BAB III

OBJEK DAN METODE PENELITIAN

3.1 Objek Penelitian

Penelitian akan di lakukan di kampus D3 FMIPA dan ilmu komputer Universitas Padjadjaran yang beralamat di Jl. Ir H. Djuanda No 4 Bandung.

3.1.1 Sejarah Singkat Objek Penelititan

Fakultas MIPA didirikan berdasarkan Keputusan Menteri PP&K RI Nomor 102333/S tanggal 22 Oktober 1958, peresmiannya dilakukan Presiden RI (Dr. Ir. Soekarno) pada tanggal 17 Nopember 1959.

Pimpinan Fakultas MIPA Universitas Padjadjaran :

Dekan	: Prof. Dr. Budi Nurani Ruchjana, MS
Pembantu Dekan I	: Prof. Dr. Toto Subroto, MS
Pembantu Dekan II	: Gatot Riwi Setyanto, Drs., M.Si
Pembantu Dekan III	: Budi Irawan, S.Si., M.Si
Kepala Bagian Tata Usaha	: Dadang Sukmana, Ir., M.Si

Tujuan Fakultas MIPA Universitas Padjadjaran :

1. Menghasilkan lulusan berkualitas dengan masa studi tepat waktu.
2. Menghasilkan lulusan dengan rata-rata waktu tunggu kerja yang singkat.

3. Menghasilkan karya ilmiah yang dipublikasikan dalam jurnal nasional dan internasional.
4. Memperoleh hak paten atau HKI atas produk ilmiah yang dihasilkan.
5. Menghasilkan produk penelitian yang diaplikasikan untuk kesejahteraan masyarakat.

Sasaran Fakultas MIPA Universitas Padjadjaran :

1. 80 persen mahasiswa lulus maksimum delapan semester dengan rata-rata IPK 3,31
2. 50 persen lulusan memiliki rata-rata waktu tunggu kerja dalam tiga bulan.
3. Menghasilkan produk ilmiah yang dipublikasikan dalam jurnal internasional 50 artikel per tahun.
4. Memperoleh lima hak paten atau HKI per tahun.
5. Menghasilkan lima karya ilmiah per tahun yang dapat dipakai untuk menyelesaikan permasalahan masyarakat.

3.1.2 Visi dan Misi Objek Penelitian

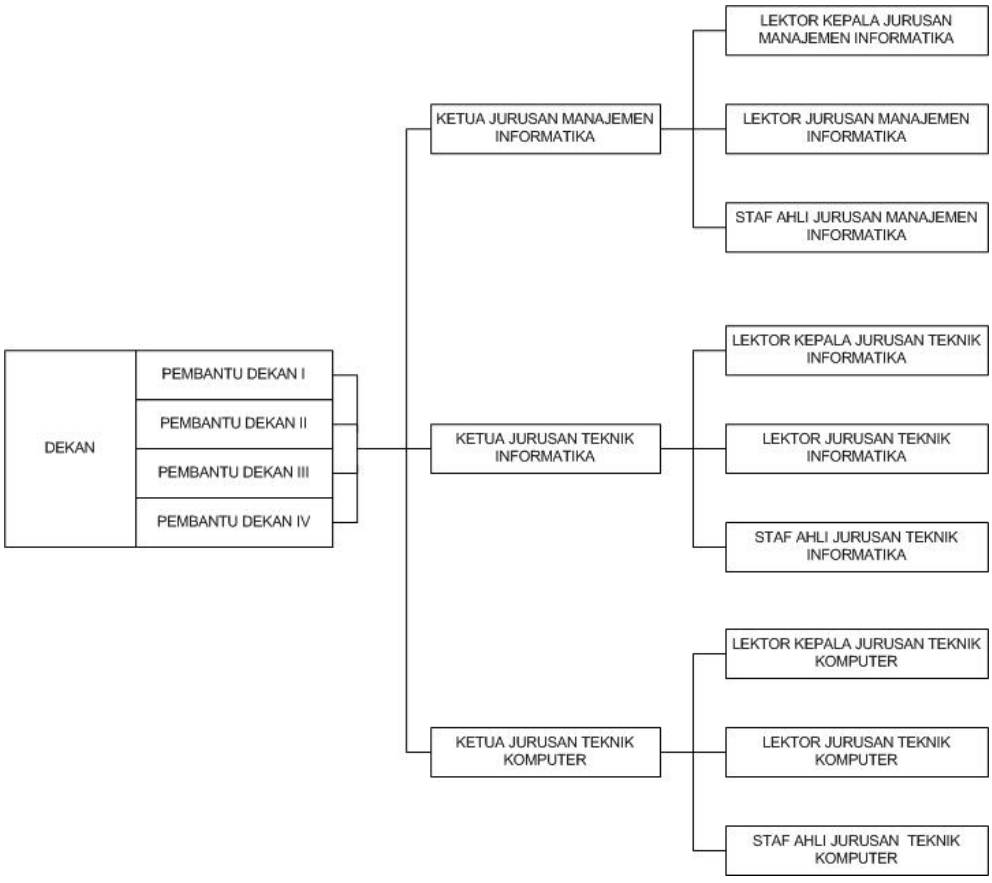
3.1.2.1 Visi Fakultas MIPA Universitas Padjadjaran

Fakultas MIPA Universitas Padjadjaran memiliki visi menjadi fakultas MIPA yang unggul di tingkat regional ASEAN pada tahun 2020.

3.1.2.2 Misi MIPA Universitas Padjadjaran

1. Melaksanakan pembelajaran berbasis penelitian untuk menghasilkan lulusan yang berdaya saing tinggi.
2. Melakukan penelitian sains dasar dan terapannya khususnya dalam pengelolaan sumber daya alam serta lingkungan yang bermanfaat bagi kesejahteraan masyarakat

3.1.3 Struktur Organisasi Fakultas MIPA Universitas Padjadjaran



Gambar 3.1 Struktur Organisasi Fakultas D3 MIPA Universitas Padjadjaran

3.2 Metode Penelitian

Penelitian adalah kegiatan untuk memperoleh fakta-fakta atau prinsip-prinsip (baik kegiatan untuk penemuan, pengujian atau pengembangan) dari suatu pengetahuan dengan cara mengumpulkan, mencatat dan menganalisa data yang dikerjakan secara sistematis berdasarkan ilmu pengetahuan (metode ilmiah).

Metode yang digunakan dalam melaksanakan penelitian merupakan dasar penyusunan rancangan penelitian dan merupakan penjabaran dari metode ilmiah secara umum.

3.2.1 Desain Penelitian

Pendekatan dalam pengembangan perangkat lunak ini menggunakan metode deskriptif, yaitu suatu metode dengan tujuan untuk membuat gambaran secara sistematis, faktual dan akurat mengenai fakta-fakta dan sifat-sifat pada suatu objek penelitian tertentu.

Pada tahap pertama dilakukan pengumpulan data dan bahan yang diperlukan. Pada tahap berikutnya mengolah dan membahas suatu kesimpulan yang pada akhirnya menghasilkan semua kebutuhan fungsional yang akan kita capai dalam pengembangan perangkat lunak.

3.2.2 Jenis dan Metode Pengumpulan Data

Dalam penelitian ini dibutuhkan data primer dan data sekunder untuk memenuhi kebutuhan dalam pengembangan perangkat lunak agar dapat memenuhi kebutuhan fungsional yang ditargetkan.

3.2.2.1 Sumber Data Primer

Data primer, yaitu data yang dikumpulkan dan diolah sendiri langsung dari objeknya. Contohnya kebutuhan fungsional dari sistem yang akan dikembangkan yang diperolah dari hasil wawancara atau observasi terhadap objek penelitian.

Dalam penelitian ini data primer diperoleh dari hasil observasi terhadap objek penelitian. Data primer tersebut berupa kebutuhan fungsional dari aplikasi *m-learning* yang akan dikembangkan. Kebutuhan fungsional didapatkan dari analisis terhadap obejek dengan sudut pandang pengguna agar didapatkan hasil yang dapat memenuhi kebutuhan pengguna.

3.2.3 Metode Pendekatan dan Pengembangan Sistem

Metode pendekatan dan pengembangan sistem menggambarkan tahapan dalam proses penelitian untuk menyelesaikan masalah penelitian dari awal perencanaan hingga tercapainya tujuan penelitian dan pengembangan sistem.

3.2.3.1 Metode Pendekatan Sistem

Metode pendekatan sistem yang digunakan dalam penelitian pengembangan perangkat lunak ini adalah metode pendekatan sistem yang berorientasi objek (*Object-Oriented Analysis and Design*).

3.2.3.2 Metode Pengembangan Sistem

Metode pengembangan sistem yang akan digunakan pada pengembangan aplikasi ini adalah model *prototype*. Karena model ini lebih memperhatikan kebutuhan fungsional pengguna, secara keseluruhan akan mengacu kepada kepuasan user. Adapun tahap-tahap dari pembuatan *prototype*, sebagai berikut :



Gambar 3.2 Metode Pengembangan *Prototype*

Sumber : (Roger S. Pressman, Ph.D., 2002, Rekayasa Perangkat Lunak Pendekatan Praktisi (BUKU SATU), Andi, Yogyakarta)

Metode pengembangan *prototype* terdiri dari tiga tahapan, yaitu pengumpulan kebutuhan, perancangan sistem serta pengujian terhadap hasil yang telah dibuat. Proses-proses tersebut dapat dijelaskan sebagai berikut :

1. Mendengarkan pelanggan (*Listen to customer*)

   Metode pengembangan *prototype* dimulai dengan pengumpulan syarat. *Devoleper* dan pelanggan bertemu dan mendefinisikan dari seluruh tujuan untuk *software,* identifikasi apa saja syarat yang dikenal dan daerah *outline* dimana definisi tersebut diperintahkan.

2. Membangun atau mengubah kembali *project* yang dibuat (*Build/revise mock-up*)

   Proses selanjutnya adalah perancangan kilat, dimana proses ini mewakili setiap aspek yang terdapat pada *software* yang terlihat oleh *customer* atau *user* (misal : pendekatan *input* dan bentuk *output*). Idealnya, *prototype* melayani sebagian mekanisme untuk mengidentifikasi syarat dari sebuah *software*.

3. Menguji hasil *project* yang dibuat (*Customer test drives mock-up*)

   Prototype dievaluasi oleh pelanggan/pengguna dan digunakan untuk penyaringan terhadap *software* yang dibangun. Proses iterasi sebagai *prototype* yang seimbang untuk memenuhi kebutuhan *customer* ketika pada saat yang sama memungkinkan *developer* untuk memperoleh pemahaman terbaik terhadap kebutuhan apa yang telah dikerjakan.

3.2.3.3 Alat Bantu Analisis dan Perancangan

Unified Modeling Language (UML) dikembangkan pada tahun 1995 oleh Grady Booch, James Rumbaugh dan Ivar Jacobson. "*UML adalah bahasa pemodelan standar pada rekayasa perangkat lunak".* (Munawar, 2005). UML terdiri dari serangkaian diagram yang dibutuhkan oleh analis untuk menggambarkan proses pengembangan sebuah sistem. Dengan UML akan bisa menceritakan apa yang seharusnya dilakukan sebuah sistem bukan bagaimana yang seharusnya dilakukan sebuah sistem.

Peran *Unified Modeling Language* (UML) dalam perancangan perangakat lunak adalah sebagai berikut :

1. Visualisasi

   Menggambarkan ide dalam notasi dan semantik yang lebih mudah dipahami oleh siapapun.

2. Spesifikasi

   Spesifikasi dari semua keputusan penting analisa, perancangan, dan penerapan yang harus diambil dalam pengembangan dan *deployment sistem*.

3. Konstruksi

   UML bukan bahasa pemrograman visual. Model UML dapat dihubungkan secara langsung dengan beberapa bahasa pemrograman. *Forward engineering*: menghasilkan kode dari model. *Reverse engineering*: membangun model dari kode.

4. Dokumentasi

   UML mencakup dokumentasi arsitektur sistem dan rincinya. UML sebagai suatu bahasa untuk menyatakan kebutuhan dan pengujian UML menyediakan bahasa untuk aktifitas perencanaan proyek dan manajemen *release.*

Unified Modeling Language (UML) terdiri dari delapan diagram untuk menggambarkan rancangan perangkat lunak dalam metode pengembangan sistem berorientasi objek.

Delapan Diagram UML tersebut adalah sebagai berikut :

1. *Use Case Diagram*

   Use case diagram digunakan untuk menggambarkan hubungan antara actor dengan *use case* dalam sebuah sistem. *Use case* diagram lebih berfokus dalam menggambarkan proses apa saja yang dikerjakan di dalam sistem. *Use case diagram* mempermudah dalam menggambarkan interaksi actor dengan proses yang ada dalam sebuah sistem.

2. *Activity diagram*

   Menggambarkan rangkaian aliran dari aktivitas, digunakan untuk mendeskripsikan aktifitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktifitas lainnya seperti *use case* atau interaksi.

3. *Sequence diagram*

Sequence Diagram digunakan untuk memberikan gambaran detail dari setiap *use case diagram* yang telah dibuat sebelumnya. *Sequence Diagram*merupakan diagram interaksi yang menekankan pada pengiriman pesan dalam suatu urutan waktu tertentu.

4. *Collaboration Diagram*

   Collaboration diagram menampilkan informasi yang sama dengan *Sequence diagram*. Hanya saja, interaksi antara *object* dan *actor* tidak ditampilkan berdasarkan urutan waktu.

5. *Class diagram*

   Menggambarkan struktur statis *class* di dalam sistem, *Class* merepresentasikan sesuatu yang ditangani oleh sistem. *Class* dapat berhubungan dengan yang lain melalui berbagai cara: *associated* (terhubung satu sama lain), *dependent* (satu *class* tergantung/menggunakan *class* yang lain), *specialed* (satu *class* merupakan spesialisasi dari *class* lainnya), atau *package* (grup bersama sebagai satu unit). Sebuah sistem biasanya mempunyai beberapa *class diagram*.

6. *Component Diagram*

   Menggambarkan arsitektur fisik dari perangkat keras dan perangkat lunak sistem, menunjukkan hubungan komputer dengan perangkat (*nodes*) satu sama lain dan jenis hubungannya. Di dalam *nodes*, *executeable component* dan *object* yang dialokasikan untuk memperlihatkan unit perangkat lunak yang dieksekusi oleh *node* tertentu dan ketergantungan komponen.

7. *Deployment diagram*

   Menggambarkan struktur fisik kode dari komponent. Komponent dapat berupa *source code*, komponent biner, atau *executable component*. Sebuah komponent berisi informasi tentang logic class atau class yang diimplementasikan sehingga membuat pemetaan dari *logical view* ke *component view.*

8. *Object Diagram*

   Object diagram berguna dalam menunjukan contoh-contoh obyek yang saling terhubung satu sama lain. Dalam banyak kasus struktur yang tepat bisa digambarkan secara tepat dengan *class diagram,* akan tetapi struktur tersebut mungkin masi susah dimengerti. Pada setuasi seperti ini pembuatan contoh degan *obyek diagram* akan sangat membantu sekali.

3.2.4 Pengujian Software

Pada pengembangan perangkat lunak ini, pengujiannya akan dilakukan dengan metode *Black Box Testing*. Metode pengujian *Black Box* lebih berfokus pada kebutuhan fungsional user terhadap perangkat lunak tersebut, sehingga diharapkan perangakat lunak yang dihasilakn sesuai dengan tujuan dan dapat memenuhi kebutuhan user.

Beberapa kategori kesalahan yang terdeteksi jika menggunakan metode *Black Box* diantaranya adalah sebagai berikut :

1. Fungsi-fungsi yang hilang atau tidak benar.

2. Kesalahan *interface.*
3. Kesalahan pada struktur data.
4. Kesalahan pada *performance*.
5. Kesalahan pada inisialisasi dan terminasi.

Pada pengujian perangkat lunak dengan metode *Black Box* terdapat beberapa faktor pengujian.

Adapun faktor-faktor pengujian pada *Black Box Testing* adalah sebagai berikut :

1. *Authorization*

   Menjamin data diproses sesuai dengan ketentuan manajemen. Authorisasi menyangkut proses transaksi secara umum yaitu otoritas bisnis, dan secara khusus otoritas pelaksanaan tindakan khusus.

2. *File Integrity*

   Menekankan pada data yang dimasukkan melalui aplikasi akan tidak bisa diubah. Prosedur yang akan memastikan bahwa file yang digunakan benar dan data dalam file tersebut akan disimpan sekuensial dan benar.

3. *Audit Trail*

   Menekankan pada kemampuan untuk mendukung proses yang terjadi. Pemrosesan data secara keseluruhan berdasarkan retensi dari kejadian yang cukup mendukung keakuratan, kelengkapan, batas waktu dan otorisai data.