BAB II
DASAR TEORI

2.1 Pengertian Kredit

Pengertian kredit mempunyai dimensi yang beraneka ragam, dimulai kata kredit yang berasal dari bahasa Yunani "*credere*" yang berarti kepercayaan. Maksudnya pemberi kredit percaya kepada penerima kredit, bahwa kredit yang disalurkannya pasti akan dikembalikan sesuai perjanjian. Sedangkan bagi penerima kredit berarti menerima kepercayaan, sehingga mempunyai kewajiban untuk membayar kembali pinjaman tersebut sesuai dengan jangka waktunya.[10]

Dalam arti yang lebih luas, pengertian Kredit adalah Kemampuan untuk melaksanakan suatu pemberian atau mengadakan suatu pinjaman dengan suatu janji pembayarannya akan dilakukan pada suatu jangka waktu yang disepakati. [8]

Menurut pasal 1 ayat 11 UU No. 10/ 1998, Kredit adalah penyediaan uang atau tagihan yang dapat disamakan dengan itu, berdasarkan persetujuan atau kesepakatan pinjam-meminjam antara bank dengan pihak lain yang mewajibkan pihak peminjam untuk melunasi utangnya setelah jangka waktu tertentu setelah pemberian bunga. [10]

2.2 Sistem Suku Bunga

Secara umum terdapat dua metode dalam perhitungan bunga, yaitu metode *Flat* dan Efektif.

2.2.1 Metode Suku Bunga Efektif (*Sliding Rate*)

Metode ini menghitung bunga yang harus dibayar setiap bulan sesuai dengan saldo pokok pinjaman bulan sebelumnya. Dengan metode ini maka angsuran (pokok + bunga) bulan berikutnya akan lebih kecil dari bulan sebelumnya. Dengan demikian angsuran akan semakin menurun dari waktu ke waktu [1].

Sistem bunga efektif akan lebih berguna untuk pinjaman jangka panjang yang tidak buru-buru dilunasi di tengah jalan, dan sistem bunga efektif ini biasanya diterapkan untuk pinjaman jangka panjang semisal KPR atau kredit investasi. [4].

2.2.2 Metode Suku Bunga Kredit *Flat Rate*

Metode *Flat Rate* (suku bunga tetap) dapat diartikan di mana bunga yang digunakan akan selalu tetap selama jangka waktu kredit. Atau dengan kata lain *Flat Rate* merupakan pembebanan bunga setiap bulan tetap dari jumlah pinjamannya, demikian juga angsuran (cicilan) pokok juga akan tetap sampai pinjaman lunas.

Perhitungan bunga didasarkan pada *plafond* kredit dan besarnya bunga yang dibebankan dialokasikan secara proporsional sesuai dengan jangka waktu kredit. Dengan cara ini, jumlah pembayaran pokok dan bunga kredit setiap bulan sama besarnya. Dengan demikian dalam suku bunga *flat*, bunga dihitung sepanjang waktu kredit dengan mengalikan tingkat bunga dengan saldo awal pinjaman sehingga walaupun kita sudah mencicil pembayaran pokok namun seolah-olah pokok pinjaman adalah masih tetap.[5]

Perhitungan menggunakan metode *Flat Rate* adalah sebagai berikut :

$$\textbf{Angsuran} = \textbf{Cicilan} + \textbf{Bunga Cicilan} \quad \text{.....……..... (2.1)}$$

Dimana:

$$\textbf{Cicilan} = \frac{\textbf{Pinjaman Pokok}}{\textbf{Jangka Waktu (Periode)}} \quad \text{..... ..... (2.2)}$$

$$\text{Bunga Cicilan} = \frac{\text{Pinjaman Pokok x \% Bunga x Tahun}}{\text{Bulan}} \quad \text{... (2.3)}$$

Keterangan :

Angsuran : Jumlah angsuran per bulan

Cicilan : Cicilan pokok per bulan

Bunga Cicilan : Bunga kredit per bulan

Pinjaman Pokok : Pinjaman kredit

% Bunga : Suku bunga pinjaman *Flat*

Jangka Waktu (Periode) : Jangka waktu pinjaman / termin

Tahun : Jangka waktu pinjaman dalam tahun

Bulan : Jangka waktu pinjaman dalam bulan

Adapun kelebihan dari metode *Flat Rate* adalah [7]:

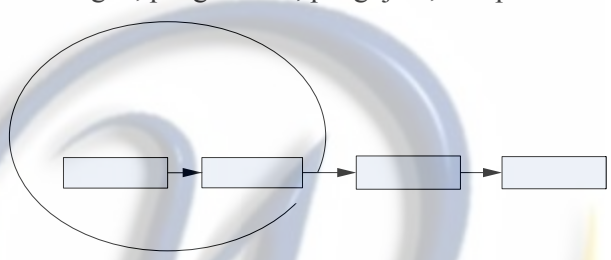
1. Proses perhitungan yang mudah dicerna sehingga memudahkan komunikasi dengan calon *customer*.
2. Prosentasi suku bunga yang lebih rendah dibandingkan dengan metode lainnya membuat *customer* lebih tertarik.
3. Dengan menggunakan metode ini total angsuran bunga dalam rupiah dari awal angsuran sampai akhir angsuran lebih menguntungkan bagi *kreditor* karena dapat memberikan total angsuran yang lebih besar walaupun metode *Flat Rate* memberikan suku bunga yang lebih rendah.

Sedangkan kelemahan dari metode *Flat Rate* adalah jika kita hendak melakukan pelunasan awal, maka besarnya jumlah bunga yang dibayarkan cukup besar karena dihitung dari pokok hutang awal [4].

2.3 Metode Pembangunan Sistem dengan Paradigma *Waterfall*

Ketika kita akan merancang atau membangun suatu sistem informasi tentunya kita membutuhkan suatu metode rekayasa perangkat lunak dengan memilih dan menggunakan paradigma yang sesuai dengan sistem yang akan dibangun tersebut. Metode yang akan digunakan pada pembangunan sistem informasi pelayanan *service* ini adalah metode *Waterfall.*

Metode "*waterfall*" atau yang sering juga disebut metode "*classic life cycle*" ini menggunakan pendekatan yang sistematis dan sekuensial dalam membangun perangkat lunak yang dimulai pada level sistem dan pengembangan melalui tahapan analisis, perancangan, pengkodean, pengujian, dan pemeliharaan.



Gambar 2.1 Metode RPL dengan Metode *Waterfall* [6].

2.3.1 Rekayasa dan pemodelan sistem/informasi

Rekayasa sistem merupakan tahapan yang pertama kali dilakukan, yaitu merumuskan sistem yang akan dibuat. Hal ini bertujuan agar pengembang benar-benar memahami sistem yang akan dibuat dan langkah-langkah serta kebijaksanaan apa saja yang berkaitan dengan pengembangan sistem tersebut.

Rekayasa sistem membutuhkan komunikasi yang intens antara pelanggan dan perekayasa informasi atau sistem. Pelanggan harus memahami sasaran sistem dan dapat menyatakannya dengan jelas. Perekayasa harus tahu pertanyaan apa yang harus dijawab, saran apa yang harus diberikan, dan penelitian seperti apa yang harus dilakukan. Bila komunikasi berhasil dan sebuah model lengkap dari sistem sudah dibuat berarti pondasi yang solid sudah dibangun bagi konstruksi sistem tersebut.

2.3.2 Analisis

Dari rumusan sistem yang diperoleh dari tahap pertama, selanjutnya dilakukan analisis yang berkaitan dengan proses dan data yang diperlukan oleh sistem serta keterkaitannya. Adapun tujuan dilakukannya tahapan ini adalah sebagai berikut :

1. Memahami sistem yang ada pada saat ini.
2. Mendefinisikan permasalahan sistem.
3. Menentukan kebutuhan sistem secara garis besar sebagai persiapan ke tahap perancangan.

2.3.3 Desain

Pada tahap perancangan atau desain ini diberikan gambaran umum yang jelas kepada pengguna dan rancang bangun yang lengkap tentang sistem yang akan dikembangkan kepada pihak-pihak yang terlibat dalam pengembangan sistem. Perancangan disini dilakukan dengan pemodelan menggunakan metode *Data Flow Oriented* dengan *tool Data Flow Diagram* (DFD).

Tahapan perancangan sistem disini dibagi menjadi dua bagian, yaitu perancangan global dan perancangan rinci. Perancangan global dilakukan untuk memberikan gambaran umum kepada pengguna tentang sistem yang dirancang dan sebagai persiapan untuk tahap perancangan rinci. Perancangan rinci dilakukan untuk memberikan gambaran rancang bangun yang lengkap kepada pemrogram dan pihak-pihak lain yang terlibat dalam pengembangan sistem sebagai persiapan untuk tahap implementasi.

2.3.4 Pengkodean

Pada tahap ini sistem yang telah di desain lengkap, diterjemahkan kedalam bahasa pemrograman. Penterjemahan ini merupakan suatu pengkodean dari bahasa tingkat manusia (*High Level Languages*) kedalam bahasa yang dimengerti oleh mesin komputer, dimana setiap kodefikasi tersebut mempunyai arti dan fungsi tertentu.

2.3.5 Pengujian

Pengujian perangkat lunak adalah elemen kritis dari jaminan kualitas perangkat lunak dan merepresentasikan kajian pokok dari spesifikasi, desain, dan pengkodean. Sejumlah aturan yang berfungsi sebagai sasaran pengujian adalah :

1. Pengujian adalah proses eksekusi suatu program dengan maksud menemukan kesalahan.
2. *Test case* yang baik adalah *test case* yang memiliki probabilitas yang tinggi untuk menemukan kesalahan yang belum pernah ditemukan sebelumnya.
3. Pengujian yang sukses adalah pengujian yang mengungkap semua kesalahan yang terjadi sebelumnya.

2.3.6 Pemeliharaan

Setelah dilakukan pengujian dan sistem diyakini telah valid, selanjutnya sistem tersebut didistribusikan kepada pengguna. *Software* akan memburuk dengan semestinya ketika pindah *platform* sistem operasi yang berbeda, kebutuhan baru dari pengguna atau *software* tidak cukup bisa menjawab kebutuhan fungsional.

Pemeliharaan lebih dari sekedar memperbaiki kesalahan, ada empat perbedaan aktifitas dan pemeliharaan yaitu [9] :

1. *Corrective Maintenance*, adalah mengkoreksi kesalahan pada perangkat lunak yang baru terdeteksi pada saat perangkat lunak dipergunakan, atau sama dengan garansi untuk *software*.
2. *Adaptive Maintenance*, adalah penyesuaian dengan lingkungan baru sebagai tuntutan atas perkembangan teknologi komputer, atau memodifikasi *software* untuk *platform* sistem operasi yang berbeda dan memodifikasi *software* untuk mengakomodasi lingkungan internal.
3. *Perfective Maintenance* atau *Enhancement*, adalah menambah dan mengenali fungsi tambahan yang bermanfaat diluar kebutuhan fungsional aslinya, atau bila perangkat lunak sukses

dipergunakan serta memuaskan pemakai maka pemeliharaan ditujukan untuk menambah kemampuannya seperti memberikan fungsi-fungsi tambahan, peningkatan kerja, dsb.

4. *Preventive Maintenance* dan *Reengineering*, adalah pembangunan kembali *software* yang sudah memburuk kinerjanya. Perubahan perangkat lunak dilakukan dengan tujuan agar perangkat lunak lebih mudah dipelihara, dan kehandalan perangkat lunak meningkat. Aktifitas ini lebih mudah disamping aktifitas pemeliharaan yang lain.

2.4 Metode Perancangan Sistem Berorientasi Aliran Data (*Data Flow Diagram*) [3].

Desain sistem adalah suatu fase dimana diperlukan suatu keahlian perencanaan untuk elemen-elemen komputer yang akan menggunakan sistem baru. Ada 2 hal yang perlu diperhatikan dalam desain sistem yaitu pemilihan peralatan dan program komputer untuk sistem yang baru. Ada beberapa alat Bantu yang digunakan dalam desain sistem yaitu DFD (*Data Flow Diagram*), Kamus Data (*Data Dictionary*), Diagram Konteks (*Contex Diagram*), dll.

DFD (*Data Flow Diagram*) adalah suatu model logika data atau proses yang dibuat untuk menggambarkan dari mana asal data dan kemana tujuan data yang keluar dari sistem, dimana data disimpan, proses apa yang menghasilkan data tersebut dan interaksi antara data yang tersimpan dan proses yang dikenakan pada data tersebut.

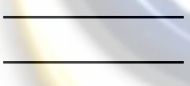
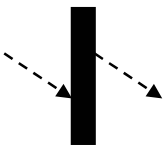
DFD adalah suatu teknik penggambaran atau pemodelan menggunakan notasi-notasi grafis yang menunjukan aliran informasi dan perubahannya yang diterapkan sebagai perubahan atau perpindahan data dari masukan (*input*) menjadi keluaran (*output*).

DFD menggambarkan penyimpanan data dan proses yang mentransformasikan data. DFD menunjukan hubungan antara data pada sistem dan proses pada sistem. Terdapat 2 teknik dasar DFD yang umum dipakai yaitu *Gane and Sarson* dan *Yourdon and De Marco*.

DFD level 0 (*Data Context Diagram*) merepresentasikan elemen-elemen perangkat lunak atau sistem secara keseluruhan sebagai suatu proses dengan data masukan dan keluaran digambarkan sebagai panah yang masuk dan keluar proses. Selanjutnya pada level yang lebih tinggi (1,2,3,… dan seterusnya), proses tersebut dipecah-pecah untuk memperoleh aliran data dan proses yang lebih rinci.

Berikut ini akan diberikan simbol-simbol yang digunakan di dalam diagram alir data (*Data Flow Diagram*).

Tabel 2.1 Simbol-simbol pada Diagram Alir Data (*Data Flow Diagram*)[2]

No.	Simbol	Keterangan
1.		*External entity* (kata benda), sebuah elemen sistem (misalnya perangkat keras, seseorang, program yang lain) atau dapat berupa sumber asal datangnya data (*source*) atau yang menerima data (*sink*).
2.		Proses (kata kerja), menggambarkan kegiatan yang dilakukan oleh sistem, dapat berupa prosedur yang memanipulasi atau mengolah data.
3.		*Data Store* (kata benda), merupakan tempat penyimpanan data untuk direferensi atau diolah lagi lebih lanjut.
4.		*Data Flow* (kata benda), menggambarkan aliran data yang menunjukkan transportasi data atau informasi.
5.		*Control Flow* (kata benda), menggambarkan aliran kontrol data atau kejadian.
6.		*Control Bar*, merupakan acuan untuk CSPEC (*control specification*) yang menjelaskan kebiasaan sistem dan mendefinisikan bagaimana proses diaktifkan sebagai konsekuensi dari suatu kejadian.

2.4.1 DFD *Leveled* [3].

Model ini menggambarkan sistem sebagai jaringan kerja antar fungsi yang berhubungan satu dengan yang lain dengan aliran dan penyimpanan data. Sebagai alat bantu desain sistem, model ini hanya memodelkan sistem dari sudut pandang yaitu sudut pandang fungsi.

Dalam DFD *Leveled* ini akan terjadi penurunan level dimana dalam penurunan level yang lebih rendah harus mampu merepresentasikan proses tersebut ke dalam spesifikasi proses yang jelas. Jadi dalam DFD *leveled* bisa dimulai dari DFD level 0 kemudian turun ke DFD level 1 dan seterusnya. Setiap penuunan hanya dilakukan bila perlu. Dalam penurunan level, tidak semua bagian dari sistem harus diturunkan dengan jumlah level yang sama. Setiap penurunan level menyatakan semakin detail akan suatu proses yang terjadi.

Aliran data yang masuk dan keluar pada suatu proses dilevel X harus berhubungan dengan aliran data yang masuk dan keluar pada level X+1 yang mendefinisikan proses pada level X tersebut.

2.5 Kamus Data *(Data Dictionary)*

Kamus data adalah kumpulan elemen-elemen atau simbol-simbol yang digunakan untuk membantu dalam penggambaran atau pengidentifikasian setiap *field* atau file di dalam sistem. Dapat dikatakan juga kamus data atau *data dictionary* atau disebut juga dengan istilah *system data dictionary* adalah katalog fakta tentang data dan kebutuhan-kebutuhan informasi dari suatu sistem informasi.

Dengan menggunakan kamus data, analis sistem dapat mendefinisikan data yang mengalir di sistem dengan lengkap. Kamus data dibuat pada tahap analisis sistem dan digunakan baik pada tahap analisis maupun pada tahap perancangan sistem. Pada tahap analisis, kamus data dapat digunakan sebagai alat komunikasi antara analis sistem dengan pemakai sistem tentang data yang mengalir di sistem, yaitu tentang data yang masuk ke sistem dan tentang informasi yang dibutuhkan oleh pemakai sistem.

Pada tahap perancangan sistem, kamus data digunakan untuk merancang *input*, merancang laporan-laporan, dan *database*. Kamus data dibuat berdasarkan arus data yang ada di DFD*(Data Flow Diagram).* Arus data di DFD sifatnya adalah global, hanya ditunjukkan nama arus datanya saja. Keterangan lebih lanjut tentang struktur dari suatu arus data di DFD secara lebih terinci dapat dilihat di kamus data.

Simbol-simbol yang ada dalam kamus data adalah sebagai berikut :

Tabel 2.2 Simbol Pada Kamus Data

Simbol	Arti
=	Terdiri dari/didefinisikan sebagai
+	Dan
(…)	Opsional
{ …}	Iterasi/ pengulangan
[…]	Pemillihan dari beberapa alternatif
…	Komentar
@	Identifikasi atribut kunci
ı	Pemisahan pada pemilihan (atau)

2.6 Konsep Basis Data

2.6.1 Model Data *Entity-Relationship* [3]

Model data *entity-relationship* adalah model data yang didasarkan pada sebuah persepsi terhadap sebuah dunia nyata yang di dalamnya terdapat sekumpulan objek dasar dan relasi antar objek-objek tersebut. Pada model *entity-relationship* data diterjemahkan dengan memanfaatkan sejumlah perangkat konseptual menjadi sebuah diagram data yang disebut sebagai *Entity-Relationship Diagram* (E-R Diagram). Tiga hal mendasar dalam model E-R, yaitu himpunan entitas, himpunan relasi dan atribut. Selain itu terdapat batasan-batasan dalam pemetaan data yaitu kardinalitas pemetaan dan ketergantungan ekstensi.

1. Himpunan Entitas

   Sebuah entitas adalah sesuatu atau sebuah objek di dunia nyata yang dapat dibedakan dari objek-objek lain. Himpunan entitas adalah sekumpulan entitas yang mempunyai tipe sama dan memiliki atribut-atribut yang sama. Sebuah entitas direpresentasikan oleh atribut-atributnya.

2. Atribut

   Atribut adalah penjelasan atau gambaran sifat yang dimiliki oleh setiap anggota dari himpunan entitas. Setiap atribut yang dimiliki oleh sebuah himpunan entitas ditunjukkan dengan adanya informasi yang sama disimpan dalam basis data pada setiap entitas anggota himpunan entitas tersebut.

3. Himpunan Relasi

   Relasi menunjukkan adanya hubungan diantara sejumlah entitas yang berasal dari himpunan entitas yang berbeda. Himpunan relasi merupakan kumpulan semua relasi diantara entitas-entitas yang terdapat pada entitas-entitas himpunan tersebut.

4. Kardinalitas Pemetaan

   Kardinalitas pemetaan menunjukkan jumlah maksimum entitas yang dapat berelasi dengan entitas pada himpunan entitas yang lain.

Dalam *entity relationship*, relasi yang bisa terjadi antara 2 file adalah sebagai berikut:

a. *One to one relationship* :

   Hubungan antara file pertama dengan file kedua adalah satu banding satu. Contohnya file guru dengan file siswa dimana guru tersebut mengajar privat. Artinya guru tersebut hanya mengajar 1 siswa itu dan siswa tersebut hanya diajar oleh 1 guru tersebut.

b. *One to many relationship* :

   Hubungan antara file pertama dengan file kedua adalah satu banding banyak. Contohnya adalah file guru dengan file siswa

dimana guru tersebut mengajar di SMU. Artinya guru tersebut mengajar banyak siswa dan siswa yang banyak tersebut hanya diajar oleh 1 guru tersebut.

c. *Many to many relationship* :

Hubungan antara file pertama dan file kedua adalah banyak banding banyak. Contoh dari hubungan *many to many relationship* adalah file dosen dengan file mahasiswa dimana dosen tersebut mengajar di Universitas. Artinya dosen tersebut mengajar banyak mahasiswa dan mahasiswa diajar oleh banyak dosen.

- Kunci

Key adalah satu gabungan beberapa atribut yang dapat membedakan sebuah entitas dengan entitas lain. Beberapa macam *key* antara lain [5] :

1. *Superkey*

   Merupakan satu atau lebih atribut (kumpulan atribut) yang dapat membedakan sebuah entitas di dalam sebuah himpunan entitas.

2. *Candidate key*

   Candidat key adalah *superkey* yang tidak mengandung *superkey* lainnya, yang merupakan subset dari *superkey* pertama.

3. *Primary key*

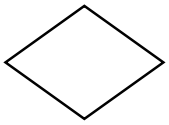
   Primary Key adalah atribut yang dapat digunakan untuk membedakan sebuah entitas dalam sebuah himpunan entitas.

2.6.2 *Entity Relationship Diagram*

Entity Relationship Diagram adalah salah satu metode pemodelan data yang mengidentifikasi objek data dengan notasi grafis. *Entity Relation Diagram* menetapkan semua data yang dimasukkan, disimpan, ditransformasikan, dan diproduksi pada suatu aplikasi. *Entity Relation Diagram* hanya berfokus pada data (sehingga memuaskan prinsip pertama analisis operasional), dengan menunjukan jaringan data yang ada untuk suatu sistem yang diberikan.

Adapun simbol-simbol yang digunakan dalam *Entity-Relationship Diagram* (Diagram E-R) adalah :

Tabel 2.3 Simbol-simbol pada *Entity-Relationship*

No.	Simbol	Keterangan
1.		Simbol *Rectangler*, menggambarkan *entity set* dari suatu basis data
2.		Simbol *Diamond*, menggambarkan *relationship* atau hubungan antar basis data
3.		Simbol Lingkaran, menggambarkan atribut atau *field* dari suatu basis data
4.		Simbol *Line*, menggambarkan hubungan antar atribut dengan *entity set* dan *entity set* dengan *relationship*