BAB II
TINJAUAN PUSTAKA DAN LANDASAN TEORI

BAB II
TINJAUAN PUSTAKA DAN LANDASAN TEORI

2.1 Tinjauan Pustaka

Penelitian sebelumnya dengan judul "Pengaplikasian Sistem Informasi Pencatatan Keuangan pada Koperasi Serba Usaha Karya Mentulik" oleh Astri Ayu Purwati dan kawan-kawan pada tahun 2020, membuat sebuah sistem informasi pencatatan keuangan paka Koperasi Serba Usaha Karya Mentulik. Penelitian ini bertujuan untuk membantu pengurus dalam mengelola kegiatan koperasi. Aplikasi yang dibangun, dapat menangani pencatatan laporan kas masuk dan kas keluar, laporan laba/keuntungan, laporan neraca, dan pembuatan laporan pertanggungjawaban. Penelitian ini menghasilkan aplikasi pelaporan keuangan berbasis desktop yang bertujuan agar pekerjaan dapat lebih aman, akurat dan efisien. Sistem aplikasi keuangan ini dibuat untuk mengoptimalkan pekerjaan dari pengurus KSU Karya Mentulik dalam pelaporan keuangan yang sebelumnya manual menjadi terkomputerisasi[3].

Terdapat penelitian lain yang berjudul "Sistem Informasi Keuangan Pada Perusahaan Koset Elit dengan Metode Waterfall" yang dilakukan oleh I Kadek Andy Asmarajaya dan kawan-kawan pada tahun 2021. Penelitian ini dilakukan untuk mempermudah administrator agar dapat memahami besarnya biaya yang harus dikeluarkan untuk menjamin kelangsungan operasional perusahaan. Aplikasi yang dibangun dapat digunakan untuk mengolah data pengguna, data akun, data transaksi pemasukan, data transaksi pengeluaran, dan menghasilkan laporan pendapatan, laporan pengeluaran, laporan jurnal umum, laporan buku besar, laporan laba rugi, dan laporan neraca. Penelitian ini telah membantu administrator dalam menghasilkan laporan keuangan[4].

Penelitian lainnya berjudul "Pengembangan Sistem Informasi Pengelolaan Keuangan Bagi Usaha Mikro Kecil dan Menengah (UMKM)" oleh Ivan Faizal dan kawan-kawan pada tahun 2021, telah membangun sistem informasi pengelolaan keuangan bagi usaha mikro kecil dan menengah. Penelitian ini bertujuan untuk membuat sistem yang dapat membantu pelaku UMKM dalam pencatatan dan pengelolaan keuangan yang dibangun menggunakan metode *Extreme Programming(XP)*. Aplikasi yang dibangun, dapat mengelola data akun, kelola jurnal umum,

menampilkan buku besar dan neraca saldo. Penelitian ini dapat menghasilkan sistem informasi yang dapat menjadi alat bantu bagi UMKM dalam mengoptimalkan pencatatan dan pengelolan keuangan UMKM[5].

Penelitian yang akan dilakukan penulis memiliki perbedaan dengan sistem yang sudah dibuat pada jurnal penelitian sebelumnya mengenai koperasi simpan pinjam yaitu, sistem yang dikembangkan pada penelitian sebelumnya, hanya menghasilkan laporan keuangan saja, tidak menghasilkan neraca keuangan sampai dengan SHU simpan pinjam. Sedangkan, sistem yang akan penulis kembangkan yaitu, Sistem Informasi Neraca Keuangan Koperasi Bagian Unit Simpan Pinjam Berbasis Desktop pada KOPEGTEL Cilacap bertujuan untuk menghasilkan hasil akhir berupa neraca keuangan dan SHU dari bagian unit simpan pinjam. Metode yang digunakan untuk membuat aplikasi ini adalah metode *prototype.*Sistem ini dibuat menggunakan *software Microsoft Visual Studio* 2010 dengan bahasa pemrograman C#.

Penelitian ini diperlukan adanya teori-teori yang mendasar untuk menunjang proses penelitian ini. Teori-teori yang digunakan dalam penelitian ini adalah :

2.2 Landasan Teori

2.2.1 Sistem Informasi

Sistem informasi merupakan suatu kombinasi teratur dari orang-orang, *hardware*, *software*, jaringan komunikasi dan sumber daya data yang mengumpulkan, mengubah, dan menyebarkan informasi dalam sebuah organisasi[6].

Terdapat pendapat lain mengenai sistem informasi yaitu, suatu sistem didalam suatu organisasi yang mempertemukan kebutuhan pengelolaan transaksi harian, mendukung operasi, bersifat manajerial, dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang dibutuhkan[7].

Sebagai sebuah sistem, sistem informasi adalah kumpulan komponen yang saling bekerja sama mengerjakan sebuah proses untuk mencapai tujuan tertentu. Proses yang dilakukan oleh sistem informasi berupa proses pencatatan transaksi, pemeliharaan basis data, serta penyediaan laporan dan informasi. Transaksi disini merupakan seluruh kegiatan yang perlu dicatat oleh perusahaan atau organisasi, sehingga suatu saat data tersebut dapat diolah menjadi sebuah informasi. Tujuan jangka

pendek dari sebuah sistem informasi yaitu dapat membantu berbagai pihak di dalam perusahaan atau organisasi untuk membuat keputusan.

Sedangkan tujuan jangka panjang dari sistem informasi yaitu menciptakan keunggulan kompetitif, yaitu keunggulan yang tidak atau belum dimiliki oleh perusahaan atau organisasi lain, sehingga dapat memenangkan persaingan. Sistem informasi dapat dilaksanakan dengan baik apabila dilengkapi dengan komponen-komponen yang baik. Salah satu komponen penting yaitu teknologi informasi. Teknologi informasi merupakan berbagai perangkat yang dapat digunakan untuk mencatat data, mengolah transaksi, dan menghasilkan informasi. Perangkat ini meliputi perangkat keras (*hardware*) dan perangkat lunak (*software*)[8].

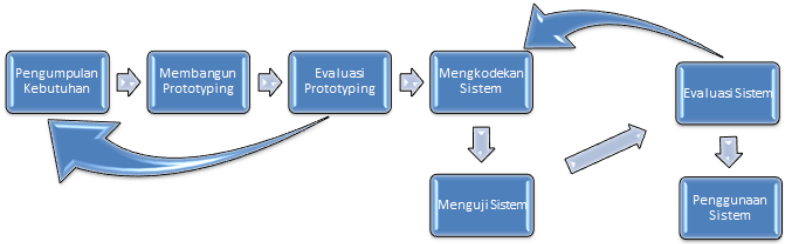
2.2.2 Rekayasa Perangkat Lunak

Rekayasa perangkat lunak (*software engineering*) adalah pembangunan dengan menggunakan prinsip atau konsep rekayasa dengan tujuan menghasilkan perangkat lunak yang bernilai ekonomi yang dipercaya dan bekerja secara efisien menggunakan mesin [9].

Rekayasa perangkat lunak lebih fokus pada bagaimana membuat perangkat lunak yang memenuhi kriteria berikut [10]:

a. Dapat terus dipelihara setelah perangkat lunak selesai dibuat seiring berkembangnya teknologi dan lingkungan (*maintainability*).
b. Dapat diandalkan dengan proses bisnis yang dijalankan dan perubahan yang terjadi (*dependability* dan *robust*).
c. Efisien dari segi sumber daya dan penggunaan.
d. Kemampuan untuk dipakai sesuai dengan kebutuhan (*usability*).

1. Metode Pengembangan Sistem



Gambar 2. 1 Metode *Prototype* Menurut Pressman

Ada beberapa model pengembangan perangkat lunak, salah satunya yaitu model *prototype*. Model *prototype* dapat digunakan untuk meyambung ketidakpahaman pelanggan mengenai hal teknis dan memperjelas spesifikasi kebutuhan yang diinginkan pelanggan kepada pengembang perangkat [11]. Terdapat beberapa tahapan dalam proses *prototype* yaitu [12]:

1. Pengumpulan Kebutuhan
   Pelanggan dan pengembang mendeskripsikan format dan seluruh kebutuhan perangkat lunak, serta garis besar dari sistem yang akan dikembangkan.
2. Membangun *Prototyping*
   Dilakukan perancangan dan pembuatan desain *prototype* sistem yang bertumpu kepada penyajian untuk pelanggan, misalkan membuat contoh *input* dan *output* sistem yang akan dikembangkan.
3. Evaluasi *Prototyping*
   Dilakukan evaluasi dari *prototyping* yang telah dibuat, apakah sudah sesuai dan memenuhi keinginan pelanggan atau belum. Jika sudah sesuai, maka akan dilanjutkan ke tahap pengembangan ke 4, namun apabila masih belum memenuhi seluruh kebutuhan pelanggan, maka akan diperbaiki dengan mengulang kembali ke tahap pengembangan 1, 2, dan 3.
4. Mengkodekan Sistem
   Tahap ini dilakukan ketika *prototype* yang dibuat telah sesuai dengan kebutuhan pelanggan, dan sudah diterima serta disepakati, dengan melanjutkan ke tahap pengkodean sistem menggunakan bahasa pemrograman yang sesuai.
5. Menguji Sistem
   Dilakukan pengujian sistem yang telah dibangun dengan menilai kinerja yang dihasilkan. Dapat menggunakan pengujian dengan metode *Black Box* atau *White Box*.
6. Evaluasi Sistem
   Setelah melakukan pengujian, dilakukan evaluasi dari hasil yang didapatkan dari pengujian sistem. Apakah sistem yang dibangun seuai dengan yang diinginkan oleh pelanggan atau belum. Jika sesuai, dapat dilanjutkan ke tahap berikutnya, jika belum seusai, maka perlu dilakukan pengulangan di tahap 4 dan tahap 5.
7. Penggunaan Sistem

Sistem yang sudah dibangun, siap untuk digunakan oleh pelanggan setelah pengujian sistem berhasil dan diterima oleh pelanggan.

Model *prototyping* ini memiliki beberapa keunggulan dan kelemahan. Adapun keunggulan dari model *prototyping* yaitu [13]:

1. Adanya komunikasi yang baik antara pengembang dan pelanggan.
2. Pengembang dapat bekerja lebih baik dalam menentukan kebutuhan pelanggan.
3. Pelanggan berperan aktif dalam pengembangan sistem.
4. Lebih menghemat waktu dalam pengembangan sistem.
5. Penerapan menjadi lebih mudah karena pemakai mengetahui apa yang diharapkannya.

Akan tetapi, model *prototyping* ini juga memiliki kelemahan seperti [14]:

1. Proses analisis dan perancangan perangkat lunak terlalu singkat.
2. Biasanya model proses *prototyping* mengesampingkan alternatif pemecahan masalah.
3. Pelanggan kadang tidak melihat atau menyadari bahwa perangkat lunak yang ada belum mencantumkan kualitas perangkat lunak secara keseluruhan dan juga belum memikirkan kemampuan pemeliharaan untuk jangka waktu lama.
4. Pengembang biasanya ingin cepat menyelesaikan proyek. Sehingga menggunakan algoritma dan bahasa pemrograman yang sederhana untuk membuat *prototyping* lebih cepat selesai.
5. *Prototype* yang dihasilkan tidak selamanya mudah diubah.
6. Biasanya model proses *prototyping* kurang fleksibel bila terjadi perubahan.

2. Pengujian Sistem

Di dalam rekayasa perangkat lunak, terdapat tahap pengujian sistem yaitu *Black-Box Testing* dan *White-Box Testing*. *Black-Box Testing* adalah pengujian yang berfokus pada kebutuhan fungsional dari perangkat lunak atau sistem. *Black-Box Testing* berusaha untuk dapat menemukan kesalahan seperti [15]:

1. Fungsi-fungsi yang salah atau hilang
2. Kesalahan antarmuka
3. Kesalahan didalam struktur data atau akses *database* eksternal
4. Kesalahan performa sistem
5. Kesalahan inisialisasi dan terminasi

Black-Box Testing hanya menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program untuk mengetahui apakah *input*, proses, dan *output* dari perangkat lunak tersebut sudah sesuai dengan spesifikasi yang dibutuhkan [16].

Sedangkan *White-Box Testing* merupakan pengujian perangkat lunak yang menggunakan struktur kontrol yang dijelaskan sebagai bagian dari perancangan perangkat komponen untuk menghasilkan uji kasus [17]. Ada beberapa tahapan pengujian menggunakan *White-Box Testing* seperti:

1. Pengujian keseluruhan keputusan yang menggunakan *logical*
2. Pengujian keseluruhan *loop* yang ada sesuai dengan batasannya
3. Pengujian pada struktur data yang sifatnya internal dan terjamin validitasnya

Metode *White-Box Testing* ini banyak digunakan untuk [18]:

1. Mengeksekusi seluruh perulangan yang ada kepada batas nilai dan operasional di setiap situasi dan kondisi
2. Mendefinisikan tentang seluruh alur logika yang ada
3. Keputusan yang sifatnya logis, dapat digunakan di semua kondisi *true* (benar) atau *false* (salah)

3. Alat Bantu

a. *Flowchart* Sistem

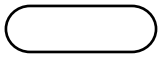
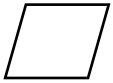
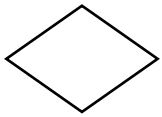
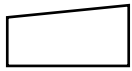
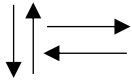
Flowchart merupakan penggambaran secara grafik dari langkah-langkah dan urut-urutan prosedur dari suatu program. *Flowchart* dapat membantu analis dalam memecahkan masalah kedalam segmen-segmen yang lebih kecil serta membantu menganalisis alternatif-alternatif lain dalam pengoperasian [19].

Untuk menggambarkan sebuah algoritma yang terstruktur dan mudah dipahami oleh orang lain (khususnya *programmer* yang bertugas mengimplementasikan program), maka dibutuhkan alat bantu yang berbentuk diagram alir (*flowchart*) [20].

Tujuan dari *flowchart* yaitu untuk menggambarkan suatu tahapan penyelesaian masalah secara sederhana, terurai, rapi dan jelas menggunakan simbol-simbol yang standar.

Simbol-simbol yang digunakan untuk menggambarkan algoritma dalam bentuk diagram alir beserta kegunaannya, sebagai berikut [21]:

Tabel 2. 1 Simbol-simbol *Flowchart* Sistem

No.	Simbol	Nama	Keterangan
1.		*Terminal*	Menyatakan permulaan atau akhir suatu program.
2.		*Input/Output*	Menyatakan proses *input* atau *output* tanpa tergantung jenis peralatannya.
3.		*Process*	Menyatakan suatu tindakan (proses) yang dilakukan oleh komputer.
4.		*Decision*	Menunjukkan suatu kondisi tertentu yang akan menghasilkan dua kemungkinan jawaban : ya / tidak.
5.		*Manual Input*	Memasukkan data secara manual *on-line keyboard*.
6.		*Document*	Mencetak keluaran dalam bentuk dokumen (melalui *printer*).
7.		*Flow*	Menyatakan jalannya arus suatu proses.

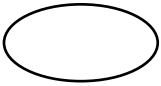
b. UML (*Unified Modeling Language*)

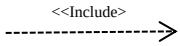
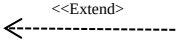
UML (*Unified Modeling Language*) merupakan salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan kebutuhan, membuat analisis dan desain, serta meggambarkan arsitektur dalam pemrograman berorientasi objek [22]. UML juga bisa didefinisikan sebagai suatu bahasa yang digunakan untuk menentukan, memvisualisasikan, membangun, dan mendokumentasikan suatu sistem informasi[23].

1. *Use Case* Diagram

Use case diagram adalah pemodelan untuk kelakuan sistem informasi yang akan dibangun. *Use case* digunakan untuk mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibangun. *Use case* diagram digunakan untuk mengetahui fungsi apa saja yang ada pada sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Berikut ini simbol-simbol yang digunakan dalam *use case* diagram [24]:

Tabel 2. 2 Simbol-simbol *Use Case Diagram*

No.	Simbol	Nama	Keterangan
1.		*Use case*	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor. Biasanya dinyatakan dengan menggunakan kata kerja diawal kalimat.
2.		*Actor*	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri.

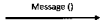
3.	———	*Association*	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau sebaliknya.
4.		*Include*	Relasi *use case* tambahan ke sebuah *use case*, dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini. *Include* bisa diartikan sebagai *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan dan *use case* tambahan akan selalu melakukan pengecekan apakah *use case* yang ditambahkan telah dijalankan sebelum *use case* tambahan dijalankan.
5.		*Extend*	Relasi *use case* tambahan ke sebuah *use case*, dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu.

6.		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas

2. *Sequence* Diagram

Sequence diagram merupakan gambaran bagaimana sistem merespon kegiatan *user* (pengguna). *Sequence* diagram juga dapat diartikan sebagai sebuah penggambaran kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* (pesan) yang dikirimkan dan diterima antar objek [25].Berikut ini simbol-simbol yang digunakan dalam *sequence* diagram [26]:

Tabel 2. 3 Simbol-simbol *Sequence Diagram*

No.	Simbol	Nama	Keterangan
1.		*LifeLine*	Menggambarkan aktifitas dari objek.
2.		*Actor*	Mewakili pengguna yang berinteraksi dengan sistem.
3.		*Message*	Komunikasi antar objek yang menggambarkan aksi yang akan dilakukan.
4.		*Boundary*	Menggambarkan sebuah *user interface* atau suatu alat yang berinteraksi dengan sistem yang lain.
5.		*Control Class*	Menggambarkan sebuah *class* untuk mengatur aliran dari informasi untuk sebuah seknario

			dan menghubungkan *boundary* dengan tabel.
6.		*Entity*	Bertanggung jawab menyimpan data atau informasi.

2.2.3 Pemrograman Berorientasi Objek

Pemrograman berorientasi objek merupakan pemrograman prosedural yang membagi aplikasi berdasarkan macam-macam fungsi yang terdapat di dalamnya. Berbagai macam fungsi tersebut saling bergantung satu sama lain sehingga sulit untuk dipisahkan. Pemrograman berorientasi objek memecah komponennya menjadi objek-objek yang saling berinteraksi.

Keuntungan dalam menggunakan pemrograman berorientasi yaitu seperti [27]:

a. *Real world programming*
   Program yang memodelkan dunia nyata, sehingga bisa menggambarkan kondisi yang ada ke dalam bentuk seakurat mungkin. Program yang menggunakan pemrograman berorientasi objek akan disusun berdasarkan objek-objek yang masing-masing memiliki fungsi sesuai dengan peran dan kebutuhan interaksinya.
b. *Reusability of code*
   Kelas yang sudah dibuat di dalam pemrograman berorientasi objek, dapat digunakan kembali oleh program yang lain. Hal tersebut bukan hanya untuk mengurangi pembuatan kelas berulang-ulang, tetapi juga meminimalisir terjadinya kesalahan jika harus mengembangkan dari awal. Sehingga dapat menghemat waktu dan biaya pengembangan program.
c. *Resilience to change*
   Program yang menerapkan memodelkan dunia nyata seperti pemrograman berorientasi objek, dapat bersifat dinamis. Program tersebut harus bisa menangani perubahan yang terjadi tanpa mengubah keseluruhan aplikasi
d. *Information hiding*
   Informasi yang terdapat di dalam sebuah objek sebisa mungkin disembunyikan dari kelas lain yang ada, untuk mengamankan data agar hanya fungsi yang berada di dalam kelas tersebut saja yang bisa

membaca, mengubah serta memanipulasi data tersebut. Namun, tetap disediakan cara supaya objek dari luar kelas, dapat mengakses dan mengubah data secara tidak langsung, sehingga kelas lain hanya menerima data yang diperlukan tanpa mengetahui bagaimana cara kerja dalam kelas tersebut.

e. *Modularity of code*

Pemrograman berorientasi objek menggunakan modularitas, yaitu setiap objek yang dibuat dikelola secara terpisah dari objek lainnya walaupun berasal dari kelas yang sama. Sehingga modifikasi terhadap sebuah objek dapat dilakukan tanpa memengaruhi fungsionalitas objek lainnya.

2.2.4 Basis Data

Basis data adalah kumpulan dari data yang saling berhubungan yang dikelompokkan dalam sebuah tabel atau beberapa tabel dan sebuah aplikasi program yang mengatur cara mengakses data tersebut.

Tujuan dari *DBMS* (*Database Management Systems*) yaitu untuk menyediakan sebuah cara untuk menyimpan dan mengambil informasi yang ada di dalam basis data [28].

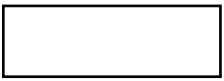
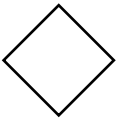
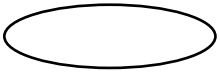
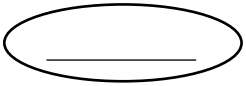
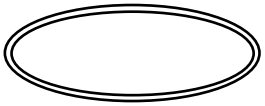
Sistem basis data menyediakan dua jenis bahasa yaitu *data definition language* (*DDL*) yang berfungsi untuk membuat suatu skema basis data dan *data manipulation language* (*DML*) yang berfungsi untuk melakukan pencarian dan perbaruan data pada basis data [29].

Basis data (*database*) juga dapat diartikan sebagai himpunan kelompok data yang saling terhubung dan diorganisasi sedemikian rupa supaya kelak dapat dimanfaatkan kembali secara cepat dan mudah. Database mempunyai delapan operasi dasar, seperti *create database*, *drop database*, *create table*, *drop table*, *insert*, *read*, *update*, dan *delete*. Sebuah database menyimpan data di dalam tabel, dimana setiap tabel memiliki baris dan kolom [30].

1. *Entity Relatonship Diagram* (ERD)

Entity Relationship Diagram (*ERD*) merupakan bentuk awal yang dibuat ketika merancang basis data relasional. *ERD* juga diartikan sebagai model *entity relationship* yang berisi berbagai komponen himpunan entitas dan himpunan relasi yang dilengkapi dengan atribut-atribut yang merepresentasikan seluruh fakta yang ditinjau[31]. Simbol-simbol *ERD* dapat di lihat pada **Tabel 2.4**

Tabel 2. 4 Simbol-simbol *ERD*

No	Simbol	Fungsi
1.	Entitas 	Entitas merupakan suatu kumpulan objek atau sesuatu yang dapat dibedakan atau diidentifikasi secara unik.
2.	Relasi 	Relasi adalah hubungan yang terjadi antara satu entitas atau lebih.
3.	Atribut 	Atribut merupakan kumpulan elemen data yang membentuk suatu entitas.
4.	Atribut Kunci Primer 	Atribut yang digunakan untuk menentukan suatu *entity* secara unik.
5.	Atribut Multinilai 	Atribut yang memiliki sekelompok nilai untuk setiap *instant entity*.
6.	Alur	Alur berfungsi untuk menghubungkan atribut dengan entitas dan entitas dengan relasi.

2.2.5 Koperasi

Koperasi merupakan usaha bersama yang berbentuk badan hukum, anggotanya sebagai pemilik dan pengguna jasa koperasi, serta mengembalikan semua penerimaan di atas biayannya kepada anggota sesuai dengan transaksi yang mereka jalankan dengan koperasi[32].

Koperasi dikelompokkan menjadi 4 faktor yaitu, jenis usaha, status anggota, tingkatan, dan fungsinya. Berdasarkan topik pembahasan yang ada yaitu koperasi bagian unit simpan pinjam, maka kaan dibahas faktor koperasi berdasarkan jenis usahanya. Pengelompokan jenis-jenis koperasi berdasarkan jenis usahanya dibagi menjadi 4, yaitu koperasi produksi, koperasi konsumsi, koperasi simpan pinjam, dan koperasi serba usaha. Berikut penjelasan lebih lanjutnya [33]:

1. Koperasi Produksi
   Koperasi yang mempunyai tujuan untuk membantu usaha para anggotanya atau melakukan usaha secara bersama-sama. Koperasi produksi memiliki berbagai macam bentuk seperti, koperasi produksi untuk para petani, peternak sapi, pengrajin, dan sebagainya. Koperasi produksi bertujuan untuk membantu usaha para anggotanya yang mengalami kesulitan dalam menjalani usahanya.
2. Koperasi Konsumsi
   Koperasi yang menjual berbagai barang kebutuhan pokok untuk para anggotanya. Harga barang-barang dari koperasi umumnya lebih murah dibanding dengan harga di pasaran. Sebagai contoh koperasi menjual beras, telur, gula, tepung, kopi, dan lainnya.
3. Koperasi Simpan Pinjam
   Koperasi yang menyediakan pinjaman uang dan tempat menyimpan uang. Uang yang dipinjamkan diperoleh dari dana yang dikumpulkan secara bersama-sama oleh para anggotanya. Koperasi simpan pinjam sekilas terlihat seperti bank, namun terdapat beberapa poin yang membedakan koperasi simpan pinjam dengan bank yaitu, bunga pinjaman yang ditawarkan lebih ringan dibandingkan dengan bank, pembayaran pinjaman dapat dilakukan dengan cara mengangsur, dan bunga yang didapatkan dari hasil pinjaman dinikmati secara bersama dengan cara bagi hasil.

4. Koperasi Serba Usaha
   Koperasi yang didalamnya memiliki berbagai macam bentuk usaha. Bentuk usaha yang dilakukan bisa berupa gabungan antara koperasi produksi dan koperasi konsumsi atau antara koperasi produksi dan koperasi simpan pinjam.

Sebuah koperasi memiliki neraca sebagai laporan keuangan yang digunakan untuk melihat keseimbangan (*balance*) keuangan dari koperasi tersebut. Neraca adalah laporan yang menunjukkan posisi keuangan perusahaan pada tanggal tertentu[34]. Penjelasan lain mengenai pengertian neraca, yaitu neraca merupakan laporan keuangan yang memberikan informasi terperinci mengenai aktiva, kewajiban perusahaan serta modal pemilik pada saat tertentu[35].Menurut KBBI, neraca diartikan sebagai catatan perbandingan untung rugi, utang-piutang, pemasukan dan pengeluaran, dan sebagainya.*Balance Sheet* (neraca) adalah laporan posisi keuangan yang menunjukkan *asset*/harta, hutang dan modal pada suatu saat tertentu. Bentuk neraca ada 2 macam :

1. Bentuk *skontro*/*horizontal*/T *account*
2. Bentuk *staffel*/*vertical*/Laporan

Perkiraan-perkiraan yang di laporkan ke dalam neraca yaitu aktiva, kewajiban dan modal sering disebut perkiraan riil, sedangkan perkiraan yang di laporkan ke dalam perhitungan laba rugi yaitu pendapatan dan beban sering di sebut perkiraan nominal atau perkiraan sementara[36].

~Halaman Ini Sengaja Dikosongkan~