

BAB II

TINJAUAN PUSTAKA

II.1. Produksi

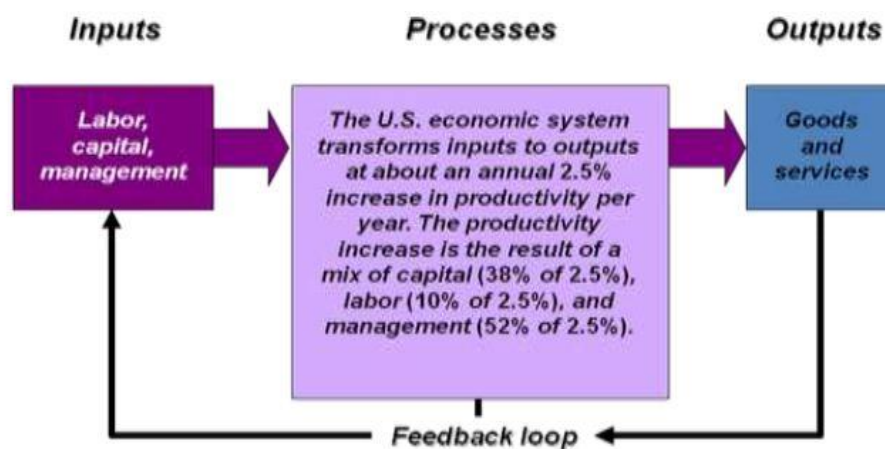
Produksi adalah suatu kegiatan yang mengubah input menjadi output. Kegiatan tersebut dalam ekonomi biasa dinyatakan dalam fungsi produk, Fungsi produk menunjukkan jumlah maksimum output yang dapat dihasilkan dari pemakaian sejumlah input dengan menggunakan teknologi tertentu. Produksi sering didefinisikan sebagai penciptaan guna, dimana guna berarti kemampuan barang atau jasa untuk memenuhi kebutuhan manusia. Menurut Sukirno (2006:6) pengertian faktor produksi adalah benda-benda yang disediakan oleh alam atau diciptakan oleh manusia yang dapat digunakan untuk memproduksi barang dan jasa. Produksi pertanian yang optimal adalah produksi yang mendatangkan produk yang menguntungkan ditinjau dari sudut ekonomi ini berarti biaya faktor-faktor input yang berpengaruh pada produksi jauh lebih kecil bila dibandingkan dengan hasil yang diperoleh sehingga petani dapat memperoleh keuntungan dari usaha taninya. (*Nova Tumoka ; 2013 : 346*)

II.2. Produktivitas

Produktivitas muncul karena adanya suatu proses transformasi yang berlangsung. Proses transformasi adalah serangkaian aktivitas yang dilakukan untuk mengolah atau mengubah sejumlah masukan (input) menjadi sejumlah keluaran (output) yang memiliki nilai tambah. Pengolahan yang terjadi bisa secara fisik maupun nonfisik. Sedangkan nilai tambah adalah nilai dari keluaran yang

bertambah dalam pengertian nilai guna atau nilai ekonomisnya. Pengertian produktivitas secara umum adalah rasio antara output dibagi dengan input. Sementara pendekatan dalam studi produktivitas tidak hanya menekankan pada aspek ekonomi tertentu saja, melainkan juga aspek-aspek non ekonomi, yang kadang-kadang lebih besar peranannya dalam peningkatan produktivitas.

(F. Priyo Suprobo1 ; 2013 : 8)



Gambar II.1. Proses Transformasi
 Sumber : (F. Priyo Suprobo1 ; 2013 : 8)

II.3. Jenis Dan Pengukuran Produktivitas

Berdasarkan pendekatan rasio output/ input, ada tiga tipe dasar produktivitas , produktivitas parsial, produktivitas factor total, dan produktivitas total. Dalam hal pengukuran produktivitas , beberapa pendekatan untuk membandingkan tingkat hasil pengukuran produktivitas dapat dibedakan sebagai berikut :

1. Membandingkan unit kerja periode yang diukur dengan unit kerja periode dasar.

2. Membandingkan unit kerja suatu organisasi dengan unit kerja organisasi yang lain.
3. Membandingkan unit kerja yang sebenarnya dengan target yang telah ditetapkan.

Dari segi teknik pengukuran produktivitas, dikenal beberapa model pengukuran sebagai berikut :

1. Model Objectives Matrix (OMAX)
2. Model David J. Sumanth
3. Model Habberstad Productivity Wheel
4. Model Marvin E. Mundel
5. Model Craig Harris
6. Model APC
7. Model Kendric Creamer
8. Model Pertambahan nilai

Diantara berbagai model diatas, yang akan diutamakan didalam artikel ini adalah Model Marvin E. Mundel. Pertimbangan menggunakannya adalah karena sistem tersebut sederhana, mudah dipahami, mampu memantau produktivitas dari tiap bagian institusi dengan kriteria produktivitas yang sesuai dengan keberadaan bagian tersebut atau lebih fleksibel, mudah dilaksanakan tanpa memerlukan keahlian khusus dan kebutuhan datanya mampu menyesuaikan sehingga mudah diperoleh.

II.4. Sistem

Sistem merupakan suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul dan bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran tertentu.

Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, saling bekerja sama membentuk satu kesatuan. Komponen-komponen sistem atau elemen-elemen sistem dapat berupa suatu sub sistem atau bagian-bagian dari sistem. Setiap sub sistem mempunyai sifat-sifat dari sistem untuk menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan.

Media penghubung antara satu sub sistem dengan sub sistem yang lainnya biasa disebut dengan penghubung (interface). Melalui penghubung ini memungkinkan sumber-sumber daya mengalir dari satu sub sistem ke sub sistem yang lainnya. Melalui penghubung keluaran (output) untuk sub sistem akan menjadi masukan (input) sub sistem yang lainnya. Dengan penghubung satu sub sistem dapat berinteraksi dengan sub sistem yang lainnya membentuk satu kesatuan.

Suatu sistem pasti mempunyai satu tujuan (goal) atau sasaran (objective). Sasaran dari sistem sangat menentukan masukan yang dibutuhkan sistem dan keluaran yang akan dihasilkan sistem. (*Anastasia Lipursari ; 2013 : 27*)

II.5. Informasi

Data yang telah diklasifikasi atau diolah atau diinterpretasi untuk digunakan dalam proses pengambilan keputusan adalah informasi. Sistem pengolahan mengolah data menjadi informasi atau tepatnya mengolah data dari

bentuk tidak berguna menjadi berguna bagi penerimanya. Nilai informasi berhubungan dengan keputusan. Nilai informasi dilukiskan paling berarti dalam konteks sebuah keputusan. Bila tidak ada keputusan, maka informasi menjadi tidak diperlukan. Keputusan dapat berkisar dari keputusan berulang sederhana sampai keputusan strategis jangka panjang.

Fungsi utama informasi adalah menambah pengetahuan atau mengurangi ketidakpastian pemakai informasi. Informasi yang disampaikan kepada pemakai mungkin merupakan hasil data yang dimasukkan ke dalam dan pengolahan suatu model keputusan. Akan tetapi, dalam pengambilan keputusan yang kompleks, informasi hanya dapat menambah kemungkinan keputusan atau mengurangi bermacam-macam pilihan. Informasi yang disediakan bagi pengambil keputusan memberikan suatu kemungkinan faktor resiko pada tingkat-tingkat pendapatan yang berbeda.

Informasi yang dapat ditangani atau dihasilkan dalam fungsi organisasi yang dapat ditentukan banyaknya sangat penting karena sistem informasi memberikan informasi formal mengenai keadaan yang memberikan tingkat kemungkinan meramalkan yang lebih besar kepada pemakai baik mengenai kejadian maupun mengenai hasil kegiatan (termasuk kegiatan pemakai sendiri) organisasi.

Hal-hal yang dapat ditentukan oleh nilai informasi adalah : manfaat dan biaya untuk mendapatkannya. Suatu informasi dikatakan bernilai bila manfaat lebih efektif dibandingkan dengan biaya mendapatkannya. Akan tetapi, perlu dipertimbangkan bahwa informasi yang digunakan untuk beberapa kegunaan

sehingga tidak memungkinkan dan sulit untuk menghubungkan suatu bagian informasi pada suatu masalah tertentu dengan biaya untuk memperolehnya karena sebagian besar informasi dinikmati tidak hanya oleh satu pihak di dalam perusahaan. (*Anastasia Lipursari ; 2013 : 28*)

Sebagian besar informasi tidak dapat persis ditafsir keuntungannya dengan suatu nilai uang, tetapi dapat ditafsir nilai efektivitasnya. Nilai informasi ini didasarkan atas 10 (sepuluh) sifat yaitu:

a. Mudah diperoleh

Sifat ini menunjukkan mudahnya dan cepatnya informasi dapat diperoleh. Kecepatan memperoleh dapat diukur, misalnya 1 menit versus 24 jam. Akan tetapi, berapa nilainya bagi pemakai informasi sulit mengukurnya.

b. Luas dan lengkap

Sifat ini menunjukkan lengkapnya isi informasi. Hal ini tidak berarti hanya mengenai volumenya, tetapi juga mengenai keluaran informasinya. Sifat ini sangat kab Sifat ini menunjuk kabur dan karena itu sulit mengukurnya.

c. Ketelitian

Sifat ini berhubungan dengan tingkat kebebasan dari kesalahan keluaran informasi. Dalam hubungannya dengan volume data yang besar biasanya dua jenis kesalahan, yakni kesalahan pencatatan dan kesalahan perhitungan.

d. Kecocokan

Sifat ini menunjukkan betapa baik keluaran informasi dalam hubungannya dengan permintaan para pemakai. Isi informasi harus ada hubungannya dengan masalah yang sedang dihadapi. Semua keluaran lainnya tidak berguna, tetapi mahal mempersiapkannya. Sifat ini sulit mengukurnya.

e. Ketepatan Waktu

Sifat ini berhubungan dengan waktu yang dilalui yang lebih pendek dari pada siklus untuk mendapatkan informasi. Masukan, pengolahan, dan pelaporan keluaran kepada para pemakai biasanya tepat waktu. Dalam beberapa hal, ketepatan waktu dapat diukur.

f. Kejelasan

Sifat ini menunjukkan tingkat keluaran informasi yang bebas dari istilah-istilah yang tidak jelas. Membenarkan laporan dapat memakan biaya yang besar.

g. Keluwesan

Sifat ini berhubungan dengan dapat disesuaikannya keluaran informasi tidak hanya dengan lebih dari satu keputusan tetapi juga dengan lebih dari seorang pengambil keputusan. Sifat ini sulit diukur, tetapi dalam banyak hal dapat diberikan nilai yang dapat diukur.

h. Dapat dibuktikan

Sifat ini menunjukkan kemampuan beberapa pemakai informasi untuk menguji keluaran informasi dan sampai pada kesimpulan yang sama.

i. Tidak ada prasangka

Sifat ini berhubungan dengan tidak adanya keinginan untuk mengubah informasi guna mendapatkan kesimpulan yang telah dipertimbangkan sebelumnya.

j. Dapat diukur

Sifat ini menunjukkan hakikat-hakikat informasi dihasilkan dari sistem informasi formal.

Nilai informasi yang sempurna adalah pengambil keputusan diijinkan untuk memilih keputusan optimal dalam setiap hal dan bukan keputusan yang rata-rata akan menjadi optimal dan untuk menghindari kejadian-kejadian yang akan mengakibatkan suatu kerugian. Akan tetapi informasi sempurna, mungkin tidak ada. Dalam hal-hal demikian, perkiraan-perkiraan hasil sebelumnya

mungkin dipengaruhi oleh informasi tambahan, meskipun informasi tersebut tidak memberikan kepastian, informasi yang tidak sempurna sesungguhnya merupakan informasi dari uji petik (sampling). Informasi ini tidak sempurna karena lebih banyak memberi perkiraan dari pada memberi suatu angka yang pasti.

Kualitas suatu informasi tergantung dari 3 (tiga) hal yaitu, informasi harus akurat, tepat waktu dan relevan.

a. Akurat

Informasi harus bebas dari kesalahan-kesalahan dan tidak bisa menyesatkan. Akurat juga berarti informasi harus jelas mencerminkan maksudnya. Informasi harus akurat karena dari sumber informasi sampai penerima informasi kemungkinan banyak terjadi gangguan (noise) yang dapat mengubah atau merusak informasi tersebut.

b. Tepat waktu

Informasi yang datang pada si penerima tidak boleh terlambat. Informasi yang sudah usang tidak akan mempunyai nilai lagi karena informasi merupakan landasan dalam pengambilan keputusan.

c. Relevan

Informasi tersebut mempunyai manfaat untuk pemakainya. Relevansi informasi untuk orang satu dengan orang yang lain berbeda, misalnya informasi sebab-sebab kerusakan mesin produksi kepada akuntan perusahaan adalah kurang relevan dan akan lebih relevan, bila ditujukan kepada ahli teknik perusahaan.

II.6. *Database*

Sebuah *database* adalah sebuah struktur yang umumnya dikategorikan dalam dua hal yaitu sebuah database flat dan sebuah database relasional. *Database* relasional lebih disukai karena lebih masuk akal dibandingkan database flat. *Database* merupakan sembarang pengumpulan data, sebuah file terdiri dari atas sejumlah *record* atau tabel. Masing-masing terbentuk dari field atau kolom dari tipe tertentu, dan kumpulan operasi yang memudahkan pencarian, penyimpanan, kombinasi ulang, dan aktivitas sejenisnya. Secara umum, pengertian database adalah sekumpulan data store (bisa dalam jumlah yang sangat besar) yang tersimpan dalam *magnetic disk*, *optical disk*, *magnetic drum* atau media penyimpanan sekunder lainnya, dapat juga diartikan sebagai sekumpulan program-program aplikasi umum yang bersifat "batch" yang mengeksekusi dan memproses data secara umum, seperti pencarian, peremajaan, penambahan, dan penghapusan terhadap data.

Pendapat lain menurut George M.Scott: "*Data Base* adalah sistem *file* komputer yang menggunakan cara pengorganisasian *file* tertentu, dimaksudkan untuk mempercepat pembaharuan masing-masing *record*, serta pembaharuan secara serempak atas *record* terkait, juga untuk mempermudah dan mempercepat akses terhadap seluruh *record* lewat program aplikasi, serta akses yang cepat terhadap data yang disimpan, yang harus digunakan secara bersama-sama untuk dibaca guna penyusunan laporan rutin atau khusus ataupun untuk penyelidikan. Sedangkan menurut Rahmat Priyanto : "*Database* dapat diartikan sebagai suatu bentuk penyimpanan informasi yang terpusat agar data-data atau informasi yang

ada didalamnya mudah dicari, dikelola dan digunakan kembali”. (R Achmad Margoyuwono ; 2012 : 45)

II.7. UML (*Unified Modelling Language*)

Unified Modelling Language (UML) adalah sebuah "bahasa" yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan *class* dan *operation* dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasa-bahasa berorientasi objek seperti C++, Java, C# atau VB.NET. Walaupun demikian, UML tetap dapat digunakan untuk modeling aplikasi prosedural dalam VB atau C. Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan *syntax*/semantik. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan UML *syntax* mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan. Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya: Grady Booch OOD (*Object-Oriented Design*), Jim Rumbaugh OMT (*Object Modeling Technique*), dan Ivar Jacobson OOSE (*Object-Oriented Software Engineering*). Sejarah UML sendiri cukup panjang. Sampai era tahun 1990 seperti kita ketahui puluhan metodologi pemodelan berorientasi objek telah bermunculan di dunia. Diantaranya adalah:

metodologi booch, metodologi coad, metodologi OOSE, metodologi OMT, metodologi shlaer-mellor, metodologi wirfs-brock, dsb. Masa itu terkenal dengan masa perang metodologi (*method war*) dalam pendesainan berorientasi objek. Masing-masing metodologi membawa notasi sendiri-sendiri, yang mengakibatkan timbul masalah baru apabila kita bekerjasama dengan group/perusahaan lain yang menggunakan metodologi yang berlainan. Dimulai pada bulan Oktober 1994 *Booch, Rumbaugh dan Jacobson*, yang merupakan tiga tokoh yang boleh dikatakan metodologinya banyak digunakan memelopori usaha untuk penyatuan metodologi pendesainan berorientasi objek. Pada tahun 1995 direlease draft pertama dari UML (versi 0.8). Sejak tahun 1996 pengembangan tersebut dikoordinasikan oleh Object Management Group (OMG – <http://www.omg.org>). Tahun 1997 UML versi 1.1 muncul, dan saat ini versi terbaru adalah versi 1.5 yang dirilis bulan Maret 2003. *Booch, Rumbaugh dan Jacobson* menyusun tiga buku serial tentang UML pada tahun 1999. Sejak saat itulah UML telah menjelma menjadi standar bahasa pemodelan untuk aplikasi berorientasi objek.

(Yuni Sugiarti ; 2013 : 33)

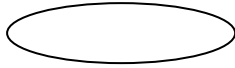
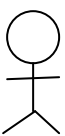
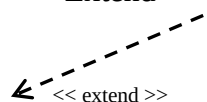
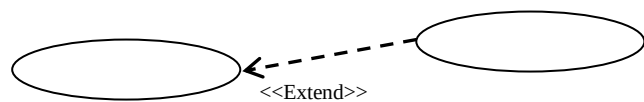
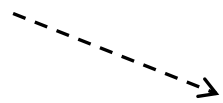
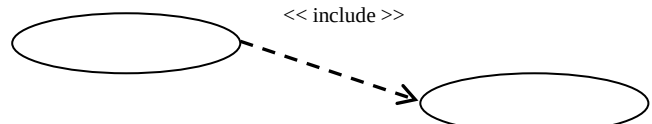
Dalam pembuatan Tugas Akhir ini penulis menggunakan diagram Use Case yang terdapat di dalam UML. Adapun maksud dari Use Case Diagram diterangkan dibawah ini.

1. Use Case Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah “apa” yang diperbuat sistem, dan bukan “bagaimana”. Sebuah *use case* merepresentasikan sebuah interaksi antara aktor

dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu. *Use case diagram* dapat sangat membantu bila kita sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang *test case* untuk semua *feature* yang ada pada sistem. Sebuah *use case* dapat meng-*include* fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa *use case* yang di-*include* akan dipanggil setiap kali *use case* yang meng-*include* dieksekusi secara normal. Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*. Sebuah *use case* juga dapat meng-*extend* *use case* lain dengan *behaviour*-nya sendiri. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain. (Yuni Sugiarti ; 2013 : 41)

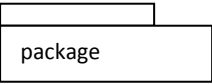
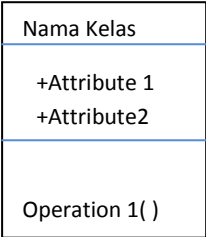
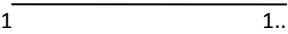
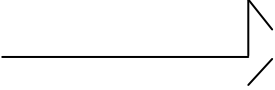
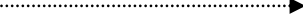
Tabel II.1. Simbol Use Case

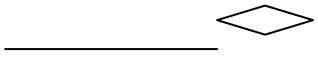
Simbol	Deskripsi
Use Case 	Fungsional yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit dan aktor, biasanya menggunakan kata kerja di awal frase nama <i>use case</i>
Aktor  Nama Aktor	Orang atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat sistem itu sendiri. Jadi walaupun simbol aktor adalah gambar orang, tapi aktor belum tentu merupakan orang. Biasanya menggunakan kata benda di awal frase nama actor
Asosiasi / Association 	Komunikasi antar aktor dan <i>use case</i> yang berpartisipasi <i>use case</i> atau <i>use case</i> berinteraksi dengan aktor.
Extend 	Relasi <i>use case</i> tambahan ke sebuah <i>use case</i> dimana <i>use case</i> yang ditambahkan dapat berdiri sendiri walupun tanpa <i>use case</i> tambahan itu, mirip dengan prinsip <i>inheritance</i> pada pemrograman berorientasi objek; biasanya <i>use case</i> tambahan memiliki nama depan yang sama dengan <i>use case</i> yang ditambahkan, arah panah menunjukan pada <i>use case</i> yang dituju. Contoh : 
Include << include >> 	Relasi <i>use case</i> tambahan ke sebuah <i>use case</i> dimana <i>use case</i> yang ditambahkan memerlukan <i>use case</i> ini untuk menjalankan fungsinya atau sebagai syarat dijalankan <i>use case</i> ini. ada dua sudut pandang yang cukup besar mengenai <i>include</i> di <i>use case</i>, <i>include</i> berarti <i>use case</i> yang ditambahkan akan selalu dipanggil saat <i>use case</i> tambahan dijalankan, contoh; 

Sumber : (Yuni Sugiarti ; 2013 : 42)

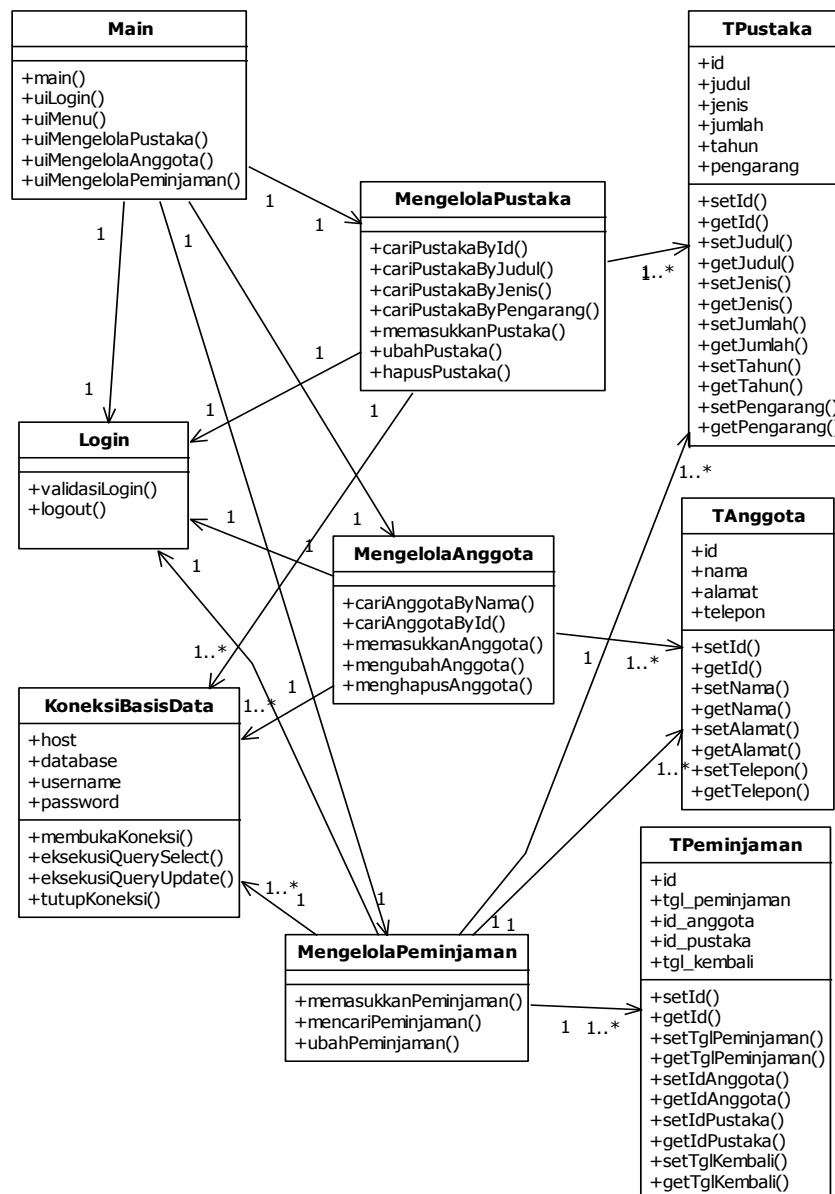
2. Class Diagram

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi. Berikut adalah simbol-simbol pada diagram kelas :

Simbol	Deskripsi
Package 	Package merupakan sebuah bungkusan dari satu atau lebih kelas
Operasi 	Kelas pada struktur sistem
Antar muka / interface  Interface	Sama dengan konsep interface dalam pemograman berorientasi objek
Asosiasi 	Relasi antar kelas dan makna umum , asosiasi biasanya juga disertai dengan multiplicity
Asosiasi berarah /Directed Asosiasi 	Relasi antar kelas dengan makna kelas yang satu di gunakan oleh kelas yang lain asosiasi biasanya juga disertai dengan multiplicity
Generalisasi 	Relasi antar kelas dengan makna generalisasi – spesialisasi (umum-khusus)
Ketergantungan/defedency 	Relasi antar kelas dengan makna ketergantungan antar kelas

Agregasi 	Relasi antar kelas dengan makna semua bagian (whole part)
---	--

Gambar II.2. Class Diagram
Sumber : (Yuni Sugiarti ; 2013 : 59)



Gambar II.3. Contoh Class Diagram
Sumber : (Yuni Sugiarti ; 2013 : 63)

3. Sequence Diagram

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

Banyaknya diagram *sequence* yang harus digambar adalah sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram *sequence* sehingga semakin banyak *use case* yang didefinisikan maka diagram *sequence* yang harus dibuat juga semakin banyak.

4. Activity Diagram

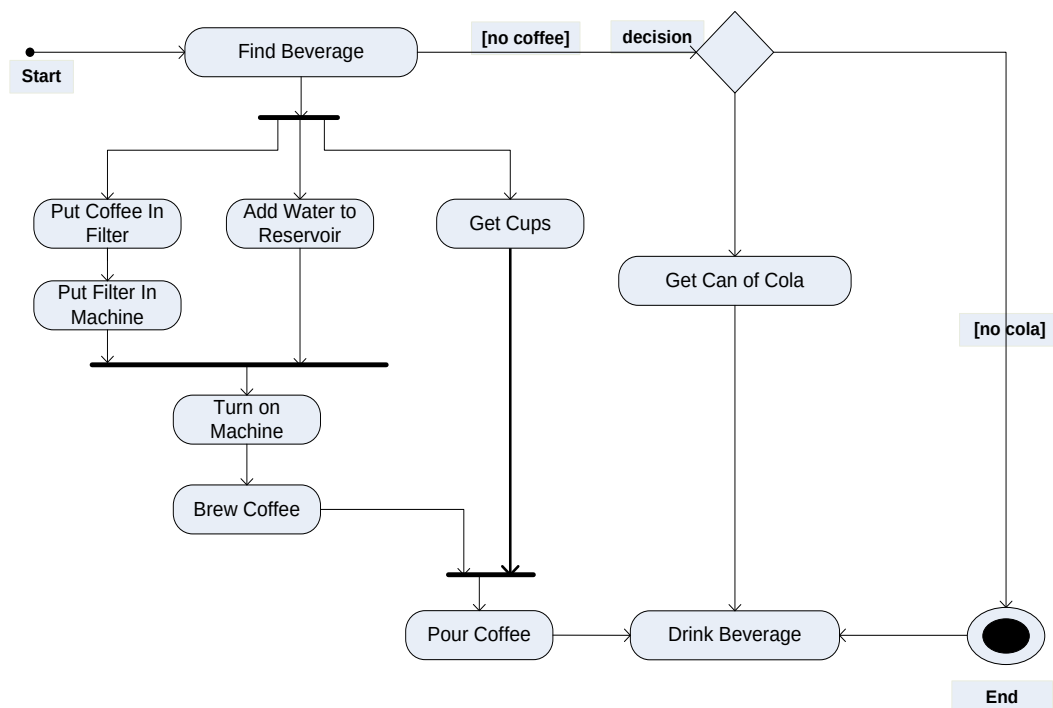
Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.

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.

Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas.

Sama seperti *state*, standar UML menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas. *Decision* digunakan untuk menggambarkan behaviour pada kondisi tertentu. Untuk mengilustrasikan proses-proses paralel (*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal.

Activity diagram dapat dibagi menjadi beberapa *object swimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu.



Gambar II.4. Activity Diagram
Sumber : (Yuni Sugiarti ; 2013 : 76)

II.8. Visual Studio

Visual Basic adalah salah satu bahasa pemrograman komputer. Bahasa pemrograman adalah perintah-perintah yang dimengerti oleh komputer untuk melakukan tugas-tugas tertentu. Bahasa pemrograman Visual Basic, yang dikembangkan oleh Microsoft sejak tahun 1991, merupakan pengembangan dari pendahulunya yaitu bahasa pemrograman BASIC (Beginner's All-purpose Symbolic Instruction Code) yang dikembangkan pada era 1950-an. Visual Basic merupakan salah satu Development Tool yaitu alat bantu untuk membuat berbagai macam program komputer, khususnya yang menggunakan sistem operasi Windows. Menurut Widodo (2004 : 86) “Visual Basic ialah bahasa pemrograman event-driven yang berasal dari BASIC. Event driven artinya program menunggu sampai adanya respon dari pemakai berupa kejadian tertentu, misalnya tombol diklik, atau menu dipilih.” (Anis nurhanafi ; 2012 : 2)

II.9. Crystal Report

Crystal Report adalah suatu program aplikasi yang dirancang untuk membuat laporan-laporan yang dapat digunakan dengan bahasa pemrograman berbasis windows, seperti *Visual Basic 6.0*, *Visual C++*, *Visual Interdev*.

3.7. *SQL Server*. (Wahyu Triyanto ; 2009 : 43)

II.10. SQL Server

Microsoft SQL Server adalah sebuah sistem manajemen basis data relasional (RDBMS) produk *microsoft*. Bahasa *query* utamanya adalah *Transact-SQL* yang merupakan implementasi dari SQL standar ANSI/ISO yang digunakan oleh *Microsoft* dan *Sybase*. (Wahyu Triyanto ; 2009 : 43)

