himpunan struktur dan teknikuntuk pemodelan desain program berorientasi objek (*OOP*) serta aplikasinya. *UML* adalah metodologi untuk mengembangkan sistem *OOP* dan sekelompok perangkat (*tool*) untuk mendukung pengembangan sistem.

Menurut Yuni Sugiarti (2013:41) "*UML* adalah sebuah bahasa yang sudah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem perangkat lunak"

Menurut Satzinger, Jackson, dan Burd (2012:46), *unified modelling language* adalah bentuk standar dari model konstruksi dan notasi yang dikembangkan secara khusus untuk pengembangan berorientasi objek yang di definisikan oleh *OMG (Object Management Group),* ialah sebuah organisasi standar dalam pengembangan *UML*.

Contoh UML:

1. *Use case Diagram*
2. *Class Diagram*
3. *Sequence diagram*
4. *Communication diagram*
5. *State machine.*

2.1.5.1 Tujuan dan Fungsi Dari Penggunaan *UML Diagram*

Berikut ini merupakan tujuan dan fungsi dari penggunaan *UML* Diagram :

a. Dapat memberikan bahasa pemodelan visual kepada pengguna dari berbagai macam pemrograman maupun proses rekayasa.
b. Dapat menyatukan praktek-praktek terbaik yang ada dalam pemodelan.
c. Dapat memberikan model yang siap untuk digunakan, merupakan bahasa visual yang ekspresif untuk mengembangkan *system* dan untuk menukar model secara mudah
d. Dapat berguna sebagi *blueprint*, sebab sangat lengkap dan deatail dalam perancangannya yang nantikan akan diketahui informasi yang detail mengenai coding dari suatu program.

e. Dapat memodelkan *system* yang berkonsep berorientasi objek, jadi tidak hanya digunakan untuk memodelkan perangkat lunak saja.
f. Dapat menciptakan suatu bahasa permodelan yang nantinya dapat dipergunakan oleh manusia maupun mesin.

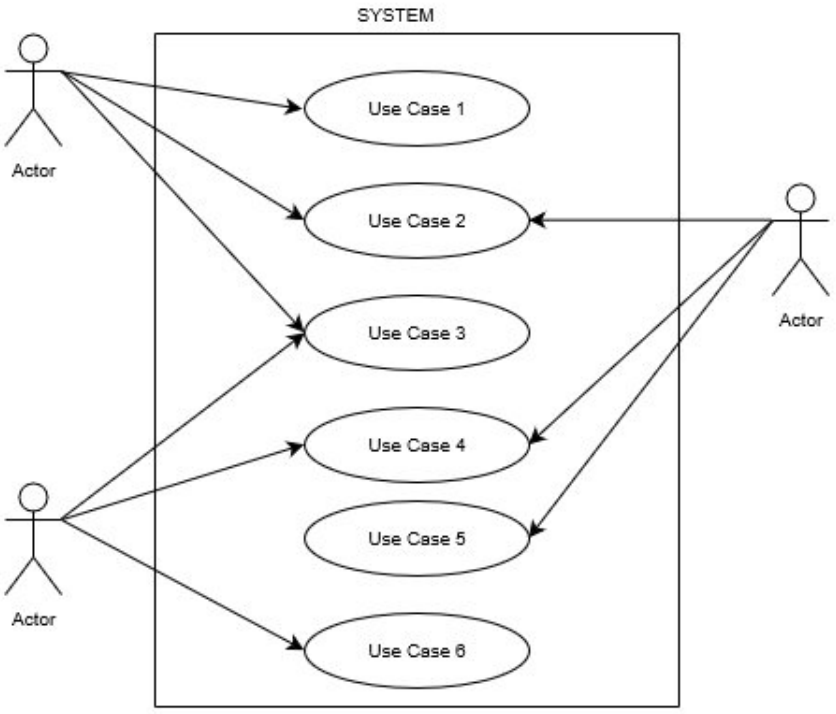
2.1.5.2 Jenis-jenis *UML diagram*

Menurut Whitten dan Bentley (2007:371*), Unified Modeling Languange* (UML) adalah satu kumpulan konsep atau aturan permodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem *software* yang berkaitan dengan objek.

UML dapat digunakan untuk menggambarkan, menentukan, membangun dan mendokumentasikan sesuatu dari sebuah sistem perangkat lunak. Ibaratnya, seperti seorang arsitek yang menciptakan sebuah desain bangunan untuk melakukan konstruksi bangunan, arsitek perangkat lunak juga dapat membuat diagram *UML* untuk membantu *developer* perangkat lunak dalam membangun perangkat lunak.

Jenis UML diagram antra lain :

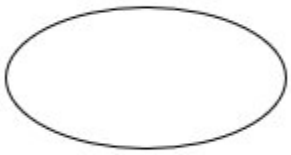
2.1.5.2.1 *Use Case Diagram*



Gambar 2.1 ***Use Case Diagram***

Menurut Whitten dan Bentley (2007: 246-252) diagram *use case* bertujuan untuk memberikan gambaran secara umum mengenai keseluruhan fungsionalitas suatu sistem dalam bentuk aktor, aktivitas, dan ketergantungannya. Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. Blok-blok diagram yang digunakan antara lain :

1. *Use Case*

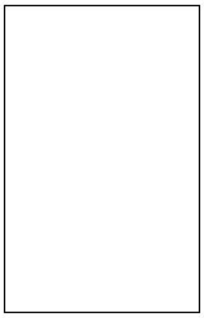


Menggambarkan suatu aturan tindakan aktor yang digambarkan dengan bentuk elips.

2. Aktor

Aktor disini adalah orang, organisasi, atau sistem eksternal yang berperan dalam satu atau lebih interaksi dengan sistem. Disimbolkan dalam bentuk manusia/orang.

3. Boundary



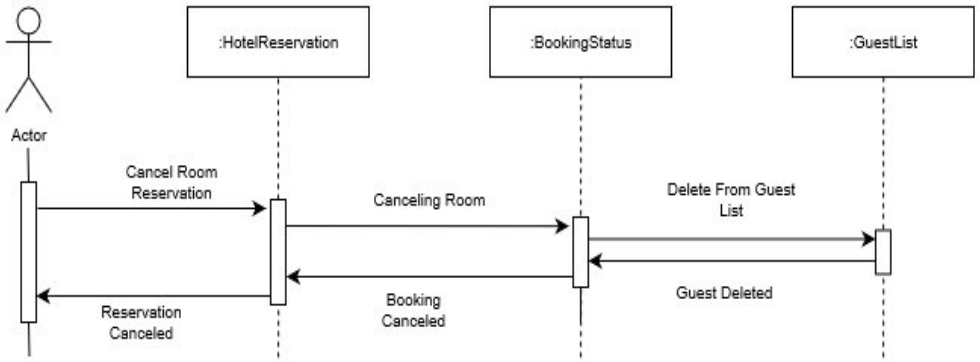
Boundary digunakan untuk menunjukkan ruang lingkup sistem. Berupa sebuah kotak besar yang akan digambarkan sebagai *container* dari *use case*.

4. Relation

Relation digunakan untuk menunjukkan hubungan antara aktor dengan *use case.* Digambarkan dalam bentuk panah.

2.1.5.2.2 *Sequence Diagram*

Menurut Carina Titus (2016:12), *"A Sequence Diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario".*



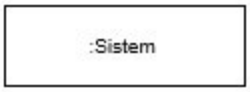
Gambar 2.2 *Sequence Diagram* (Peneliti,2018)

Menurut Whitten dan Bentley (2007:394-396) didalam *UML*, representasi dari prilaku disebut *sequence diagram*. *Sequence* menggambarkan interaksi antar objek didalam dan diluar sekitar sistem. *Sequence* diagram biasa digunakan untuk menggambarkan *scenario* atau rangkaian langkah yang dilakukan sebagai respon dari sebuah *event* untuk menghasilkan *output* tertentu. Berikut ini adalah komponen-komponen yang terdapat dana sequence diagram :

1. Aktor

Digunakan untuk mewakili user dalam berinteraksi denngan interface.

2. Sistem



Digambarkan dengan kotak disertai dengan tanda titik dua (:)

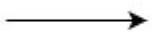
3. Lifelines

Garis *vertical* yang terdapat dibawah aktor dan sistem. *Lifelines* mengindikasi keberadaan sebuah *object* atau aktor dalam basis waktu.

4. Activation Bars

Kotak segi empat yang digambar dibawah *lifelines*. Mengindikasikan sebuah objek akan melakukan sebuah aksi.

5. Input Messages

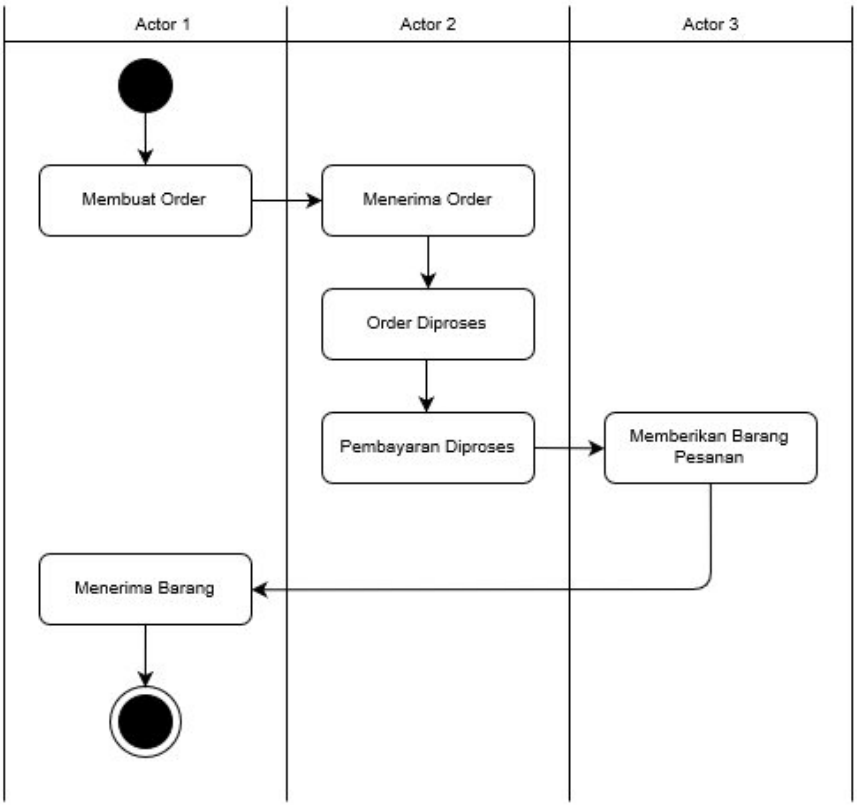


Gambar panah *horizontal* dari aktor menuju sistem yang menunjukkan adanya input pesan.

6. Output Messages

Gambar panah *horizontal* dari sistem menuju aktor.

2.1.5.2.3 *Activity Diagram*



Gambar 2.3 *Activity Diagram* (Peneliti,2018)

Menurut Satzinger, Jackson dan Burd (2012:57), "*an Activity Diagram describes user (or system) activities, the person who does each activity, and the sequential flow of these activities*". Yang terjemahannya adalah Activity Diagram yang menggambarkan beberapa aktivitas pengguna (atau sistem), orang yang melakukan setiap aktivitas dan arus yang berurutan dari aktivitas

Menurut Whitten dan Bentley (2007:390) *activity diagram* adalah diagram yang digunakan untuk menggambarkan alur sebuah proses bisnis, proses-

proses dalam *use case*, dan logika perilaku *object (method).* Berikut adalah elemen-elemen yang terdapat dalam activity diagram :

1. *Initial Node*

Menandakan dimulainya suatu proses. Digambarkan dengan bentuk lingkaran yang *solid.*

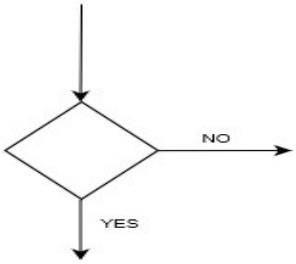
2. *Action*

Digunakan untuk menunjukkan sebuah aktivitas, digambarkan dengan bentuk persegi panjang dengan ujung yang melingkar.

3. *Flow*

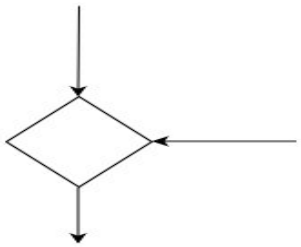
Menunjukkan jalur antar aktivitas, dari sebuah aktivitas ke aktivitas lainnya. Digambarkan dengan bentuk panah.

4. *Decision*



Menunjukkan keadaan kondisional. Digambarkan dengan bentuk wajik dimana ada satu alur yang masuk dan dua atau lebih alur yang mengarah keluar.

5. *Merge*



Menunjukkan adanya penggabungan alur yang terpisah. Digambarkan dengan bentuk wajik dimana ada dua atau lebih alur yang masuk dan satu alur yang mengarah keluar

6. *Fork*

Adanya dua atau lebih *actions* yang berlangsung secara bersamaan.

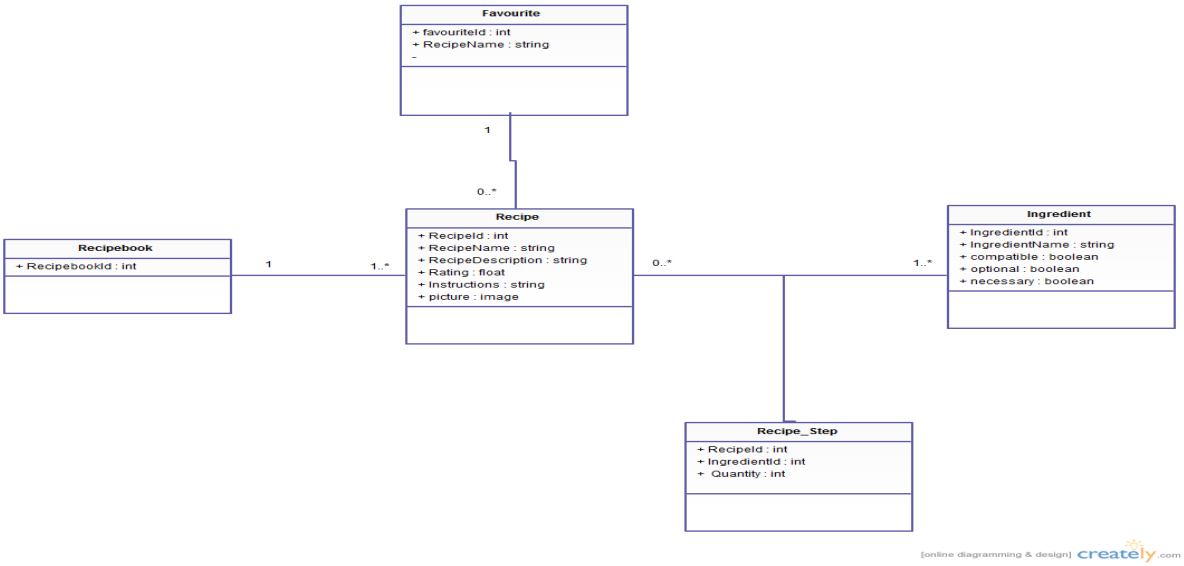
7. *Join*

Semua *actions* yang menuju pada join harus selesai sebelum proses berlanjut.

8. *Activity Final*

Menandakan akhir dari proses. Digambarkan dengan lingkaran *solid* disertai dengan lingkaran kosong dibagian luarnya.

2.1.5.2.4 Class Diagram



Gambar 2.4 *Class Diagram* (Carina,2016)

Menurut Carina Titus (2016,12), *"This is static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes."*

Menurut Whitten dan Bentley (2007:400-409), *Class Diagram* (kelas diagram) yaitu gambaran mengenai objek-objek atau struktur yang menyusun sebuah sistem dengan penggambaran hubungan antar kelas-kelas yang ada. Sebuah kelas diagram disusun oleh tiga bagian, yaitu :

1. Nama Kelas

   Digunakan untuk membedakan antara suatu kelas dan kelas lainnya.
2. Atribut Kelas

   Digunakan untuk menyimpah sebuah *state*
3. Operasi Kelas

   Menyimpan operasi-operasi yang digunakan.

2.1.5.2.5 *Package Diagram*

Package diagram utamanya digunakan untuk mengelompokkan elemen UML yang berlainan secara bersama-sama ke dalam tingkat pembangunan lebih tinggi yaitu berupa sebuah paket, disamping kelas, ada hubungan ketergantungan. Sebagai contoh, jika kita memiliki sistem pendaftaran untuk kantor dokter, mungkin masuk akal untuk kelompok kelas pasien dengan kelas sejarah medis pasien bersama-sama untuk membentuk paket kelas pasien. Selain itu, dapat berguna untuk membuat paket perawatan yang mengandung segala penyait, obat-obatan khusus yang diresepkan untuk mereka.

2.1.5.2.6 Statechart Diagram

Statechart diagram digunakan untuk memodelkan perilaku dinamis satu kelas atau objek. *Statechart diagram* memperlihatkan urutan keadaan sesaat *(state*) yang dilalui sebuah objek. Kejadian yang menyebabkan sebuah transisi dari suatu *state* atau aktivitas kepada yang lainnya.

2.1.2.5.7 *Communication/collaboration diagram*

Collaboration diagram menggambarkan interaksi antara objek seperti *sequence diagram*, namun *collaboration* diagram lebih menekankan pada peran masing-masing objek. Setiap *message* memiliki *sequence* number, dimana *message* dari level tertinggi diberi angka 1.

2.1.2.5.8 *Composite Structure Diagram*

Diagram struktur komposit adalah diagram yang menunjukkan struktur *internal classifier*, termasuk poin interaksinya ke bagian lain dari sistem. Hal ini menunjukkan konfigurasi dan hubungan bagian, yang bersama-sama melakukan perilaku *classifier*. Diagram struktur komposit merupakan jenias diagram dengan struktur yang statis.

2.1.2.5.9 *Object Diagram*

Object diagram merupakan sebuah gambaran tentang objek-objek dalam sebuah sistem pada suatu titik waktu. Karena lebih menonjolkan perintah dari *class*, objek diagram sering disebut juga sebagai diagram perintah.

2.1.2.5.10 *Timing Diagram*

Timing diagram adalah bentuk lain *dari interaction diagram*, dimana fokus dari *timing diagram* adalah waktu. *Timing diagram* akan menunujukkan faktor pembatas waktu diantara perubahan *state* pada objek yang berbeda.

2.1.2.5.11 *Component diagram*

Diagram ini dapat digunakan unutk menggambarkan distribusi fisik dari modul perangkat lunak melalui jaringan. Misalnya, ketika merancang *system client-server*, maka diagram ini akan berguna untuk menunjukkan kelas yang berada pada *server* dan kelas yang berada pada *client*.

2.1.2.5.12 *Deployment Diagram*

Deployment diagram menggambarkan detail bagaimana komponen *dideploy* dalam infrastruktur sebuah sistem, dimana komponen yang akan terletak pada perangkat keras, bagaimana spesifikasi *server* dan menjabarkan hal-hal lain yang bersifar fisik (perangkat keras)

2.1.6 *Eight Golden Rules* (Delapan Aturan Emas)

Menurut Shneiderman (2010:88-89) untuk merancang suatu *user interface* yang baik terdapat delapan aturan emas yang dapat digunakan. Hal yang perlu diperhatikan, yaitu:

a. Berusaha untuk konsisten.

Seluruh desain antarmuka harus konsisten mulai dari jenis tulisan, warna tulisan, bentuk menu yang digunakan, peletakan menu dalam

aplikasi dan sebagainya. Terminologi yang ada dalam *prompt*, menu dan layar bantuan harus identik.

b. Memberikan penggunaan yang *universal*.

Untuk memenuhi kebutuhan *user* yang beragam maka perancangan *user interface* harus mempertimbangkan hal-hal berikut yang meliputi perbedaan umur, keterbatasan fisik dan keberagaman teknologi.

c. Memberikan umpan balik yang informatif.

Untuk setiap aksi yang dilakukan, seharusnya ada suatu sistem yang memberikan umpan balik yang interaktif kepada *user*. Tujuan dari umpan balik tersebut juga berfungsi agar *user* mengetahui apakah aksi yang diberikan sudah benar dan diterima dengan baik oleh sistem.

d. Merancang dialog untuk menghasilkan keadaan akhir.

Dalam merancang komunikasi yang baik dengan pengguna, urutan tindakan harus diatur menjadi bagian awal, tengah dan akhir. *Feedback* yang informatif dapat memberikan kepuasan kepada *user* dan memberikan suatu indikasi untuk mempersiapkan aksi selanjutnya.

e. Menyediakan pencegahan kesalahan.

Apabila *user* melakukan kesalahan, *user interface* harus mendeteksi kesalahan dan memberikan instruksi yang sederhana serta spesifik kepada *user* bagaimana memperbaiki kesalahan tersebut.

f. Memungkinkan pembalikan aksi yang mudah.

Perancancangan *user interface* harus dapat mengembalikan aksi yang dilakukan sebelumnya. Aksi ini dapat menghilangkan kekhawatiran *user*. Contohnya *user* tidak sengaja melakukan aksi yang tidak diinginkan, maka user dapat melakukan pembatalan aksi dengan mengembalikan user interface sebelumnya. Selain itu, aksi ini dapat mendorong user untuk melakukan aksi-aksi yang belum pernah dijelajahi.

g. Menyediakan *internal locus of control*.

Pengguna mempunyai kemampuan untuk mengatur program yang ada di dalam sistem atau *user interface*.

h. Mengurangi beban ingatan jangka pendek.

User interface harus dibuat sederhana karena keterbatasan manusia untuk mengingat informasi dalam jangka pendek. *User Interface* yang sederhana dapat membuat *user* tidak perlu banyak mengingat dalam penggunaanya.

2.1.7 Lima Faktor Manusia Terukur

Menurut Shneiderman (2010:32) untuk merancang sebuah sistem yang membuat *user* lebih nyaman sebelumnya harus memperhatikan 5 (lima) faktor berikut :

1. Waktu untuk belajar

   Waktu yang dipelukan oleh seorang *user* dalam belajar menggunakan aksi-aksi yang ada pada sistem dan relevan untuk mengatur suatu pekerjaan.
2. Kecepatan bekerja

   Waktu untuk melaksanakan atau menyelesaikan suatu target tugas yang diperlukan oleh *user.*
3. Tingkat kesalahan pada user

   Berapa banyak dan apa saja jenis kesalahan yang akan dilakukan oleh *user* dalam menyelesaikan suatu pekerjaan.
4. Daya ingat

   Kemampuan *user* dapat mempertahankan pengetahuannya setelah jangka waktu tertentu. Seperti : beberapa jam kemudian, beberapa hari kemudian, ataupun beberapa minggu kemudian.

5. Kepuasan subjektif

   Jumlah tingkat kepuasan *user*, tertarik dengan aspek-aspek *interface* yang terdapat dalam sistem yang sudah dibuat. Hal tersebut dapat dikatahui dari hasil wawancara atau *survey* tertulis yang meliputi skala kepuasan dan ruang untuk berkomentar secara bebas.

2.1.8 Android

Menurut Setyorini (2014:11) android adalah sebuah sistem operasi untuk perangkat *mobile* berbasis *Linux* yang mencakup sistem operasi, *middleware* dan aplikasi. *Android* menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka. Dalam Kamus Dunia Komputer dan Internet, *open source* adalah perintah-perintah program atau bahasa pemrograman yang tersedia gratis untuk digunakan oleh kalangan luas. Boleh dimodifikasi dan digunakan oleh siapa saja.

2.1.9 GPS (*Global Positioning System*)

Menurut International Journal of Computer Science and Mobile Computing (2014:5), *"GPS is a satellite based system which is used in navigation. The GPS system positional accuracy is reduced by using ionospheric error. Ionospheric delay is a function of Total Electron Content (TEC)"*

Sistem ini terdiri dari tiga bagian : *space segment*, *control segment*, dan *user segment*. Tiga bagian tersebut dapat dijelaskan sebagai berikut :

1. *Space Segment*

   Bagian ini terdiri dari kumpulan satelit yang mentransmisi sinyal radio kepada *user* dengan menggunakkan satelit luar angkasa yang diatur oleh Departemen Pertahanan Amerika Serikat.
2. *Control Segment*

   Bagian ini terdiri dari jaringan global dan fasilitas untuk melacak satelit GPS, moitor *transmisi*, melakukan analisis, serta mengirim perintah dan data kembali ke satelit.
3. *User Segment*

   User segment terdiri dari peralatan penerima GPS, yang menerima sinyal dari satelit GPS dan menggunakan informasi yang ditransmisikan untuk menghitung posisi tiga dimensi dan si *user* dan waktu.

   Dari ketiga bagian tersebut, angkatan udara Amerika Serikat mengembangkan, menjaga, dan mengoperasikan *space segment* dan *control segment*

2.1.10 *Project Management Body Of Knowledge*

Gambar 2.5 The Five PMBOK Process Group (PMBOK Guide 5th, 2013)

Project Management Body of Knowledge (PMBOK) adalah kumpulan proses dan bidang pengetahuan umun diterima sebagai praktek terbaik dalam disiplin manajemen proyek. Sebagai standard yang diakui secar internsional (IEEE Std 1490-2003) memberikan dasar-dasar manajemen proyek, terlepas dari jenis proyek baik itu konstruksi, perangkar lunak, teknik, otomotif juga konstruksi. PMBOK mengakui lima kelompok proses dasar dan Sembilan bidang pengetahuan khas dari hampir semua proyek. Konsep dasar yang berlaku untuk proyek-proyek, program dan kegiatan. Kelima kelompok dasar tersebut berupa :

1. Memulai
2. Perencanaan
3. Pelaksanaan
4. Pemantauan dan Pengendalian
5. Penutupan

Proyek integrasi manajemen yang berfokus pada pengembangan perencanaan, eksekusi dan kontrol perubahan. Efektif integrasi proses yang diperlukan untuk mencapai tujuan proyek.

Proyek lingkup manajemen yang menyediakan jaminan bahwa proyek didefinisikan dengan akurat dan lengkap dan semua akan sesuai dengan rencana. Mendefinisikan dan mengontrol apa yang dapat dan tidak dapat dilakukan didalam proyek.

Proyek manajemen waktu, penting untuk mengembangkan, memantau dan mengolah jadwal proyek. Termasuk proses yang diperlukan untuk menyelesaikan proyek dalam waktu yang telah ditentukan.

Biaya proyek manajemen, jaminan *budget* dalam mengembangkan dan melengkapi dalam persetujuan pembuatan proyek. Perencanaan, memperkirakan, penganggaran dan pengendalian biaya untuk memastikan proyek tersebut dapat diselesaikan dalam anggaran yang disetujui.

2.1.11 Perancangan Sistem Informasi Berbasis *Objek Oriented*

2.1.11.1 Objek Oriented

Menurut Satzinger, Jackson dan Burd (2012:290), "*Object-Oriented an approach to system developments that views an information system as a collection of interacting object that work together to accomplish task*"

Yang artinya *Object-Oriented* adalah pendekatan pengembangan sistem yang memandang sistem informasi sebagai kumpulan objek yang saling berinteraksi dan bekerja sama untuk menyelesaikan tugas.

2.1.11.2 *Object Oriented Analysis* (OOA)

Menurut Shelly dan Rosebland (2012:21), "*Object Orientes Analysis combines data and the process that act on the data into things called object*"

Yang artinya *Object Oriented Analysis* adalah berorientasi untuk menggabungkan data dan proses yang berhubungan dengan analisis sistem ke hal-hal yang terkait dengan penggunaan objek.

2.1.11.2 *Object Oriented Design* (OOD)

Menurut Satzinger, Jackson dan Burd (2012;95), "*Object Oriented Design is a process by which a set of detailed models are build and then used by the programmers to write and test the new system*"

Yang artinya *Object Oriented Design* adalah suatu proses dimana serangkaian model rinci dibangun dan kemudian digunakan oleh *programmer* untuk menulis dan menguji sistem baru.

2.1.11.3 *Object Oriented Analysis and Design* (OOAD)

Menurut Dennis, Wixom dan Roth (2009,48), "*our method uses object and classes as its key concepts and builds on four general principles for analysis and design : model the system's context, emphasize architecturel considerations, reuse patterns that express well established design ideas, and tailor the method to each development situation*"

Yang *artinya Object Oriented Analysis and Design* adalah metode yang menggunakan *object* dan *class* sebagai kunci utama dan membangun atas dasar empat prinsip dalam menganalisa dan merancang : yakni ruang lingkup sistem model, konsiderasi arsitektur sistem, penggunaan ulang pola yang dapat menggambarkan ide-ide perancangan, menyatukan ide tersebut ke dalam situasi pengembangan. Dengan pemahaman :

- *Object* : "*an entity with identity, state, and behaviour*", yang artinya object adalah sebuah entitas yang memiliki identitas, status dan perilaku.
- *Class* : "*A description of an collection of objects sharing structure, behavioural pattern, and attribute*" yang artinya *class* adalah suatu deskripsi dari kumpulan *object* yang memiliki struktur, pola perilaku, dan pangkat yang sama

Menurut Bodnar dan Hopwood (2009:412) *Objek Oriented Analysis and Design* adalah pendekatan analisis dan perancangan sistem yang dimulai dengan deskripsi yang sangat umum dari sebuah sistem tertentu dan kemudian terus melalui serangkaian langkah-langkah logis yang rinci dan masing-masing meningkat secara rinci.

Dari beberapa pengertian diatas, dapat disimpulkan bahwa *Object Oriented Analysis and Design* adalah pendekatan analisis dan perancangan sistem yang menggunakan *object* dan *class* sebagai kunci utama dimulai dangan deskripsi yang sangat umun dari suatu sistem tertentu kemudian melalui serangkaian langkah-langkah logis yang rinci.

2.2 TEORI YANG BERKAITAN DENGAN MANAJEMEN PROYEK

2.2.1 Manajemen Proyek

Menurut Heizer dan Render (2012:92) Manajemen proyek merupakan suatu pemikiran tentang manajemen yang ditujukan untuk mengelola kegiatan yang berbentuk proyek. Manajemen proyek adalah tantangan yang sulit bagi manajer operasi. Risiko pada manajemen proyek sangat tinggi. Kelebihan biaya dan keterlambatan yang tidak diperlukan terjadi karena penjadwalan dan pengendalian yang buruk. Pada umumnya kegiatan manajemen berfokus pada kegiatan perencanaan, pengorganisasian, dan pengendalian dari proses yang akan berlangsung seperti proses produksi atau penghantaran jasa. Manajemen proyek memiliki perbedaan dari kegiatan manajemen pada umumnya, karena sebuah proyek memiliki batasanbatasan seperti adanya batasan ruang lingkup dan biaya untuk suatu kegiatan yang penting, yang dibatasi oleh waktu. Ada empat komponen penting dari sebuah proyek, yaitu ruang lingkup (*scope*), waktu, biaya dan kualitas. Keempat komponen tersebut menjadi batasan dalam pelaksanaan proyek. Dengan kata lain, dapat dikatakan bahwa kriteria yang harus dipenuhi dari produk yang dihasilkan dari proyek meliputi kriteria atau batasan waktu, batasan ruang lingkup, batasan biaya, dan batasan kualitas. Jadi empat keharusan dalam sebuah proyek adalah (Dimyati & Nurjaman, 2014:41-42):

1. Diselesaikan dan diserahkan dengan tepat waktu
2. Cukup dibiayai dengan dana yang telah ditentukan
3. Sesuai dengan ruang lingkup yang disepakati
4. Memiliki kualitas hasil sesuai dengan kriteria yang disepakati antara pelaksana dan pemberi proyek.

2.2.2 Proyek Manajer

A Project manager is the person assigned by the performing organization to lead the team that is responsible for achieving the project objectives.

Menurut Marchewka (2017), Seorang manajer proyek adalah orang yang ditugaskan oleh organisasi yang melakukan untuk memimpin tim yang bertanggung jawab untuk mencapai tujuan proyek .

2.2.3 Profil Kegiatan Proyek

Kegiatan proyek dapat diartikan sebagai suatu kegiatan sementara yang berlangsung dalam jangka waktu terbatas, dengan alokasi sumbar daya tertentu dan dimaksudkan untuk melaksanakan tugas yang sasarannya telah digariska dengan jelas. Tugas tersebut dapat berupa pembangunan gedung, membuat produk baru atau melakukan penelitian dan pengembangan.

Dari pengertian diatas terlihat bahwa ciri pokok dari proyek adalah :

a. Memiliki tujuan yang khusus, produk akhir atau hasil kerja akhir
b. Jumlah biaya, sasaran jadwal serta kriteria mutu dalam proses mencapai tujuan diatas yang telah ditentukan
c. Bersifat sementara, dalam arti umumnya dibatasi oleh selesainya tugas. Titik awal dan akhir ditentukan dengan jelas
d. Nonrutin, tidak berulang-ulang. Jenis dan insensitas kegiatan berubah sepanjang proyek berlangsung.

2.2.4 Sasaran Proyek dan Tiga Kendala

Didalam proses mencapai tujuan tersebut telah ditentukan batasan sebagai berikut :

a. Anggaran

   Proyek harus diselesaikan dengan biaya yang tidak melenihi anggaran. Untuk proyek-proyek yang melibatkan dana dalam jumlah besar dan jadwal bertahun-tahun, anggaran bukan hanya ditentukan untuk total proyek, tapi dipecah lagi berdasarkan komponen-komponennya, atau perperiode tertentu yang jumlahnya disesuaikan dengan keperluan. Dengan demikian, penyelesaian bagian-bagian proyek pun harus memenuhi sasaran anggaran per periode.

b. Jadwal

   Proyek harus dikerjakan sesuai dengan kurun waktu dan tanggal akhir yang telah ditentukan. Bila hasil akhir adalah produk baru, maka penyerahannya tidak boleh melewati batas waktu yang ditentukan.

c. Mutu

   Produk atau hasil kegiatan proyek harus memenuhi spesifikasi dan kriteria yang disyaratkan. Sebagai contoh, bila hasil kegiatan proyek berupa instalasi pabrik,

maka kriteria yang harus dipenuhi adalah pabrik harus mampu beroperasi secara memuaskan dalam kurun waktu yang telah ditentukan. Jadi, memenuhi persyaratan mutu berarti mampu memnuhi tugas yang dimaksudkan atau sering disebut sebagai *fit for intended use.*

Ketiga batasan diatas disebut juga sebagai tiga kendala *(triple constraint)*

2.2.5 Konsep dan Fungsi Manajemen Proyek

"Konsep manajemen proyek merupakan buah pemikiran tentang manajemen yang ditujukan untuk mengelola kegiatan yang berbentuk proyek. Perumusannya disusun sedemikian rupa sehingga dapat menghadapi dan mengakomodir perilaku dan dinamika yang melekat pada kegiatan proyek." (Iman Soeharto, 2001:76)

2.2.6 Fungsi Dasar Manajemen Proyek

Fungsi dasar manajemen proyek terdiri dari pengelolaan-pengelolaan lingkup kerja, waktu, biaya, dan mutu. Pengelolaan aspek-aspek tersebut dengan benar merupakan kunci keberhasilan penyelenggaraan proyek.

a. Pengelolaan Lingkup Proyek

   Lingkup proyek adalah total jumlah kegiatan atau pekerjaan yang harus dilakukan untuk menghasilkan produk yang diinginkan oleh proyek tersebut. Dalam hubungan ini dokumen yang berisi batasan lingkup proyek yang memuat kuantitas, kulitas dan spesifikasi amatlah penting.

   b. Pengelolaan Waktu

      Waktu atau jadwal merupakan sasaran utama proyek. Keterlambatan akan mengakibatkan berbagai bentuk kerugian, misalnya penambahan biaya, kehilangan kesempatan untuk memasuki pasar, dan lain-lain. Pengelolaan waktu meliputi perencanaan, penyusunan, dan pengendalian jadwal. Salah satu teknik yang spesifik untuk maksud tersebut adalah mengelola cadangan waktu.

   c. Pengelolaan Biaya

Pengelolaan biaya meliputi segala aspek yang berkaitan dengan hubungan antara dana dan kegiatan proyek. Mulai dari proses memperkirakan keperluan jumlah dana, mencari dan memilih sumber serta macam pembiayaan, perencanaan serta pengendalian alokasi pemakaian biaya sampai kepada akuntansi dan administrasi pinjaman dan keuangan. Agar pengelolaan bisa efektif terutama dalam aspek perencanaan dan pengendalian biaya proyek maka disusun bermacam-macam teknik dan metode. Misalnya teknik menyusun anggaran biaya proyek, identifikasi *varians*, konsep nilai hasil dan lain-lain.

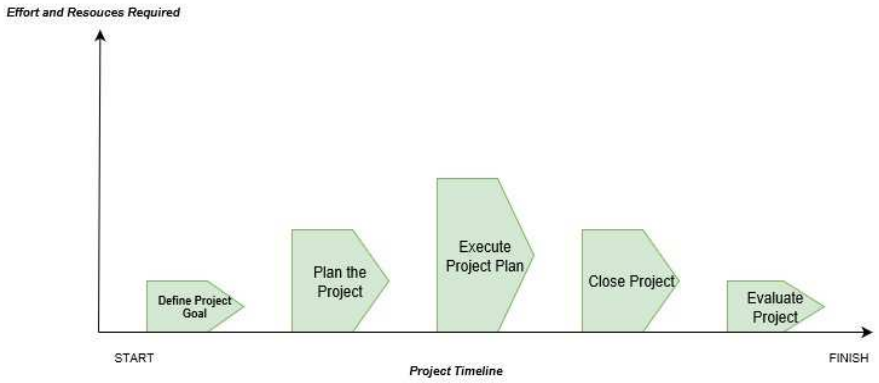
d. Mengelola Kualitas Mutu

Mutu, dalam kaitannya dengan proyek dartikan sebagai sebuah hasil yang telah memenuhi syarat untuk penggunaan yang telah ditentukan (*fit for intended use*). Agar suatu produk atau jasa hasil proyek memenuhi syarat penggunaan, diperlukan suatu proses yang panjang dan kompleks, mulai dari mengkaji apa saja, syarat dan penggunaan yang dikehendaki oleh pemilik proyek atau pemesan produk (klien), menjabarkan persyaratan tersebut menjadi kriteria dan spesifikasi, serta menuangkannya menjadi gambar-gambar instalasi atau produksi. Juga termasuk daya serta jadwal, sampai kepada merencanakan dan mengendalikan aspek mutu pada tahap implementasi atau produksi.

2.2.7 Milestone

Milestone adalah suatu bagian item pekerjaan yang dibuat seolah-olah menjadi *temporary finish* atau hasil jadi sementara atas sekelompok atau serangkaian pekerjaan-pekerjaan yang menjadi bagian dari *schedule* utama. *Item* pekerjaan yang dijadikan *milestone* haruslah *item* pekerjaan yang dianggap menjadi bagian penting sebelum melanjutkan pekerjaan berikutnya atau berpengaruh atas kelangsungan pekerjaan berikutnya.

2.2.8 *Project Life Cycle*



Gambar 2.7 A Generic Project Life Cycle (Marchewka,2017)

Terlepas dari apakah suatu proyek besar atau kecil, proyek disusun dalam fase sekuensial untuk membuat proyek lebih mudah dikelola dan mengurangi risiko. *Phase exit*, *stage gates* atau *kill points* adalah tinjauan akhir fase yang memungkinkan organisasi untuk mengevaluasi kinerja proyek dan untuk mengambil tindakan sesegera mungkin untuk memperbaiki masalah atau bahkan membatalkan proyek. *Fast tracking*, atau memulai tahap berikutnya sebelum fase saat ini selesai terkadang dapat mempercepat selalinya suatu proyek, tetapi fase yang tumpang tindih dapat berisiko dan seharusnya hanya dilakukan ketika risiko dianggap dapat diterima.

Gambar 2.7 menyediakan siklus hidup umum yang menggambarkan tahapan umum atau tahapan yang dibagi oleh sebagian besar proyek. Terlepas apakah Anda sedang membangun rumah, mendesain produk pelanggan baru, mengembangkan *website*, atau meluncurkan pesawat ruang angkasa ke Mars, setiap proyek dapat dikelola menggunakan siklus hidup proyek yang sama.

1. *Defining Project Goal*

   Semua proyek memiliki permulaan. Meskipun sebuah proyek dimulai ketika seseorang muncul dengan ide baru untuk mungkin produk, layanan, atau sistem baru, langkah pertama dalam memulai proyek haruslah untuk menentukan tujuan proyek. Sasaran proyek harus menyatakan secara eksplisit nilai bisnis yang diimpikan proyek karena proyek adalah investasi organisasi yang memerlukan

waktu dan sumber daya dan melibatkan risiko. Tujuan yang terdefinisi dengan baik akan menentukan harapan *stakeholders* dan mendorong fase lain dari proyek. Sasaran proyek juga harus menjawab pertanyaan : bagaimana kita tahu jika proyek ini berhasil diberikan waktu, uang, dan sumber daya yang diinvestasikan? Setelah tujuan proyek telah ditetapkan dengan jelas, tujuan tersebut harus disetujui oleh para *stakeholders* proyek sebelum proyek dapat memulai tahap perencanaan.

2. *Plan Project*

   Tujuan proyek memberikan arahan untuk merencanakan proyek, jika tidak, itu akan seperti mengendarai mobil tanpa tujuan dalam pikiran. Rencana proyek mendefinisikan:

   a. *Project objectives* – Sasaran proyek mencakup ruang lingkup (pekerjaan proyek), jadwal, *objectives* mendukung tujuan proyek dengan mendefinisikan pekerjaan apa yang harus diselesaikan, kapan perlu diselesaikan, berapa banyak biaya yang harus dikeluarkan, dan apakah pekerjaan dapat diterima oleh *stakeholders*.

   b. *Resources* – Sumber daya dibutuhkan untuk menyelesaikan pekerjaan proyek dan mencakup hal-hal seperti orang, fasilitas, dan teknologi

   c. *Controls* – Inti dari mengelola sebuah proyek termasuk memastikan bahwa tujuan dan sasaran proyek terpenuhi dan sumber daya digunakan secara efisien dan efektif. Selain itu, risiko, perubahan, dan komunikasi di antara para *stakeholders* proyek harus dikelola secara proaktif di seluruh proyek.

3. *Execute project plan*

   Persetujuan atas rencana proyek (*project plan*) diperlukan sebelum pindah ke tahap eksekusi. Meskipun tahap *project plan* menguraikan kemajuan proyek yang diharapkan atau direncanakan, *fase execute project plan* berkonsentrasi pada desain, pengembangan, dan penyajian produk, layanan atau sistem proyek. Selain itu, kontrol yang didefinisikan dalam tahap perencanaan sekarang memungkinkan para *stakeholders* proyek untuk membandingkan kemajuan yang direncanakan proyek dengan kemajuan aktual dalam hal pekerjaan yang selesai tepat waktu, sesuai anggaran, dan dalam standar kualitas sehingga mencapai nilai bisnis yang

diharapkan. Pada akhir fase ini, tim mengimplementasikan atau menyajikan produk, layanan atau sistem yang telah selesai kepada organisasi.

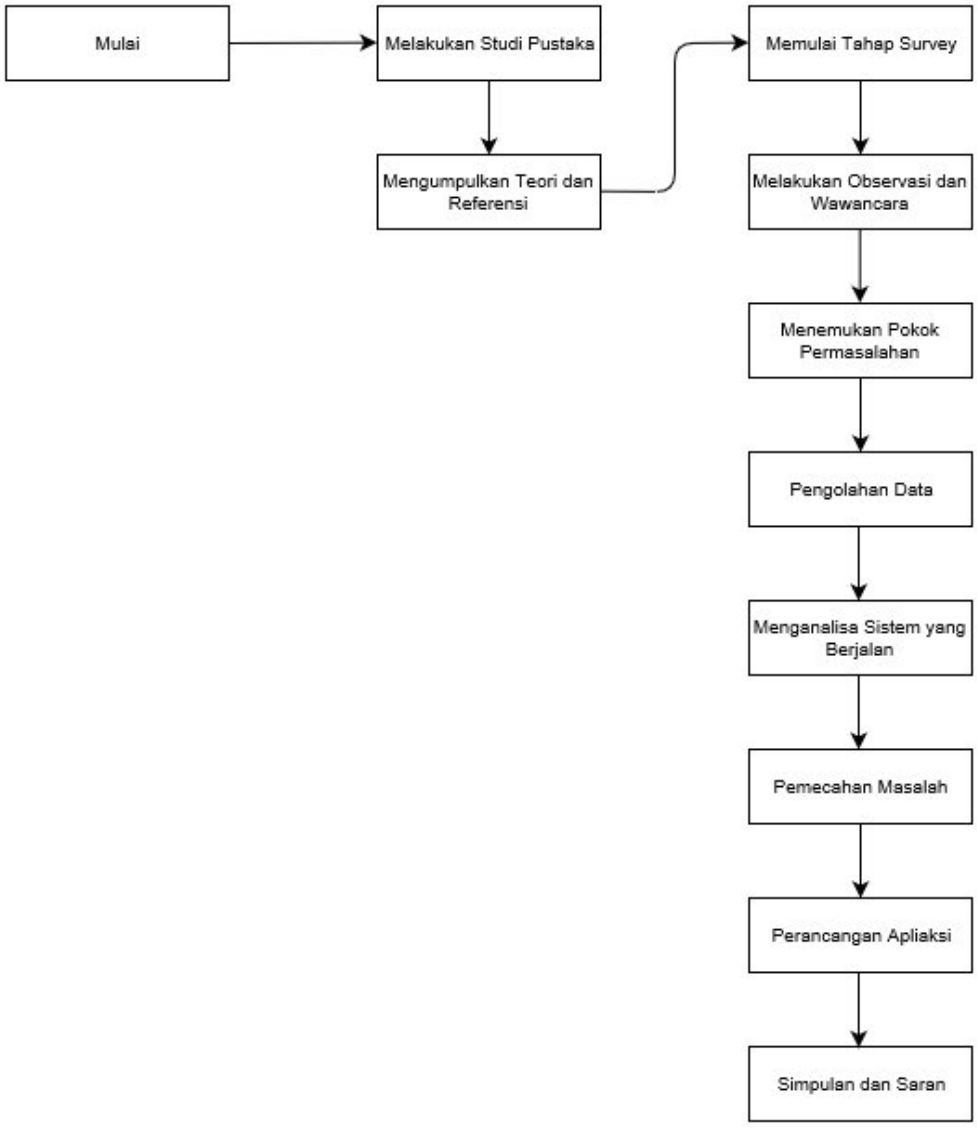
4. *Close and Evaluate Project*

   Sebuah proyek harus memiliki tujuan yang pasti. Fase terakhir memastikan bahwa semua pekerjaan diselesaikan sesuai kesepakatan oleh tim, sponsor, atau pemangku kepentingan lainnya. Namun, proyek dan tim proyek harus dievaluasi selama masa tinjauan untuk menentukan apakah tujuan proyek yang ditetapkan dalam fase awal tercapai. Selain itu, praktik terbaik berdasarkan pengalaman dan pelajaran yang didapat harus didokumentasikan dan tersedia untuk proyek di masa depan.

   Selain itu, sebagian besar proyek memiliki karakteristik berikut:

   a. Sebuah upaya (pekerjaan), dalam hal biaya dan tingkat *staf*, rendah pada awal proyek, tetapi kemudian meningkat ketika pekerjaan proyek sedang dilakukan, dan kemudian menurun pada akhir saat proyek selesai.
   b. Resiko dan ketidakpastian berada pada level tertinggi pada awal project.
   c. Untuk banyak proyek, kemampuan *stakeholders* untuk memengaruhi fitur atau fungsi produk paling tinggi pada awal proyek. Biaya mengubah fitur atau fungsi produk dan memperbaiki kesalahan menjadi lebih mahal seiring dengan kemajuan proyek.

2.3 Kerangka Berfikir



Gambar 2.8 Kerangka Berfikir