



## BAB II

### TINJAUAN PUSTAKA

#### 2.1. Teori Umum

##### 2.1.1. Pengertian Komputer

Arospudin (2013:19), menjelaskan bahwa “Komputer adalah alat bantu pemrosesan data secara elektronik dan cara pemrosesan datanya berdasarkan urutan instruksi atau program yang tersimpan dalam memori masing-masing komputer.”

Hartono (2013:27), menjelaskan bahwa “Komputer adalah sebuah mesin yang dapat dikendalikan melalui perintah (*programmable machine*) yang dirancang untuk secara otomatis melakukan serangkaian urutan perhitungan (*arithmetic*) atau proses-proses yang diurutkan secara logis. Urutan-urutan tersebut dapat diubah seketika oleh komputer, sehingga komputer dapat menyelesaikan lebih dari satu tugas.”

Jadi, dapat disimpulkan bahwa komputer adalah sebuah alat elektronik yang digunakan untuk memproses data, yang dikendalikan melalui perintah untuk membantu menyelesaikan tugas-tugas yang ada.

##### 2.1.2. Pengertian Aplikasi

Arospudin (2013:7), “Aplikasi (*application*) merupakan *software* yang dibuat oleh suatu perusahaan komputer untuk mengerjakan tugas-tugas tertentu misalnya *Ms-Word*, *Ms-Excel*.”

Sujatmiko (2012:23), “Aplikasi (*application*) adalah program komputer yang dibuat oleh suatu perusahaan komputer untuk membantu manusia dalam mengerjakan tugas-tugas tertentu, misalnya *Ms-Word*, *Ms-Excel*. Aplikasi (*application*) berbeda dengan sistem operasi (yang menjalankan komputer), *utility* (yang melaksanakan perawatan atau tugas-tugas umum) dan bahasa (yang digunakan untuk, e, buat program-program komputer).

Secara keseluruhan, dapat disimpulkan bahwa aplikasi adalah *software* atau program yang dibuat untuk membantu para pemakai dalam mengerjakan tugas-tugas tertentu.



### 2.1.3. Pengertian Sistem

Hartono (2013:10), Sistem adalah suatu himpunan dari berbagai bagian atau elemen, yang saling berhubungan secara teorganisasi berdasarkan fungsi-fungsinya, menjadi suatu kesatuan.

Sutabri (2012:3), Sistem adalah suatu kumpulan dari unsur, komponen, atau variabel yang terorganisasi, saling berinteraksi, saling tergantung satu sama lain, yang berfungsi bersama-sama untuk mencapai tujuan tertentu.

Jadi, dapat disimpulkan bahwa sistem adalah komponen yang saling berinteraksi satu sama lain yang bersama sama untuk mencapai tujuan tertentu.

### 2.1.4. Karakteristik sistem

Sutabri (2012:13), Model umum sebuah sistem terdiri dari input, proses, dan output. Selain itu sebuah sistem juga memiliki karakteristik atau sifat-sifat tertentu, yang mencirikan bahwa hal tersebut dikatakan sebagai suatu sistem. Adapun karakteristik sistem yaitu :

#### 1. Komponen Sistem (*Components*)

Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang bekerja sama membentuk suatu kesatuan.

#### 2. Batasan Sistem (*Boundary*)

Ruang lingkup sistem merupakan daerah yang membatasi antara sistem dengan sistem lainnya atau sistem dengan lingkungan luarnya.

#### 3. Lingkungan Luar Sistem (*Environment*)

Bentuk apapun yang ada diluar ruang lingkup atau batasan sistem yang mempengaruhi operasi sistem tersebut disebut dengan lingkungan luar sistem.

#### 4. Penghubung Sistem (*Interface*)

Media yang menghubungkan sistem dengan subsistem yang lain disebut dengan penghubung sistem. Penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem ke subsistem lain.

#### 5. Masukan Sistem (*Input*)

Energi yang dimasukkan ke dalam sistem disebut masukan sistem, yang dapat berupa pemeliharaan (*maintenance input*) dan sinyal (*signal input*).



#### 6. Keluaran Sistem (*Output*)

Hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna. Keluaran ini merupakan masukan bagi subsistem yang lain.

#### 7. Pengolah Sistem (*Process*)

Suatu sistem dapat mempunyai suatu proses yang akan mengubah masukan menjadi keluaran.

#### 8. Sasaran Sistem (*Objective*)

Suatu sistem memiliki tujuan dan sasaran yang pasti dan bersifat deterministik. Suatu sistem dikatakan berhasil bila mengenai sasaran atau tujuan yang telah direncanakan.

### 2.1.5. Pengertian Informasi

Davis dalam Hartono (2013:15) mengemukakan, “Informasi adalah data yang telah diolah menjadi sebuah bentuk yang berarti bagi penerimanya dan bermanfaat dalam pengambilan keputusan saat ini atau mendatang.”

Sutabri (2012:22) menjelaskan bahwa, “Informasi adalah data yang telah diklasifikasikan atau diolah atau diinterpretasikan untuk digunakan dalam proses pengambilan keputusan.”

Jadi, informasi merupakan sekumpulan data yang telah diolah untuk diberikan kepada pihak yang membutuhkannya.

### 2.1.6. Pengertian Data

Thompson & Handelman dalam Hartono (2013:80) mengemukakan, “Data adalah hasil pengukuran dan pencatatan terhadap fakta tentang sesuatu, keadaan, tindakan, atau kejadian.”

Arospudin (2013:22), menjelaskan bahwa “Data adalah kumpulan angka-angka maupun karakter-karakter yang tidak memiliki arti. Data dapat diolah sehingga menghasilkan informasi.”

Secara keseluruhan, dapat disimpulkan bahwa data merupakan kumpulan angka-angka maupun karakter-karakter yang belum memiliki arti sehingga data tersebut diolah untuk menghasilkan suatu informasi.



## 2.1.7. Metode Pengembangan Sistem

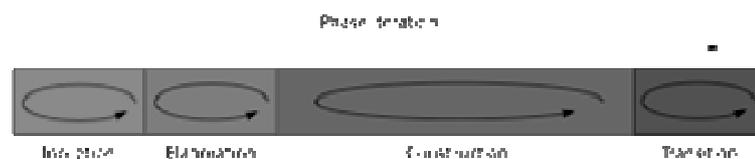
### 2.1.7.1. Pengertian RUP (*Rational Unified Process*)

Siahaan (2012:183), RUP (*Rational Unified Process*) adalah salah satu kerangka kerja untuk melakukan proses rekayasa kebutuhan. Tujuan utama standar RUP (*Rational Unified Process*) adalah untuk memastikan bahwa perangkat lunak yang sampai pada pengguna adalah perangkat lunak yang berkualitas baik.

Sukamto dan Shalahuddin (2013:124), RUP (*Rational Unified Process*) adalah pendekatan pengembangan perangkat lunak yang dilakukan berulang-ulang (*iterative*), fokus pada arsitektur (*architecture-centric*), lebih diarahkan berdasarkan penggunaan kasus (*use case driven*). RUP merupakan proses rekayasa perangkat lunak dengan pendefinisian yang baik (*well defined*) dan penstrukturan yang baik (*well structured*). RUP menyediakan pendefinisian struktur yang baik untuk alur hidup proyek perangkat lunak.

### 2.1.7.2. Fase RUP (*Rational Unified Process*)

RUP (*Rational Unified Process*) memiliki empat buah tahap fase yang dapat dilakukan pula secara iteratif. Berikut ini adalah gambar alur hidup RUP.



**Gambar 2.1.** Alur Hidup RUP (*Rational Unified Process*)

Berikut ini penjelasan untuk setiap fase pada RUP (*Rational Unified Process*) :

#### 1. *Inception* (permulaan)

Tahap ini lebih pada memodelkan proses bisnis yang dibutuhkan (*business modeling*) dan mendefinisikan kebutuhan akan sistem yang akan dibuat (*requirements*).



Berikut adalah tahap yang dibutuhkan pada tahap ini :

- a. Memahami ruang lingkup dari proyek termasuk pada biaya, waktu, kebutuhan, resiko dan lain sebagainya.
  - b. Membangun kasus bisnis yang dibutuhkan.
2. *Elaboration* (perluasan atau perencanaan)
- Tahap ini lebih difokuskan pada perencanaan arsitektur sistem. Tahap ini juga dapat mendeteksi apakah arsitektur sistem yang diinginkan dapat dibuat atau tidak. Mendeteksi resiko yang mungkin terjadi dari arsitektur yang dibuat. Tahap ini lebih pada analisis dan desain sistem serta implementasi sistem.
3. *Construction* (konstruksi)
- Tahap ini fokus pada pengembangan komponen dan fitur-fitur sistem. Tahap ini lebih pada implementasi dan pengujian sistem yang fokus pada implementasi perangkat lunak pada kode program.
4. *Transition* (transisi)
- Tahap ini lebih pada *deployment* atau instalasi sistem agar dapat dimengerti oleh *user*. Aktifitas pada tahap ini termasuk pada pelatihan *user*, pemeliharaan dan pengujian sistem.

### 2.1.8. Studi Kelayakan

Sutabri (2012:71), setelah mengumpulkan dan mendokumentasi fakta, sistem analis mengetahui apa yang sesungguhnya dilakukan oleh sistem. Selanjutnya sistem analis melakukan studi kelayakan untuk memperhitungkan apakah organisasi atau instansi di mana sistem tersebut dibuat dapat melakukan ke tahap berikutnya dalam proses pembangunan sistem atau tidak. Untuk melakukan hal tersebut sistem analis menggunakan kriteria utama, yaitu studi kelayakan.

Yang dimaksud dengan studi kelayakan adalah proses mempelajari dan menganalisis masalah yang telah ditentukan sesuai dengan tujuan akhir yang dicapai. Ada beberapa aspek yang perlu dipertimbangkan dalam menilai suatu studi kelayakan, yaitu dilihat dari segi:

1. Kelayakan Teknis



Sebuah masalah mempunyai kelayakan teknis, jika tim perancang sistem dapat menyelesaikan masalah tersebut dengan menggunakan *hardware* dan *software* yang tersedia, yaitu yang ada atau yang dapat dimiliki. Dengan kata lain, sistem yang nanti akan diterapkan menggunakan teknologi yang lama atau menggunakan teknologi yang baru.

## 2. Kelayakan Operasional

Sebuah masalah mempunyai kelayakan operasi jika tim perancang sistem dapat menyelesaikan masalah dengan menggunakan personel dan prosedur yang tersedia, yaitu yang ada atau yang dapat dimiliki, dengan kata lain apakah sistem yang baru akan mengubah cara kerja dan struktur organisasi yang telah ada dan telah berjalan saat ini, sehingga dalam memeriksa aspek kelayakan operasi, sistem analisis semestinya memperhitungkan reaksi perubahan sistem.

## 3. Kelayakan Ekonomis

Sebuah masalah mempunyai kelayakan ekonomis jika tim perancang sistem dapat menyelesaikan masalah tersebut dalam waktu dan anggaran biaya yang masuk akal, misalnya keuntungan sistem melebihi biaya penyusunan sistem. Dengan kata lain apakah sistem yang baru lebih menguntungkan dari segi ekonomi.

## 2.2. Teori Judul

### 2.2.1. Pengertian Penentuan

Kamus Besar Bahasa Indonesia (2011:1252), “Penentuan adalah perbuatan (hal tersebut) menentukan; penetapan; pembatasan (arti dsb).”

### 2.2.2. Pengertian Penjurusan

Kamus Besar Bahasa Indonesia (2011:500), “Penjurus adalah baris dsb untuk memberi arah tujuan. Jurusan adalah arah; tuju(an).”

### 2.2.3. Pengertian Bidang Studi

Kamus Besar Bahasa Indonesia (2011:156), “Bidang merupakan permukaan rata dan tentu batasannya.”



Kamus Besar Bahasa Indonesia (2011:1146), “Studi merupakan pembelajaran; penggunaan waktu dan pikiran untuk memperoleh ilmu pengetahuan.”

#### **2.2.4. Pengertian Sekolah Menengah Atas (SMA)**

Kamus Besar Bahasa Indonesia (2011:1054), “Sekolah lanjutan merupakan bangunan atau lembaga untuk belajar dan mempelajari (menurut pelajaran atau tujuannya ada).”

#### **2.2.5. Metode *Naive Bayes***

Naive bayes adalah sebuah pengklasifikasian probabilitas sederhana yang mengaplikasikan teorema bayes. Teori keputusan *bayes* adalah pendekatan statistik (memprediksi peluang dimasa depan, berdasarkan pengalaman di masa sebelumnya). Ide dasar dari bayes adalah menangani masalah yang bersifat hipotesis yakni mendesain suatu klasifikasi untuk memisahkan objek. Misalkan terdapat dua jenis objek dengan kemungkinan kemunculan random, selanjutnya ingin diprediksi objek apa yang akan lewat selanjutnya.

Supranto (2008:350), Seorang ahli matematika dari Inggris, Thomas Bayes (1702-1761), mengembangkan teori untuk menghitung probabilitas tentang sebab-sebab terjadinya suatu kejadian (*causes*) berdasarkan pengaruh yang dapat diperoleh sebagai hasil observasi.

#### **2.2.6. Pengertian Probabilitas**

Arhami (2005:137), Suatu hal yang sudah lama sekali tetapi masih tetap sangat penting sebagai alat dalam penyelesaian masalah AI adalah probabilitas (Frley: 1983). Probabilitas merupakan suatu cara kuantitatif yang berhubungan dengan ketidakpastian yang telah ada sejak abad ke 17.

Supranto (2008:319), Kata probabilitas sering dipertukarkan dengan istilah lain seperti peluang dan kemungkinan.



Secara umum, probabilitas merupakan peluang bahwa sesuatu akan terjadi. Secara lengkap, probabilitas didefinisikan sebagai berikut:

*“Probability” is a measure of a likelihood of the occurrence of a random event.* (Mendenhall dan Reinmuth, 1982).

Terjemahan bebasnya :

*“Probabilitas” ialah suatu nilai yang dipergunakan untuk mengukur tingkat terjadinya suatu kejadian yang acak.*

### **2.2.7. Pengertian Penentuan Penjurusan Bidang Studi Pada Sekolah Menengah Atas (SMA) dengan Metode *Naive Bayes***

Dari uraian diatas dapat disimpulkan bahwa penentuan penjurusan bidang studi pada Sekolah Menengah Atas (SMA) dengan Metode *Naive Bayes* merupakan suatu aplikasi yang berfungsi untuk menentukan penjurusan bidang studi pada Sekolah Menengah Atas (SMA) dengan menggunakan beberapa probabilitas yang ada, sehingga penentuan penjurusan dilakukan berdasarkan kriteria yang telah ditetapkan.

## **2.3. Teori Khusus**

### **2.3.1. Pengertian Rekayasa Perangkat Lunak**

Pressman (2010:15), Rekayasa perangkat lunak merupakan pada dasarnya merupakan aplikasi dari suatu pendekatan yang sistematis, disiplin, dan dapat diukur pada pengembangan, operasi, dan perawatan perangkat lunak. Proses perancangan lunak mencakup lima kegiatan kerangka kerja, diantaranya, komunikasi, perencanaan, pemodelan, konstruksi, dan penyerahan perangkat lunak ke pengguna .

Sukanto dan Shalahuddin (2013:4), Rekayasa perangkat lunak merupakan pembangunan dengan menggunakan prinsip atau konsep rekayasa dengan tujuan menghasilkan perangkat lunak yang bernilai ekonomi yang dipercaya dan bekerja secara efisien menggunakan mesin.



## 2.3.2. Pemrograman Berorientasi Objek

### 2.3.2.1. Pengertian Pemrograman Berorientasi Objek

Sukamto dan Shalahuddin (2013:100), Metodologi berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya. Metodologi berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara sistematis.

### 2.3.2.2. Konsep Dasar Berorientasi Objek

Sukamto dan Shalahuddin (2013:103), Pendekatan berorientasi objek merupakan suatu teknik atau cara pendekatan dalam melihat permasalahan dan sistem atau sistem perangkat lunak, sistem informasi, atau sistem lainnya. Pendekatan berorientasi objek akan memandang sistem yang akan dikembangkan sebagai suatu kumpulan objek yang berkorespondensi dengan objek-objek dunia nyata.

Sistem berorientasi objek merupakan sebuah sistem yang dibangun dengan berdasarkan metode berorientasi objek adalah sebuah sistem yang komponennya dibungkus atau dienkapsulasi menjadi kelompok data dan fungsi. Setiap komponen dalam sistem tersebut dapat mewarisi atribut dan sifat dan komponen lainnya, dan dapat berinteraksi satu sama lain.

Berikut ini adalah beberapa konsep dasar yang harus dipahami tentang metodologi berorientasi objek :

1. Kelas atau *class*

Kelas adalah kumpulan objek-objek dengan karakteristik yang sama. Kelas merupakan definisi statik dan himpunan objek yang sama yang mungkin lahir atau diciptakan dari kelas tersebut.

2. Objek atau *object*

Objek merupakan suatu entitas yang mampu menyimpan informasi (status) dan mempunyai operasi (kelakuan) yang dapat diterapkan atau dapat berpengaruh pada suatu objeknya.



### 3. Metode atau *method*

Operasi atau metode atau *method* pada sebuah kelas hampir sama dengan fungsi atau prosedur pada metodologi struktural. Operasi atau metode merupakan fungsi atau transformasi yang dapat dilakukan terhadap objek atau dilakukan oleh objek.

### 4. Atribut atau *attribute*

Atribut dari sebuah kelas adalah variabel global yang dimiliki sebuah kelas. Atribut dipunyai secara individual oleh sebuah objek, misalnya berat, jenis, nama, dan sebagainya. Atribut sebaiknya bersifat privat untuk menjaga konsep enkapsulasi.

### 5. Abstraksi atau *abstraction*

Prinsip untuk merepresentasikan duni nyata yang kompleks menjadi satu bentuk model yang sederhana dengan mengabaikan aspek-aspek lain yang tidak sesuai dengan permasalahan.

### 6. Enkapsulasi atau *encapsulation*

Pembungkusan atribut data dan layanan (operasi-operasi) yang dipunyai objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerjanya.

### 7. Pewarisan atau *inheritance*

Mekanisme yang memungkinkan satu objek mewarisi sebagian atau seluruh definisi dan objek lain sebagai bagian dan dirinya.

### 8. Antarmuka atau *interface*

Antarmuka atau *interface* sangat mirip dengan kelas, tapi tanpa atribut kelas dan memiliki metode yang dideklarasikan tanpa isi.

### 9. *Reusability*

Pemanfaatan kembali objek yang sudah didefinisikan untuk suatu permasalahan pada permasalahan lainnya yang melibatkan objek tersebut.

### 10. Generalisasi dan Spesialisasi

Menunjukkan hubungan antara kelas dan objek yang umum dengan kelas dan objek yang khusus.

### 11. Komunikasi Antar Objek

---



Komunikasi antar-objek dilakukan lewat pesan (*message*) yang dikirim dan satu objek ke objek lainnya.

12. Polimorfisme atau *polymorphism*

Kemampuan suatu objek untuk dihunukan dibanyak tujuan yang berbeda dengan nama yang sama sehingga menghemat baris program.

13. *Package*

*Package* adalah sebuah kontainer atau kemasan yang dapat digunakan untuk mengelompokkan kelas-kelas sehingga memungkinkan beberapa kelas yang bernama sama disimpan dalam *package* yang berbeda.

### 2.3.3. Kamus Data

Kamus data (*data dictionary*) dipergunakan untuk memperjelas aliran data yang digambarkan pada DFD. Rosa dan Shalahuddin (2013:73), Kamus data adalah kumpulan daftar elemen data yang mengalir pada sistem perangkat lunak sehingga masukan (*input*) dan keluaran (*output*) dapat dipahami secara umum (memiliki standar cara penulisan). Kamus data dalam implementasi program dapat menjadi parameter masukan atau keluaran dari sebuah fungsi atau prosedur.

Kamus data biasanya berisi :

1. Nama-nama dari data
2. Digunakan pada-merupakan proses-proses yang terkait data
3. Deskripsi-merupakan deskripsi data
4. Informasi tambahan-seperti tipe data, nilai data, batas nilai data, dan komponen yang membentuk data.

Kamus data memiliki beberapa simbol untuk menjelaskan informasi tambahan sebagai berikut:

**Tabel 2.1.** Simbol – Simbol Kamus Data

No.	Simbol	Keterangan
1	=	Disusun atau terdiri dari
2	+	Dan
3	[   ]	Baik .... atau ....
4	{ } <sup>n</sup>	n kali diulang atau bernilai banyak
5	( )	Data opsional
6	*.....*	Batas komentar

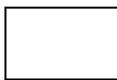
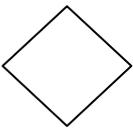
Sumber : Rosa dan Shalahuddin (2013:73)

### 2.3.4. Pengertian ERD (*Entity Relationship Diagram*)

Fatta (2007:121), *Entity Relationship Diagram (ERD)* adalah gambar atau diagram yang menunjukkan informasi dibuat, disimpan, dan digunakan dalam system bisnis. ERD digunakan untuk memodelkan struktur data dan hubungan antar data, karena hal ini relatif kompleks. Dengan ERD kita dapat menguji model dengan mengabaikan proses yang harus dilakukan. Dan dengan ERD kita mencoba menjawab pertanyaan seperti ; data apa yang kita perlukan? Bagaimana data yang satu berhubungan dengan yang lain?

Pada dasarnya ERD menggunakan 3 macam simbol yang digunakan yaitu sebagai berikut :

**Tabel 2.2.** Simbol – Simbol ERD (*Entity Relationship Diagram*)

No.	Simbol	Nama	Deskripsi
1		Entity	Suatu objek yang dapat diidentifikasi dalam lingkungan pemakai, sesuatu yang penting bagi pemakai dalam konteks sistem yang akan dibuat.
2		Hubungan	Sebagaimana halnya entity maka dalam hubungan harus dibedakan antara hubungan atau bentuk hubungan antar entiti dengan isi dari hubungan itu sendiri.



**Lanjutan Tabel 2.2.** Simbol – Simbol ERD (*Entity Relationship Diagram*)

No.	Simbol	Nama	Deskripsi
3		Atribut	Entity mempunyai elemen yang disebut atribut, dan berfungsi mendeskripsikan karakter entity.
4		Link	Baris sebagai penghubung antara himpunan, relasi dan himpunan entitas dan atributnya

Sumber : Fatta (2007 : 121)

### 2.3.5. Pengertian UML (*Unified Modelling Language*)

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

*UML (Unified Modelling Language)* menyediakan serangkaian gambar dan diagram yang sangat baik. Beberapa diagram memfokuskan diri pada ketangguhan teori *object-oriented* dan sebagian lagi memfokuskan pada detail rancangan dan konstruksi. Semua dimaksudkan sebagai sarana komunikasi antar *team programmer* maupun dengan pengguna.

Widodo dan Herlawati (2011:6), “*UML* diaplikasikan untuk maksud tertentu”, biasanya antara lain untuk:

1. Merancang perangkat lunak.
2. Sarana komunikasi antara perangkat lunak dengan proses bisnis.
3. Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem.
4. Mendokumentasi sistem yang ada, proses-proses dan organisasinya.

**Tabel 2.3.** Tipe Diagram *UML (Unified Modelling Language)*

No.	Diagram	Tujuan
1	<i>Class</i>	Memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi, serta relasi-relasi
2	<i>Package</i>	Memperlihatkan kumpulan kelas-kelas, merupakan dari diagram komponen
3	<i>Use case</i>	Diagram ini memperlihatkan himpunan <i>use case</i> dan aktor-aktor (suatu jenis khusus dari kelas)
4	<i>Sequence</i>	Diagram interaksi yang menekankan pada pengiriman pesan dalam suatu waktu tertentu
5	<i>Communication</i>	Sebagai pengganti diagram kolaborasi <i>UML 1.4</i> yang menekankan organisasi struktural dari obyek-obyek yang menerima serta mengirim pesan
6	<i>Statechart</i>	Diagram status memperlihatkan keadaan-keadaan pada sistem, memuat status ( <i>state</i> ), transisi, kejadian serta aktivitas
7	<i>Activity</i>	Tipe khusus dari diagram status yang memperlihatkan aliran dari suatu aktivitas ke aktivitas lainnya dalam suatu sistem
8	<i>Component</i>	Memperlihatkan organisasi serta kebergantungan sistem atau perangkat lunak pada komponen-komponen yang telah ada sebelumnya
9	<i>Deployment</i>	Memperlihatkan konfigurasi saat aplikasi dijalankan ( <i>run-time</i> )

Sumber: Widodo dan Herlawati (2011:10-12)

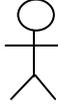
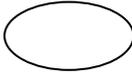
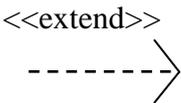
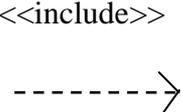
### 2.3.6. Jenis-Jenis Diagram UML (*Unified Modelling Language*)

#### 2.3.6.1. Use Case Diagram

Sukamto dan Shalahuddin (2013:155), “*use case* atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat”.

Adapun simbol-simbol yang digunakan dalam *use case* adalah sebagai berikut:

Tabel 2.4. Simbol-simbol *Use case* Diagram

No.	Simbol	Nama	Deskripsi
1		<i>Actor</i>	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri.
2		<i>Use case</i>	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.
3		<i>Association</i>	Komunikasi antara aktor dan <i>use case</i> yang berpartisipasi pada <i>use case</i> atau <i>use case</i> memiliki interaksi dengan aktor.
4		<i>Extend</i>	Relasi <i>use case</i> tambahan ke sebuah <i>use case</i> yang ditambahkan dapat berdiri sendiri walau tanpa <i>use case</i> tambahan itu.
5		<i>Generalization</i>	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah <i>use case</i> dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.
6		<i>Include</i>	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.

Sumber: Sukanto dan Shalahuddin (2013:156-158)



### 2.3.6.2. Class Diagram

Sukanto dan Shalahuddin (2013:141), “*class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem”.

Diagram kelas dibuat agar pembuat program atau *programmer* membuat kelas-kelas sesuai rancangan di dalam diagram kelas agar antara dokumentasi perancangan dan perangkat lunak sinkron. Adapun simbol-simbol yang digunakan dalam *class diagram* adalah sebagai berikut:

**Tabel 2.5.** Simbol-simbol *Class Diagram*

No	Gambar	Nama	Deskripsi
1		<i>Class</i>	Kelas pada stuktur sistem.
2		<i>Interface</i>	Sama dengan konsep <i>interface</i> dalam pemrograman berorientasi objek.
3		<i>Association</i>	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan <i>multiplicity</i> .
4		<i>Directed association</i>	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain.
5		<i>Generalization</i>	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum khusus).
6		<i>Dependency</i>	Relasi antar kelas dengan makna kebergantungan antar kelas.
7		<i>Aggregation</i>	Relasi antar kelas dengan makna semua bagian ( <i>whole-part</i> ).

Sumber: Sukanto dan Shalahuddin (2013:146-147)



### 2.3.6.3. Activity Digaram

Sukanto dan Shalahuddin (2013:161) “*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 aktor, jadi aktivitas yang dapat dilakukan oleh sistem. Adapun simbol-simbol yang digunakan dalam *activity diagram* adalah sebagai berikut:

**Tabel 2.6.** Simbol-simbol *Activity Diagram*

No.	Simbol	Nama	Deskripsi
1		Status awal	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
2		Aktivitas	Aktivitas yang dilakukan sistem, biasanya diawali dengan kata kerja.
3		Decision	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
4		<i>Join</i>	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
5		Status akhir	Status akhir yang dilakukan sebuah sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
6		<i>Swimlane</i>	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

Sumber: Sukanto dan Shalahuddin (2013:162-163)

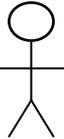
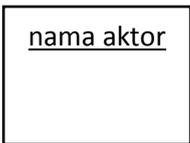


#### 2.3.6.4. Sequence Diagram

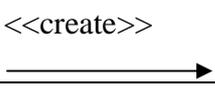
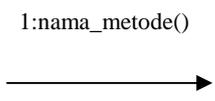
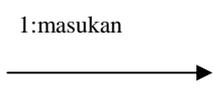
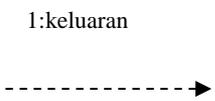
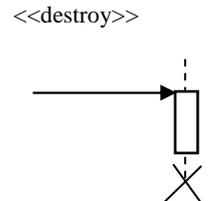
Sukanto dan Shalahuddin (2013:165), “diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek”.

*Sequence* diagram menunjukkan urutan *event* kejadian dalam suatu waktu. Komponen *sequence* diagram terdiri atas obyek yang dituliskan dengan kotak segiempat bernama. *Message* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan progress vertikal. Simbol-simbol yang digunakan dalam *sequence* diagram adalah:

**Tabel 2.7.** Simbol-simbol *Sequence* Diagram

No.	Simbol	Nama	Keterangan
1	 atau  nama aktor	Aktor atau Tanpa Waktu Aktif	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
2		Garis hidup atau <i>lifeline</i>	Menyatakan kehidupan suatu objek.
3		Objek	Menyatakan objek yang berinteraksi pesan.
4		Waktu Aktif	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya.

Lanjutan Tabel 2.7. Simbol-simbol *Sequence* Diagram

No.	Simbol	Nama	Keterangan
5		Pesan tipe <i>create</i>	Menyatakan suatu objek membuat objek lain, arah panah mengarah pada objek yang dibuat.
6		Pesan tipe <i>call</i>	Arah panah mengarah pada objek yang memiliki operasi atau metode, karena ini memanggil operasi atau metode maka operasi atau metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi.
7		Pesan tipe <i>send</i>	Menyatakan bahwa suatu objek mengirimkan data atau masukan atau informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim.
8		Pesan tipe <i>return</i>	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.
9		Pesan tipe <i>destroy</i>	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada <i>create</i> maka ada <i>destroy</i> .

Sumber: Sukamto dan Shalahuddin (2013:165-167)

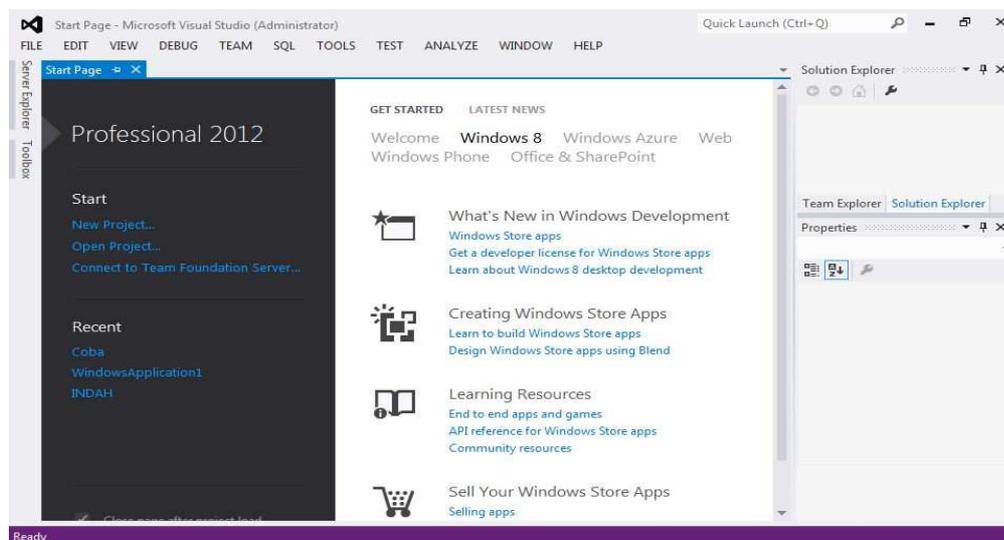


## 2.4. Teori Program

### 2.4.1. Pemrograman *Visual Basic 2012*

#### 2.4.1.1. Pengertian Pemrograman *Visual Basic 2012*

Komputer (2013:3), “*Visual basic* merupakan bahasa pemrograman yang dikeluarkan oleh *Microsoft*. Bahasa pemrograman *visul basic* digunakan untuk membuat aplikasi Windows yang berbasis *Graphical User Interface (GUI)*.”



**Gambar 2.2. Tampilan *Visual Basic 2012***

#### 2.4.1.2. *Visual Studio Toolbox*

Komputer (2013:3), *Toolbox* dapat diartikan sebagai kotak alat. Di dalam *toolbox* terdapat banyak ikon di mana tiap ikon tersebut melambangkan objek yang disebut kontrol. Kontrol pada *toolbox* bisa digunakan untuk membuat *instance* atau kejadian baru di form yang digunakan selama pemrograman.

Penjelasan fungsi-fungsi dari kontrol *toolbox* yang ada di *visual basic 2012* seperti berikut:

Tabel 2.8. *Visual Studio Toolbox*

No.	Toolbox	Keterangan
1	<i>Pointer</i>	Untuk memilih pointer mouse atau melepaskan dari memilih objek-objek <i>toolbox</i> .
2	<i>BackgroundWorker</i>	Mengeksekusi operasi tertentu di <i>thread</i> lain.
3	<i>BindingNavigator</i>	Menyediakan antarmuka untuk navigasi dan manipulasi data yang dikaitkan dengan form.
4	<i>Button</i>	Untuk menerima klik dari pengguna. Jika pengguna mengkliknya maka kode tertentu bisa dieksekusi.
5	<i>CheckBox</i>	Untuk menerima tanda centang dari pengguna. Jika pengguna mencentangnya maka item di kotak tersebut otomatis bisa dipakai untuk program.
6	<i>ComboBox</i>	Menampilkan item-item namun dalam bentuk hanya satu yang terlihat. Jika pengguna mengklik <i>ComboBox</i> barulah semua item didalamnya akan kelihatan.
7	<i>DateTimePicker</i>	Memungkinkan pengguna menampilkan tanggal dan memilihnya. Selanjutnya tanggal dan waktu yang dipilih dapat dimodifikasi menggunakan format tertentu.
8	<i>FontDialog</i>	Menampilkan kotak dialog yang meminta pengguna memilih font yang ada.
9	<i>GroupBox</i>	Mengelompokkan kontrol dalam satu kotak untuk memudahkan pengaturan dan manajemen antarmuka.
10	<i>HscrollBar</i>	Memungkinkan komponen induk untuk menggulirkan isinya secara horizontal. Kontrol ini tidak diperlukan jika komponen induknya telah memiliki atribut <i>Autoscroll</i> dan atribut tersebut sudah dalam keadaan <i>enabled</i>
11	<i>ImageList</i>	Menampilkan daftar gambar. Banyak gambar yang bisa disimpan di <i>imagelist</i> yang nantinya bisa digunakan oleh kontrol lain seperti <i>ListView</i> , <i>TreeView</i> , dan <i>ToolStrip</i> .

Lanjutan Tabel 2.8. *Visual Studio Toolbox*

No.	<i>Toolbox</i>	Keterangan
12	<i>ListBox</i>	Menampilkan item-item dalam bentuk kotak dimana semua item ditampilkan di dalam daftar. Jika jumlah item lebih banyak dari ukuran <i>ListBox</i> maka <i>ListBox</i> otomatis memiliki <i>scrollbar vertikal</i> yang dapat dipakai untuk menampilkan semua item tersebut.
13	<i>ListView</i>	Menampilkan item-item dalam satu dari lima tampilan yang berbeda.
14	<i>OpenFileDialog</i>	Menampilkan jendela dialog yang meminta pengguna untuk memilih file tertentu untuk dibuka.
15	<i>PictureBox</i>	Menampilkan gambar yang tampilannya bisa diatur sedemikian rupa.
16	<i>PrintPreviewDialog</i>	Menampilkan kotak dialog yang menampilkan preview kepada pengguna berupa apa yang akan dicetak.
17	<i>RadioButton</i>	Memungkinkan pengguna untuk memilih satu opsi dari beberapa pilihan yang dikaitkan dengan tombol radio lainnya.
18	<i>TextBox</i>	Memungkinkan pengguna memasukkan teks yang bisa lebih dari satu baris dan menyediakan kemampuan pengeditan dan <i>masking</i> kata sandi.
19	<i>ToolStrip</i>	Komponen yang menyediakan toolbar dan elemen antarmuka lain yang dapat mendukung berbagai opsi tampilan.
20	<i>WebBrowser</i>	Memungkinkan pengguna membuka halaman web dari dalam form.

Sumber: *Komputer (2013:4-9)*



### 2.4.1.3. Panel Properties

Komputer (2013:17), *Panel properties* menampilkan properti dari kontrol yang dipilih. Tiap kontrol akan menyebabkan pemilihan nilai di jendela ini akan berubah. Berikut ini merupakan contoh tampilan jendela properties untuk kontrol form dan arti-arti dari nilai *properties*-nya :

**Tabel 2.9. Panel Properties**

No.	Panel Properties	Keterangan
1	<i>AccessibleDescription</i>	Deskripsi yang akan ditampilkan ke <i>Accessibility Client</i> .
2	<i>AccessibleName</i>	Nama yang akan ditampilkan ke <i>Accessibility Client</i> .
3	<i>AccessibleRole</i>	Peran yang akan ditampilkan di <i>Accessibility Client</i> .
4	<i>AutoScroll</i>	Mengindikasikan apakah scrollbar otomatis diaktifkan jika isi kontrol lebih besar dari area yang terlihat.
5	<i>AutoSize</i>	Menentukan apakah kontrol akan mengatur ulang ukuran dirinya sendiri secara otomatis sesuai dengan kontennya.
6	<i>BackColor</i>	Menentukan warna <i>background</i> dari komponen.
7	<i>BackgroundImage</i>	Menentukan gambar <i>background</i> dari komponen.
8	<i>ControlBox</i>	Menentukan apakah <i>form</i> memiliki Control atau System <i>Menu Box</i> atau tidak.
9	<i>DoubleBuffered</i>	Menentukan apakah kontrol akan di- <i>buffer</i> ganda atau tidak.
10	<i>Font</i>	Menentukan jenis dan ukuran <i>font</i> yang akan ditampilkan di kontrol.
11	<i>HelpButton</i>	Menentukan apakah form memiliki tombol help dibagian caption bar.
12	<i>Icon</i>	Mengindikasikan ikon untuk sebuah form. Ikon ini akan ditampilkan dibagian menu box sistem dan ketika form sedang di- <i>minimize</i> .

Lanjutan Tabel 2.9. *Panel Properties*

No.	<i>Panel Properties</i>	Keterangan
13	<b><i>Locked</i></b>	Menentukan apakah kontrol bisa dipindahkan atau diresize.
14	<b><i>MainMenuStrip</i></b>	Menentukan <i>MenuStrip</i> utama untuk foem. Properti ini berguna untuk aktivasi keyboard dan merger otomatis di MDI.
15	<b><i>Padding</i></b>	Menentukan spasi interior dari sebuah kontrol.
16	<b><i>RightToLeft</i></b>	Menentukan apakah komponen akan menggunakan bahasa <i>Right to Left</i> atau tidak.
17	<b><i>ShowInTaskbar</i></b>	Menentukan apakah form akan muncul di taskbar dari windows.
18	<b><i>Size</i></b>	Menentukan ukuran kontrol dalam satuan pixel.
19	<b><i>Text</i></b>	Menentukan teks yang diasosiasikan untuk kontrol tersebut.
20	<b><i>WindowState</i></b>	Menentukan kondisi awal jendela dari form.

Sumber: *Komputer (2013:17-19)*

#### 2.4.2. Basis Data (*Database*)

Sukanto dan Shalahuddin (2013:43) mengemukakan “*Database* adalah sistem terkomputerisasi yang tujuan utamanya adalah memelihara data yang sudah diolah atau informasi dan membuat informasi tersedia saat dibutuhkan. Pada intinya *database* adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat.”

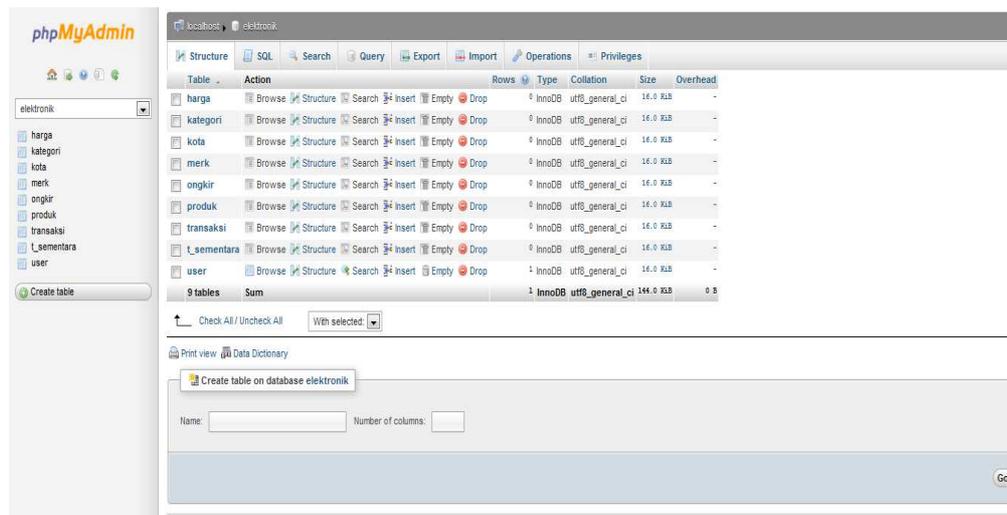
Anhar (2010:45), “*Database* adalah sekumpulan tabel-tabel yang berisi data dan merupakan kumpulan dari *field* atau kolom. Struktur file yang menyusun sebuah *database* adalah *Data Record* dan *Field*.”

Jadi, dapat disimpulkan bahwa *database* merupakan tempat penyimpanan data dari sekumpulan data yang telah diolah.



### 2.4.3. Pengertian MySQL

Nugroho (2013:15), PHP MyAdmin adalah aplikasi manajemen database server MySQL berbasis web. Dengan aplikasi ini kita bisa mengelola database sebagai *root* (pemilik server) atau juga sebagai user biasa, kita bisa membuat database baru, mengelola database dan melakukan operasi perintah-perintah database secara lengkap seperti saat kita di *MySQL Prompt* (versi DOS).



Gambar 2.3. Tampilan PHP MyAdmin

### 2.4.4. XAMPP

Nugroho (2013:1), XAMPP adalah paket program web lengkap yang dapat anda pakai untuk belajar pemrograman web, khususnya PHP dan MySQL, paket ini dapat didownload secara gratis dan legal.

Berikut ini adalah gambar tampilan awal XAMPP.



Gambar 2.4. Tampilan XAMPP