BAB II

KAJIAN TEORI

Kajian teori dalam penyusunan penelitian ini sangat diperlukan untuk menunjang atau memperdalam pemahaman terhadap informasi-informasi yang disajikan. Berikut teori-teori yang digunakan:

A. Absensi

Absen menurut kamus besar Bahasa Indonesia merupakan tidak hadirnya seseorang dalam sebuah instansi. Sedangkan absensi biasa disebut sebagai proses penandaan atau pencatatan waktu hadir seseorang dalam sebuah dokumen yang dibuat sebagaimana mestinya guna sebagai acuan dalam menentukan sebuah keputusan dalam lingkup penilaian. Jenis-jenis absen terbagi menjadi dua yaitu:

1. Absensi Manual

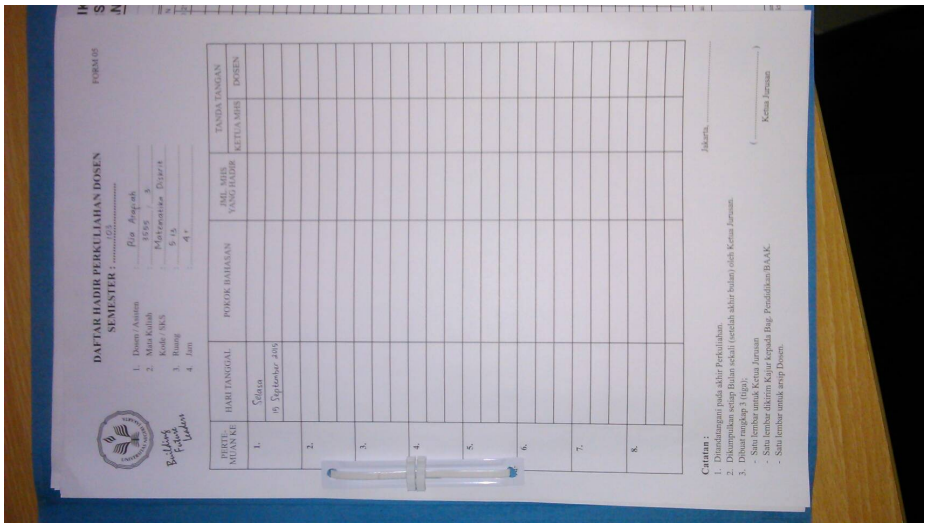
   Absensi manual adalah cara pengentrian kehadiran dengan cara menggunakan pena (tanda tangan).

2. Absensi non Manual (dengan menggunakan alat)

   Absensi non manual adalah suatu cara pengentrian kehadiran dengan menggunakan sistem terkomputerisasi, bisa menggunakan kartu dengan *barcode*, *finger print* ataupun dengan mengentrikan nomor identitas dan sebagainya.

Jenis absensi yang diterapkan di Universitas Negeri Jakarta ialah absensi manual. Dosen menunjuk salah satu mahasiswa sebagai penanggung jawab kelas dan bertugas mengambil *form* absensi dan mengembalikannya. Proses absensi dilakukan secara bergilir pada saat mata kuliah berlangsung. Terdapat dua *form* pada satu map

absen, yaitu *form* 05 dan *form* 06. *Form* 05 berisikan tanggal pertemuan, materi, jumlah mahasiswa, tanda tangan dosen dan penanggung jawab kelas. *Form 06* merupakan daftar absensi mahasiswa setiap pertemuan.



DAFTAR HADIR PERKULIAHAN DOSEN
SEMESTER : .....

FORM 05

1. Dosen / Asisten
2. Mata Kuliah
3. Kode / SKS
4. Ruang
5. Jam

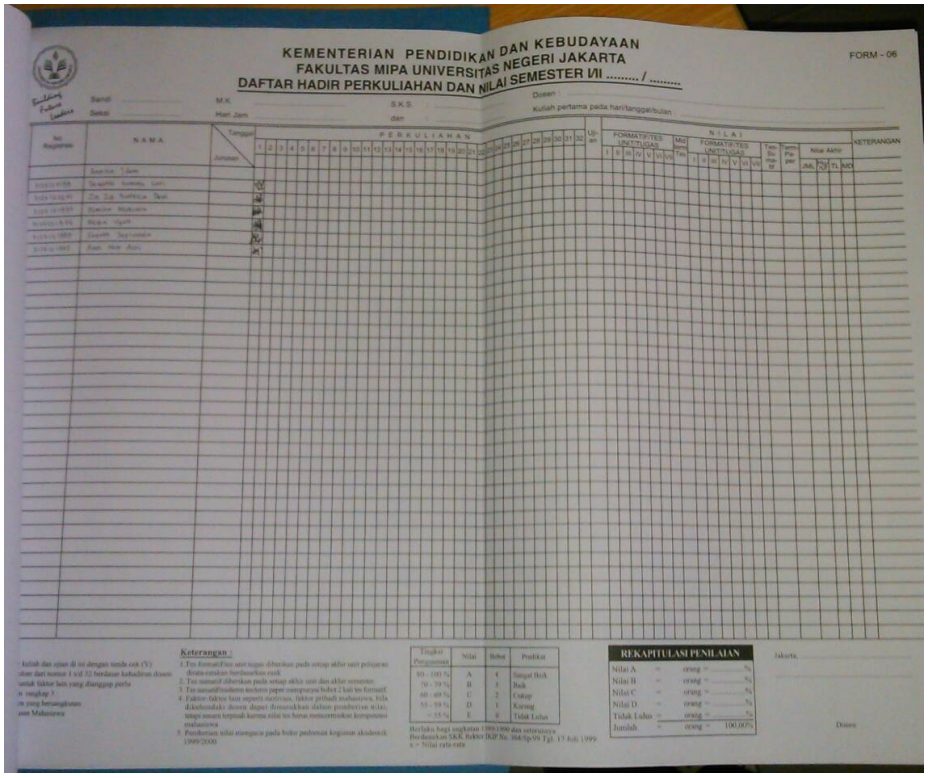
PERTE-MUAN KE	HARI TANGGAL	POKOK BAHASAN	PARAF MHS YANG HADIR	TANDA TANGAN KETUA MHS	TANDA TANGAN DOSEN
1.	Selasa 19 September 2006				
2.					
3.					
4.					
5.					
6.					
7.					
8.					

Jakarta, ...

(.....)
Ketua Jurusan

Catatan :
1. Ditandatangani pada akhir Perkuliahan.
2. Dikumpulkan setiap Bulan sekali / setelah akhir bulan oleh Ketua Jurusan.
3. Dibuat rangkap 3 (tiga):
   - Satu lembar untuk Ketua Jurusan
   - Satu lembar untuk Kasubag Pengajaran
   - Satu lembar untuk arsip Dosen

Gambar 2.1: *Form* 05 Absensi Mahasiswa



KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN
FAKULTAS MIPA UNIVERSITAS NEGERI JAKARTA
DAFTAR HADIR PERKULIAHAN DAN NILAI SEMESTER I/II ...../.....

FORM - 06

Sandi : .....
Sekst : .....
M.K : .....
Hari Jam : .....
S.K.S : .....
dan : .....
Dosen : .....
Kuliah pertama pada hari/tanggal/bulan : .....

No Registrasi	NAMA	Tanggal	PERKULIAHAN 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32	Jumlah	Ujian	NILAI: FORMATIF/TES UNSTTUGAS I II III IV V VI	MID Test	FORMATIF/TES UNSTTUGAS I II III IV V VI	UTS	UAS	Nilai Akhir: ANGKA	HURUF	KETERANGAN

Keterangan :
1. Tes formatif/tes unit tugas diberikan pada setiap akhir unit pelajaran dalam rentang berdasarkan unit.
2. Tes sumatif diberikan pada setiap akhir unit dan akhir semester.
3. Tes sumatif/midtest/uts/uas paper/ujian praktek / lisan/ tes formatif.
4. Faktor-faktor lain seperti motivasi, faktor pribadi mahasiswa, tida dikelompokkan dalam dapat dimasukkan dalam pemberian nilai, tetapi secara terpisah karena nilai ini harus menacerminkan kompetensi mahasiswa
5. Pemberian nilai mengacu pada buku pedoman kegiatan akademik 1999/2000

Tingkat Penguasaan	Nilai	Bobot	Predikat
90 - 100 %	A	4	Sangat Baik
70 - 79 %	B	3	Baik
60 - 69 %	C	2	Cukup
55 - 59 %	D	1	Kurang
< 55 %	E	0	Tidak Lulus

Berlaku bagi angkatan 1993/1994 dan seterusnya berdasarkan SK. Rektor IKIP No. 384/Sp.93 Tgl. 17 Juli 1993
x = Nilai yang ada

REKAPITULASI PENILAIAN

Nilai A	=	orang = ...	%
Nilai B	=	orang = ...	%
Nilai C	=	orang = ...	%
Nilai D	=	orang = ...	%
Tidak Lulus	=	orang = ...	%
Jumlah	=	orang =	100.00%

Jakarta, ...
Dosen

Gambar 2.2: *Form* 06 Absensi Mahasiswa

B. Teori Pengembangan Perangkat Lunak

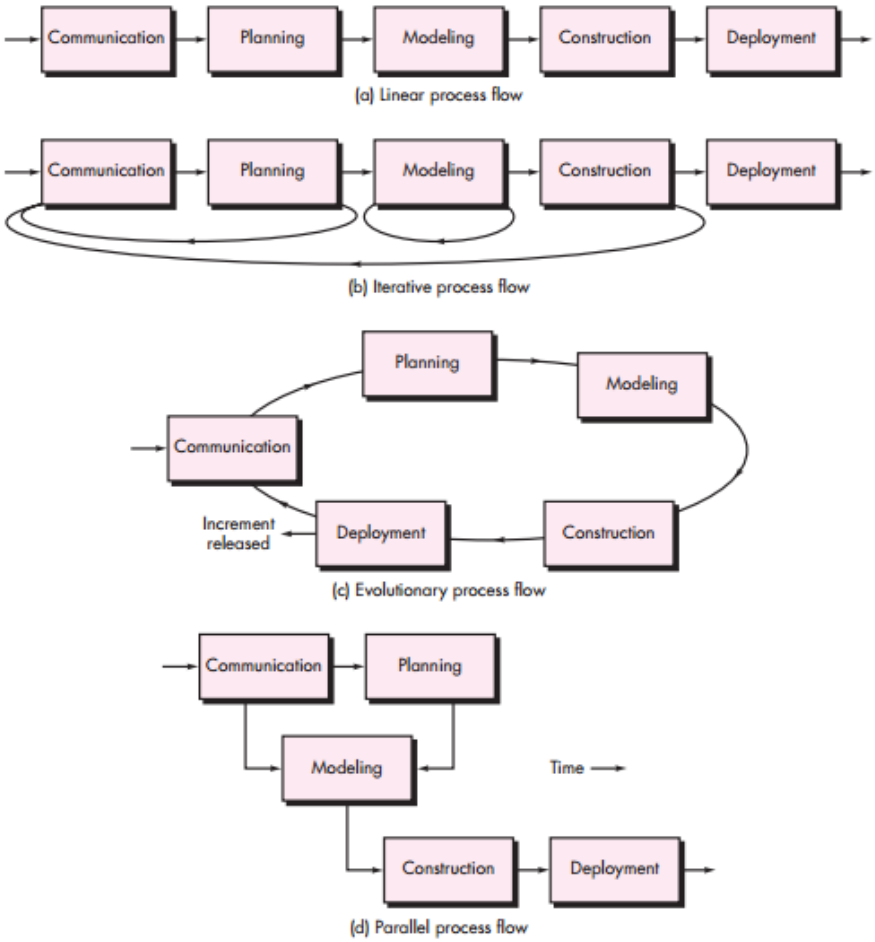
Dalam mengembangkan sebuah perangkat lunak yang ekonomis, handal, dan bekerja secara efisien dibutuhkan serangkaian langkah-langkah yang dapat diprediksi sebagai acuan dalam mengembangkan *software*. Proses pengembangan sebuah aplikasi dapat dikatakan sebagai kumpulan aktivitas kerja, tindakan dan tugas yang dilakukan untuk mengerjakan sebuah produk. Masing- masing aktivitas, tindakan dan tugas memiliki kerangka *(framework)* atau model yang mendefinisikan suatu hubungan proses dengan satu sama lain. *Framewok* dapat disebut juga sebagai tahapan pengembangan. [7] Pada umumnya terdapat lima tahapan dalam sebuah proses, yaitu: *communication, planning, modeling, construction, and deployment*. Terdapat lima aliran atau *flow* dalam teori rekayasa perangkat lunak, yaitu:

1. *Linear process flow* : Lima *framework* dilakukan secara berurutan, dimulai dari *communication* hingga mencapai puncaknya *deployment*.

2. *Iterative process flow* : Mengulangi satu atau lebih *framework* sebelum melanjutkan ke tahap berikutnya.

3. *Evolutionary process flow* : Tahapan *framework* dilakukan secara melingkar.

4. *Parallel process flow* : Melakukan satu *framework* atau *framework* lainnya secara pararel.



Gambar 2.3: *Process Flow* Rekayasa Perangkat Lunak

Menurut Pressman, selain *process flow* terdapat beberapa model pengembangan dalam *software process* yang juga menjadi acuan dalam mengembangkan perangkat lunak, yaitu: *The Waterfall Model, V-Model, Incremental Process Model, Spiral Model, RAD (Rapid Application Development) Model*, dan lainnya. Pemilihan model pengembangan juga bergantung pada *process flow* yang digunakan. Dalam pengembangan aplikasi ini, peneliti menggunakan *Evolutionary process flow* dan *Spiral Model*. Seperti pada penelitian *Insitute of Technology Hydreabad* [6] yang mengangkat tema *Mobile Phone Conversion and Location Application for Android* menggunakan Spiral Model sebagai model pengembangan *software*. Penelitian ini mengemukakan

bahwa setiap fase atau *cycle* dimulai dengan tujuan desain dan diakhiri dengan klien yang menerima progress kemajuan. Analisis dan upaya pengaplikasian juga diterapkan pada setiap tahap proyek. Spiral Model menjadi acuan dalam pengembangan aplikasi absen ini dikarenakan fleksibilitas yang terdapat pada model ini. Fleksibilitas yang dimaksud ialah jika terjadi kesalahan dalam pengembangan, penulis dapat kembali ke tahapan yang terdapat kesalahan dan memperbaiki kesalahan tersebut. Model ini juga dapat menangani perubahan yang sering terjadi pada pengembangan perangkat lunak.[7]

Spiral Model adalah bagian dari *Evolutionary process model* yang menggabungkan aspek pengendalian dari *Model Iterative* dan aspek sistematis dari *Model Waterfall*. Penggabungan kedua model ini dapat mengatasi kelemahan yang terdapat pada model *iterative* dan model waterfall seperti *user* yang sulit menyatakan analisis kebutuhan secara rinci pada tahapan awal, kesalahan kecil akan menjadi masalah besar jika tidak diketahui sejak awal pengembangan yang berakibat pada tahapan selanjutnya dan proses pengembangan tidak dapat dilakukan secara berulang sebelum terjadinya suatu produk. Pada spiral model terdapat lima tahapan pengembangan perangkat lunak yaitu:

1. Komunikasi *(Communication)*

   Tahapan dimana peneliti berkomunikasi mencari tahu tentang kebutuhan user atau biasa disebut dengan *User Requirement*. *User Requirement* dapat diperoleh dengan metode wawancara, kuesioner ataupun observasi. *User requirement* merupakan hal yang penting dalam sistem informasi, agar dapat menentukan batasan-batasan dan kebutuhan dalam suatu sistem.

2. Perencanaan *(Planning with Estimation, Scheduling)*

   Pada tahap ini peneliti menganalisa dan merencanakan hasil dari *User Requirement* yang telah diperoleh.

3. Pemodelan dan Desain *(Modelling and Design)*

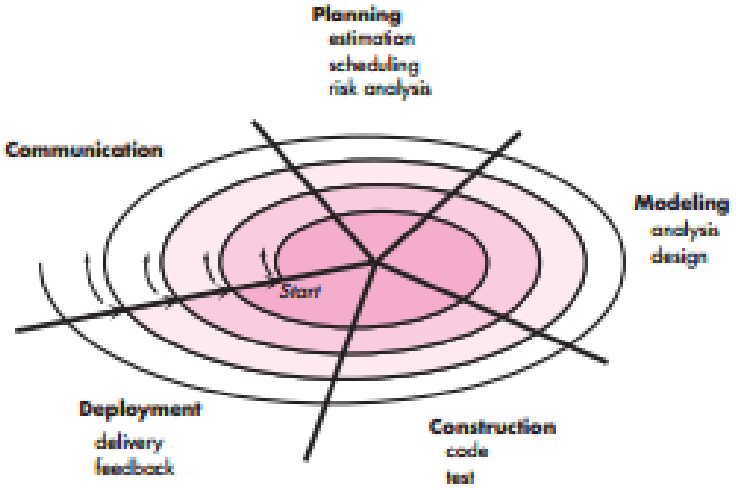
   Tahapan ini berisi dengan diagram UML, tampilan awal *(mockup)* dan *user interface* yang telah didesain sedemikian rupa.

4. Konstruksi dan Pembangunan *(Code and Test)*

   Tahap ini peneliti mulai membangun sistem dengan berbagai langkah-langkah dalam pelaksanaannya seperti membuat basis data, implementasi desain sistem dan implementasi pemrograman. Setelah selesai dalam proses *coding*, peneliti akan melakukan *testing* untuk menguji sistem.

5. *Deployment (Delivery Feedback)*

   Tahapan ini hanya akan dilaksanakan jika sistem ditanam di dalam server dan akan dilakukan penyebaran untuk jangka panjang.



Gambar 2.4: Spiral Model

Model spiral dapat dibagi dalam beberapa *cycle*, satu *cycle* bisa hanya berisi satu tahapan atau hingga lima tahapan sekaligus. Jumlah *cylce* dapat ditentukan oleh kesepakatan *user* dan peneliti. [7]

C. Android

Android merupakan perangkat bergerak pada sistem operasi untuk telepon seluler yang berbasis linux. *Platform* Android dibuat terbuka sehingga sebuah aplikasi dapat membuat panggilan, mengambil gambar, memainkan musik dan lain-lain. Android juga merupakan *open source* dimana para *developer* dapat secara bebas memperluas teknologi baru. Aplikasi android dibuat untuk memiliki akses yang sama terhadap kemampuan sebuah telepon dalam menyediakan layanan dan aplikasi yang luas terhadap para pengguna. Android juga dapat membangun aplikasi yang inovatif dengan menggabungkan informasi yang ada di web dan dengan data yang ada pada ponsel pengguna. *Platform* Android memiliki sekumpulan *tools* yang dapat digunakan sehingga membantu para pengembang dalam meningkatkan produktivitas pada saat membangun aplikasi yang dibuat. Android saat ini dapat dijumpai dimana saja karena tampilannya yang menarik dan *user friendly* atau sangat mudah mengoperasikannya.[1] Versi android yang digunakan dalam sistem absensi online ini ialah Android versi *Lollipop* (Android 5.0) dengan API 21.

D. Android Studio

Dalam pembangunan dan pengembangan aplikasi Android dibutuhkan IDE *(Integrated Development Environment)*. Aplikasi perangkat lunak yang menyediakan fasilitas lengkap bagi *programmer* komputer untuk pengembangan perangkat lunak, salah satunya ialah Android Studio. Android Studio merupakan IDE dari Google yang menyediakan fitur baru dan perbaikan atas Eclipse ADT. Android Studio merupakan IDE resmi untuk mengembangkan aplikasi android yang berbasi IntelliJ IDEA. Kemampuan yang Android Studio tawarkan antara lain, diperluasnya dukungan template untuk layanan Google dan berbagai jenis perangkat, layar editor kaya dengan

dukungan untuk mengedit tema, dan fitur Built-in mendukung untuk *Google Cloud Platform*, sehingga mudah untuk mengintegrasikan *Google Cloud Messaging*.[3]

E. *Unified Modeling Languange*

Unified Modeling Language atau disingkat UML merupakan bahasa standar yang digunakan untuk memvisualisasikan, menspesifikasikan, menkonstruksikan, serta mendokumentasikan sebuah sistem *software*. UML menawarkan sebuah standar untuk merancang model sebuah sistem dan sudah digunakan secara luas dan menggunakan notasi yang sudah dikenal untuk analisa dan desain berorientasi objek. Ada beberapa diagram yang dapat dipergunakan untuk memperjelas penggunaan UML dalam pemrograman berorientasi objek diantaranya *use case diagram, class diagram,* dan *activity diagram*. [10]

1. *Use Case* Diagram

Use Case Diagram merupakan gambaran fungsionalitas dari sebuah sistem. *Use case* diagram digunakan untuk memodelkan proses berdasarkan perspektif pengguna sistem. *Use case* menggambarkan kebutuhan sistem dari sudut pandang luar sistem. Terdapat empat komponen pada *Use Case* Diagram, yaitu:

1. Aktor

Gambar 2.5: Aktor *Use Case*

Aktor merupakan gambaran dari seseorang atau sesuatu yang akan berinteraksi

dengan sistem.

2. *Oval Use Case*

Gambar 2.6: *Oval Use Case*

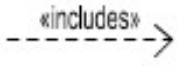
Representasi *Oval* dari suatu *Use Case* merupakan sebuah bentuk fungsionalitas dari sistem yang mempresentasikan kegunaan atau aktivitas sistem. Gambaran tugas dan apa yang akan dikerjakan oleh sistem juga diilustrasikan oleh *use case*.

3. Tanda Penghubung

   Tanda penghubung terbagi menjadi dua yaitu:

   - Garis Penghubung : Tanda penghubung yang menghubungkan link antar elemen.
   - «*Include*» : Tanda penghubung dengan kondisi tertentu, yaitu kelakuan yang harus terpenuhi agar sebuah event dapat terjadi, dimana pada kondisi ini sebuah *use case* adalah bagian dari *use case* lainnya.

Gambar 2.7: Garis Penghubung



Gambar 2.8: *Include*

4. Relasi

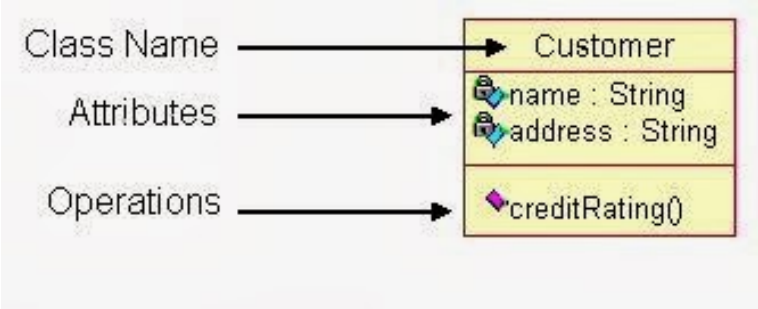
   (a) *Association* adalah sebuah relasi antara aktor dan *use case*, dimana sebuah interaksi terjadi diantara aktor dan *use case* tersebut.

   (b) *Generalization*, disebut juga *inheritance* (pewarisan), sebuah elemen dapat merupakan spesialisasi dari elemen lainnya.

   (c) *Dependency*, sebuah elemen bergantung dalam beberapa cara ke elemen lainnya.[10]

2. *Class* Diagram

Class diagram menggambarkan struktur objek yang terdapat pada sebuah sistem. Diagram ini menunjukkan objek-objek yang terdapat pada suatu sistem dan relasi antar objek-objek tersebut. *Class* memiliki tiga area pokok yaitu : Nama, Atribut dan Metoda/Operation.

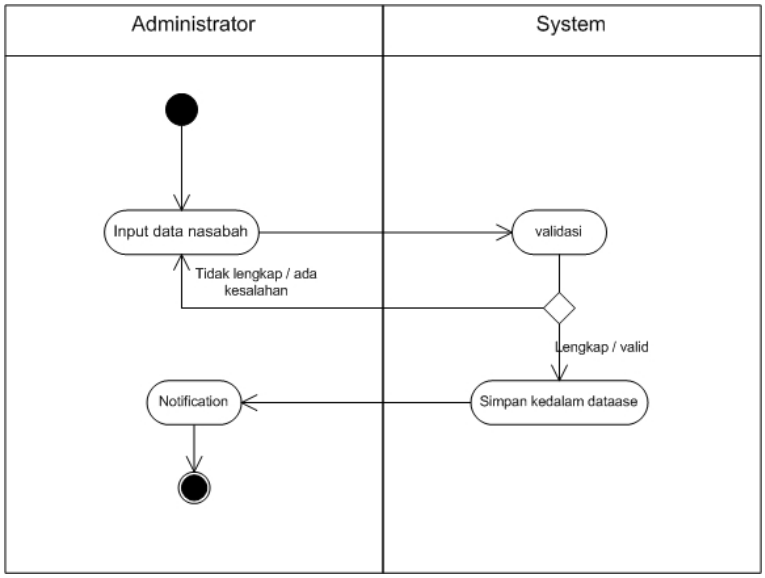


Gambar 2.9: Contoh *Class* Diagram

Atribut dan Metoda dapat mempunyai sifat *Private* (tidak dapat dipanggil dari luar *class*), *Protected* (hanya dapat dipanggil oleh *class* yang bersangkutan dan anak yang mewarisinya) dan *Public* (dapat dipanggil siapa saja). [10]

3. *Activity* Diagram

Activity diagram digunakan untuk menggambarkan alur dari proses bisnis atau langkah-langkah *usecase* secara berurutan. Diagram ini juga digunakan untuk menggambar *action* (tindakan) yang akan dieksekusikan ketika suatu proses sedang berjalan dan berserta hasil dari proses eksekusi tersebut. *Initial node* yang digambarkan dengan simbol lingkaran padat, merupakan titik yang mengawali *activity* diagram. *Activity* diagram dapat diakhiri dengan memberikan *activity final* diagram yang digambarkan dengan lingkaran padat dengan mempunyai cincin dibagian luarnya. [10]



Gambar 2.10: Contoh *Activity* Diagram

F. *Entity Relationship Diagram*

Terdapat banyak cara dalam memodelkan data. Namun yang paling sering digunakan adalah *Entity Relationship Diagram*. ERD adalah diagram yang menggambarkan entitas-entitas dan hubungan yang terbentuk berdasarkan data. Terdapat beberapa elemen pada ERD diantaranya ialah:

1. Entitas

   Entitas adalah sesuatu yang dibutuhkan untuk menyimpan data. Dalam pemodelan sistem, entitas dibutuhkan untuk memetakan suatu konsep yang abstrak menjadi suatu bentuk.

2. Atribut

   Jika entitas adalah sesuatu yang dibutuhkan untuk menyimpan data, dibutuhkan juga sesuatu yang dapat mengidentifikasi potongan data untuk disimpan pada entitas tertentu. Potongan data tersebut disebut Atribut. Atribut dapat juga digambarkan sebagai karakteristik dari sebuah entitas. Sifat dari atribut ialah sederhana, jelas, terpisah dan berkarakteristik tunggal.

3. *Keys*

   Keys berfungsi untuk menghubungkan suatu objek dengan objek lainnya. Keys diletakkan pada atribut kemudian dapat dihubungkan dengan atribut di entitas lain.

4. Relasi

   Relasi adalah asosiasi atau hubungan antara satu entitas atau lebih. Relasi bersifat dua arah, merupakan hubungan timbal balik antar entitas yang dihubungkan dengan relasi tersebut. Derajat yang biasa digunakan pada relasi adalah *binary relationship* yang terdiri atas *one to one* (1:1), *one to many* (1:*) dan *many to many* (*:*). [10]

G. Database MySQL

Basis data (database) adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut. Database diguna-

kan untuk menyimpan informasi atau data yang terintegrasi dengan baik di dalam komputer.[9]

Structured Query Language (SQL) adalah sekumpulan perintah khusus yang digunakan untuk mengakses data dalam database relasional. SQL merupakan sebuah bahasa komputer yang mengikuti standar ANSI (*American Nasional Standard Institute*) yang digunakan dalam manajemen database relasional. Hingga saat ini hampir seluruh server database atau *software* database mengenal dan mengerti bahasa SQL.

MySQL adalah sebuah *software database open source* sistem manajemen basis data SQL (*database management system*) atau DBMS yang *multithread*,*multi-user*,yang paling populer di dunia. Dengan kehandalan, kecepatan dan kemudahan penggunaannya, MySQL menjadi pilihan utama bagi banyak pengembang *software* dan aplikasi baik di *platform* web maupun desktop.[9]

H. *Application Programming Interface*

API atau *Application Programming Interface* adalah sekumpulan perintah, fungsi, *class* dan protokol yang disediakan oleh sistem operasi atau bahasa pemrograman tertentu agar memungkinkan suatu *software* berhubungan dengan *software* lainnya. API menyediakan sebuah mekanisme dalam penggunaan kembali sebuah kode yang telah dibuat sebelumnya sehingga *programmer* dapat memanfaatkannya kembali untuk keperluan yang berbeda. Penggunaan API ini lebih efektif dibanding *programmer* harus membuat kode untuk setiap programnya dari awal. API juga merupakan sebuah jembatan atau media perantara antara basis data dengan aplikasinya. API yang dapat mengakses basis data dapat memberikan kebutuhan aplikasi seperti membaca atau menulis data. [5]

Gambar 2.11: Cara Kerja API

I. *Wireless Fidelity*

Wireless Fidelity atau biasa disebut Wi-Fi adalah salah satu teknologi *Wireless Network* tanpa kabel berdasarkan spesifikasi dari IEEE 802.11. WiFi menggunakan gelombang elektromagnetik untuk mengirimkan data melalui gelombang udara. Daerah yang mempunyai sinyal WiFi adalah daerah yang berada pada radius 100 meter dari titik akses yang sering disebut *hotspot*. Adapun komponen yang terdapat dalam *hotspot* area mencakup beberapa hal, salah satunya adalah *Access Point* (Titik Akses). *Access Point* merupakan perangkat yang menghubungkan teknologi *Wireless* LAN dengan ethernet yang terdapat di komputer. Titik akses memiliki kemampuan untuk melayani pengguna sebanyak 128 orang. Luas daerah yang dapat dijangkau oleh sebuah titik akses mencapai 25-1000 meter. [2]