

## **BAB II**

### **LANDASAN TEORI**

#### **2.1 Tinjauan Pustaka**

Tinjauan pustaka dalam penelitian ini menganalisa dari beberapa jurnal untuk membangun sistem portal informasi kampus berbasis Android, antara lain :

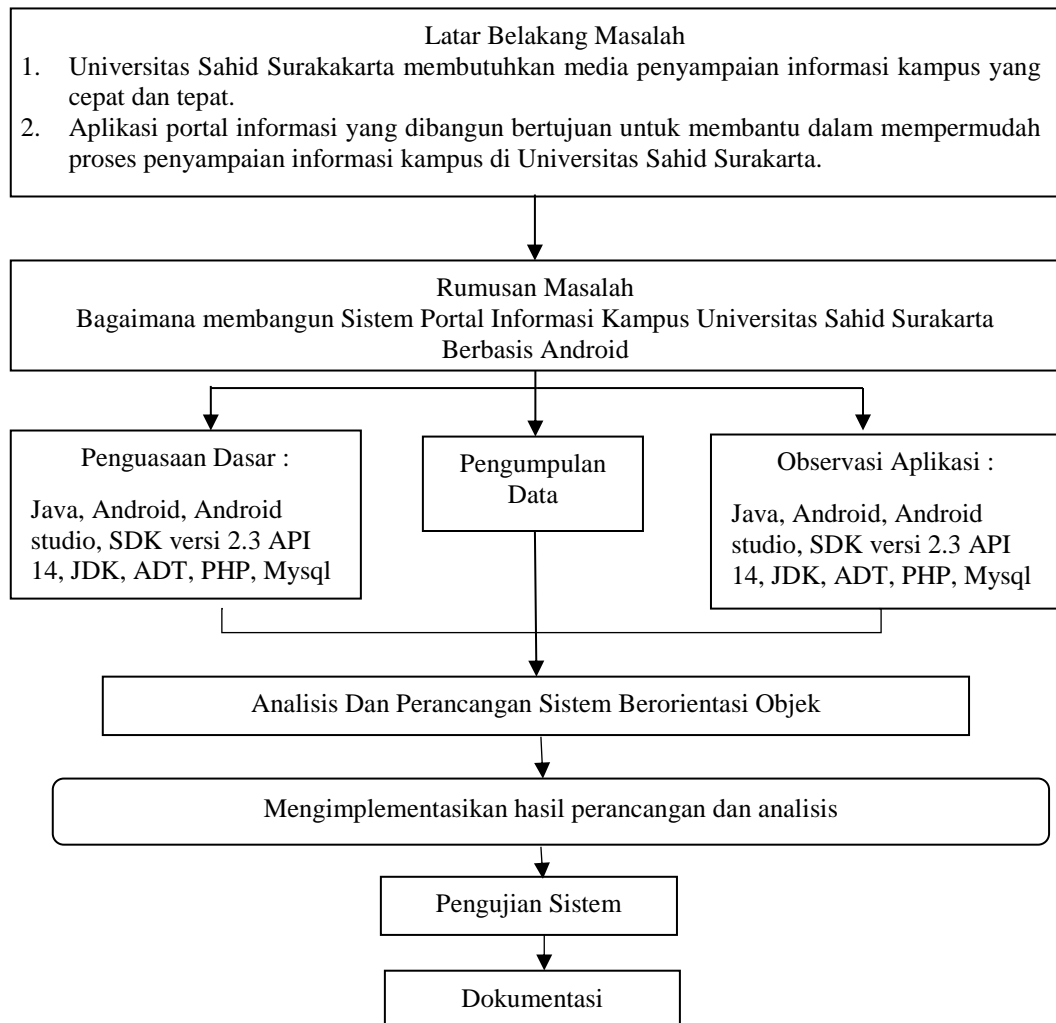
Achyarudin dan Zulkarnaen (2013) menjelaskan bahwa dengan adanya Sistem Informasi Akademik berbasis Android dapat meningkatkan fungsionalitas dari *website* simponi dengan pengoptimalan fitur pada perangkat *mobile*. Informasi-informasi penting pada STMIK Global Informatika Multi Data Palembang dapat diketahui secara langsung dengan adanya fitur pemberitahuan atau notifikasi seperti materi perkuliahan, tugas, informasi pengumuman dan forum diskusi antara dosen dan mahasiswa pada aplikasi simponi *mobile* berbasis Android.

Apriyanto, dkk (2015) menjelaskan bahwa berdasarkan penelitian yang penulis lakukan di Universitas Bina Darma Palembang, dihasilkan suatu aplikasi berbasis *mobile* yang terintegrasi dengan *database* portal *web* berita dan pengumuman Universitas Bina Darma yang dapat membantu dalam penyampaian informasi seputar berita dan pengumuman yang ada pada Universitas Bina Darma.

Wijaya (2013) menjelaskan bahwa pengembangan sistem informasi akademik berbasis *mobile* sangat penting dilakukan. Mahasiswa dapat dengan mudah mengakses sistem informasi akademik melalui perangkat *mobile* yang ada.

#### **2.2 Kerangka Berfikir**

Kerangka berfikir adalah dasar pemikiran dari penelitian yang disintesiskan dari fakta-fakta, observasi dan telah dilakukan penelitian. Kerangka berpikir memuat teori, dalil atau konsep - konsep yang akan dijadikan dasar dalam penelitian. Uraian dalam kerangka pikir ini menjelaskan hubungan antar variabel (Riduwan, 2014).



Gambar 2.1. Diagram Kerangka Berfikir

Kerangka pemikiran menjelaskan alur proses pembuatan laporan Tugas Akhir

#### 1. Latar Belakang Masalah

Universitas Sahid Surakarta khususnya prodi Teknik Informatika membutuhkan media penyampaian informasi kampus yang cepat dan tepat.

Aplikasi portal informasi yang dibangun bertujuan untuk membantu dalam mempermudah proses penyampaian informasi kampus di Universitas Sahid Surakarta.

## 2. Rumusan Masalah

Bagaimana membangun Sistem Portal Informasi Kampus Universitas Sahid Surakarta Berbasis Android ?

## 3. Pengumpulan Data Tertulis dan Tidak Tertulis

Mengumpulkan data yang diperlukan untuk penelitian, baik melalui *interview*, observasi maupun dokumentasi.

## 4. Penguasaan Dasar

Melakukan beberapa percobaan membuat aplikasi android dengan tujuan agar dapat lebih menguasai pemrograman android menggunakan Andoid studio yang merupakan tool lengkap yang terdiri dari SDK android dan API sebagai alat untuk desain tampilan, coding, dan emulasi.

## 5. Observasi Aplikasi

Mencari beberapa contoh aplikasi atau tinjauan pustaka yang berkaitan dengan pemrograman Android, karya ilmiah, buku yang dapat dijadikan referensi dalam membangun sistem portal informasi kampus berbasis Android untuk Universitas Sahid Surakarta.

## 6. Analisis dan Perancangan Sistem Berbasis Objek

Menganalisa dan merancang aplikasi yang akan dibangun, yang meliputi desain aplikasi, dan isi dari aplikasi yang akan dibangun.

## 7. Implementasi

Membangun Aplikasi sistem portal informasi kampus berbasis Android sesuai dengan data-data yang didapatkan dari Universitas Sahid surakarta.

## 8. Pengujian Sistem

Pada tahap ini dimana sistem yang telah siap digunakan kemudian dilakukan pengujian untuk mengetahui jika ternyata masih ada kesalahan atau kekurangan pada sistem yang telah dibuat.

## 9. Dokumentasi

Pada tahap akhir dimana sistem telah siap digunakan di Universitas Sahid Surakarta dan membuat dokumentasi dari keseluruhan penelitian tugas akhir ini.

## 2.3 Teori Pendukung

### 2.3.1 Pengertian Sistem Informasi

Sistem informasi adalah cara-cara yang diorganisasi untuk mengumpulkan, memasukkan, dan mengolah serta menyimpan data, dan cara-cara yang diorganisasi untuk menyimpan, mengelola, mengendalikan, dan melaporkan informasi sedemikian rupa sehingga sebuah organisasi dapat mencapai tujuan yang telah ditetapkan (Krismiaji, 2015).

### 2.3.2 Android

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 (Safaat, 2014).

Seiring pembentukan *Open Handset Alliance*, OHA mengumumkan produk perdana mereka, Android, perangkat bergerak (Mobile) yang merupakan modifikasi kernel Linux 2.6. Sejak Android dirilis telah dilakukan berbagai pembaharuan berupa perbaikan bug dan penambahan fitur baru.

### 2.3.3 Web Portal

Sebuah web portal adalah sebuah halaman yang mengijinkan user untuk mengkostumisasi *home page*-nya dengan melakukan *drag* dan *drop widget*. Pendekatan ini memberikan kontrol penuh kepada *user* atas konten apa yang dilihat pada *home page*-nya, di mana halaman *web* tersebut adalah halaman yang ingin dilihat oleh *user*, dan bagaimana *user* tersebut berinteraksi dengan konten tersebut (Zabir, 2008).

### 2.3.4 Android Studio

Android Studio adalah Lingkungan Pengembangan Terpadu - *Integrated Development Environment* (IDE) untuk pengembangan aplikasi Android, berdasarkan IntelliJ IDEA. Selain merupakan editor kode IntelliJ dan alat pengembang yang berdaya guna, Android Studio menawarkan fitur lebih banyak untuk meningkatkan produktivitas Anda saat membuat aplikasi Android (Supardi, 2015).

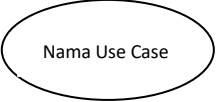
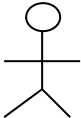
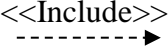
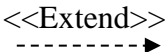

### 2.3.5 UML

UML merupakan sintaks umum untuk membuat model logika dari suatu sistem dan digunakan untuk menggambarkan sistem agar dapat dipahami selama fase analisis dan desain. UML biasanya disajikan dalam bentuk diagram atau gambar yang meliputi *class* beserta atribut dan operasinya, serta hubungan antar *class* yang meliputi *inheritance*, *association* dan komposisi (Sugiarti, 2013).

#### 2.3.5.1 Use Case Diagram

*Use case diagram* yang menggambarkan *actor*, *use case* dan relasinya sebagai suatu urutan tindakan yang memberikan nilai terukur untuk aktor. Sebuah *use case* digambarkan sebagai elips horizontal dalam suatu diagram UML *use case*. Simbol – simbol *Use Case Diagram* dapat dilihat pada Tabel 2.1.




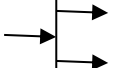
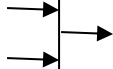

Tabel 2.1 Simbol-simbol *Use Case Diagram*

No	Gambar	Nama	Keterangan
1		<i>Use Case</i>	Fungsional yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.
2		<i>Actor</i>	Orang proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri.
3		<i>Include</i>	Menspesifikasikan bahwa pemanggilan <i>use case</i> oleh <i>use case</i> lain
4		<i>Extend</i>	Menspesifikasikan bahwa perluasan dari <i>use case</i> lain jika kondisi atau syarat terpenuhi
5		<i>Generalisasi/ Generalization</i>	Hubungan generalisai dan spesialis antara dua buah <i>use case</i> .

#### 2.3.5.2 Activity Diagram

*Activity Diagram* adalah diagram yang menggambarkan aliran kerja atau aktifitas dari suatu sistem. Perlu diperhatikan bahwa *activity diagram* menggambarkan aktifitas sistem bukan apa yang dilakukan aktor. Simbol-simbol *Activity Diagram* dapat dilihat pada Tabel 2.2.

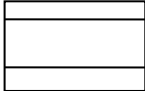
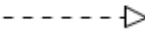
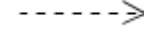

Tabel 2.2 Simbol-simbol *Activity Diagram*

No	Gambar	Nama	Keterangan
1		<i>Activity</i>	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		<i>Activity Initial Node</i>	Bagaimana object dibentuk atau diawali.
3		<i>Activity Final Node</i>	Bagaimana object diakhiri atau dihancurkan
4		<i>Fork Node</i>	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran
5		<i>Join Node</i>	Beberapa aliran yang pada tahap tertentu menjadi satu aliran.
6		<i>Decision node</i>	Suatu titik/point pada activity diagram yang mengindikasikan suatu kondisi dimana ada kemungkinan perbedaan transisi

### 2.3.5.3 Class Diagram

*Class Diagram* menggambarkan struktur dan deskripsi *class*, *package* dan object beserta hubungan satu sama lain seperti *contaiment*, pewarisan, asosiasi, dan lain-lain. *Class diagram* membantu dalam visualisasi struktur kelas-kelas dari suatu sistem yang merupakan tipe *diagram* yang paling banyak dipakai. Simbol-simbol *Class Diagram* dapat dilihat pada Tabel 2.3.


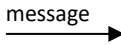




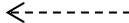
Tabel 2.3 Simbol-simbol *Class Diagram*

No	Gambar	Nama	Keterangan
1		<i>Class</i>	Himpunan dari object-object yang berbagi atribut serta operasi yang sama.
2		<i>Realization</i>	Operasi yang benar-benar dilakukan oleh suatu object.
3		<i>Dependency</i>	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri ( <i>independent</i> ) akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri
4		<i>Association</i>	Apa yang menghubungkan antara object satu dengan object lainnya

#### 2.3.5.4 *Sequence Diagram*

*Sequence Diagram* mendeskripsikan bagaimana entitas dalam sistem berinteraksi, termasuk pesan yang digunakan saat interaksi. *Diagram* ini juga menunjukkan serangkaian pesan yang diperlukan oleh objek-objek yang melakukan suatu tugas atau aksi tertentu. Simbol – simbol *Sequence Diagram* dapat dilihat pada Tabel 2.4.

Tabel 2.4 Simbol – simbol *Sequence Diagram*

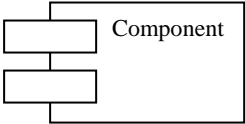


No	Gambar	Nama	Keterangan
1		<i>LifeLine</i>	Object <i>entity</i> , antarmuka yang saling berinteraksi.
2		<i>Line Message</i>	Spesifikasi dari komunikasi antar object yang memuat informasi-informasi tentang aktifitas yang terjadi.
3		<i>Actor</i>	Pengguna diluar sistem.
4		<i>Boundary</i>	Boundary biasanya berupa tepi dari system, seperti user interface, atau suatu alat yang berinteraksi dengan system lain.
5		<i>Control element</i>	Control element mengatur aliran dari informasi untuk sebuah scenario.
6		<i>Entity</i>	Entity biasanya elemen yang bertanggung jawab menyimpan data atau informasi.
7		<i>Return</i>	Spesifikasi dari komunikasi antar object yang memuat informasi-informasi tentang aktifitas yang terjadi

### 2.3.5.5 Deployment Diagram

*Deployment Diagram* menggambarkan detail bagaimana komponen di-sebar (*di-deploy*) kedalam infrastruktur sistem, dimana komponen akan terletak (pada mesin, *node*, *server*, atau piranti kertas apa), bagaimana kemampuan jaringan pada lokasi tersebut, spesifikasi *server*, dan hal-hal lain yang bersifat fisik. Simbol-simbol *Deployment Diagram* dapat dilihat pada Tabel 2.5.



Tabel 2.5 Simbol-simbol *Deployment Diagram*

No	Nama	Gambar	Keterangan
1		<i>Component</i>	Pada <i>deployment diagram</i> , <i>component</i> - <i>component</i> yang ada diletakkan didalam node untuk memastikan keberadaan posisi mereka.
2		<i>Node</i>	<i>Node</i> menggambarkan bagian-bagian <i>hardware</i> dalam sebuah sistem. Notasi untuk node digambarkan sebagai sebuah kubus 3 dimensi.
3		<i>Association</i>	Sebuah <i>association</i> digambarkan sebagai sebuah garis yang menghubungkan dua node yang mengindikasikan jalur komunikasi antara element-elemen <i>hardware</i> .

## 2.4 Metode Pengujian *Blackbox*

*Black Box Testing* atau Pengujian Kotak Hitam atau juga disebut *Behavioral Testing*, berfokus pada persyaratan fungsional dari perangkat lunak. Artinya, teknik *Black Box Testing* memungkinkan untuk mendapatkan *set* kondisi masukan yang sepenuhnya akan melaksanakan semua persyaratan fungsional untuk suatu program. *Black Box Testing* bukan merupakan alternatif dari pengujian *White Box Testing*. Sebaliknya, *Black Box Testing* adalah pendekatan *komplementer* yang mungkin untuk mengungkap kelas yang berbeda dari kesalahan daripada metode *White Box Testing* (Pressman, 2010).

### 2.4.1. Teknik-teknik dalam *black box testing*

Berikut ini merupakan beberapa teknik yang bisa digunakan untuk merancang pengujian *black box* ini:

**a) *Equivalence partitioning***

Teknik ini merupakan teknik pengujian *software* yang melibatkan pembagian nilai *input* kedalam bagian nilai *valid* dan tidak *valid* dan memilih nilai perwakilan dari masing-masing sebagai *data test*.

**b) *Boundary value analysis***

Teknik ini merupakan teknik pengujian *software* yang melibatkan penentuan-penentuan nilai *input* dan memilih beberapa nilai dari batasan-batasan tersebut baik luar maupun dalam dari batasan-batasan tersebut sebagai *data test*.

**c) *Cause Effect Graphic***

Teknik ini merupakan teknik pengujian *software* yang melibatkan penidentifikasi sebab-sebab (kondisi *input*) dan akibat-akibat (kondisi *output*), menghasilkan grafik sebab-akibat dan kasus-kasus *test*.

## 2.5 Kuesioner

Kuesioner merupakan teknik pengumpulan data dengan cara memberikan daftar pertanyaan tertulis yang ditujukan kepada responden yang jumlahnya banyak sehingga tidak memungkinkan untuk dilakukan pengumpulan data melalui wawancara (Sugiyono, 2009).

Terdapat 4 buah pilihan jawaban yang disediakan untuk pertanyaan pada kuesioner, yaitu:

- a. Sangat Setuju (SS)
- b. Setuju (S)
- c. Kurang Setuju (KS)
- d. Tidak setuju (TS)

Berdasarkan data yang dihasilkan dari kuesioner, dapat dihitung nilainya berdasarkan persentase masing-masing jawaban dengan menggunakan rumus:

$$Y = P/Q * 100\% \quad (I) \dots\dots\dots (2.1)$$

Keterangan: Y = Nilai presentase

P = Banyak jawaban responden dari tiap soal

Q = Jumlah responden

Sedangkan untuk menarik kesimpulan dari setiap pertanyaan, dilakukan perhitungan menggunakan skala likert. Skala likert adalah metode perhitungan

yang digunakan untuk keperluan riset atas jawaban setuju atau tidaknya seorang responden terhadap suatu pernyataan. Untuk menghitung skor maksimum tiap jawaban, dengan mengalikan skor dengan jumlah keseluruhan responden, yaitu skor responden. Skor jawaban responden disajikan pada Tabel 2.6.

Tabel 2.6 Skor Jawaban Responden

Jawaban	Skor
Sangat Setuju	4
Setuju	3
Kurang Setuju	2
Tidak Setuju	1