BAB III

METODE PENELITIAN

3.1 Metode Penelitian

Metode penelitian berisi uraian yang menjelaskan secara rinci tentang bagaimana suatu penelitian dilakukan. Penjelasan ini juga bisa digunakan untuk menilai apakah hasil suatu penelitian dapat dipercaya, dan apakah kesimpulan penelitian ini dapat digunakan untuk kepentingan secara praktis atau digunakan dalam kajian pustaka dalam penelitian berikutnya.

Dalam penelitian untuk perancangan Aplikasi Sejarah Kristen Protestan Di Tanah Batak Toba berbasis Mobile Android menggunakan metodologi *Deskriptif* dan *Action* (Tindakan).

Metodologi *Deskriptif* merupakan suatu metode yang ingin mencari jawab secara mendasar tentang sebab akibat, dengan menganalisa faktor-faktor penyebab terjadinya ataupun munculnya suatu fenomena tertentu. Tujuan dari penelitian deskriptif ini adalah untuk membuat deskipsi, gambaran atau lukisan secara sistematis, factual dan akurat mengenai fakta-fakta, sifat-sifat serta hubungan antar fenomena yang diselidiki. Sedangkan metodologi *Action* (Tindakan) merupakan pendekatan untuk meningkatkan pemahaman dan penindak lanjutan dari hasil analisis untuk diimplementasikan dalam perancangan sebuah aplikasi.

3.1.1 Desain Penelitian

Desain penelitian menurut Moh. Nazir, Ph.D (2003:84) memaparkan bahwa “Desain penelitian adalah proses yang diperlukan dalam perencanaan dan pelaksanaan penelitian”.

Dari definisi di atas maka dapat ditarik sebuah kesimpulan bahwa desain penelitian suatu penelitian untuk perancangan aplikasi ini sangat perlu dilakukan perencanaan dan perancangan penelitian, agar penelitian yang dilakukan dapat berjalan dengan baik dan sistematis.

3.1.2 Jenis dan Metode Pengumpulan Data

Dalam penelitian ini jenis dan metode pengumpulan data yang digunakan adalah sebagai berikut :

3.1.2.1 Sumber Data Primer

Data primer merupakan data yang didapat dari sumber informan pertama. Pemberi informan tersebut diharapkan dapat menjawab semua pertanyaan dengan jelas dan lengkap. Untuk mendapatkan data primer metode pengumpulan data yang dipakai adalah wawancara dan observasi.

1. Observasi

   Observasi merupakan suatu cara pengumpulan data melalui pengamatan dan pencatatan oleh pengumpul data terhadap gejala atau peristiwa yang diselidiki pada obyek penelitian secara langsung. Penulis melakukan tinjauan langsung ke lapangan untuk mencari data-data dan fakta yang diperlukan dalam membuatan aplikasi Sejarah Kristen Protestan di tanah Batak Toba.

2. Wawancara (*interview*)

   Wawancara merupakan proses memperoleh keterangan untuk tujuan penelitian dengan cara tanya jawab dengan responden dan metode pengumpulan data ini dimana peneliti bertemu langsung (tatap muka) atau

tidak langsung (telepon, konferensi video, online melalui jaringan internet). Dalam hal ini penulis melakukan wawancara dengan yaitu Bapak Gr. H. Sitompul, S.PAK sebagai pimpinan jemaat HKBP Balige. Wawancara ini dilakukan untuk mendapatkan informasi tentang perkembangan agama Kristen Protestan di daerah Batak Toba.

3.1.2.2 Sumber Data Sekunder

Data sekunder merupakan sumber data penelitian yang diperoleh peneliti secara tidak langsung melalui media perantara (diperoleh dan dicatat oleh pihak lain). Data sekunder umumnya berupa bukti, catatan atau laporan historis yang telah tersusun dalam arsip (data dokumenter/ Dokumentasi) yang dipublikasikan dan yang tidak dipublikasikan. Adapun data sekunder yang digunakan yaitu sumber tertulis yang antara lain terdiri dari:

a. Sumber buku, Jurnal Ilmiah dan Internet

   Buku-buku, jurnal ilmiah, dan data yang berasal dari internet sangat berguna untuk menunjang penelitian, yakni sebagai tinjauan pustaka untuk menganalisis penelitian yang dilakukan.

b. Dokumentasi

   Dokumentasi adalah teknik pengumpulan data dengan cara mengumpulkan dokumen-dokumen yang berhubungan dengan objek penelitian. Dalam hal ini, dokumen-dokumen yang diperoleh dianalisis sehingga diperoleh data-data yang sesuai untuk kegiatan pengembangan sistem.

3.1.3 Metode Pendekatan dan Pengembangan Sistem

Berikut ini merupakan metode pendekatan dan pengembangan system yang dipakai.

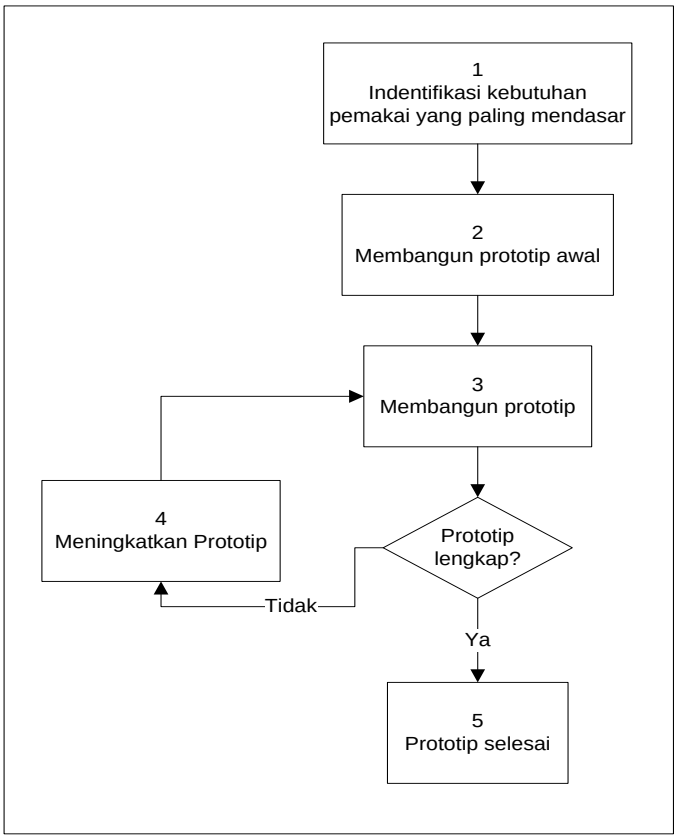
3.1.3.1 Metode Pendekatan Sistem

Metode pendekatan sistem adalah metode yang memandang software atau aplikasi sebagai sebuah interaksi antar bagian dalam sebuah sistem. Dalam melakukan penelitian ini metode pendekatan yang dipakai adalah pemodelan berorientasi objek karena metode ini dikembangkan sebagai suatu alat analisis dan desain yang sering dipakai dalam pengembangan system berbasis *mobile.*

3.1.3.2 Metode Pengembangan Sistem

Metode pengembangan sistem adalah metode-metode, prosedur-prosedur, konsep-konsep pekerjaan dan aturan-aturan untuk mengembangkan suatu sistem informasi. Dengan mengikuti metode dan prosedur-prosedur yang diberikan oleh suatu metode, maka pengembangan sistem diharapkan akan dapat diselesaikan dengan berhasil.

Metode penelitian khusus untuk perangkat lunak adalah metode *protoype*. Menurut Abdul Kadir (2003:416), suatu *prototype* merupakan suatu metode dalam pengembangan sistem yang menggunakan pendekatan untuk membuat sesuatu program dengan cepat dan bertahap sehingga segera dapat dievaluasi oleh pemakai.



Gambar 3.1 Membangun Prototype

Sumber: Jogiyanto HM (2005)

Prototyping merupakan suatu model yang memperlihatkan fitur-fitur suatu produk, layanan, atau sistem usulan. Modelnya dikenal dengan sebutan metode *prototype*. Metode penelitian ini dipilih karena alasan sebagai berikut :

Protoype memberikan ide bagi disainer sistem maupun *user* tentang cara sistem akan berfungsi dalam bentuk lengkapnya. Langkah dalam *prototyping* adalah seperti berikut :

1. Mengidentifikasi kebutuhan-kebutuhan dasar pemakai. Dalam hal ini perancangan sistem bekerja dengan pemakai untuk menangkap informasi dasar yang diperlukan pemakai.

2. Mengembangkan sebuah *prototype*. Perancangan sistem menciptakan sebuah *prototype* dengan cepat. Dengan menggunakan perangkat lunak generasi keempat atau menggunakan perangkat lain. *Prototype* dapat hanya mancakup fungsi-fungsi yang paling penting atau mencakup seluruh sistem.
3. Menggunakan *prototype*. Pada tahapan ini, pemakai diminta untuk bekerja dengan sistem untuk menentukan cocok-tidaknya *prototype* terhadap kebutuhan pemakai dan diharapkan pemakai memberi saran-saran untuk perbaikan *prototype*.
4. Memperbaiki dan meningkatan *Prototype*. *Prototype* diperbaiki sesuai dengan semua perubahan yang diminta atau yang disarankan oleh pemakai. Setelah diperbaiki, langkah 3 dan 4 dilakukan secara terus menerus sampai pemakai merasa puas.

3.2 Alat Bantu Analisis dan Perancangan

1. UseCase Diagram

Use Case Diagram atau merupakan pemodelan untuk menggambarkan kelakukan (behavior) sistem yang akan dibuat. Diagram use case mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. Dengan pengertian yang cepat, diagram use casi digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Terdapat beberapa simbol dalam menggambarkan diagram use case, yaitu use case, aktor dan relasi. Hal yang perlu diingat mengenai diagram use case adalah diagram use

case bukan menggambarkan tampilan antarmuka (use interface), arsitektur dari sistem, kebutuhan nonfungsional, dan tujuan performasi. Sedangkan untuk penamaan use case adalah nama didefinisikan sesimpel mungkin, dapat dipahami dan menggunakan kata kerja.

2. Activity Diagram

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. Dan Activity diagram menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.

Activity diagram merupakan state diagram khusus, di mana sebagian besar state adalah action dan sebagian besar transisi di-trigger oleh selesainya *state* sebelumnya (internal processing). Oleh karena itu activity diagram tidak menggambarkan *behaviour internal* sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

3. Class Diagram

Diagram class mendeskripsikan jenis-jenis objek dalam sistem dan berbagai hubungan statis yang terdapat di antara mereka. Diagram class juga menunjukkan properti dan operasi sebuah class dan batasan-batasan yang terdapat dalam hubungan-hubungan objek tersebut.

Kotak adalah notasi UML untuk *class*. Nama, *attribute, operation* dan *responsibility* dari *class* ada pada kotak tersebut. *Stereotype* bisa dipergunakan untuk mengorganisasikan daftar *attribute* dan *operation*. Dalam beberapa kasus,

kadangkala hanya perlu ditampilkan sebagian saja dari *attribute* dan *operation*. Tipe *attribute* dan nilai *default* bisa dimunculkan sebagaimana pada *operation*. Untuk mengurangi ambiguitas pada pendeskripsian *class*, *constraint* bisa ditambahkan. Bahkan kalau perlu bisa ditambahkan *attached notes* ke dalam kotak tersebut.

4. Sequence Diagram

Diagram sequence menggambarkan kelakuan/perilaku objek pedause case dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antara objek. Oleh karena itu untuk menggambar diagram sequence maka harus diketahui objek-objek yang terlibat dalam sebuah use case beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu.

Pada diagram ini *participant* diletakkan di atas dan waktu ditunjukkan dari atas ke bawah. *Lifeline participant* diurutkan dari setiap participant. Kotak kecil pada *Lifeline* menyatakan activation, yaitu menjalankan salah satu operation dari participant. State bias ditambahkan dengan menambahkannya sepanjang life line.

Message (sederhana, *synchronous* atau *asynchronous*) adalah tanda panah yang menghubungkan suatu *life line* ke *life line* yang lain. Lokasi *life line* dalam dimensi vertikal mewakili urutan waktu dalam *sequence* diagram. *Message* yang pertama terjadi adalah yang paling dekat dengan bagian atas diagram dan yang terjadi belakangan adalah yang dekat dengan bagian bawah.

5. Component Diagram

Component diagram menggambarkan struktur dan hubungan antar komponen piranti lunak, termasuk ketergantungan (*dependency)* di antaranya.

Component piranti lunak adalah modul berisi *code*, baik berisi *source code* maupun *binary code*, baik *library* maupun *executbale*, baik yang muncul pada *compile time, link time*, maupun *run time.* Umumnya *component* terbentuk dari beberapa *class* atau *package*, tapi dapat juga dari *component- component* yang lebih kecil.

Component bisa diakses melalui *interface* nya yaitu koleksi operasi-operasi. Relasi antara *component* dan interfacenya disebut *realization*. Suatu *component* bias mengakses *service-service* yang ada di *component* lain dengan cara *import interface*. Sedangkan *component* yang menyediakan *service* menggunakan *export interface*.

6. Collaboration Diagram

Collaboration diagram menggambarkan interaksi antar objek seperti *sequence* diagram, tetapi lebih menekankan pada peran masing-masing objek dan bukan pada waktu penyampaian *message.*

Collaboration diagram merupakan asosiasi diantara obyek-obyek. Panah di dekat garis asosiasi menunjukkan *message*, sedangkan content *message* ditunjukkan dengan label. Angka pada *message* menunjukkan urutan *message.* Dengan *Collaboration* diagram memungkinkan untuk memodelkan pengiriman sebuah *message* ke banyak obyek pada *class* yang sama. Demikian juga halnya untuk menunjukkan adanya obyek aktif yang mengendalikan aliran dari *message*.

7. Deployment Diagram

Deployement diagram menyediakan gambaran bagaimana sistem secara fisik akan terlihat. Sistem terdiri dari node-node dimana setiap node diwakili untuk sebuah kubus. Garis yang menghubungkan antara 2 kubus menunjukkan

hubungan diantara kedua node tersebut. Tipe node bisa berupa *device* yang berwujud *hardware* dan bisa juga *processor* (yang mengeksekusi *component*) atau *execution environment (software* yang menjadi *host* atau mengandung *software* yang lain).

3.3 Pengujian Software

Pada pengujian *black-box,* kasus-kasus pengujian berdasarkan pada spesifikasi sistem. Rencana pengujian dapat dimulai sedini mungkin di proses pengembangan perangkat lunak.

Pada pengujian *black box,* mencoba beragam masukan dan memeriksa keluaran yang dihasilkan. Teknik pengujian *black box* juga dapat digunakan untuk pengujian berbasis skenario, dimana isi dalam sistem mungkin tidak tersedia untuk diinspeksi tapi masukan dan keluaran yang didefinisikan dengan *use case* dan informasi analisis yang lain.

Adapun faktor-faktor pengujian *black-box* adalah :

1. ***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.

2. ***Service levels***

   Menekankan bahwa hasil yang diinginkan didapat dalam waktu yang diinginkan oleh *user*. Untuk mencapai keinginan tersebut, harus dilakukan penyesuaian antara keinginan *user* dengan sumber daya yang ada.

3. *Ease of use*

Menekankan perluasan usaha yang diminta untuk belajar, mengoprasikan dan menyiapakan inputan, dan menginterpretasikan output dari sistem. Faktor ini tersangkut dengan *usability* system terhadap interaksi antara manusia dan system.

4. *Authorization*

Menjamin data diproses sesuai dengan ketentuan manajemen. *Authorization* menyangkut proses transaksi secara umum dan khusus.

Fokus Pengujian *Black box testing* yaitu sebagai berikut :

1. Menguji fungsi-fungsi khusus dari aplikasi.
2. *Test input* dan *output* untuk fungsi yang ada tanpa memperhatikan prosesnya.

Beberapa jenis kesalahan yang dapat di identifikasi :

1. Fungsi tidak benar atau hilang,
2. Kesalahan interface,
3. Kesalahan pada struktur data atau akses database external,
4. Kesalahan pada kinerja,
5. Kesalahan pada saat inisialisasi dan terminasi,
6. Kesensitifan sistem terhadap nilai input tertentu,
7. Batasan dari suatu data.