BAB II

LANDASAN TEORI

2.1 Tinjauan Pustaka

A. Konsep Dasar Sistem Informasi

Menurut Supriyanto (2007:243), ”Informasi adalah data yang telah diolah menjadi suatu bentuk yang berarti bagi penerimanya dan bermanfaat dalam mengambil keputusan saat ini atau mendatang”. Suatu informasi dapat dikatakan memiliki manfaat dalam proses pengambilan keputusan apabila informasi tersebut mempunyai kualitas dan nilai. Berikut ini criteria informasi yang memiliki kualitas yang baik diataranya :

1. Akurat, yang berarti informasi harus tidak bisa atau menyesatkan dan bebas dari kesalahan.
2. Tepat pada waktunya, yang berarti informasi yang sampai kepada penerima tidak boleh terlambat. Mahalnya nilai informasi saat ini adalah karena harus cepatnya informasi tersebut didapatkan, sehingga diperlukan teknologi mutakhir untuk mendapatkan, mengolah, dan mengirimkan secara cepat.
3. Relevan, yang berarti informasi harus mempunyai manfaat bagi pihak yang menerimanya.

Suatu informasi dikatakan tidak berkualitas dapat disebabkan oleh beberapa hal berikut ini:

1. Metode pengukuran dan pengumpulan data yang salah.
2. Tidak mengikuti prosedur pengolahan data yang benar. Data hilang atau tidak terolah
3. Kesalahan mencatat atau megoreksi data.
4. File historis/induk yang salah (atau keliru memilih file gistoris).
5. Kesalahan dalam prosedur pengolahan (misal kesalahan program komputer).
6. Kesalahan yang disengaja.

B. Konsep Dasar Model Pengembangan Sistem

Model pengembangan sistem yang penulis gunakan dalam skripsi ini adalah *waterfall.* Menurut Pressman (2010:39) "*Waterfall* adalah model klasik yang bersifat sistematik, berurutan dalam membangun *software*."

Kelebihan dari model ini adalah selain karena mengaplikasikan menggunakan model ini mudah, kelebihan dari model ini adalah ketika semua kebutuhan sistem dapat didefinisikan secara utuh, eksplisit, dan benar diawal proyek, maka *Software Engineering* (SE) dapat berjalan dengan baik dan tanpa masalah. Meskipun seringkali kebutuhan sistem tidak dapat didefinisikan seeksplisit yang diingikan, tetapi paling tidak, problem pada kebutuhan sistem diawal proyek lebih ekonomis dalam hal uang (lebih murah), usaha, dan waktu yang terbuang sedikit jika dibandingkan problem yang muncul pada tahap-tahap selanjutnya.

Kekurangan yang utama dari model ini adalah kesulitan dalam mengakomodasi perubahan setelah proses dijalani. *Fase* sebelumnya harus lengkap dan selesai sebelum mengerjakan *fase* berikutnya.

Masalah dengan *waterfall* :

1. Perubahan sulit dilakukan karena sikapnya yang kaku.

2. Karena sifat kakunya model ini cocok ketika kebutuhan dikumpulkan secara lengkap sehingga perubahan bisa ditekan sekecil mungkin. Tapi pada kenyataanya jarang sekali konsumen/ pengguna yang bisa memberikan kebutuhan secara lengkap, perubahan kebutuhan adalah sesuatu yang wajar terjadi**.**

3. *Waterfall* pada umumnya digunakan untuk rekayasa sistem yang besar yaitu dengan proyek yang dikerjakan dibeberapa tempat berbeda, dan dibagi menjadi beberapa proyek.

B. Konsep Dasar Programan

Menurut Sukamto dan Shalahuddin (2013:67) “Konsep atau paradigma atau sudut pandang programan yang membagi-bagi program berdasarkan fungsi-fungsi atau prosedur-prosedur yang dibutuhkan program computer.” Modul-modul (pembagian program) biasanya dibuat dengan mengelompokan fungsi-fungsi dan prosedur-prosedur ditulis secara sekuensial atau terurut dari atas ke bawah sesuai dengan kebergantungan antar fungsi atau prosedur (fungsi atau prosedur yang dapat dipakai oleh fungsi prosedur di bawahnya harus yang sudah ditulis atau dideklarasikan di atasnya).

Tujuan teknik pemerograman dalam pemerograman terstruktur adalah :

a. Meningkatkan kehandalan program.

b. Agar program mudah di baca dan di telusuri.

c. Menyederhanakan kerumitan dan pemeliharaan program.

d. Meningkatkan produktivitas pemerogram.

Standar teknik pemecahan masalah dalam pemerograman terstruktur adalah sebagai berikut :

1. Modular

Dalam pemrograman secara modular, suatu program akan dipilih kedalam sejumlah modul, dimana setiap modul menjalankan fungsinya sendiri. Tentunya fungsi yang dijalankan oleh setiap modul sangat terbatas sesuai dengan ruang lingkup yang akan dikerjakan. Dengan adanya sejumlah modul program ini tentu saja kesalahan yang timbul dapat dikurangi. Setiap program tentu akan memiliki program utamanya, yang kemudian akan memanggil sejumlah modul-modul yang ada.

2. *Top Down* Analis

Menurut Sukamto dan Shalahuddin (2013:297) "Pengujian *top-down integration* atau integrasi dari atas ke bawah merupakan pendekatan bertahap (*incremental*) untuk mengontruksi struktur program. Modul diitegrasi berdasarkan hirarki,

dimulai dari modul yang lebih besar lalu didekomposisi ke modul yang lebih kecil."

3. Buttom-Up

Menurut Sukamto dan Shalahuddin (2013:281) "Pengujian integrasi dari bawah ke atas (*bottom-up integration*) memulai pengujian dari modul yang paling kecil ke modul yang lebih besar. Pengujian dari bawah ke atas sering dilakukan untuk pengembangan perangkat lunak yang tidak menggunakan alur prototype sehingga perangkat lunak dibangun dari modul-modul yang besar sesuai dengan hirarki pemakainnya."

C. UML (*unfied Modelling Language*)

Menurut Sukamto dan Shalahuddin (2013:133) "UML (*unified Modelling Language*) adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement*, membuat analisis & desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek."

UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak. UML merupakan bahasa visual untuk melakukan pemodelan. Jadi pengguna UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataanya UML paling banyak digunakan pada metodologi berorientasi objek.

UML (*unified Modelling Language*) untuk pemerograman terstruktur mendefinisikan diagram-diagram sebagai berikut :

1. ***Use Case Diagram***

Menurut Sukamto dan Shalahuddin (2013:155) "*Use case* atau diagram *use case* merupakan pemodelan untuk melakukan (*behavior*) sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu."

2. *Activity Diagram*

Menurut Sukamto dan Shalahuddin (2013:161) "Diagram aktivitas atau *activity* diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan actor, jadi aktivitas yang dapat dilakukan oleh sistem."

3. *Component Diagram*

Menurut Sukamto dan Shalahuddin (2013:148) "Diagram komponen atau *component* diagram dibuat untuk menunjukan organisai dan ketegantungan diantara kumpulan

komponen dalam sebuah sistem. Diagram komponen focus pada komponen sistem yang dibutuhkan dan ada di dalam sistem."

4. Diagram *Deployment* (*Deployment Diagram*)

Menurut Sukamto dan Shalahuddin (2013:154) "Diagram *deployment* atau *deployment* diagram menunjukan konfigurasi komponen dan proses eksekusi aplikasi.

D. ERD (*entity Relationship Diagram*)

Pada dasarnya "ERD (*Entity Relationship Diagram*) adalah sebuah diagram yang secara konseptual memetakan hubungan antar penyimpanan pada diagram DFD (*Data Flow Diagram*)" (Wardani dan Kusuma, 2010:30). Dimana DFD digunakan untuk menggambarkan aliran data dalam sebuah sistem informasi yang dikembangkan menggunakan computer. ERD ini digunakan untuk melakukan permodelan terhadap struktur data dan hubungannya. Penggunaan ERD ini dilakukan untuk mengurangi tingkat kerumitan penyusun sebuah *database* yang baik.

Entity dapat berarti sebuah objek yang dapat dibedakan dengan objek lainnya. Dalam dunia *databse entity* memiliki atribut yang menjelaskan karakteristik dari *entity* tersebut. Ada dua macam atribut yang dikenal dalam *entity* yaitu atribut yang berperan sebagai kunci primer dan atribut deskriptif. Hal ini berarti setiap *entity*

memiliki himpunan yang diperlukan sebuah *primarykey* untuk membedakan anggota-anggotanya dalam himpunan tersebut.

Atribut dalam ERD dapat memiliki sifat-sifat menurut Wardani dan Kusuma (2010:30) sebagai berikut:

1. *Atomic*, menggambarkan atribut berisi nilai yang spesifik dan tidak dapat dipecah lagi.
2. *Multivalued*, menandakan atribut bisa memiliki lebih dari satu nilai untuk tiap *entity* tertentu.
3. *Composite*, gabungan dari beberapa atribut yang bersifat atomic.

Ada beberapa derajat relasi yang dapat terjadi pada ERD menurut Wardani dan Kusuma (2013:35), yaitu:

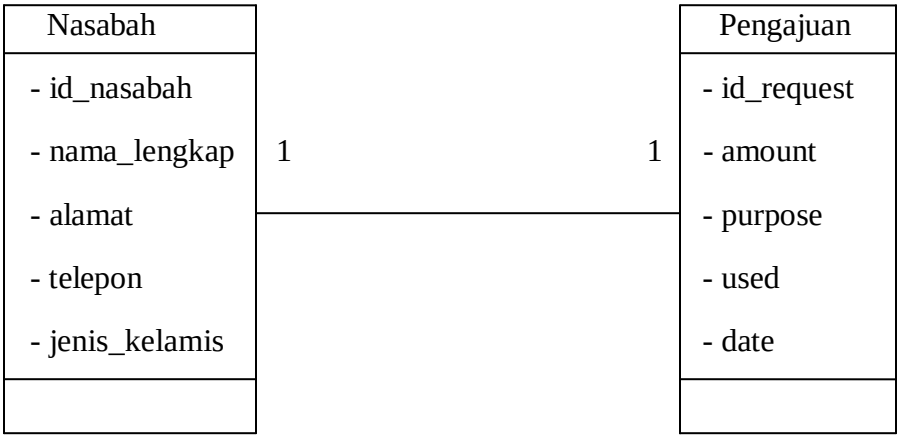
1. *One to one* (1-1), menggambarkan bahwa antara satu anggota *entity* A hanya dapat berhubungan dengan satu anggota *entity* B.
2. *One to many* (1-N), menggambarkan bahwa satu anggota *entity* A dapat memiliki hubungan dengan lebih dari satu anggota *entity* B.
3. *Many to many* (N-N), Menggambarkan bahwa lebih dari satu anggota *entity* A dapat memiliki hubungan dengan lebih dari satu anggota *entity* B.

E. LRS (*Logical Record Structure*)

LRS (*logical Record Structure*) adalah respresentasi dari struktur *record-record* pada table-tabel yang terbentuk dari hasil antar himpunan entitas. Menentukan kardinalitas, jumlah *table* dan *Foreign Key* (FK).

Berikut contoh serta cara kerja dari LRS berdasarkan kardinalitas antar table:

1. *One to one*

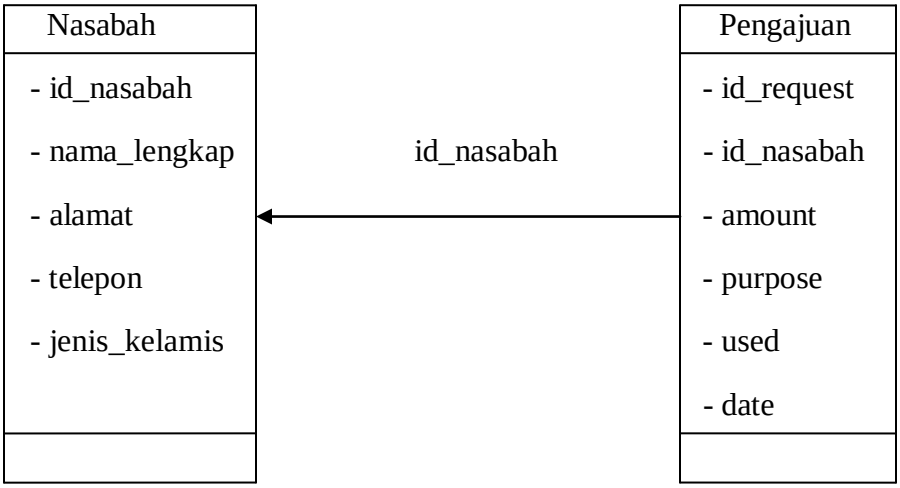


Sumber: Wardani dan Kusuma (2010:31)

Gambar II.1

Relasi *One to one*

Gambar di atas menunjukkan relasi dengan kardinalitas *one to one* Karen 1 nasabah hanya bisa melakukan 1 pengajuan pembiayaan. Relasi 1:1 yang terjadi pada contoh di atas membentuk table LRS sebagai berikut:

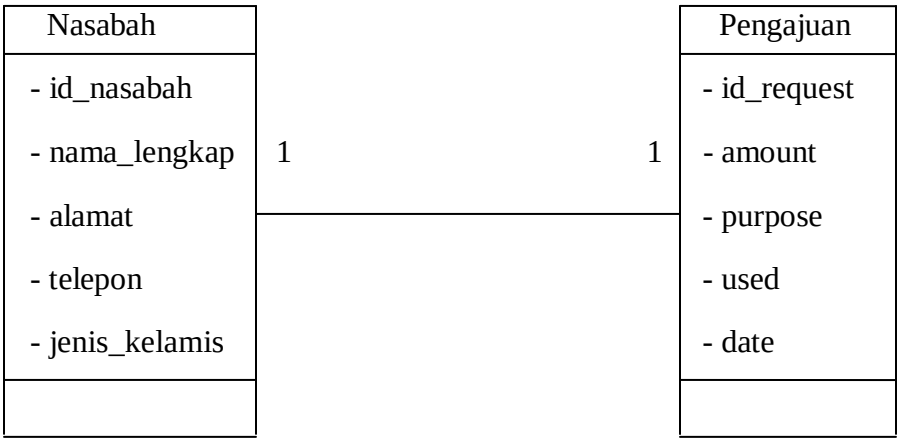


Sumbe: Wardani dan Kusuma (2010:31)

Gambar II.2

LRS *One To One*

2.*One to Many*

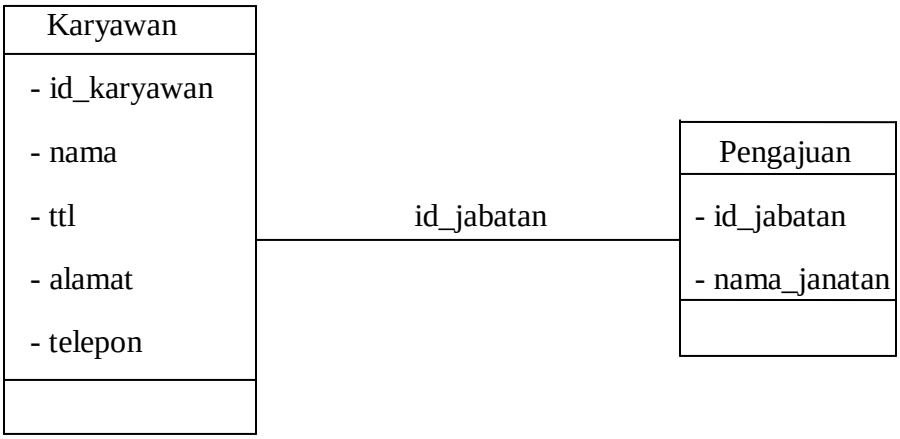


Sumber: Wardani dan Kusuma (2010:31)

Gambar II.3

Relasi *One To Many*

Gambar di atas menunjukkan relasi dengan kardinalitas 1 jabatan dapat dimilki oleh banyak karyawan, akan tetapi karyawan hanya bisa mengakses atau memiliki satu jabatan saja. Relasi 1:* yang terjadi pada contoh di atas membentuk table LRS sebagai berikut:

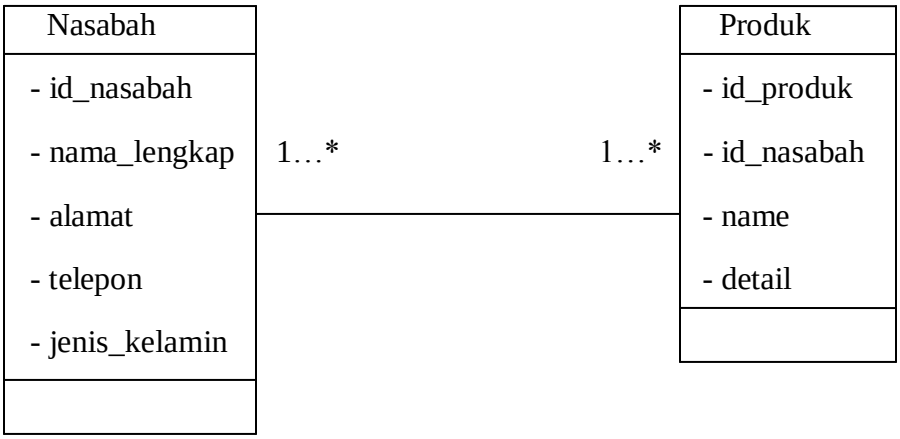


Sumber: Wardani dan Kusuma (2010:31)

Gambar II.4

LRS *One To Many*

3. *Many To Many*

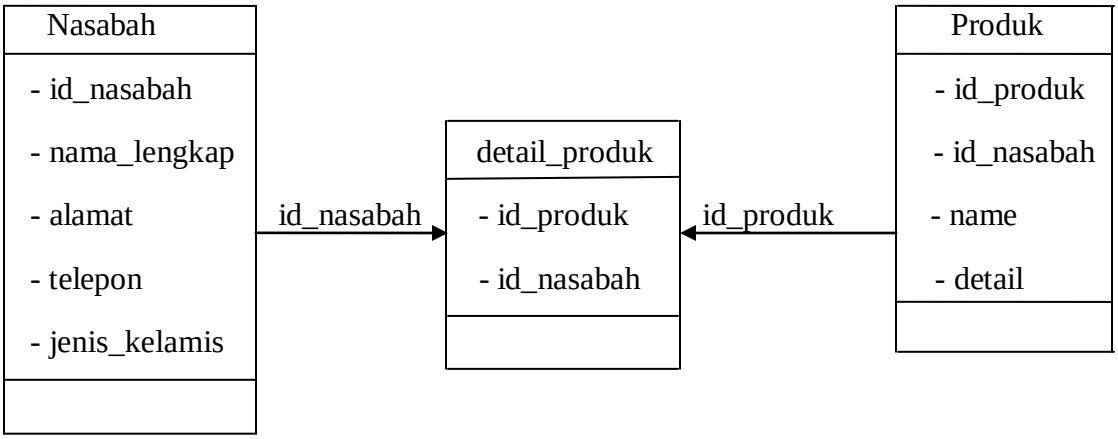


Sumber: Wardani dan Kusuma (2013:31)

Gambar II.5

Relasi *Many To Many*

Gambar di atas menunjukkan relasi dengan kardinalitas 1 nasabah dapat melihat banyak produk dan 1 produk juga dapat diakses oleh banyak nasabah. Relasi *:* yang terjadi pada contoh di atas menghasilkan sebual table baru dapat dilihat pada table LRS sebagai berikut:



Sumber: Wardani dan Kusuma (2010:31)

Gambar II.6

LRS *Many To Many*

2.2 Penelitian Terkait

Adapun tinjauan jurnal untuk penelitian terkait yang penulis lakukan dalam pembuatan skripsi ini yaitu:

Sistem pendidikan yang diterapkan kebanyakan masih menggunakan cara-cara manual dalam aktivitas pendidikannya baik mengenai cara pengolahan data maupun sistem akademik pendidikannya. Untuk memperbaiki kinerja pengolahan data dan akademik maka dibuatkannya sistem yang diharapkan akan menjadi kemudahan untuk sarana pengolahan data akademik sehingga dapat diperoleh hasil yang efektif dan efisien. (Setiawan, dkk, 2013:1)

Dalam Sistem Informasi Akademik Sekolah Menengah Pertama Negeri 2 Talang Empat berbasis *Web* terbagi menjadi beberapa Menu yaitu menu utama, menu *home*, menu Informasi, menu materi, menu berita, menu *galeri*, menu buku tamu dan menu *admin*. Untuk informasi terdiri dari sub menu data kelas, sub menu informasi data Guru, sub menu informasi data Siswa, sub menu infromasi data Nilai, sub menu informasi data Alumni dan sub menu informasi data Pelajaran. (Membara, dkk, 2014:79)

Sistem informasi berbasis *web* ini dirancang sebagai solusi bagi Jurusan Sistem Informasi Fakultas Ilmu Komputer untuk mengelola bagian akademik dalam penyajian laporan nilai serta keaktifan siswa secara cepat dan tepat dibandingkan secara manual sehingga kinerja dalam mencapai pekerjaan dapat diwujudkan secara lebih maksimal.