BAB II

TINJAUAN PUSTAKA DAN LANDASAN TEORI

2.1 Tinjauan Pusataka

Alfian.s, Mgs. Abd. Rahman Faja dan M. Haviz Irfani, (2013), mengembangkan aplikasi tentang *"Aplikasi Koperasi Simpan Pinjam Pada PT Elnusa Petrofin Palembang"*. Mereka mengembangkan aplikasi dengan menggunakan metodologi *iterasi*. Pembuatan aplikasi tersebut menggunakan bahasa pemrograman *Visual Basic* dan *Microsoft SQL Server* sebagai basis data. Sistem yang dibagun bertujuan agar dapat meningkatkan kinerja *staff* dan meminimalkan kesalahan dalam mencatat data anggota, data transaksi, serta pembuatan laporan.

Skripsi yang berjudul "*Pengembangan Aplikasi Desktop Sistem Informasi Bmt (Studi Kasus Di Bmt Insan Madani Kalibayem)*", Oleh Hilda Helty Pratiwi (2016). Sistem yang dibangun dengan bahasa pemrograman *Java* dan *Mysql* sebagai database ini dapat mencatat dan menampilkan seluruh transaksi simpanan anggota, pinjaman anggota, dan dapat mencatat berapa kali transaksi angsuran dari pinjaman anggota, serta dapat mencetak laporan transaksi secara berkala.

Penelitian yang berjudul "*Pengembangan Sistem Informasi Ksp Di Kpri Makmur Sejahtera Berbasis Desktop*" Metode pengembangan sistem dalam penelitian ini menerapkan metodologi berorientasi objek yaitu *Unified Approach* (UA) dengan menggunakan *Unified Modelling Language* (UML) untuk memodelkan sebuah sistem. Penelitian ini bertujuan untuk pengembangan sistem komputerisasi dan mempecepat pekerjaan dan pencarian data serta mempermudah dalam pembuatan berbagai laporan. Gin Gin Ichwaniadi Ginanjar , Asep Deddy Supriatna (2015)

Berdasarkan penelitian yang sudah ada, maka penulis menambah beberapa spesifikasi yang menjadi perbedaan dengan sistem yang dikembangkan. Perbedaan tersebut adalah sebagai berikut:

1. Aplikasi dapat melakukan perhitungan transaksi pinjaman dan angsuran secara otomatis
2. Aplikasi dapat menampilkan jumlah saldo anggota pada transaksi simpanan wajib dan simpanan sukarela
3. Aplikasi dapat melakukan transaksi peranikan simpanan
4. Aplikasi dapat menampilkan dan mencetak laporan transaksi penarikan simpanan anggota.

2.2 Landasan Teori

Untuk mendukung pembuatan laporan ini, maka perlu dikemukakan hal atau teori-teori yang berkaitan dengan permasalahan dan ruang lingkup pembahasan. Teori-teori yang berkaitan akan dibahas sebagai landasan dalam membangun laporan ini.

2.2.1 Pengertian Aplikasi

Aplikasi adalah suatu subkelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan suatu tugas yang diinginkan pengguna. Biasanya dibandingkan dengan perangkat lunak sistem yang mengintegrasikan berbagai kemampuan komputer, tapi tidak secara langsung menerapkan kemampuan tersebut untuk mengerjakan suatu tugas yang menguntungkan pengguna. Aplikasi adalah penerapan, penggunaan atau penambahan. Dari pengertian diatas dapat disimpulkan bahwa aplikasi adalah berbagai bentuk pekerjaan atau tugas-tugas tertentu seperti penerapan, penggunaan dan penambahan data. (Anisyah, 2000:3).

2.2.2 Aplikasi Berbasis *Desktop*

Menurut Konixbam (2009) *Desktop Based Application* adalah suatu aplikasi yang dapat berjalan sendiri atau independen tanpa menggunakan *browser* atau koneksi Internet di suatu komputer otonom dengan sistem operasi atau *platform* tertentu. Aplikasi *Desktop* difokuskan kepada aplikasi yang lebih independen. Tentu Tujuannya adalah untuk mempermudahkan para pengguna aplikasi *desktop* dalam hal memodifikasi pengaturan aplikasi sehingga efektifitas, efesinsi waktu, dana, dan tenaga dapat lebih ditekankan semaksimal mungkin.

2.2.3 Pengertian Sistem Informasi

Bodnar dan Hopwood (2000:4) mengungkapkan bahwa istilah sistem informasi menganjurkan penggunaan teknologi komputer di dalam organisasi untuk menyajikan informasi kepada pemakai. Sistem informasi "berbasis komputer" merupakan sekelompok perangkat keras dan perangkat lunak yang dirancang untuk mengubah data menjadi informasi yang bermanfaat.

Penulis lain, Janner Simarmata (2004) menyatakan: informasi adalah pengumpulan atau pengolahan data untuk memberikan pengetahuan atau keterangan. Informasi merupakan sesuatu yang nyata atau setengah nyata yang dapat mengurangi derajat ketidakpastian tentang suatu keadaan atau kejadian. Dapat juga di artikan sebagai data yang telah di manipulasi sehingga dapat berguna bagi seseorang. Informasi juga meliputi data atau sumber daya yang tersedia dalam suatu perusahaan yang dapat mempengaruhi hasil kinerja bagian-bagian atau elemen-elemen yang ada dalam perusahaan. Adapun sumber daya utama suatu perusahaan dapat terdiri dari manusia, material, mesin, uang yang memiliki wujud fisik dan dapat di sentuh dan jenis sumber daya informasi yang memiliki nilai dari apa yang diwakili (bukan dalam bentuk wujudnya). Informasi yang berkualitas memiliki kriteria: lengkap, akurat, tepat waktu dan relevan. Janner Simarmata (2004) menjelaskan sistem informasi dengan kriteria sebagai berikut:

1. Lengkap berarti dengan informasi yang diterima, seorang penerima informasi tersebut mendapat gambaran yang di hadapi atau solusinya.

2. Akurat berarti bebas dari kesalahan- kesalahan yang menyesatkan yang dapat di timbulkan oleh gangguan-gangguan yang dapat merusak informasi pada saat penyampaiannya.
3. Tepat waktu berarti informasi tersedia pada saat di butuhkan karena informasi merupakan dasar dari pengambilan keputusan.
4. Relevan berarti memberikan manfaat bagi penerimanya.

2.2.4 *Systems Development Life Cycle* (SDLC)

Dalam alur penelitian, metode yang digunakan adalah model SDLC (*Software Development Life Cycle*). SDLC adalah suatu kerangka yang menggambarkan beberapa kegiatan yang dilakukan melalui beberapa tahap dalam pembuatan sebuah *software* (Fatta, 2007). Selain itu, SDLC juga penting untuk proses *maintenance software* itu sendiri. Model SDLC yang dipakai dalam pengembangan aplikasi adalah model *Waterfall*. *Waterfall* model adalah sebuah contoh dari proses perencanaan dimana semua proses kegiatan harus terlebih dahulu direncanakan dan dijadwalkan sebelum dikerjakan. *Waterfall* Model atau *ClassicLife Cycle* merupakan model yang paling banyak dipakai dalam *SoftwareEngineering* (SE) (Sommerville 2011). Disebut *waterfall* karena tahap demi tahap yang dilalui harus menunggu selesainya tahap sebelumnya dan berjalan berurutan.

SDLC membagi pengembangan perangkat lunak dalam 6 fase, yaitu:

a. Analisis kebutuhan dan spesifikasi (*Requirement analysis and specification*).
b. Desain (*design*).
c. Pengodean (*coding*).
d. Pengujian (*testing*).
e. Pemeliharaan (*maintenance).*

Dalam membangun sebuah web atau aplikasi, setiap jenisnya memiliki persyaratan yang berbeda-beda. Hal ini diperlukan untuk menyesuaikan fase SDLC memenuhi kebutuhan yang lebih spesifik dari web atau aplikasi. Dalam proses

penyesuaian fase SDLC memunculkan berbagai pendekatan pengembangan perangkat lunak. Berikut pendekatan-pendekatan dalam SDLC:

a. *Waterfall approach*, yaitu pendekatan yang menjelaskan proses pengembangan perangkat lunak dalam aliran linier berurutan.
b. *Prototyping approach*, yaitu pendekatan yang juga dikenal sebagai pendekatan evolusioner.
c. *Spiral approach*, yaitu pendekatan untuk mengembangkan perangkat lunak yang diliris dalam berbagai versi.
d. *Win-win spiral approach*, yaitu pendekatan dari pendalaman pendekatan spiral approach, yang digunakan saat perangkat lunak memiliki tenggat waktu rilis.
e. *Incremental approach*, yaitu pendekatan yang membagi persyaratan (*requirement*) menjadi beberapa unit fungsional.

2.3 Teknologi Pengembangan Aplikasi

2.3.1 *Unified Markup Language* (UML)

Menurut Rosa, Shalahuddin (2011a:118) "*Unified Modelling Language* (UML) merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung". Menurut munawar (2005 : 17), *Unified Modeling Language* (UML) adalah salah satu alat bantu yang sangat handal dalam perkembangan sistem beroritasi objek. Hal ini disebabkan karena UML menyediakan bahasa pemodelan *visual* yang memungkinkan bagi pengembang *system* untuk membuat cetak biru atau visi mereka dalam bentuk yang baku, mudah dimengerti serta dilengkapi dengan mekanisme yang efektif untuk berbagi (*sharing*) dan mengkomunikasikan rancangan mereka dengan yang lain.

Model UML yang dipakai dalam pengembangan aplikasi antara lain adalah *Use Case* Diagram, *Activity* Diagram, dan *Class* Diagram

a. *Use Case* Diagram

   Use Case Diagram mendeskripsikan sebuah interaksi antara satu atau lebih actor dengan sistem yang dibuat. Dapat dikatakan *use case* digunakan untuk mengetahui fungsi yang ada di dalam sistem dan siapa saja yang berhak menggunakan.

b. *Activity* Diagram

   Activity Diagram merupakan diagram yang digunakan untuk menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis.

c. *Class* Diagram

   Class Diagram merupakan diagram yang digunakan untuk menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan constraint yang berhubungan dengan objek yang dikoneksikan. Class Diagram secara khas meliputi: Nama Kelas (*Class Name*), Atribut *(Attribute*s), Operasi *(Operations),* dan Relasi *(Relationships).*

2.3.2 Bahasa Pemrograman C#

C# (dibaca: C *sharp*) adalah bahasa pemrograman yang diciptakan oleh *Microsoft* yang dikembangkan dibawah kepemimpinan Anders Hejlsberg yang telah menciptakan berbagai macam bahasa pemrograman termasuk Borland Turbo C++ dan orland Delphi. Bahasa C# juga telah di standarisasi secara internasional oleh ECMA. Seperti halnya bahasa pemrograman yang lain, C# bisa digunakan untuk membangun berbagai macam jenis aplikasi, seperti aplikasi berbasis windows (*desktop*) dan aplikasi berbasis *web* serta aplikasi berbasis *web services*.

Kelebihan C#

a. Sederhana (*Simple*)

   C# bersifat sederhana, karena bahasa ini di dasarkan kepada bahasa C dan C++. Jika anda familiar dengan C dan C++ atau bahkan Java, anda akan menemukan aspek aspek yang begitu *familiar*, seperti *statements, expression, operators,* dan beberapa fungsi yang diadopsi langsung dari C

dan C++, tetapi dengan berbagai perbaikan yang membuat bahasanya menjadi lebih sederhana.

b. Object Oriented Language

C# memenuhi syarat-syarat sebagai sebuah bahasa pemrograman yang bersifat *Object Oriented*, yaitu *encapsulation, inheritance* dan *polymorphism*.

c. Powerfull dan Fleksibel

C# bisa di gunakan untuk membuat berbagai macam aplikasi, seperti aplikasi pengolah kata, grafik, *spreadsheets*, atau bahkan membuat kompiler untuk sebuah bahasa permrograman.

d. Efisien

C# tidak memiliki terlalu banyak *keyword,* sehingga dapat mengurangi kerumitan.

e. Modular Kode (Abid Alfan Syakir, 2015)

C# di tulis dengan pembagian masing *Class-Class* (*classes*) yang terdiri dari beberapa *routines* yang disebut sebagai *member methods*. *Class-Class* dan metode metode ini dapat digunakan kembali oleh program atau aplikasi lain. Hanya dengan memberikan informasi yang di butuhkan oleh *Class* dan metode yang di maksud, maka kita akan dapat membuat suata kode yang dapat di gunakan oleh satu atau beberapa aplikasi dan program (*reusable code*).

2.3.4 *Microsoft SQL Server*

SQL Server adalah sebuah sistem manajemen *database* relasional yang memiliki kegunaan merancang sebuah aplikasi yang berhubungan dengan arsitektur *server* atau *client*. Pada umumnya SQL *Server* selalu di pergunakan di dunia bisnis dengan kelengkapan basis yang jauh lebih banyak namun memiliki skala kecil hingga skala menengah, akan tetapi sekarang ini lebih berkembang lagi sehingga menggunakan basis data dengan skala yang cukup besar.

Kegunaan dan fungsi dari SQL Antara lain :

a. Bisa membuat tabel baru dalam sebuah *database* tertentu
b. Mampu membuat prosedur sederhana yang bisa di simpan dalam *database*
c. Membuat pemandangan yang jauh lebih baik dalam *database* yang tersedia
d. Mampu mengatur dan melaksanakan hak akses yang terdapat pada tabel, pandangan maupun pada prosedur *database* tertentu
e. Mampu membuat *database* yang baru dan bisa di *upgrade*
f. Menghapus catatan terdahulu dari *database*
g. Mampu dalam memperbaharui catatan yang terdapat pada *database*
h. Memperbaharui catatan yang sudah tersedia dari *database*
i. Dengan cepat memperbaiki catatan yang terdapat pada *database*
j. Bisa dengan mudah menyisipkan catatan-catatan yang terdapat pada *database*

2.3.5 *Visual Studio*

Microsoft Visual Studio merupakan sebuah perangkat lunak lengkap (*suite*) yang dapat digunakan untuk melakukan pengembangan aplikasi, baik itu aplikasi bisnis, aplikasi personal, ataupun komponen aplikasinya, dalam bentuk aplikasi *console*, aplikasi *windows* ataupun aplikasi web.

Visual Studio mencakup kompiler, SDK, *Integrated Development Environmen*t (IDE), dan dokumentasi (umumnya berupa MSDN *Library*). Kompiler yang dimasukkan ke dalam paket *Visual Studio* antara lain *Visual C++, Visual c#, Visual Basic, Visual Basic.NET, Visual InterDev, Visual J++, Visual J#,*

Visual FoxPro, dan *Visual SourceSafe. Microsoft Visual Studio* dapat digunakan untuk mengembangkan aplikasi dalam *native code* (dalam bentuk bahasa mesin yang berjalan diatas *Windows*) ataupun *managed code* (dalam bentuk *Microsoft Intermediate Languange* diatas *.NET Framework*) selain itu, *visual studio* juga dapat digunakan untuk 28 mengembangkan aplikasi *Silverlight*, aplikasi *Window Mobile* (yang berjalan diatas *.NET Compact Framework*).

2.3.6 *Crystal Report*

Cristal Report dirancang untuk untuk membuat laporan yang dapat digunakan dalam bahasa pemograman berbasis windows, seperti Borland Delphi, Visual basic, visual C/ C++, dan visual Interdev. Menurut Hadi S, (2003) ada beberapa kelebihan dari *Crystal Report* ini adalah :

1. Dari segi pembuatan laporan, tiadak terlalu rumit yang memungkinkan para programmer pemula sekalipun dapat membuat laporan yang sederhana tanpa melibatkan banyak kode pemograman.
2. *Integrasi* dengan bahasa pemograman lain yang memungkinkan dapat digunakan oleh banyak *programmer* dengan masing-masing keahlian.
3. Fasilitas impor hasil laporan yang mendukung format-format popular seperti *Microsoft Word, Microsoft Excell, Access, Adobe Acrobat Reader, HTML* dan sebagainya.

2.3.7 Pengujian Aplikasi

Software testing adalah aktivitas menjalankan serangkaian eksekusi yang dinamis pada program *software* setelah *source code software* tersebut telah dikembangkan. *Software testing* dilakukan untuk menemukan dan memperbaiki sebanyak mungkin potensi kesalahan sebelum *software* tersebut digunakan oleh pelanggan atau *end user*. Dari definisi di atas, *testing* merupakan aktivitas atau proses memeriksa dan mengevaluasi sistem dengan tujuan untuk menemukan kesalahan pada sistem tersebut.

Menurut Perry (2006:69), *functional testing* juga dapat disebut sebagai *black box testing* karena tidak ada pengetahuan dari logika internal sistem yang

digunakan untuk membuat *test case*. Biasanya dalam pengujian fungsional, teknik validasi lebih digunakan untuk melakukan pengujian. Tim penguji melakukan validasi terhadap *function key* yang ada dan mengobservasi hasilnya.

Kelebihan dari *functional testing*:

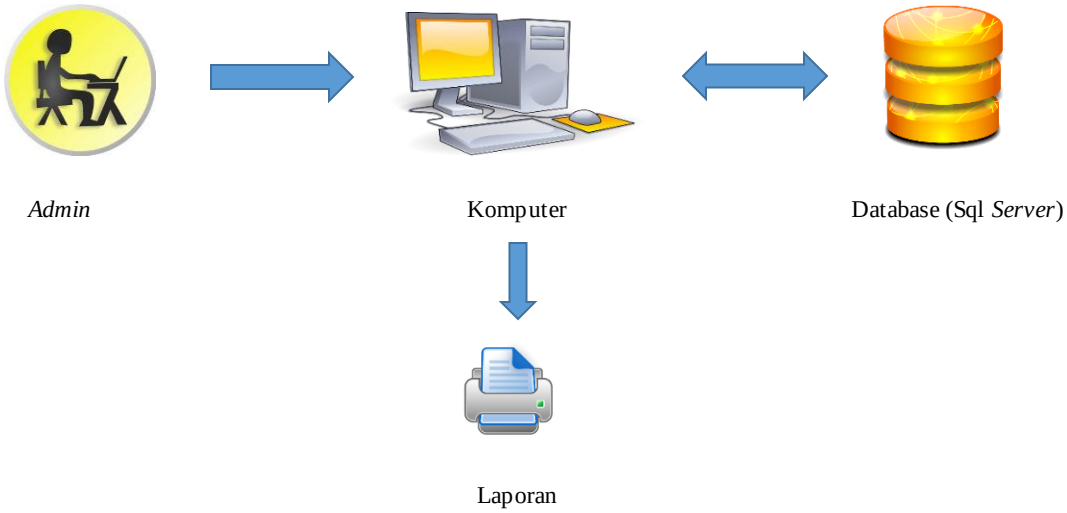
- Melakukan simulasi terhadap kegunaan sistem yang sebenarnya.
- Tidak membuat asumsi terhadap struktur sistem.

Kekurangan dari *functional testing*:

- Dapat berpotensi menghilangkan logika yang salah dalam *software*.
- Memungkinkan terjadinya pengujian yang redundan.

2.3.8 Arsitektur Sistem

Dalam mengembangkan aplikasi diperlukan perangcangan arsitektur perangkat lunak yang bertujuan untuk menggambarkan bagaimana sistem dikembangkan dan dijalankan. Arsitektur perangkat lunak pada aplikasi dapat dilihat pada Gambar 2.1.



Gambar 2 .1 Arsitektur Sistem