BAB II
TINJAUAN PUSTAKA

2.1 Teori Umum

2.1.1 Pengertian Komputer

Pangera dan Ariyus (2010:3), Komputer adalah perangkat elektronik, beroperasi di bawah perintah pengendali yang disimpan pada memori komputer, dapat menerima data, memproses data berdasarkan aturan tertentu, mencetak hasilnya, dan menyimpan data untuk penggunaan di masa depan.

Sanders dalam Sutarman (2012:2), Komputer adalah sistem elektronik untuk memanipulasi data yang cepat dan tepat serta di rancang dan di organisasikan agar secara otomatis menerima dan menyimpan data *input*, memprosesnya dan menghasilkan *output* dibawah pengawasan suatu langkah-langkah instruksi program yang tersimpan pada memori (*stored program*).

Fuori dalam Sutarman (2012:2), Komputer adalah suatu pemroses data (*data processor*) yang dapat melakukan perhitungan yang besar dan cepat, termasuk perhitungan aritmatika yang besar atau operasi logika, tanpa campur tangan dari manusia mengoperasikan selama pemrosesan.

Sutarman (2012:3), Komputer adalah alat yang dapat membaca *input* data dan mengolahnya sesuai dengan program yang ditetapkan untuk menghasilkan informasi yang merupakan *output* hasil pemrosesan *input* data.

Sujatmiko (2012:156), Komputer merupakan mesin yang dapat mengolah digital dengan mengikuti serangkaian perintah atau program.

Asropudin (2013:19), *Computer* merupakan alat bantu pemrosesan data secara elektronik dan cara pemrosesan datanya berdasarkan urutan instruksi atau program yang tersimpan dalam memori masing-masing komputer.

Kadir dan Triwahyuni (2013:2), Komputer adalah mesin serbaguna yang dapat dikontrol oleh program, digunakan untuk mengolah datamenjadi informasi.

Dari beberapa definisi komputer diatas penulis menyimpulkan bahwa komputer adalah suatu alat elektronik yang digunakan untuk memroses data yang menghasilkan informasi yang bermanfaat dan dapat bekerja secara otomatis.

2.1.2 Pengertian Perangkat Lunak (*Software*)

Sujatmiko (2012:256), *Software* adalah kumpulan beberapa perintah yang di eksekusi oleh mesin komputer dalam menjalankan pekerjaannya.

Sutarman (2012:88), Perangkat lunak (*software*) merupakan suatu program yang berisi barisan instruksi (perintah) yang ditulis dalam bahasa komputer yang dimengerti oleh *hardware* komputer.

Sutarman (2012:145), secara khusus *software* terbagi menjadi lima jenis:

1. Sistem operasi (*operating system*),
2. Bahasa pemrograman,
3. Program aplikasi,
4. Alat bantu (*utility*), dan
5. *User* program.

Sukamto dan Shalahuddin (2014:2), Perangkat lunak (*software*) adalah program komputer yang terasosiasi dengan dokumentasi perangkat lunak dokumentasi kebutuhan, model desain, dan cara penggunaan (*user manual*).

Dari beberapa definisi diatas penulis menyimpulkan bahwa *software* adalah serangkaian instruksi elektronik yang dapat mengoperasikan suatu pekerjaan sesuai dengan kebutuhan.

2.1.3 Pengertian Program

Sujatmiko (2012:223), Program merupakan serangkaian petunjuk berupa perintah-perintah yang disusun sedemikian rupa melaksanakan suatu tugas yang akan dikerjakan oleh komputer.

Sutarman (2012:3), Program adalah barisan perintah/instruksi yang disusun sehingga dapat dipahami oleh komputer dan kemudian dijalankan sebagai barisan perhitungan numerik, di mana barisan perintah tersebut berhingga, berakhir dan menghasilkan *output*.

Kadir dan Triwahyuni (2013:2), Program adalah deretan instruksi yang digunakan untuk mengendalikan komputer sehingga komputer dapat melkukan tindakan sesuai yang dikehendaki pembuatnya.

Dari beberapa definisi diatas penulis menyimpulkan bahwa program adalah serangkaian instruksi yang dioperasikan untuk melaksanakan suatu tugas tertentu.

2.1.4 Pengertian Data

Sujatmiko (2012:76), Data merupakan kumpulan dari angka-angka maupun karakter-karakter yang tidak memiliki arti.

Sutarman (2012:3), Data adalah fakta dari suatu pernyataan yang berasal sari kenyataan, dimana pernyataan tersebut merupakan hasil pengukuran atau pengamatan.

Asropudin (2013:22), Data adalah kumpulan dari angka-angka maupun karakter-karakter yang tidak memiliki arti.

Kadir dan Triwahyuni (2013:2), Data adalah bahan mentah bagi komputer yang dapat berupa angka maupun gambar.

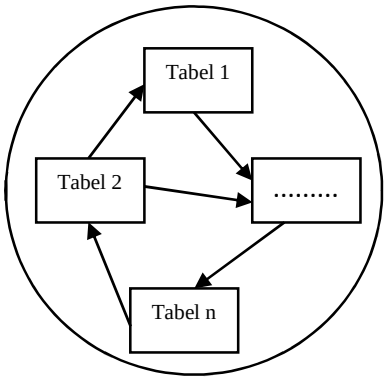
Dari beberapa definisi diatas dapat disimpulkan bahwa data adalah sekumpulan fakta yang harus diolah lebih lanjut untuk menghasilkan suatu informasi.

2.1.5 Pengertian *Database*

Sujatmiko (2012:76), *Database* adalah representasi kumpulan fakta yang saling berhubungan disimpan secara bersama sedemikian rupa dan tanpa pegulangan (*redundansi*) yang tidak perlu, untuk memenuhi berbagai kebutuhan.

Sukamto dan Shalahuddin (2014:43), Basis data adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat.

Kadir dan Triwahyuni (2013:339), Basis data adalah suatu pengorganisasian sekumpulan data yang saling terkait sehingga memudahkan aktivitas untuk memperoleh informasi.



Gambar 2.1 Ilustrasi Basis Data

(*Sumber: Sukamto dan Shalahuddin, 2014:44*)

2.2 Teori Judul

2.2.1 Pengertian Aplikasi

Asropudin (2013:7), Aplikasi adalah program komputer yang dibuat untuk mengerjakan atau menyelesaikan masalah-masalah khusus."

Sujatmiko (2012:23), *Aplication* adalah program komputer yang dibuat oleh suatu perusahaan komputer untuk membantu manusia dalam mengerjakan tugas-tugas tertentu, misalnya *Ms-Word* dan *Ms-Excel.*

Dari beberapa definisi diatas dapat disimpulkan bahwa aplikasi adalah perangkat lunak yang digunakan untuk mengerjakan masalah-masalah tertentu.

2.2.2 Pengertian Pengolahan Data

Hutahaean (2012:8), Pengolahan data adalah serangkaian operasi atas informasi yang direncanakan guna mencapai tujuan atau hasil yang diinginkan.

Berikut adalah unsur pokok pengolahan data, yaitu:

1. Membaca
2. Menulis, mengetik
3. Mencatat atau mencetak
4. Menyortir
5. Menyampaikan atau memindahkan
6. Menghitung
7. Membandingkan
8. Menyimpan.

Sutarman (2012:4), Pengolahan data (*data processing*) adalah proses perhitungan/transformasi data *input* menjadi informasi yang mudah dimengerti ataupun sesuai dengan yang diinginkan.

Gambar 2.2 Siklus Pengolahan Data

Dapat disimpulkan bahwa pengolahan data adalah serangkaian proses yang mengubah data menjadi informasi yang memiliki nilai dan manfaat bagi sekelompok orang tertentu.

2.2.3 Pengertian Pasien

Berdasarkan UU RI No 44 Tahun 2009 tentang Rumah Sakit dalam BAB 1 Pasal 1 Butir 4, Pasien adalah setiap orang yang melakukan konsultasi masalah kesehatannya untuk memperoleh pelayanan kesehatan yang diperlukan, baik secara langsung maupun tidak langsung di Rumah Sakit.

Pasien adalah orang sakit atau orang yang menjalani pengobatan untuk kesembuhan penyakitnya. (www.depkes.go.id, diakses pada 1 Juni 2015 pukul 18.39 WIB)

Dari beberapa definisi diatas dapat disimpulkan bahwa pasien adalah orang yang menderita,orang yang sakit, yang membutuhkan pelayanan kesehatan tertentu.

2.2.4 Pengertian Rawat Jalan

Menurut Surat Keputusan Menteri Kesehatan RI no 560/MENKES/SK/IV/2003 tentang Pola Trif Perjan Rumah Sakit pada BAB 1 Pasal 1 Butir 4, bahwa Rawat Jalan adalah pelayanan pasien untuk observasi, diagnosis, pengobatan, rehabilitasi medik dan pelayanan kesehatan lainnya tanpa menginap di Rumah Sakit.

2.2.5 Pengertian Rawat Inap

Menurut Surat Keputusan Menteri Kesehatan RI no 560/MENKES/SK/IV/2003 tentang Pola Trif Perjan Rumah Sakit pada BAB 1 Pasal 1 Butir 6, bahwa Rawat Inap adalah pelayanan pasien untuk observasi diagnosis, pengobatan, rehabilitasi medik dan atau upaya pelayanan kesehatan lainnya dengan menginap di Rumah Sakit.

2.2.6 Pengertian Rumah Sakit

Berdasarkan UU RI No 44 Tahun 2009 tentang Rumah Sakit dalam BAB 1 Pasal 1 Butir 1, Rumah Sakit adalah institusi pelayanan kesehatan yang menyelenggarakan pelayanan kesehatan perorangan secara paripurna yang menyediakan pelayanan rawat inap, rawat jalan, dan gawat darurat.

Rumah Sakit adalah suatu fasilitas yang menyediakan rawat inap dan rawat jalan yang memberikan pelayanan kesehatan jangka pendek dan jangka panjang yang terdiri dari observasi, diagnostik, terapeutik dan rehabilitatif untuk orang-

orang yang menderita sakit, cidera dan melahirkan. (www.depkes.go.id, diakses pada 1 Juni 2015 pukul 18.34 WIB)

Menurut Surat Keputusan Menteri Kesehatan RI no 560/MENKES/SK/IV/2003 tentang Pola Trif Perjan Rumah Sakit pada BAB 1 Pasal 1 Butir 1, bahwa Rumah Sakit adalah sarana kesehatan yang menyelenggarakan pelayanan kesehatan kepada masyarakat, baik dalam bentuk promotif, kuratif maupun rehabilitatif secara paripurnayang mempunyai status sebagai Perusahaan Jawatan.

Dari beberapa definisi diatas dapat disimpulkan bahwa rumah sakit adalah suatu tempat penyediaan jasa untuk merawat orang sakit.

2.2.7 Pengertian Rumah Sakit Mata

Rahayu (2014:476), Rumah Sakit Mata adalah rumah sakit yang khusus merawat orang-orang atau penderita sakit mata.

2.2.8 Pengertian Rumah Sakit Khusus Mata Masyarakat Provinsi Sumatera Selatan

Rumah Sakit Khusus Mata Masyarakat Provinsi Sumatera Selatan merupakan rumah sakit khusus mata sebagai instansi kesehatan untuk memenuhi kebutuhan Masyarakat Provinsi Sumatera Selatan dalam layanan kesehatan khusus mata yang profesional.

2.2.9 Pengertian Aplikasi Pengolahan Data Pasien Rawat Jalan dan Rawat Inap pada Rumah Sakit Khusus Mata Masyarakat Provinsi Sumatera Selatan

Merupakan suatu perangkat lunak yang dikembangkan untuk difungsikan secara khusus dalam mengelolah data pasien baik itu pasien rawat inap maupun rawat jalan serta mampu membuat laporan data pasien yang melakukan rawat inap dan rawat jalan di Rumah Sakit Khusus Mata Masyarakat Provinsi Sumatera Selatan, sehingga pengolahan data pasien dapat dilakukan secara terkomputerisasi dengan baik.

2.3 Teori Khusus

2.3.1 Pemrograman Berorientasi Objek

2.3.1.1 Pengertian Pemrograman Berorientasi Objek

Asropudin (2013:75), *OOP* adalah singkatan dari "*Object-Oriented-Programming*". *OOP* mengacu pada metodologi pemrograman berbasis objek.

Sutarman (2012:165), Bahasa pemrograman berorientasi objek adalah suatu program komputer yang dapat dipandang sebagai sekumpulan dari unit tunggal atau objek yang dapat melakukan aksi atau tindakan satu sama lain.

Sukamto dan shalahuddin (2014:100), Metodologi Berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya.

Keuntungan menggunakan metodologi berorientasi objek adalah:

1. Meningkatkan produktivitas

   Karena kelas dan objek yang ditemukan dalam suatu masalah masih dapat dipakai ulang untuk masalah lainnya yang melibatkan obek tersebut (*reusable*).

2. Kecepatan pengembangan

   Karena sistem yang dibangun dengan baik dan benar pada saat analisis dan perancangan akan menyebabkan bekurangnya kesalahan pada saat pengkodean.

3. Kemudahan pemeliharaan

   Karena dengan model objek, pola-pola yang cenderung tetap dan stabil dapat dipisahkan dan pola-pola yang mungkin sering berubah-ubah.

4. Adanya konsistensi

   Karena sifat pewarisan dan penggunaan notasi yang sama pada saat analisis, perancangan maupun pengkodean.

5. Meningkatkan kualitas perangkat lunak

   Karena pendekatan pengembangan lebih dekat dengan dunia nyata dan adanya konsistensi pada saat pengembangannya, perangkat lunak yang

dihasilkan akan mampu memenuhi kebutuhan pe,akai serta mempunyai sedikit kesalahan.

2.3.1.2 Karakteristik Pemrograman Berorientasi Objek

Supardi (2015:151), Untuk mengetahui apakah program yang dibuat oleh pengembang (programmer) tergolong perogram berorientasi objek atau tidak. Berikut ini terdapat tiga karakteristik yang termasuk dalam karakteristik PBO.

1. Pewarisan (*Inheritance*)

Pewarisan merupakan suatu teknik membuat *class* atau objek (kelas turunan/*derived class*) dengan mewariskan kelas, sehingga memiliki sifat dan karakteristik yang sama dengan kelas induknya (kelas dasar/*base class*). Bentuk umum pewarisan sebagai berikut.

```
Public Class DerivedClass
     Inherits BaseClass
End Class
```

2. Pengapsulan (*Encapsulation*)

Pengapsulan adalah sebuah teknik di mana data dan fungsi/metode disatukan (dibungkus).

3. *Polymorphism*e

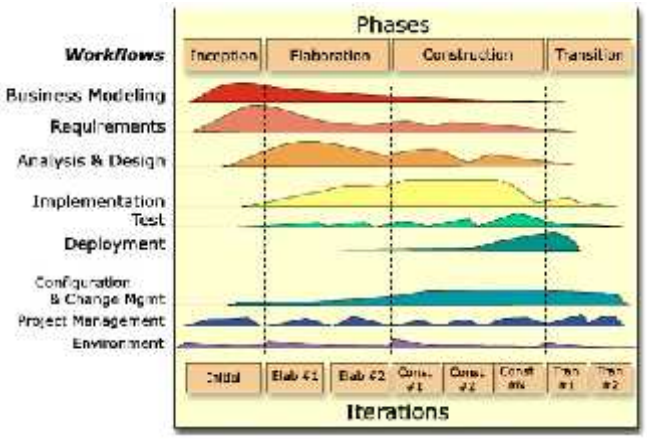
Polimorfisme merupakan suatu teknik dimana satu metode digunakan oleh objek yang berbeda. Biasanya menggunakan teknik *overloading*.

2.3.2 Metodologi *Rational Unified Process* (RUP)

Siahaan (2012:183), *Rational Unified Process* (RUP) adalah salah satu kerangka kerja untuk melakukan proses rekayasa kebutuhan.

Sukamto dan Shalahuddin (2014:125), *Rational Unified Process* (*RUP*) adalah pendekatan pengembangan perangkat lunak yang dilakukan berulang-ulang (*iterative*), fokus pada arsitektur (*architecture-centric*), lebih diarahkan berdasarkan penggunaan kasus (*use case driven*).

Rational Unified Process (*RUP*) menggunakan konsep *object oriented*, dengan aktifitas yang berfokus pada pengembangan model dengan menggunakan *Unified Model Language* (*UML*).



Gambar 2.3 Arsitektur *Rational Unified Process*

(*Sumber: http://sae-semlinda.blogspot.com*)

Melalui gambar diatas dapat dilihat bahwa *RUP* memiliki, yaitu:

1. Dimensi Pertama

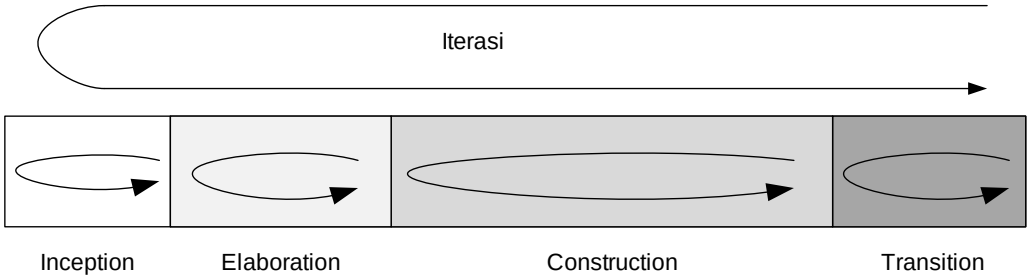
Digambarkan secara horizontal. Dimensi ini mewakili aspek-aspek dinamis dari pengembangan perangkat lunak. Aspek ini dijabarkan dalam tahapan pengembangan atau fase. Setiap fase akan memiliki suatu *major milestone* yang menandakan akhir dari awal dari *phase* selanjutnya. Setiap *phase* dapat terdiri dari satu beberapa iterasi. Dimensi ini terdiri atas *Inception, Elaborationn, Construction*, dan *Transition.*

2. Dimensi Kedua

Digambarkan secara vertikal. Dimensi ini mewakili aspek-aspek statis dari proses pengembangan perangkat lunak yang dikelompokkan ke dalam beberapa disiplin. Proses pengembangan perangkat lunak yang dijelaskan kedalam beberapa disiplin terdiri dari empat elemen penting, yakni *who is doing*, *what, how,* dan *when.* Dimensi ini terdiri atas:

Bussines Modeling, Requirement, Analysis and Design, Implementation, Test, Deployment, Configuration, dan Change Management, Project Management,Environment.

2.3.2.1 Penerapan Tahap Metodologi Pengembangan Perangkat Lunak dengan *RUP*



Gambar 2.4 Alur hidup RUP

(*Sumber: Sukamto dan Shalahuddin, 2014:128*)

Sukamto dan Shalahuddin (2014:129), Dalam *Rational Unified Process* terdapat empat tahap pengembangan perangkat lunak yaitu:

1. *Inception* (permulaan)

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

2. *Elaboration* (perluasan/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 yang fokus pada purwarupa sistem (*prototype*).

3. *Construction* (kontruksi)

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. Tahap ini menghasilkan produk perangkat lunak dimana menjadi syarat dari *Initial Operational Capability Milestone* atau batas/tonggak kemampuan operational awal.

4. *Transition* (transisi)

Tahap ini lebih pada deployment atau instalasi sistem agar dapat dimengerti oleh *user*. Tahap ini menghasilkan produk perangkat lunak dimana menjadi syarat

dari *Initial Operational Capability Milestone* atau batas/tonggak kemampuan operasional awal. Akitfitas pada tahap ini termasuk pada pelatihan *user*, pemeliharaan dan pengujian sistem apakah sudah memenuhi harapan *user.*

2.3.2.2 Aliran Kerja Utama *RUP*

Adapun aliran kerja utama pada Metodologi *RUP* adalah sebagai berikut:

1. Pemodelan Bisnis (*Bussines Modeling*)
   Mendeskripsikan struktur dan proses-proses bisnis organisasi.
2. Kebutuhan (*Requirement*)
   Mendefinisikan kebutuhan perangkat lunak dengan menggunakan metode *use case.*
3. Analisis dan Perancangan (*Analysis and Design*)
   Mendeskripsikan berbagai arsitektur perangkat lunak dari berbagai sudut pandang.
4. Implementasi (*Implementation)*
   Menuliskan kode-kode program, menguji, dan mengintegrasikan unit-unit programnya.
5. Pengujian (*Test*)
   Mendeskripsikan kasus uji, prosedur, dan alat ukur pengujian.
6. *Deployment*
   Menangani konfigurasi sistem yang akan diserahkan.

2.3.2.3 Aliran Kerja Pendukung *RUP*

Adapun aliran kerja pendukung *RUP* adalah sebagai berikut:

1. Manajemen konfigurasi dan perubahan (*configuration and change management*), mengendalikan perubahan dan memelihara artifak-artifak proyek.
2. Manajemen proyek (*Project Management*), mendeskripsikan berbagai stategi pekerjaan dengan proses yang berulang.
3. Lingkungan (*Environment*), menangani infrastuktur yang dibutuhkan untuk mengembangkan sistem.

2.3.3 *Unified Modelling Language* (*UML*)

Asropudin (2013:103), *UML* adalah singkatan dari "*Unfied Modeling Language*" Ini adalah bahasa pemrograman yang digunakan untuk perangkat lunak berorientasi objek.

Siahaan (2012:184), UML adalah sebuah bahasa standar industri yang membantu komunitas kebutuhan perangkat lunak saling berkomunikasi.

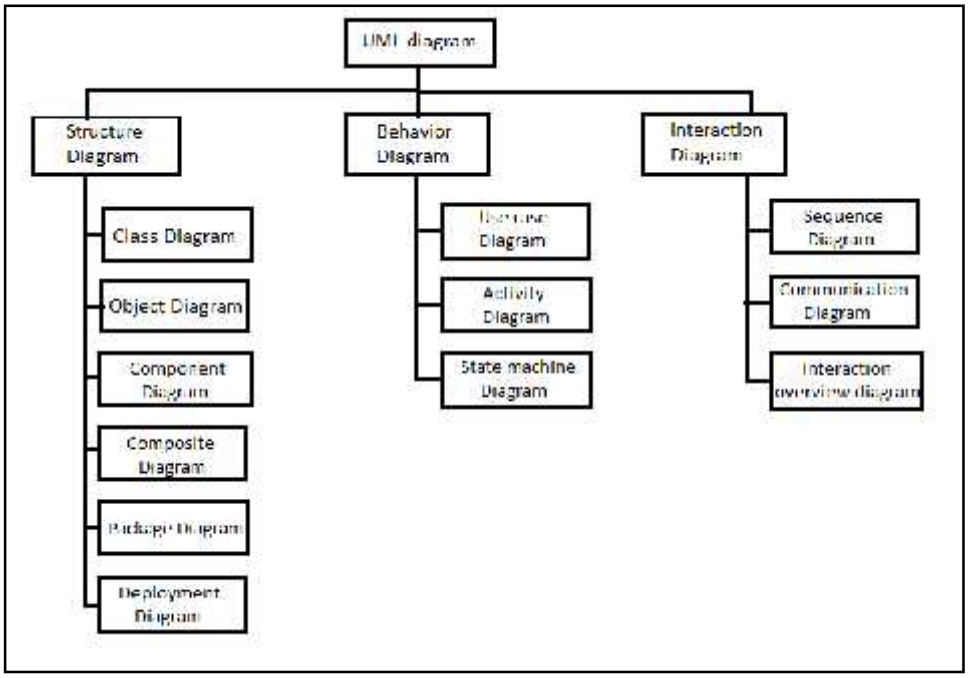
Sukamto dan Shalahuddin (2014:137), *UML* adalah bahasa virtual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan tek-teks pendukung.

Widodo dan Herlawati (2011:6), UML merupakan alat komunikasi yang konsisten dalam mensuport para pengembang sistem saat ini.

Supardi (2015:152), UML (*Unfied Modelling Language*) merupakan suatu bahasa pemodelanstandar internasional, yang memiliki beberapa diagram.

2.3.3.1 Macam-macam Diagram *UML*

Sukamto dan Shalahuddin (2014:140), Pada *UML* terdiri dari 13 macam diagram yang dikelompokkan dalam 3 kategori. Pembagian kategori dan macam-macam diagram tersebut dapat dilihat pada gambar dibawah ini:



Gambar 2.5 Macam-macam Diagram UML

(*Sumber: Sukamto dan Shalahuddin, 2014:140*)

Berikut ini penjelasan singkat dari pembagian kategori tersebut.

a. *Structure Diagram*

Yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.

b. *Behavior Diagram*

Yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.

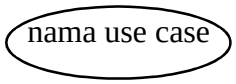
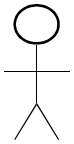
c. *Interaction Diagram*

Yaitu kumpulan diagram yang digunakan untuk menggambarkan interkasi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.

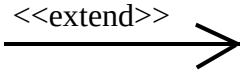
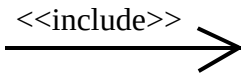
2.3.4 Diagram *Use Case* (*Use Case Diagram*)

Sukamto dan Shalahuddin (2014:155), *Use case* atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat.

Tabel 2.1 Simbol-simbol Diagram *Use Case* (*Use Case Diagram*)

Simbol	Keterangan
Use case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case*
Actor  nama aktor	Orang, proses, atau sistem lain yang berinterksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor

Lanjutan Tabel 2.1 Simbol-simbol Diagram *Use Case* (*Use Case Diagram*)

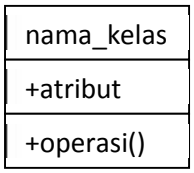
Simbol	Keterangan
Asosiasi/*association*	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor
Ekstensi/*extend* 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu; mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek; biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan
Generalisasi/*generalization* 	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya
Menggunakan/*include*/*uses* 	Relasi *use case* tambahan ke sebuah *use case* di mana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini

(*Sumber: Sukamto dan Salahuddin, 2014:156*)

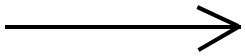
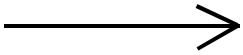
2.3.5 Diagram Kelas (*Class Diagram)*

Sukamto dan Shalahuddin (2014:141), Diagram kelas atau class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem.

Tabel 2.2 Simbol-simbol Diagram Kelas (*Class Diagram*)

Simbol	Deskripsi
Kelas 	Kelas pada struktur sistem

Lanjutan Tabel 2.2 Simbol-simbol Diagram Kelas (*Class Diagram*)

Simbol	Deskripsi
Antarmuka/*interface*  **nama_interface**	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
Asosiasi/*association* 	Relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*
Asosiasi berarah/*directed association* 	Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*
Generalisasi 	Relasi antarkelas dengan makna generalisasi-spesialisasi (umum khusus)
Kebergantungan/*dependency* 	Kebergantungan antarkelas
Agresasi/*aggregation* 	Relasi antarkelas dengan makna semua-bagian (*whole-part*)

(*Sumber: Sukamto dan Shalahuddin, 2014:146*)

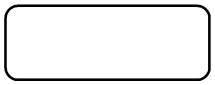
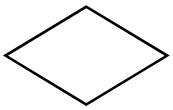
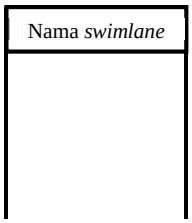
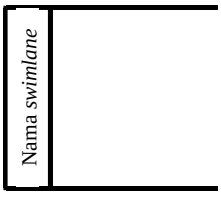
2.3.6 Diagram Aktivitas (*Activity Diagram*)

Sukamto dan Shalahuddin (2014:161), Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak.

Tabel 2.3 Simbol-simbol Diagram Aktivitas (*Activity Diagram*)

Simbol	Keterangan
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal

Lanjutan Tabel 2.3 Simbol-simbol Diagram Aktivitas (*Activity Diagram*)

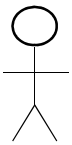
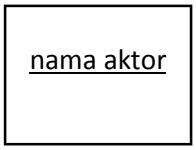
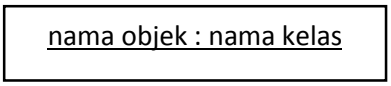
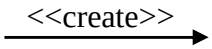
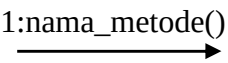
Simbol	Keterangan
Aktivitas  aktivitas	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
Percabangan/*decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
Penggabungan/*join* 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
Swimlane  Atau 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

(*Sumber: Sukamto dan Shalahuddin, 2014:162*)

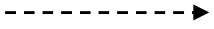
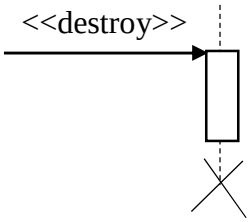
2.3.7 Diagram Sekuen (*Sequence Diagram*)

Sukamto dan Shalahuddin (2014:165), Diagram sekuan menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek.

Tabel 2.4 Simbol-simbol Diagram Sekuen (*Sequence Diagram*)

Simbol	Keterangan
Aktor  atau  Tanpa Waktu	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 gamabr orang, belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
Garis hidup/*lifeline*	Menyatakan kehidupan suatu objek
Objek 	Menyatakan objek yang berinteraksi pesan
Waktu aktif 	Mmenyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya
Pesan tipe *create* 	Menyatakan suatu objek membuat objek lain, arah panah mengarah pada ojek yang dibuat
Pesan tipe *call* 	Arah panah mengarah pada objek yang memiliki operasi/metode, operasi/metode yang dipanggil harus ada pada diagram kelas sesuai denga kelas objek yang berinteraksi

Lanjutan Tabel 2.4 Simbol-simbol Diagram Sekuen (*Sequence Diagram*)

Simbol	Keterangan
Pesan tipe *send* 1:masukan 	Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi
Pesan tipe *return* 1:keluaran 	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
Pesan tipe *destroy* 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada *create* maka ada *destroy*

(*Sumber: Sukamto dan Shalahuddin, 2014:165*)

2.4 Teori Program

2.4.1 Pemograman *Visual Basic .Net*

2.4.1.1 Pengertian Pemrograman *Visual Basic. Net*

Sujatmiko (2012:307), *Visual Basic* adalah bahasa komputer yang digunakan untuk membuat program interaktif di dalam sebuah *web page*, yang dikembangkan oleh perusahaan Microsoft Corp.

Menurut Asropudin (2013:105), *Visual Basic* adalah bahasa komputer yang digunakan untuk membuat program interaktif di dalam sebuah webpage.

Dari beberapa definisi diatas dapat disimpulkan bahwa *Visual Basic .Net* adalah bahasa komputer yang digunakan untuk membuat program interaktif.

2.4.1.2 Tipe Data dalam *Visual Basic .Net*

Darmayuda (2014:21), Setiap bahasa pemrograman termasuk *Visual Basic .Net* tentunya memiliki tipe data. *Variabel* dan *Konstanta* pada *Visual Basic. Net*

memiliki tipe data yang menentukan suatu nilai yang dapat ditampung oleh tipe *variabel* dan *konstanta* itu sendiri.

Tabel 2.5 Tipe Data Dalam *Visual Basic .Net*

Tipe Data	Ukuran (dalam bytes)	Deskripsi Jangkauan
Byte	1	Nilai antara; 0 s.d. 255
Boolean	1	Bernilai; True (benar) atau False (salah)
Char	2	Menampung karakter *Unicode*
Datetime	8	Nilai tanggal; 1/1/0001 jam 11:59:59 s.d. Tanggal 12/21/9999
Decimal	16	Untuk nilai negatif, antara: -29228162514264222592542950225 s.d. 0.00000000000000000000000001 Untuk nilai positif, antara: 0.00000000000000000000000001 s.d. 29228162514264222592542950225
Double	8	Untuk nilai negatif, antara: -1.29269212486222 E208 s.d. 4.94065645841242 E-224 Untuk nilai positif, antara: 4.94065645841242 E-224 s.d. 1.29269212486222 E208
Int16	2	-22268 s.d. 22262
Int32		-21424826048 s.d. 2142482642
Int64	8	-9222222026854225808 s.d. 9222222026854225808
Integer	4	-21424826048 s.d. 2142482642
Long	8	-9222222026854225808 s.d. 9222222026854225808
Short	2	-22268 s.d. 22262

Lanjutan Tabel 2.5 Tipe Data Dalam *Visual Basic .Net*

Tipe Data	Ukuran (dalam bytes)	Deskripsi Jangkauan
Single	4	Untuk nilai negatif, antara: -2.402822 E28 s.d. -1.401298 E-45 Unyuk nilai positif, antara: 1.401298 E-45 s.d. 2.402822 E28
String		Deretan dari karakter-karakter *Unicode*
UInt16	2	0 s.d. 65525
UInt32	4	0 s.d. 429462295
UInt64	8	0 s.d. 1844624400222095551615

(*Sumber: Darmayuda, 2014:21*)

2.4.1.3 Mendeklarasi Variabel *Visual Basic .Net*

Darmayuda (2014:26), Variabel dapat dideklarasikan dengan dua cara, yaitu:

1. Deklarasi *Eksplisit*

Deklarasi *Eksplisit* mengandung arti bahwa program harus menggunakan statement atau pernyataan untuk mendeklarasikan *variabel*. Berikut adalah *statemen-statemen* atau pernyataan untuk mendeklarasikan suatu *variabel* sebagai berikut:

Dim VarName [As DataType]

Private VarName [As DataType]

Static VarName [As DataType]

Public VarName [As DataType]

Contoh :

```
Dim i As Integer
Dim nama As String
Dim nama_brg As String
Dim banyakbarang As Long
```

2. Deklarasi *Implisit*

Deklarasi *Implisit* dilakukan tanpa menggunakan kata kunci, statemen atau pernyataan untuk mendeklarasikan sebuah variabel. Berikut adalah daftar tipe variabel *implisit:*

Tabel 2.6 Tabel Daftar Tipe *Variable Implisit*

Tipe Variabel	*Suffix*
Integer	%
Long	&
Single	!
Double	#
Currency	@
String	$

(*Sumber: Darmayuda, 2014:27*)

Contoh: `Dim i%` ‘deklarasi variabel i adalah *Integer*

`Dim nama$` ‘deklarasi variabel nama adalah *String*

`Dim banyak&` ‘deklarasi variabel banyak adalah *Long*

2.4.1.4 Operator dalam *Visual Basic .Net*

a. Operator Aritmatika

Darmayuda (2014:30), Operator aritmatika berfungsi untuk melakukan operasi matematika. Operator aritmatika juga merupakan operator yang memiliki hirarki tertinggi dibandingkan operator lainnya.

Tabel 2.7 Operator Aritmatika

Operator	**Keterangan**
+	Penjumlahan
-	Pengurangan
*	Perkalian
/	Pembagian dengan hasil bulat (integer)
^	Pangkat
Mod	Menghitung sisa pembagian (modulus)
+=	Inkremen penambahan

Lanjutan Tabel 2.7 Operator Aritmatika

Operator	Keterangan
-=	Inkremen pengurangan
***=**	Inkremen perkalian

(*Sumber: Darmayuda, 2014:31*)

b. Operator Penugasan

Darmayuda (2014:31), Operator penugasan berfungsi untuk memasukkan nilai suatu ekspresi ke ekspresi yang lainnya. Simbol yang digunakan untuk operator penugasan adalah simbol sama dengan (=).

c. Operator Pembanding

Darmayuda (2014:32), Operator pembanding berfungsi untuk membandingkan suatu ekspresi/nilai1 dengan ekspresi/nilai2 yang lain, dimana hasilnya merupakan sebuah nilai logika True atau False.

Tabel 2.8 Operator Pembanding

Operator	Keterangan
=	Sama dengan
>	Lebih besar dari
<	Lebih kecil dari
<>	Tidak sama dengan
>=	Lebih besar atau sama dengan
<=	Lebih kecil atau sama dengan
Between	Menentukan antara nilai
Like	Pencarian dengan pola
In	Mencari data dalam nilai-nilai tertentu

(*Sumber: Darmayuda, 2014:32*)

d. Operator Logika

Darmayuda (2014:32), Operator logika berfungsi untuk melakukan operasi logika (*boolean*) dan menghasilkan nilai yang bertipe true (benar) atau false (salah).

Tabel 2.9 Operator Logika

Operator	Ekspresi	Hasil	Keterangan
And	X1=2 AND X2=5	False	Nilai true, bila X1 dan X2 bernilai true
Or	X1=2 OR X2=5	True	Nilai true, bila X1 atau X2 bernilai true
Xor	X1=2 Xor X2=5	False	Nilai true, bila X1 atau X2 bernilai true
Not	Not X1=2	True	Nilai true, bila X1 atau X2 bernilai false

(*Sumber: Darmayuda, 2014:33*)

2.4.2 Pengertian *Graphical User Interface* (GUI)

Sujatmiko (2012:119), *Graphical User Interface* (GUI) adalah antarmuka pengguna grafis. Metda interaksi secara grafis antara pengguna dan komputer.

Asropudin (2013:39), *Graphical User Interface* (GUI) adalah *user interface* yang didasarkan pada grafik. Termasuk dalam GUI adalah *icons, pulldown menus* dan *mouse.*

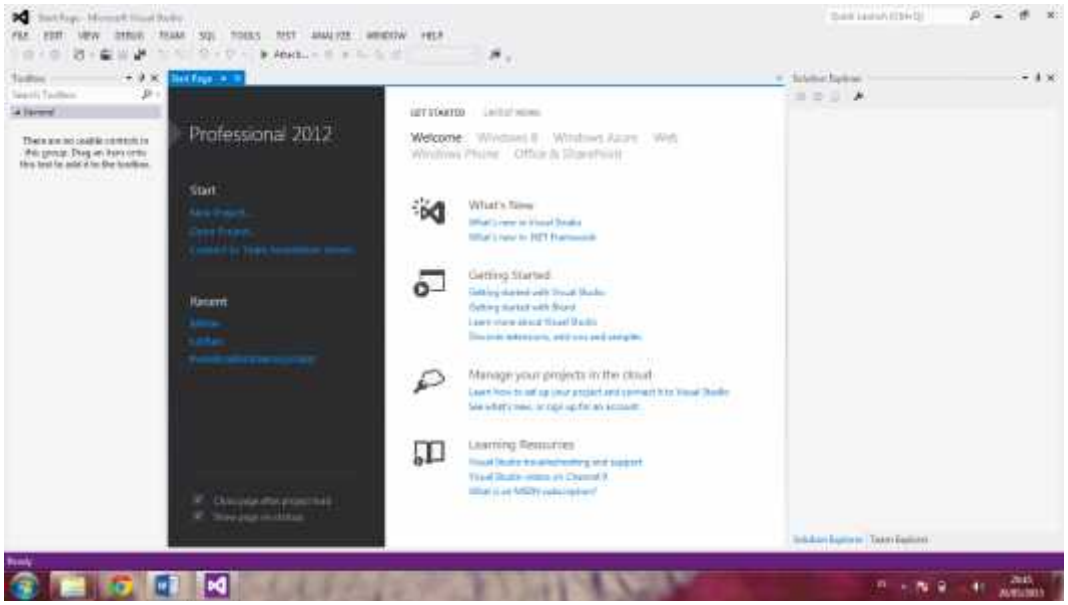
2.4.3 Mengenal *Integrated Development Environment* (IDE)

Sujatmiko (2012:137), *Integrated Development Environment* (IDE) adalah perangkat kontrol yang digunakan sebagai antarmuka yang biasa lazim digunakan untuk *hard drive*, CDROM *drive*, dan lain-lain

Asropudin (2013:47), *Integrated Development Environment* (IDE) adalah sebuah tipe *hardware interface* yang berfungsi untuk menghubungkan *hard disk*, CD-ROM, dan *drive tape* pada sebuah PC.

2.4.3.1 Tampilan *Visual Basic .Net*

Berikut tampilan dari *Visual Baic .Net*:



Gambar 2.6 Tampilan Utama *Visual Basic .Net*

(*Sumber: Supardi, 2015:5*)

2.4.3.2 Komponen-Komponen *Visual Basic .Net*

a. Jendela Menu Utama *(Main Menu)*

Menu Utama merupakan perintah lengkap Visual Basic 2012, yang berupa menu pulldown, seperti terlihat pada gambar di bawah ini.

Gambar 2.7 Jendela Menu Utama

(*Sumber: Supardi, 2015:16*)

b. Jendela *ToolBar*

Jendela *ToolBar* merupakan perintah yang sering dipakai, ditampilkan dengan gambar atau ikon tertentu, sehingga pemakai dengan mudah memakainya, seperti terlihat pada gambar dibawah ini.

Gambar 2.8 Jendela *ToolBar*

(*Sumber: Supardi, 2015:17*)

c. Jendela *ToobBox*

Jendela *ToolBox* merupakan jendela tempat objek-objek atau komponen-komponen pembentuk program, seperti terlihat pada gambar dibawah ini.

Gambar 2.9 Jendela *ToolBox*

(*Sumber: Supardi, 2015:17*)

d. Jendela Properties

Jendela Properties merupakan jendela tempat mengatur karakteristik, kejadian (*Event*) dari objek atau komponen yang diletakkan pada *Form Design.*

Gambar 2.10 Jendela Properties

(*Sumber: Supardi, 2015:17*)

e. Jendela *Form Design*

Jendela *Form Design* merupakan tempat membuat program (tempat meletakkan komponen-komponen program), seperti dibawah ini.

Gambar 2.11 Jendela *Form Design*

(*Sumber: Supardi, 2015:18*)

f. Jendela Kode Editor

Jendela Kode Editor merupakan jendela tempat mengetik *script* atau program Visual Basic .Net, seperti dibawah ini.

Gambar 2.12 Jendela Kode Editor

(*Sumber: Supardi, 2015:18*)

g. Jendela *Solution Explorer*

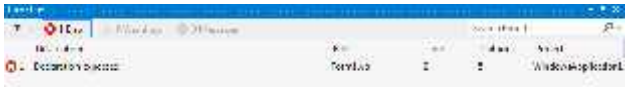
Jendela *Solution Explorer* merupakan jendela file-file yang berhubungan dengan projek yang dibuat. Spserti di bawah ini.

Gambar 2.13 Jendela *Solution Explorer*

(*Sumber: Supardi, 2015:18*)

h. Jendela Informasi Kesalahan

Jendela Informasi Kesalahan merupakan jendela yang memberi informasi kesalahan program pada waktu di-*build* (dikompilasi), seperti gambar di bawah ini.



Gambar 2.14 Jendela Informasi Kesalahan

(*Sumber: Supardi, 2015:18*)

2.4.3 *SQL Server*

2.4.3.1 Pengertian *SQL Server*

Sujatmiko (2012:258), *SQL Server* adalah sebuah sistem manajemen basis data *relational* (RDBMS) produk *Microsoft*.

Wahana Komputer (2013:2), *SQL Server* adalah *software* RDBMS kelas *enterprise* yang cukup banyak digunakan oleh dunia korporat.

2.4.3.2 *Database SQL Server*

2.4.3.3 Tipe Data dalam *SQL Server*

Wahana Komputer (2013:12), *SQL Server* adalah *software database RDBMS* yang berjenis *clinet server*.

Tabel 2.10 Tipe Data pada *SQL Server 2012*

Tipe Data	Min	Max	Storage
Bigint	-9,22337E+18	2^63-1	8 bytes
Int	-2,147,483,648	2,147,483,647	4 bytes
Smallint	-32,768	32,767	2 bytes
Tinyint	0	255	1 bytes
Bit	0	1	1 byte tiap 8 bit
Decimal	1E+38	10^38-1	Tergantung kepresisian, tiap range 5 byte
Numeric	No		
Money	-922337E+14	2^63-1/10000	8 bytes
Smallmoney	-214,748.3648	214,748.3647	4 bytes
Float	-1.79E+308	1.79E+308	Tergantung kepresisian, tiap range 4 byte

Lanjutan Tabel 2.10 Tipe Data pada *SQL Server 2012*

Tipe Data	Min	Max	Storage
Real	-3.40E+38	3.40E+38	4 bytes
Datetime	1753-01-01 00:00:00:000	9999-12-31 13:59:59:997	8 bytes
Smalldatetime	01/01/1900 0:00	06/06/2079 23:59	
Date	0001-01-01	31/12/9999	
Time	00:00:00.0000000	23:59:59.9999999	
Datetime2	0001-01-01 00:00:00.0000000	9999-12-31 23:59:59.9999999	Precision 1-2=6 bytes precision 3-4=7 bytes precision 5-7=8 bytes
Datetimeoffset	0001-01-01 00:00:00.0000000-14:00	9999-12-31 23:59:59.9999999 +14:00	Precision 1-2=8 bytes precision 3-4=9 bytes precision 5-7=10 bytes
Char	0 chars	8000 chars	Defined width
Varchar	0 char	8000 chars	2 bytes + number of chars
Varchar (max)	0 char	2^31 chars	2 bytes + number of chars
Text	0 chars	2,147,483,647 chars	4 bytes + number of chars
Nchar	0 chars	4000 chars	Defined width x 2
Nvarchar	0 chars	4000 chars	
Nvarchar(max)	0 chars	2^30 chars	
Ntext	0 chars	1,073,741,823 chars	
Binary	0 bytes	8000 bytes	

Lanjutan Tabel 2.10 Tipe Data pada *SQL Server 2012*

Tipe Data	Min	Max	Storage
Varbinary	0 bytes	8000 bytes	
Varbinary(max)	0 bytes	2^31 chars	
Image	0 bytes	2,147,483,647 bytes	
SQL_variant			
Timestamp			
Uniqueidentifier			
Xml			
Cursor			
Table			

(*Sumber: Wahana Komputer, 2013:62*)

2.4.3.4 Penggunaan SQL pada *SQL Server*

Darmayuda (2014:97), SQL (*Structure Query Language*) sering disebut dengan *query* merupakan pengembangan SQL sebagai *database query* dan bahasa pemrograman yang digunakan untuk mengakses *database*, melakukan *query*, *update database*, serta mengelola hubungan *system database.*

Wahana Komputer (2013:26), SQL adalah sebuah bahasa yang didesain khusus untuk manajemen data pada sebuah *database relational* atau lengkapnya *relational database management system (RDBMS)*.

Pada *Structure Query Language* (SQL) memiliki tiga jenis, yaitu:

1. *Database Definition Language* (DDL)

Database Definition Language digunakan untuk membangun objek seperti *databases*, *tables*, dan *views*.

Tabel 2.11 Perintah *Database Definition Language* (DDL)

Tipe	Perintah	Keterangan
DDL	CREATE	Digunakan untuk membuat *database, table,* dan *index*
	DROP	Digunakan untuk menghapus *database, table,* dan *index*

Lanjutan Tabel 2.11 Perintah *Database Definition Language* (DDL)

Tipe	Perintah	Keterangan
	ALTER	Digunakan untuk memodifikasi struktur table

(*Sumber: Darmayuda, 2014:98*)

2. *Data Manipulation Languange* (DML)

Data Manipulation Languange berfungsi untuk mengelola atau memanipulasi objek *database*.

Tabel 2.12 Perintah *Data Manipulation Languange* (DML)

Tipe	Perintah	Keterangan
DML	SELECT	Berfungsi memilih/menyeleksi menampilkan data yang diambil dari suatu *table*, dan dapat menggunakan klausa seperti (*)
	INSERT	Berfungsi untuk manipulasi data untuk menambah data (baris) baru pada *table* atau *view*
	UPDATE	Berfungsi untuk mengubah isi data pada suatu *table*. Perintah ini juga dapat menggunakan kondisi tertentu, atau dengan menggunakan klausa seperti (*)
	DELETE	Berfungsi untuk menghapus data per baris berdasarkan *criteria*/kondisi tertentu, yaitu dengan menggunakan klausa seperti (*)

(*Sumber: Darmayuda, 2014:103*)

3. *Data Control Language* (DCL)

Data Control Language berfungsi untuk mengontrol hak-hak pada objek *database.*

Tabel 2.13 Perintah *Data Control Language* (DCL)

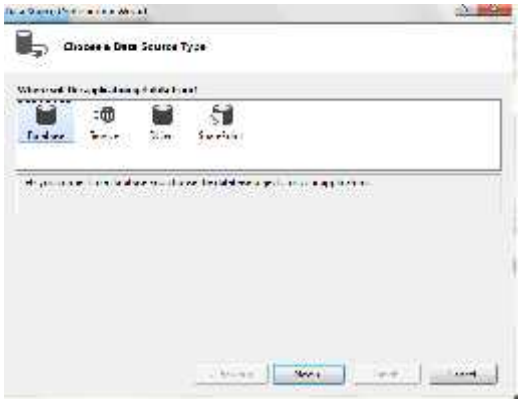
Tipe	Perintah	Keterangan
DCL	GRANT	Berfungsi untuk memberikan hak kepada user untuk mengakses sebuah database
	DENY	Berfungsi untuk membuat sebuah *entry* dalam *system* sekuritas yang melarang sebuah izin pada sebuah account melalui *group* atau keanggotaan *role*
	REVOKE	Berfungsi untuk membebaskan hak yang telah diberikan (seperti pada perintah GRANT) atau hak yang telah dilarang (seperti pada perintah DENY)

(*Sumber: Darmayuda, 2014:119*)

2.4.3.5 Cara Mengkoneksikan *SQL Server dengan Visual Basic .Net*

Mengkoneksikan *SQL Server* dengan *Visual Basic .Net* dapat dilakukan cara sebagai berikut:

1. Pilih menu Project ›› Add New Datasource, pilih *database* kemudian next.



Gambar 2.15 *Data Source Configuration Wizard,* untuk Pemilihan *Data Source Type*

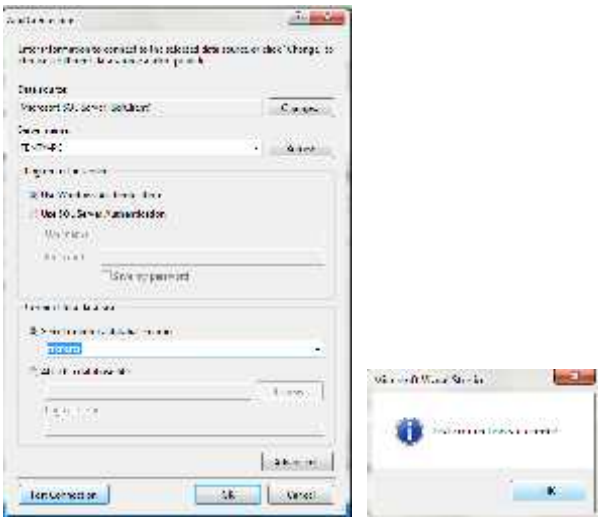
(*Sumber: Darmayuda, 2014:142*)

2. Selanjutnya pilih tab *New Connection*.

Gambar 2.16 *Data Source Configuration Wizard,* untuk Pemilihan *Data Connection*

(*Sumber: Darmayuda, 2014:143*)

Akan tampil dialog *Add Connection* dan lengkapi beberapa isian sebagai berikut, lalu *Test Connection* dan OK.



Gambar 2.17 *Add Connection* dan *Test Connection Succeeded*

(*Sumber: Darmayuda, 2014:144*)

3. Selanjutnya pilih tab *Change* untuk memilih *Data Source* dari *database* yang digunakan kemudian pilih tab OK.

Gambar 2.18 *Data Source Configuration Wizard*, untuk Pemilihan *Data Provider*

(*Sumber: Darmayuda, 2014:146*)

4. Isikan nama *server* pada *combo*, *Server Name*, kemudian
5. Pada *Log on to the server*, pilih salah satu metode koneksi ke *database*.
6. Pilih *Combo* pada *Select or enter a database name*, untuk memilih *database.*
7. Pilih tab *Test Connection* untuk mengetahui koneksi berhasil atau gagal.
8. Selanjutnya pilih *Next*, akan tampil dialog layar sebagai berikut.

Gambar 2.19 *Data Source Wizard*, untuk Menyimpan *Connection*

(*Sumber: Darmayuda, 2014:147*)

9. Selanjutnya pilih *Next*, akan tampil dialog layar untuk menentukan *table, view, stored procedure*, dan *fungctions* sebagai berikut.

Gambar 2.20 *Data Source Configuration Wizard,* untuk Pemilihan *Table*

(*Sumber: Darmayuda, 2014:148*)

10. Selanjutnya pilih tab *Finish* untuk mengakhiri pembuatan koneksi *database.*

11. Pilih Menu Project ›› *Add Class* lalu tambahkan *listing* dibawah ini.

```
Public Class connect
    Private Koneksi As New OleDb.OleDbConnection
    Public SQL As New OleDb.OleDbCommand
    Public DA As New OleDb.OleDbDataAdapter
    Public DS As New DataSet

    Public Sub BukaDatabase() 'fungsi menghubungkan database
        Koneksi.ConnectionString =
        "Data Source=FENTY-PC;Initial Catalog=rskmata;Integrated
Security=True"
        'membuka database
        Koneksi.Open()
        'koneksi database
        SQL.Connection = Koneksi
    End Sub

    Public Sub Tutupdatabase()
        Koneksi.Close()
        Koneksi.ConnectionString = Nothing
    End Sub
End Class
```