

**TESIS**

**IMPLEMENTASI GOOGLE API TRANSLATE PADA  
APLIKASI CHAT MULTI BAHASA**

Oleh:

**SURYADI HOZENG  
(P2700209005)**



**PROGRAM PASCA SARJANA  
JURUSAN TEKNIK ELEKTRO/INFORMATIKA  
UNIVERSITAS HASANUDDIN  
MAKASSAR  
2012**

TESIS

IMPLEMENTASI GOOGLE API TRANSLATE PADA  
APLIKASI CHAT MULTI BAHASA

Disusun dan diajukan oleh

SURYADI HOZENG

Nomor Pokok P2700209005

Menyetujui,  
Komisi Penasihat,

Dr. Ir. Zulfajri B Hasanuddin, M.Eng.  
Ketua

Dr. Armin Lawi, S.Si., M. Eng.  
Anggota

Mengetahui,  
Ketua Program Studi  
Teknik Elektro

Prof. Dr. Ir. Salama Manjang, M.T.

## PRAKATA

Puji syukur penulis panjatkan kehadiran Tuhan Yang Maha Esa, atas segala rahmat dan hidayah-Nya sehingga penulisan tesis yang berjudul “Implementasi Google Api Translate pada Aplikasi Chat Multi Bahasa” dapat terselesaikan.

Ucapan terima kasih dan penghargaan setinggi – tingginya penulis haturkan dengan penuh rasa hormat kepada :

1. Dr. Ir. Zulfajri B Hasanuddin, M.Eng selaku Ketua komisi penasehat dan Dr. Armin Lawi, S.Si., M. Eng selaku anggota komisi penasehat atas bimbingan dan arahan yang telah diberikan mulai dari tahap persiapan hingga selesainya penulisan tesis ini.
2. Prof. Dr. Ir. H. Salama Manjang, MT, Dr. Ir. Rhiza S.Sadjad, MSEE, Dr. Ir. Zahir Zainuddin, MSc selaku tim penguji yang telah meluangkan waktu untuk memberikan masukan dan saran dalam rangka penyempurnaan tesis ini.
3. Kedua Orang Tua Tercinta yang terus mencurahkan cinta kasih, semangat baik materil maupun moril kepada Penulis.
4. Para Guru yang telah membimbing dan memberikan pengetahuan kepada Penulis.
5. Mahasiswa Program Pascasarjana Unhas Teknik Elektro 2009 terkhusus kepada saudaraku di Teknik Informatika 2009 yang selalu memberi keceriaan kepada penulis.

6. Drs. Suarga, M.Math, M.Sc, P.hd dan Rekan kerja di STMIK Dipanegara Makassar yang senantiasa berbagi pengalaman dan memberikan motivasi bagi penulis.
7. Seluruh orang yang telah berjasa kepada penulis yang tidak dapat penulis sebutkan satu persatu.

Penulis menyadari bahwa tesis ini masih jauh dari kesempurnaan tapi semuanya telah penulis lakukan dengan sebaik-baiknya, oleh Karena itu, Penulis membuka diri terhadap kritik dan saran demi kesempurnaan tesis ini dan demi kemajuan ilmu pengetahuan nantinya.

Akhir kata, semoga tesis ini dapat bermanfaat bagi kita semua terutama bagi diri penulis sendiri. Amien.

Makassar, Agustus 2012

Penulis

Suryadi Hozeng

## ABSTRAK

**SURYADI HOZENG.** *Implementasi Google Api Translate pada Aplikasi Chating Multi Bahasa* (dibimbing oleh Dr. Ir. Zulfajri B Hasanuddin, M.Eng dan Dr. Armin Lawi, S.Si., M. Eng).

Penelitian ini bertujuan untuk mengimplementasikan Google API Translate ke dalam aplikasi chat sehingga apabila user melakukan percakapan text tidak mengalami kesulitan bahasa apabila menggunakan bahasa yang berbeda. Selanjutnya melakukan uji coba sistem yang telah dibangun dengan melakukan pengujian sistem dengan melakukan percakapan antar user menggunakan bahasa yang berbeda.

Dalam penelitian ini digunakan sistem pengujian *whitebox* dan pengujian *blackbox*. Pengujian dilakukan dengan memasukkan parameter-parameter pada sisi client kemudian server mengolah data yang dikirim oleh client dengan menerjemahkan pada sisi server kemudian mengirimnya kembali ke sisi client. Penelitian ini dilakukan di Laboratorium Teknik Informatika Universitas Hasanuddin. Pengambilan data dilakukan dari tanggal Januari 2011 sampai Juli 2012 sehingga diperoleh hasil dari penelitian .

Hasil penelitian menunjukkan bahwa percakapan berupa teks antar user berjalan sebagaimana aplikasi di buat serta berhasilnya mengimplementasikan Google API Translate ke dalam aplikasi chat yang di bangun sehingga percakapan berupa teks dapat di terjemahkan langsung sesuai dengan kebutuhan bahasa yang di gunakan oleh user.

Kata kunci :

*Google API Translate, Chat, Multi Bahasa.*

## ABSTRACT

SURYADI HOZENG. Implementation of Google Translate API on the Application Chating Multi Language (led by Dr. Ir. Zulfajri B Hasanuddin, M. Eng and Dr. Armin Lawi, S.Si., M. Eng.)

This study aims to implement the Google Translate API into the application so that when the user chat text conversations do not have language difficulties when using different languages. Subsequently tested system that has been constructed by testing the system with a conversation between users using different languages.

This study used the system whitebox testing and blackbox testing. Testing is done by entering parameters on the client side and server to process the data sent by the client to translate on the server side and then send it back to the client side. The research was conducted at the Laboratory of Information Engineering University of Hasanuddin. Data is collected from the date of January 2011 until July 2012 in order to obtain the results of the study.

The results showed that text-based conversations between the user runs as an application is made and the success of implementing the Google Translate API into a chat application that is built so that the conversation in the form of text can be translated directly in line with the needs of the language used by the user.

Key Words :

*Google Translate API, Chat, Multi Language.*

## DAFTAR ISI

	Halaman
SAMPUL .....	i
LEMBAR PENGESAHAN .....	ii
PERNYATAAN KEASLIAN TESIS .....	iii
KATA PENGANTAR .....	iv
ABSTRAK .....	vi
DAFTAR ISI .....	viii
DAFTAR TABEL .....	x
DAFTAR GAMBAR.....	xi
BAB I. PENDAHULUAN	1
A. Latar Belakang	1
B. Rumusan Masalah	2
C. Tujuan Penelitian	2
D. Manfaat Penelitian	3
E. Batasan Penelitian	3
BAB II. TINJAUAN PUSTAKA	4
A. PHP	5
B. Kriptografi	7
C. Teknik AJAX	7

D. Unified Modeling Language (UML)	9
E. Uses Case Diagram	10
F. Activity Diagram	12
G. Deployment Diagram	12
H. Black Box Testing	15
I. Kerangka Konseptual	17
BAB III. METODE PENELITIAN	18
A. Jenis Penelitian	18
B. Lokasi dan Waktu Penelitian	18
C. Perancangan Sistem	18
D. Bahan dan Alat Digunakan	19
E. Analisis Sistem	20
F. Teknik Pengumpulan Data	21
G. Teknik Pengujian	21
H. Jadwal Pelaksanaan Kegiatan	22
BAB IV Hasil dan Pembahasan	28
A. Hasil Penelitian	28
B. Pengujian Aplikasi	45
C. Hasil Pengujian Aplikasi	45
D. Simulasi Sistem	62
E. Pembahasan	67
BAB V Kesimpulan dan Saran	68
A. Kesimpulan	68
B. Saran	68
DAFTAR PUSTAKA	



## DAFTAR TABEL

No	Nama Tabel	Halaman
2.1	Simbol-Simbol <i>Use Case Diagram</i>	11
2.2	Simbol-Simbol <i>Class Diagram</i>	13
2.3	Simbol-Simbol <i>Sequence Diagram</i>	15
2.4	Simbol-Simbol <i>Activity Diagram</i>	17
2.5	Simbol <i>White Box</i>	18
4.1	Hasil Pengujian <i>White Box</i>	55
4.2	Rencana Pengujian	56
4.3	Hasil Pengujian Dari Rencana Pengujian	56

## DAFTAR GAMBAR

No	Nama Gambar	Halaman
2.1	Struktur Kerja Aplikasi Chat Multi Bahasa	6
2.2	Sebuah Kelas dari Model UML	7
2.3	Sebuah <i>Interface</i> antar-muka	7
2.4	Collaborations	8
2.5	Use Case	8
2.6	<i>Dependency</i>	9
2.7	<i>Association</i>	9
2.8	<i>Generalizations</i>	9
2.9	<i>Realizations</i>	9
2.10	Contoh Aktivitas Aktor dan <i>Use case</i>	11
2.11	Contoh <i>Class Diagram</i>	12
2.12	Contoh <i>Sequence Diagram</i>	14
2.13	Contoh <i>Activity Diagram</i>	16
2.14	Kerangka Konseptual	25
3.1	Diagram <i>Use Case</i> Perancangan Sistem	27
4.1	Arsitektur sistem	31
4.2	Diagram <i>Use Case</i>	32
4.3	<i>Class Diagram</i>	33
4.4	<i>Activity Diagram</i> Aplikasi	34
4.5	<i>Flowchart</i> dan <i>output Login Admin</i>	35

4.6	<i>Flowgraph Form Login Admin</i>	36
4.7	<i>Flowchart dan Output Form Registrasi User</i>	37
4.8	<i>Flowgraph Form Regitrasi User</i>	37
4.9	<i>Flowchart dan Output Form Chating</i>	39
4.10	<i>Flowgraph Form Chating</i>	39
4.11	<i>Output Form Chating User Language Default</i>	45
4.12	<i>Output Form Chating User Language English</i>	46
4.13	<i>Output Form Chating User Language Chinese</i>	46
4.14	<i>Form Chating User Language French</i>	47
4.15	<i>Form Chating User Language Korean</i>	47
4.16	<i>Output Form Index / Login</i>	48
4.17	<i>Output Form Registrasi</i>	48
4.18	<i>Output Form Notification Registrasi</i>	48
4.19	<i>Form Room Chating</i>	49
4.20	<i>Output Form Select Language</i>	49
4.21	<i>Output Form Hasil Chating</i>	50
4.22	<i>Output Form Hasil Chating</i>	50
4.23	<i>Form Translate</i>	51
4.24	<i>Form Translate</i>	51

# **BAB I**

## **PENDAHULUAN**

### **A. LATAR BELAKANG**

Perkembangan Teknologi Informasi telah menunjukkan jati dirinya dalam peradaban manusia dewasa ini. Sudah tentu tidak dapat diingkari dan dipandang sebelah mata, peran perkembangan teknologi informasi telah memberikan share yang signifikan terhadap nilai tambah ekonomi. Efisiensi dalam berbagai bidang, khususnya dalam masalah waktu, tenaga, dan biaya melalui kecepatan dan ketepatan informasi, serta performa fisik telah dapat ditingkatkan dengan sangat drastis, sekaligus berarti telah mampu mengefisienkan penggunaan tempat dalam artian kapasitas ruang.

Dewasa ini di zaman globalisasi, perkembangan teknologi informasi tidak terbatas pada Industri wireline tetapi juga pada Industri nirkabel. Industri nirkabel telah tumbuh secara dramatis dalam 10 tahun terakhir, melebihi jumlah pelanggan wireline sejak tahun 2001.

Dalam dunia IT (Information Technology), para pengguna IT acapkali berkomunikasi melalui media chatting. Dimana dalam komunikasi chat menggunakan bahasa lokal atau bahasa International umum seperti bahasa Inggris. Sehingga para pengguna aplikasi chatting dituntut kemampuan berbahasa asing. Acap kali pengguna IT menemukan

masalah pada kosakata jika sedang berkomunikasi dalam chatting dengan orang dari Negara tidak serumpun maka tidak memungkinkan untuk terus menerus membawa kamus yang tebal dan berat.

Maka dari masalah diatas, maka penulis akan mengangkat dan melakukan penelitian dengan judul **“IMPLEMENTASI GOOGLE API TRANSLATE PADA APLIKASI CHAT MULTI BAHASA”**.

## **B. RUMUSAN MASALAH**

Berdasarkan latar belakang yang telah dipaparkan, maka kami dapat merumuskan permasalahan yaitu: Bagaimana mengimplementasikan Google API Translate dalam rancangan sistem aplikasi chat multi bahasa yang dapat menerjemahkan kata dari satu bahasa ke bahasa lainnya.

## **C. TUJUAN PENELITIAN**

Adapun tujuan yang ingin dicapai dalam penelitian ini adalah membangun perangkat lunak chat multi bahasa sehingga pengguna aplikasi jika sedang melakukan percakapan chat tidak kesulitan bahasa.

## **D. MANFAAT PENELITIAN**

Adapun manfaat yang diharapkan dari penelitian ini adalah sebagai berikut : Pengguna IT tidak kesulitan berbahasa ketika sedang chat dengan pengguna bahasa lain.

## **E. BATASAN PENELITIAN**

Penelitian yang dilakukan meliputi studi literatur, analisis, desain, dan implementasi sistem dengan memperhatikan hal-hal sebagai berikut :

1. Membangun aplikasi chatting.
2. Mengimplementasikan Google Api Translate ke dalam aplikasi chatting tersebut sehingga user dapat berkomunikasi walau menggunakan berbagai bahasa.
3. Aplikasi mampu menerjemahkan minimal 3 jenis bahasa.

## **F. Sistematika Penulisan**

### **Bab I Pendahuluan.**

Bab ini membahas mengenai Latar Belakang, Rumusan Masalah, Tujuan Penelitian, Manfaat Penelitian, Batasan Penelitian, dan Sistematika Penulisan.

### **Bab II Tinjauan Pustaka.**

Bab ini membahas tentang Landasan Teori tentang Sistem, Konsep dasar Google API Translate, Latar belakang Google API Translate, Konsep Dasar Website, Metode perancangan aplikasi, REST, JSON, dan kerangka Konseptual.

### **Bab III Metode Penelitian.**

Bab ini membahas tentang Jenis Penelitian, Lokasi dan Waktu Penelitian, Sumber Data, Tahap Penelitian, Alat dan Bahan Penelitian, Rancangan Sistem, dan Jadwal Penelitian.

### **Bab IV Hasil Penelitian Dan Pembahasan.**

Bab ini membahas tentang , Implementasi system, Perancangan Basis Data, pengujian *white box* dan *black box* sistem.

### **Bab V Simpulan Dan Saran**

## **BAB II**

### **TINJAUAN PUSTAKA**

Tahukah Anda, dari sekian macam aplikasi, Aplikasi chatting merupakan favorit para pengembang perangkat lunak. Selain layanan IM atau chat sendiri yang sudah sedemikian meluas, pertumbuhan pengguna yang kian padat menjadi faktor penentu kepopuleran aplikasi chat tersebut. Banyak variasi aplikasi chat yang tersedia di pasaran. Kabar gembiranya, sebagian aplikasi ini disediakan cuma-cuma alias gratis. Diantaranya ada yang digarap serius dengan konvergensi tambahan fasilitas-fasilitas penting seperti VOIP, mail, hingga video streaming.

Beberapa aplikasi yang lazim di gunakan pengguna chat yaitu Yahoo messenger, AOL, MSN messenger atau GoogleTalk. Akan tetapi aplikasi tersebut tidak menyediakan layanan bahasa sehingga para pengguna harus menguasai bahasa tertentu jika ingin berkomunikasi atau dengan kata lain hanya bisa berkomunikasi dengan orang yang berbahasa yang sama. Dari masalah tersebut maka penulis melakukan penelitian membuat aplikasi chat di mana terdapat dua pengguna aplikasi chat dengan penggunaan bahasa yang berbeda.

Dimisalkan orang Indonesia yang tidak mengerti bahasa Jepang akan berkomunikasi chat dengan orang Jepang yang tidak mengerti bahasa Indonesia setelah menggunakan aplikasi chat yang kami bangun.





**Gambar 2.1** : Struktur Kerja Aplikasi Chat Multi Bahasa

## **A. KONSEP GOOGLE API TRANSLATE**

### **1. Google API Translate**

Dokumen ini ditujukan untuk pengembang yang ingin menulis aplikasi yang dapat berinteraksi dengan Google Translate API. Google Translate adalah sebuah alat yang secara otomatis menerjemahkan teks dari satu bahasa ke bahasa lain (misalnya Prancis ke Inggris). Anda dapat menggunakan Google Translate API untuk pemrograman menerjemahkan teks dalam halaman Web atau aplikasi.

Anda memerlukan account Google untuk menggunakan API ini. Jika Anda sudah memiliki Google Account, maka Anda dapat mengunjungi Google API konsol untuk membuat sebuah proyek baru dan mendapatkan kunci API Anda.

Sebuah antarmuka pemrograman aplikasi (API) adalah sebuah spesifikasi dimaksudkan untuk digunakan sebagai antarmuka dengan komponen software untuk berkomunikasi satu sama lain. API dapat

mencakup spesifikasi untuk rutinitas, struktur data, kelas objek, dan variabel. Sebuah spesifikasi API dapat mengambil banyak bentuk, termasuk Standar POSIX, seperti dokumentasi atau vendor seperti Microsoft Windows API, atau perpustakaan bahasa pemrograman, misalnya Standard Template Library di C++ atau Java API.

API berbeda dari sebuah antarmuka aplikasi biner (ABI) di bahwa mantan kode sumber berdasarkan sedangkan yang kedua adalah antarmuka biner. Misalnya POSIX adalah sebuah API, sedangkan Standard Base Linux adalah ABI [1].

Bahasa yang digunakan API dapat berupa tergantung pada bahasa, yang berarti hanya tersedia dengan menggunakan sintaks dan unsur-unsur bahasa tertentu, yang membuat API lebih nyaman digunakan bahasa-independen, yang ditulis sehingga dapat dipanggil dari beberapa bahasa pemrograman. Ini adalah fitur yang diinginkan untuk API berorientasi-layanan yang tidak terikat dengan proses tertentu atau sistem dan dapat diberikan sebagai prosedur panggilan jarak jauh atau layanan web. Sebagai contoh, sebuah website yang memungkinkan pengguna untuk meninjau restoran lokal mampu lapisan review mereka di atas peta diambil dari Google Maps, karena Google Maps memiliki API yang memfasilitasi fungsi ini. Google Maps 'API kontrol informasi apa situs pihak ketiga dapat menggunakan dan bagaimana mereka dapat menggunakannya.

API panjang mungkin digunakan untuk merujuk ke antarmuka lengkap, fungsi tunggal, atau bahkan satu set API yang disediakan oleh organisasi. Dengan demikian, ruang lingkup makna biasanya ditentukan oleh konteks penggunaan.

API dapat menggambarkan cara di mana tugas tertentu dilakukan. Dalam bahasa prosedural seperti bahasa C tindakan biasanya dimediasi oleh pemanggilan fungsi. Oleh karena itu API biasanya meliputi keterangan tentang semua fungsi / rutinitas yang disediakan. Misalnya: math.h include file untuk bahasa C mengandung definisi prototipe fungsi fungsi matematika yang tersedia di perpustakaan bahasa C untuk pemrosesan matematis (biasanya disebut libm). File ini menjelaskan cara menggunakan fungsi termasuk dalam perpustakaan diberikan: prototipe fungsi adalah tanda tangan yang menggambarkan jumlah dan jenis parameter yang akan dilewatkan ke fungsi dan jenis nilai kembali. Perilaku fungsi biasanya digambarkan secara lebih rinci dalam format yang dapat dibaca manusia dalam buku-buku dicetak atau dalam format elektronik seperti halaman manual.

Setiap permintaan aplikasi Anda mengirimkan ke Google Translate API harus mengidentifikasi aplikasi Anda ke Google, menggunakan kunci API.

Untuk informasi tentang cara menggunakan kunci API, lihat Mengidentifikasi aplikasi Anda ke Google dalam dokumen REST.

## 2. Latar Belakang API Translate

### a. Konsep Translate

Google Translate adalah sebuah alat yang secara otomatis menerjemahkan teks dari satu bahasa ke bahasa lain.

Teks sumber adalah teks yang akan diterjemahkan. Sumber bahasa adalah bahasa yang sumber teks tertulis masuk Target bahasa adalah bahasa bahwa teks sumber diterjemahkan ke dalam.

### b. Operasi API Translate

Ada tiga metode untuk invoke dalam Google Translate API:

Operation	Description	REST HTTP mapping
<a href="#">translate</a>	Translates source text from source language to target language	GET
<a href="#">languages</a>	List the source and target languages supported by the translate methods	GET
<a href="#">detect</a>	Detect language of source text	GET

### c. Teknik Pemanggilan

Ada beberapa cara untuk memanggil API:

- Menggunakan teknik REST secara langsung
- Menggunakan REST dari JavaScript (tidak ada kode server-side yang diperlukan)

### 3. REST

REST, dalam Google Translate API agak berbeda dari REST tradisional. Alih-alih menyediakan akses ke sumber daya, API menyediakan akses ke layanan. Akibatnya, API menyediakan URI tunggal yang bertindak sebagai titik akhir layanan.

Anda mengakses Google Translate API endpoint layanan menggunakan REST HTTP GET kata kerja, seperti yang dijelaskan dalam operasi API. Anda lulus dalam rincian semua permintaan layanan sebagai parameter permintaan.

#### a. Translate

Format spesifik untuk URL Google API Translate tunggal adalah :

```
https://www.googleapis.com/language/translate/v2?{parameters}
```

dimana parameter parameter apapun untuk diterapkan ke query.

Untuk rincian, lihat di dalam dokumen REST.

Berikut adalah contoh tentang bagaimana ini bekerja dalam API

Translate.

```
https://www.googleapis.com/language/translate/v2?key=INSERT-YOUR-KEY&q=hello%20world&source=en&target=de
```

b. Temukan bahasa yang didukung

Format khusus untuk mengembalikan daftar kode bahasa adalah:

```
https://www.googleapis.com/language/translate/v2/languages?{parameters}
```

dimana {parameter} parameter apapun untuk diterapkan ke query.

Untuk rincian, lihat di dalam dokumen REST.

Berikut adalah contoh bagaimana bahasa metode bekerja dalam API Translate.

```
https://www.googleapis.com/language/translate/v2/languages?key=INSERT-YOUR-KEY&target=zh-TW
```

c. Mendeteksi bahasa sumber

Format khusus untuk mendeteksi bahasa teks adalah:

```
https://www.googleapis.com/language/translate/v2/detect?{parameters}
```

dimana parameter parameter apapun untuk diterapkan ke query.

Untuk rincian, lihat di dalam dokumen REST.

Berikut adalah contoh tentang bagaimana mendeteksi karya metode dalam API Translate.

```
https://www.googleapis.com/language/translate/v2/detect?key=INSERT-YOUR-KEY&q=google+translate+is+fast
```

#### d. REST dari JavaScript

Anda dapat memanggil API Translate menggunakan REST dari JavaScript, menggunakan parameter callback query dan fungsi callback. Namun, harap diperhatikan bahwa kunci API Anda akan dapat dilihat dalam sumber HTML untuk halaman Anda. Secara default kunci dapat digunakan pada situs. Kami sangat menyarankan Anda membatasi penggunaan kunci Anda hanya untuk domain Anda mengelola, untuk mencegah penggunaan di situs yang tidak sah. Anda dapat menentukan domain yang diperbolehkan untuk menggunakan kunci API Anda dengan mengklik tombol Edit Referensi diperbolehkan ... link di bagian Akses API sederhana dari panel akses API di Console API.

Contoh berikut ini menggunakan pendekatan untuk menerjemahkan beberapa teks dan menaruhnya di bawah teks sumber:

```
<html>
  <head>
    <title>Contoh API Translate</title>
  </head>
  <body>
    <div id="sourceText">Hello world</div>
    <div id="translation"></div>
    <script>
      function translateText(response) {
```

```

        document.getElementById("translation").innerHTML += "<br>"
+ response.data.translations[0].translatedText;
    }
</script>
<script>
    var newScript = document.createElement('script');
    newScript.type = 'text/javascript';
    var sourceText =
escape(document.getElementById("sourceText").innerHTML);
    // WARNING: be aware that YOUR-API-KEY inside html is
viewable by all your users.
    // Restrict your key to designated domains or use a proxy to hide
your key
    // to avoid misuse by other party.
    var source =
'https://www.googleapis.com/language/translate/v2?key=YOUR-API-
KEY&source=en&target=de&callback=translateText&q=' +
sourceText;
    newScript.src = source;

    // When we add this script to the head, the request is sent off.

document.getElementsByTagName('head')[0].appendChild(newScript
);
</script>
</body>
</html>

```

#### 4. Fomat Data JSON

JSON (JavaScript Object Notation) adalah format data interchange ringan. Sangat mudah bagi manusia untuk membaca dan menulis. Sangat mudah bagi mesin untuk mengurai dan menghasilkan. Hal ini didasarkan pada subset dari Bahasa Pemrograman JavaScript, Standar ECMA-262



Edisi 3 – Desember 1999. JSON merupakan format teks yang benar-benar bahasa independen tetapi menggunakan konvensi yang akrab bagi programmer keluarga C-bahasa, termasuk C, C + +, C #, Java, JavaScript, Perl, Python, dan banyak lainnya. Properti ini membuat JSON bahasa data interchange ideal.

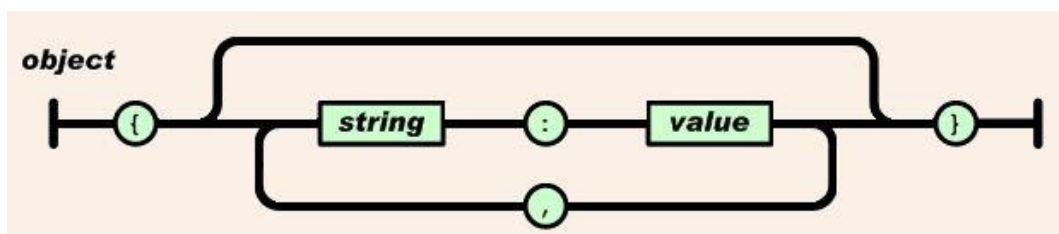
JSON dibangun di atas dua struktur:

- Kumpulan dari pasangan nama / nilai. Dalam berbagai bahasa, ini direalisasikan sebagai catatan, objek, struct, kamus, 25esam hash, daftar kunci, atau array asosiatif.
- Ordered list nilai. Dalam kebanyakan bahasa, ini direalisasikan sebagai, vector, array, daftar atau urutan.

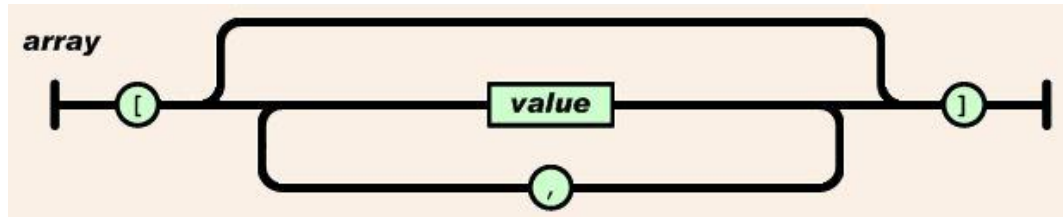
Ini adalah universal data struktur. Hampir semua bahasa pemrograman modern mendukung mereka dalam satu bentuk atau lain. Masuk akal bahwa format data yang dipertukarkan dengan bahasa pemrograman juga didasarkan pada struktur ini.

Dalam JSON, mereka mengambil bentuk-bentuk:

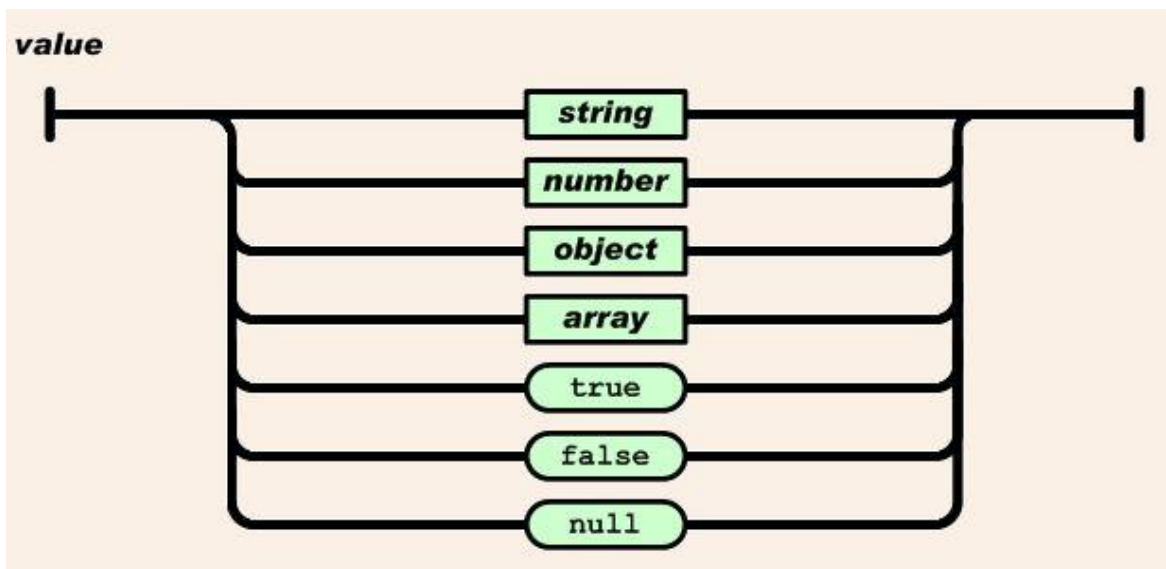
Sebuah objek adalah unordered set pasangan nama / nilai. Sebuah objek dimulai dengan {(kurung kurawal kiri) dan diakhiri dengan} (kurung kurawal kanan). Setiap nama diikuti oleh: (usus besar) dan pasangan nama / nilai dipisahkan dengan, (koma).



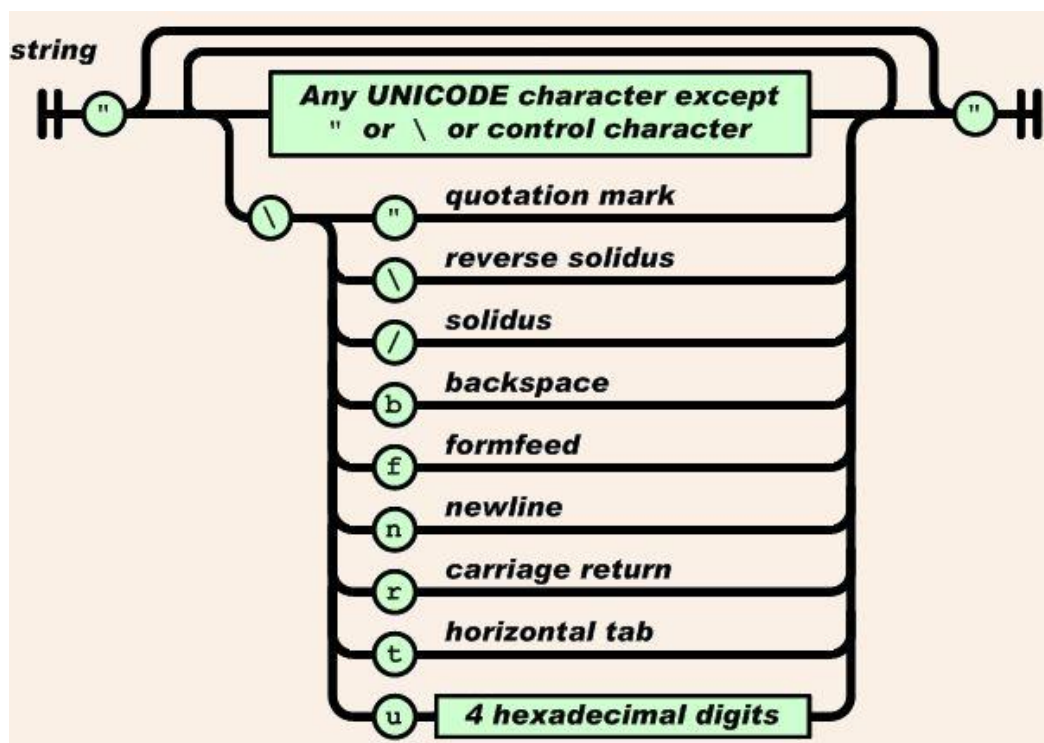
Array adalah susunan nilai yang di pesan. Array dimulai dengan [(braket kiri) dan diakhiri dengan] (kurung siku kanan). Nilai dipisahkan dengan, (koma).



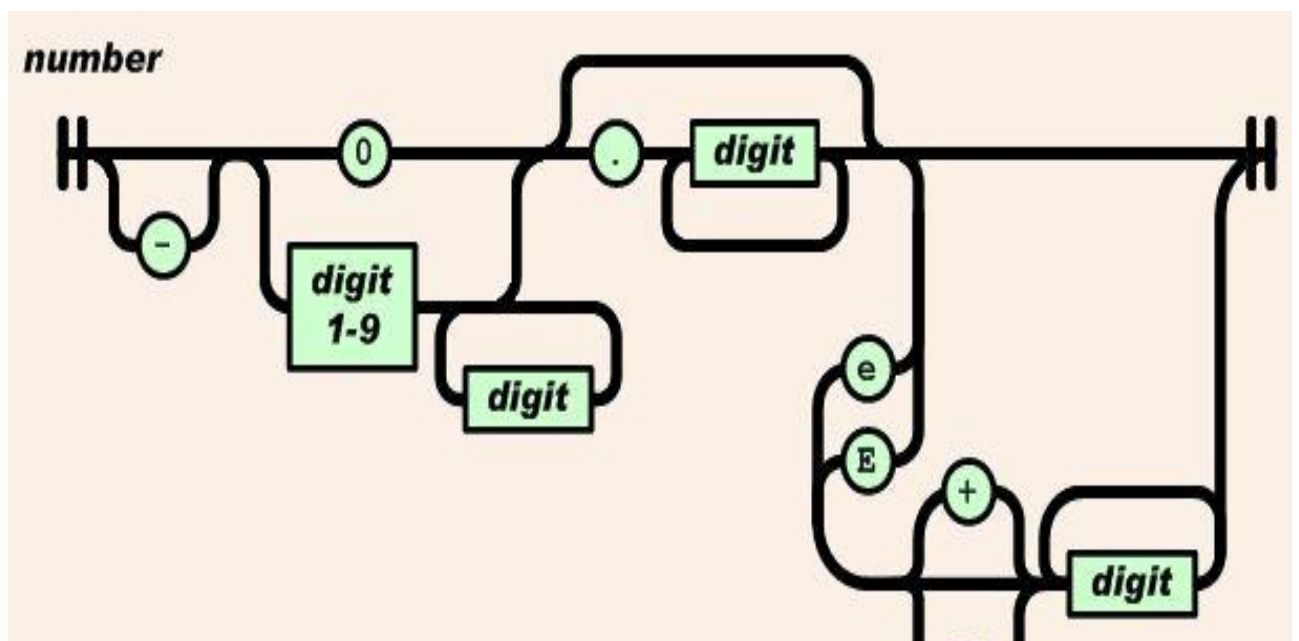
Nilai dapat berupa string dalam tanda kutip ganda, atau nomor, atau benar atau salah atau null, atau objek atau array. Struktur ini dapat dikelompokkan.



String adalah urutan nol atau lebih karakter Unicode, dibungkus dalam tanda kutip ganda, menggunakan lolos backslash. Karakter individu direpresentasikan sebagai string karakter tunggal. String adalah sangat banyak seperti C atau string Java.



Sebuah angka sangat banyak seperti nomor C atau Java, kecuali bahwa 27esam dan heksadesimal format tidak digunakan.



## **B. KONSEP APLIKASI CHATING MULTI BAHASA**

### **1. Konsep Aplikasi Chating**

Chating adalah suatu feature dalam Internet untuk berkomunikasi sesama pemakai Internet yang sedang online (yang sedang sama-sama menggunakan Internet). Komunikasi bisa berupa teks (text chat) atau suara (voice chat). Anda mengirim pesan dengan teks atau suara kepada orang lain yang sedang online, kemudian orang yang dituju membalas pesan Anda dengan teks atau suara, demikian seterusnya...

Agar Anda dapat chatting, maka Anda harus memiliki perangkat lunak chatting terlebih dahulu, misalnya mIRC, ICQ, TurboIRC, Yahoo Messenger, Google Talk dan lain-lain.

### **2. Konsep Aplikasi Chating Multi Bahasa**

Aplikasi chat multi bahasa adalah sebuah aplikasi chating dimana dapat menerjemahkan isi percakapan antar user kedalam bahasa yg diinginkan atau yang digunakan oleh user sehingga memudahkan komunikasi antar user yang berbeda bahasa tanpa perlu menggunakan kamus ataupun bantuan dari perangkat lunak penerjemah lainnya

## **C. KONSEP DASAR PEMBANGUNAN WEBSITE**

## 1) Database Management System

### a. MySQL

MySQL merupakan *software* sistem manajemen basis data (DBMS –*Database Management System*) yang sangat populer di kalangan pemrograman web, terutama di lingkungan Linux dengan menggunakan bahasa script PHP dan Perl. Software DBMS ini juga sudah tersedia untuk platform sistem operasi windows (98/me, nt/xp/2000 DAN Windowa Vista).

MySQL adalah *software* open source, artinya memungkinkan semua orang untuk menggunakan dan memodifikasi *software*.(Adi Nugroho:2011:97).

SQL (*Structured Query Language*) adalah salah satu bahasa generasi level ke-4 (4<sup>th</sup> GL) yang awalnya dikembangkan oleh IBM di *San Jose Research Laboratory*. Berbeda dengan bahasa pemrograman level ke-3 (3<sup>th</sup> GL). SQL adalah bahasa yang bersifat *request oriented* dan bersifat non-prosedural sehingga lebih mudah untuk dipelajari karena sintaks yang digunakan hampir menyerupai bahasa yang digunakan oleh manusia untuk berkomunikasi.

SQL sendiri terbagi atas beberapa bagian, yaitu :

- a. DDL (*Data Daefinition Language*), yaitu bahasa yang mempunyai kemampuan untuk mendefinisikan data yang

berhubungan dengan pembuatan dan penghapusan objek seperti tabel, bahkan basis datanya sendiri.

- b. DML (*Data Manipulation Language*), yaitu bahasa yang berhubungan dengan proses manipulasi data pada tabel, record.
- c. DCL (*Data Control Language*), yaitu bahasa yang berhubungan dengan pengendalian akses ke database.
- d. DTL (*Data Transaction Language*), yaitu bahasa yang berhubungan dengan pengaturan transaksi yang terjadi di dalam database.

## **2) Bahasa Pemrograman**

### **a. HTML**

HTML (*Hyper Text Markup Language*) atau Dokumen HTML adalah sebuah file teks murni. Dokumen HTML diberikan nama sembarang dengan tambahan extension “.htm” atau “.html”.

Bermula dari sebuah bahasa yang sebelumnya banyak digunakan di dunia penerbitan dan percetakan yang disebut dengan SGML (*Standard Generalized Markup Language*), HTML adalah sebuah standar yang digunakan secara luas untuk menampilkan halaman web. (Wahana Computer, 2009:3).

### **b. PHP**

PHP (Personal / Home Page) merupakan bahasa script yang digunakan untuk membuat halaman web yang dinamis. Dinamis berarti halaman yang akan ditampilkan dibuat saat halaman itu diminta oleh client. Mekanisme ini menyebabkan informasi yang diterima client selalu yang terbaru. Semua script PHP dieksekusi pada server dimana script tersebut dijalankan. (MADCOMS, 2011 : 20).

### **3) Editor Pembangun Aplikasi**

#### **a. Macromedia Dreamweaver**

*Macromedia Dreamweaver* adalah Text Editor 32 bit, Editor HTML dan Editor Program berbasis windows yang dapat juga menggantikan aplikasi text editor seperti notepad dengan berbagai dan banyaknya fitur yang disediakan bagi *web page author* dan programmer.

*Macromedia Dreamweaver* dapat digunakan untuk penulisan sintaks HTML, CSS, PHP, ASP Perl, C/C++, Java, Javascript dan VBScript yang difasilitasi dengan *web browser* untuk menampilkan halaman HTML dan perintah-perintah FTP untuk meng-*upload file local* ke FTP Server. (Adi Prasetyo:54).

### **4) Software Server**

#### **a. Appserv**

Sebagaimana dikemukakan oleh perusahaab pendirinya,"AppServ telah menjadi web server terpopuler di internet". Server AppServ bekerja pada hampir semua platform yang terkenal. Selain itu, web server apache selalu menawarkan fitur-fitur bervariasi sehingga memberi sarana bagi para developer untuk menciptakan dan memperluas desain situs web secara cepat. Web server apache menawarkan harga jual terbaik, yaitu dapat diperoleh secara gratis. (Stuart McClure, dkk : 56).

#### **D. METODE PERANCANGAN APLIKASI**

##### **1. Unified Modeling language (UML)**

Menurut Janner Simarmata (2010 : 73), *Unified Modelling Language (UML)* adalah :

*"Sebuah bahasa pemrograman yang telah menjadi standar untuk merancang dan mendokumentasikan sistem perangkat lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem dan sudah digunakan secara luas dan menggunakan notasi yang sudah dikenal untuk analisa dan desain berorientasi objek. Adapun kondisi yang sering terjadi adalah penggabungan beberapa kontribusi ke sebuah metode berorientasi objek".*

Untuk dapat mengetahui UML membutuhkan bentuk konsep dari sebuah bahasa model dan mempelajari 3 elemen utama dari UML seperti *building block*, aturan-aturan menyatakan bagaimana *building block* diletakkan secara bersamaan dan beberapa mekanisme umum.

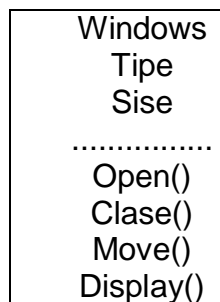
Tiga macam yang terdapat dalam *building block* adalah katagori



benda/*things*, hubungan, dan diagram. Benda/*things* adalah abstraksi yang pertama dalam sebuah model, hubungan sebagai alat komunikasi dari benda-benda, dan diagram sebagai kumpulan/*group* dari benda-benda/*things*.

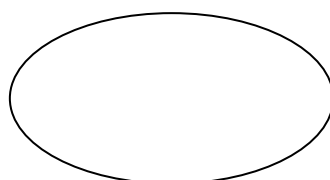
#### a. Benda/*Things*

Adalah hal yang sangat mendasar dalam model UML, juga merupakan bagian paling statik dari sebuah model, serta menjelaskan elemen-elemen lainnya dari sebuah konsep dan atau fisik. Bentuk dari beberapa benda/*thing* adalah sebagai berikut: Pertama, adalah sebuah kelas yang diuraikan sebagai sekelompok dari object yang mempunyai atribut, operasi, hubungan yang semantik.



**Gambar 2.2** Sebuah Kelas dari Model UML

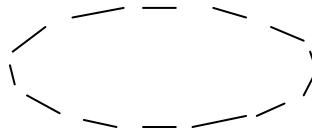
Kedua yang menggambarkan *interface* merupakan sebuah antar muka yang menghubungkan dan melayani antar kelas dan atau elemen. *Interface/antarmuka* mendefinisikan sebuah set/kelompok dari spesifikasi pengoperasian, umumnya digambarkan dengan sebuah lingkaran yang disertai dengan namanya. Sebuah antar-muka berdiri sendiri dan umumnya merupakan pelengkap dari kelas atau komponen.



Ispelling

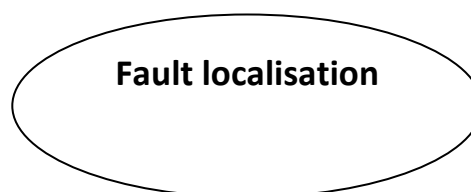
**Gambar 2.3.** Sebuah *Interface* antar-muka

Ketiga adalah *collaboration* yang didefinisikan dengan interaksi dan sebuah kumpulan/kelompok dari kelas-kelas/elemen-elemen yang bekerja secara bersama-sama. *Collaborations* mempunyai struktur dan dimensi. Pemberian sebuah kelas memungkinkan berpartisipasi di dalam beberapa *collaborations* dan digambarkan dengan sebuah *elips* dengan garis terpotong.



**Gambar 2.4.** Collaborations

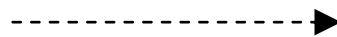
Keempat, sebuah *use case* adalah rangkaian/uraian sekelompok yang saling terkait dan membentuk sistem secara teratur yang dilakukan atau diawasi oleh sebuah aktor. *use case* digunakan untuk membentuk tingkah-laku benda/things dalam sebuah model serta direalisasikan oleh sebuah *collaboration*.



**Gambar 2.5** *Use Case*

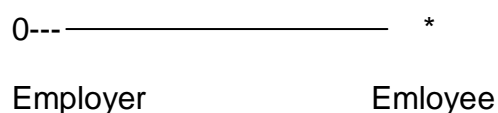
## b. Hubungan / Relationship

Ada 4 macam hubungan didalam penggunaan *UML*, yaitu: *dependency*, *association*, *generalization*, dan *realization*. Pertama, sebuah *dependency* adalah hubungan semantik antara dua benda/*things* yang mana sebuah benda berubah mengakibatkan benda satunya akan berubah pula. Umumnya sebuah *dependency* digambarkan sebuah panah dengan garis terputus-putus.



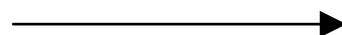
**Gambar 2.6** *Dependency*

Kedua, sebuah *association* adalah hubungan antar benda struktural yang terhubung diantara objek. Kesatuan objek yang terhubung merupakan hubungan khusus, yang menggambarkan sebuah hubungan struktural diantara seluruh atau sebagian.



**Gambar 2.7** *Association*

Ketiga, sebuah *generalization* adalah menggambarkan hubungan khusus dalam objek anak/*child* yang menggantikan objek *parent*/induk. Dalam hal ini, objek anak memberikan pengaruhnya dalam hal struktur dan tingkah lakunya kepada objek induk.



**Gambar 2.8** *Generalizations*

Keempat, sebuah *realization* merupakan hubungan semantik

antara pengelompokan yang menjamin adanya ikatan diantaranya, hubungan ini dapat diwujudkan diantara *interface* dan kelas atau *elements*, serta antara *use cases* dan *collaborations*.

### **Gambar 2.9 Realizations**

#### **c. Diagram**

UML sendiri terdiri atas pengelompokan diagram-diagram sistem menurut aspek atau sudut pandang tertentu. Diagram adalah yang menggambarkan permasalahan maupun solusi dari permasalahan suatu model. UML mempunyai macam-macam diagram, yaitu: *Use Case Diagram*, *Class Diagram*, *Sequence Diagram*, *Activity Diagram*, *Component*, *Deployment Diagram*, *Object Diagram*, *Statechart Diagram*, dan *Communication Diagram*.

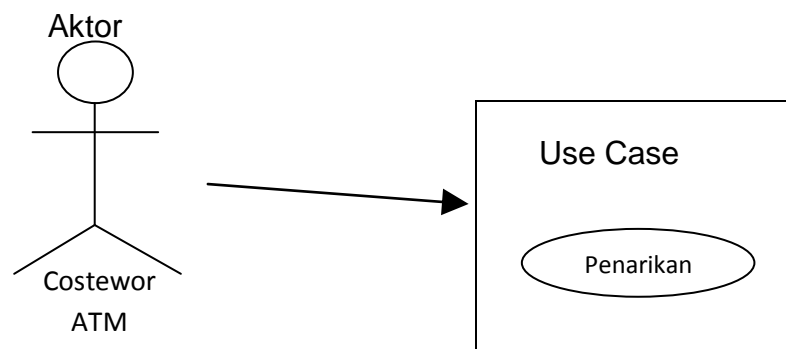
##### **1. Use Case Diagram**

Menurut Rosa A.S dan M. Shalahuddin (2011 : 130), *use case diagram* merupakan pemodelan untuk kelakuan sistem informasi yang akan di buat.

*Use case diagram* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Syarat penamaan

pada *use case* adalah nama didefinisikan sesimpel mungkin dan dipahami. Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case*.

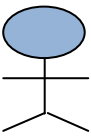
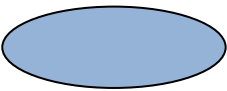


- 1) Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
- 2) *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antarunit atau aktor.



**Gambar 2.10** Contoh Aktivitas Aktor dan *Use case*

Berikut kami gambarkan simbol-simbol yang digunakan dalam perancangan *use case diagram* pada tabel 2.1:

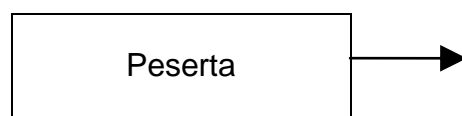
**Tabel 2.1** Simbol-simbol *Use case Diagram*

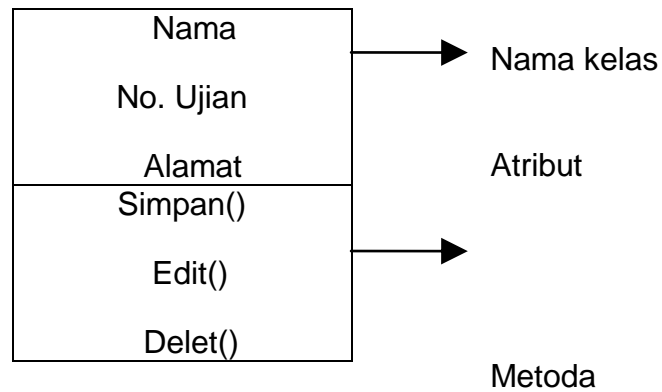
No.	Simbol	Keterangan
1.		<i>Actor</i> , merupakan segala sesuatu yang berinteraksi dengan sistem komputer. Biasa merupakan orang, perangkat keras atau mungkin orang lain
2.		<i>Use case</i> , merupakan penggambaran proses yang dilakukan oleh actor terhadap sistem
3.		<i>Uni directional association</i> , merupakan penggambaran arah aliran proses antara actor dan sistem
4.		<i>Communication</i> , merupakan hubungan actor dengan jenis interaksinya dengan sistem

## 2. Class Diagram

Menurut Rosa A.S dan M. Shalahuddin (2011 : 122), *class diagram* menggambarkan struktur sistem dari segi pendefisian kelas-kelas yang akan dibuat untuk membangun sistem.

Kelas menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi), beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain.

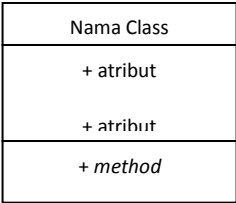



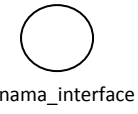





**Gambar 2.11** Contoh *Class Diagram*

Berikut kami gambarkan simbol-simbol yang digunakan dalam perancangan *Class Diagram* pada tabel 2.2:

**Tabel 2.2** Simbol-simbol *Class Diagram*

No	Simbol	Keterangan
1		<p><i>Class</i>, adalah blok-blok pembangun pada pemrograman berorientasi objek. Sebuah <i>class</i> digambarkan sebagai sebuah kotak yang terbagi atas 3 bagian.</p>
2		<p><i>Asosiasi</i>, merupakan sebuah <i>relationship</i> paling umum antara 2 <i>class</i> dan dilambangkan oleh sebuah garis yang menghubungkan antara 2 <i>class</i>.</p>

3		Sama dalam konsep <i>interface</i> dalam pemrograman berorientasi objek
4		<i>Dependency</i> , digunakan untuk menunjukkan operasi pada suatu <i>class</i> yang menggunakan <i>class</i> yang lain.
5		<i>Aggregation</i> , mengindikasikan keseluruhan bagian <i>relationship</i> dan biasanya disebut sebagai relasi.
6		<i>Composition</i> , mengindikasikan <i>class</i> yang saling berhubungan dengan <i>class</i> lain.

### 3. *Sequence Diagram*

Menurut Rosa A.S dan M. Shalahuddin (2011 : 137), *sequence diagram* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antarobjek.

Oleh karena itu untuk menggambar *sequence diagram* maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Diagram ini secara khusus berasosiasi dengan *use case*. *Sequence diagram* memperlihatkan tahap demi tahap apa yang harus terjadi untuk menghasikan suatu didalam *use case*





diagram. Tipe diagram yang digunakan sebaiknya digunakan diawal tahap desain atau analisis karena kesederhanaannya dan mudah untuk di mengerti. Diagram ini adalah yang berinteraksi dengan penekanan pada pengiriman pesan dalam suatu waktu tertentu. Diagram ini bersifat dinamis dan bisa dilihat pada objek-objek dan pesan-pesan. Objek ini berperan aliran diperlihatkan pada kotak persegi panjang yang melintas pada bagian atas diagram.

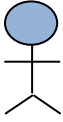


Display

**Gambar 2.12** Contoh *Sequence Diagram*

Berikut kami gambarkan simbol-simbol yang digunakan dalam perancangan *Sequence Diagram* pada tabel 2.3:

**Tabel 2.3** Simbol-simbol *Sequence Diagram*

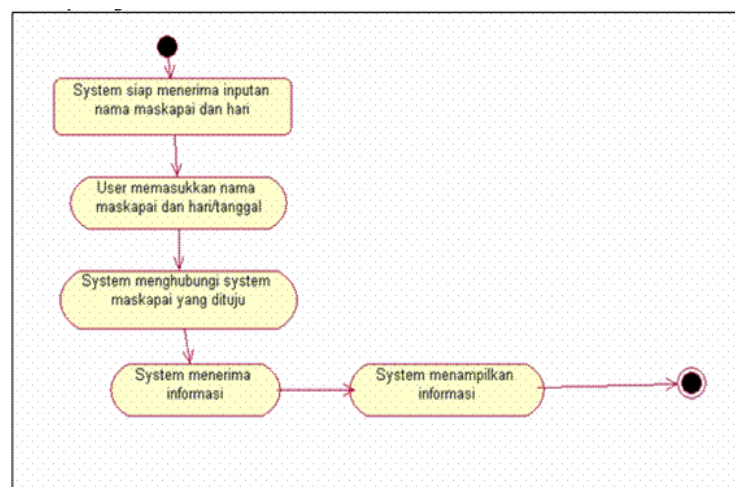
No	Simbol	Keterangan
1		<i>Object line</i> , menggambarkan object yang terlibat dalam system
2		<i>Activation</i> , menggamabarkan kegiatan yang dilakukan oleh masing masing object

3		<i>Actor</i> , merupakan segala sesuatu yang berinteraksi dengan computer
4		<i>Communication</i> , merupakan hubungan actor dengan jenis interaksinya dengan system
5		<i>Lifeline</i> , menyatakan kehidupan suatu objek

#### 4. Activity Diagram

Menurut Rosa A.S dan M. Shalahuddin (2011 : 134), *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis.





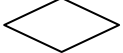
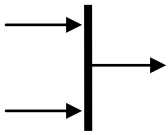
Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem.

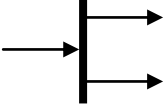


### Gambar 2.13 Contoh Activity Diagram

Berikut kami gambarkan simbol-simbol yang digunakan dalam perancangan *Sequence Diagram* pada tabel 2.3:

**Tabel 2.4** Simbol-simbol *Activity Diagram*

No	Simbol	Keterangan
1		<i>Initial state</i> , menggambarkan awal proses
2		<i>Final state</i> , menggambarkan akhir proses
3		<i>Activity state</i> , menggambarkan kegiatan yang terjadi pada sistem
4		<i>Transition</i> atau <i>link</i>
No	Simbol	Keterangan
5		<i>Decision</i> atau Ketegasan, menggambarkan ketegasan dimana jika ada pilihan aktivitas lebih dari satu
6		<i>Join</i> atau Penggabungan, menggambarkan penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu

7		<i>Fork</i> atau Percabangan, menggambarkan percabangan dimana satu aktivitas dicabangkan menjadi dua atau lebih
8	Swimlane	Sebuah cara untuk mengelompokkan activity berdasarkan aktor (mengelompokkan activity dalam sebuah urutan yang sama)

## E. METODE PENGUJIAN APLIKASI

### 1. Metode Pengujian Aplikasi

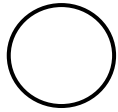

#### 1) *White Box*

Pengujian *white box* adalah metode desain test case yang menggunakan struktur kontrol desain prosedural untuk memperoleh test case, (Janner Simarmata, 2010:30). Dengan menggunakan metode *white box testing* penulis dapat melakukan *test case* yaitu :

- 1) Memberikan jaminan bahwa semua jalur independen pada suatu modul telah digunakan paling tidak satu kali.
- 2) Mengerjakan semua keputusan logis pada sisi *true* dan *false*.
- 3) Mengeksekusi semua perulangan pada batasannya dan pada operasionalnya.
- 4) Menggunakan struktur data internal untuk menjamin validitasnya.

Simbol yang digunakan dalam pengujian white box, seperti pada tabel 2.4 sebagai berikut :

**Tabel 2.5** Simbol *White Box*

No	SIMBOL	KETERANGAN
1		<i>NODE</i> , digunakan untuk mendefenisikan sebuah kegiatan
2		<i>EDGE</i> , digunakan sebagai jalur yang menghubungkan antara kegiatan.

Metode pegujian pada white box dapat menghasilkan *basis path* dan jalur *independen*

- *Basis Path*

*Basis Path* merupakan pengujian *white box* yang diusulkan pertama kali oleh Tom McCabe. Metode ini memungkinkan penguji dapat mengukur kompleksitas logis dari desain prosedural dan menggunakannya sebagai pedoman untuk menetapkan himpunan basis,. (Janner Simarmata, 2010:47).

- Jalur Independen (*Independent Path*)

Jalur Independen adalah jalur yang melalui program yang memperkenalkan sedikitnya satu rangkaian statemen proses baru atau suatu kondisi baru, (Janner Simarmat, 2010:138).

## 2) **Black Box**

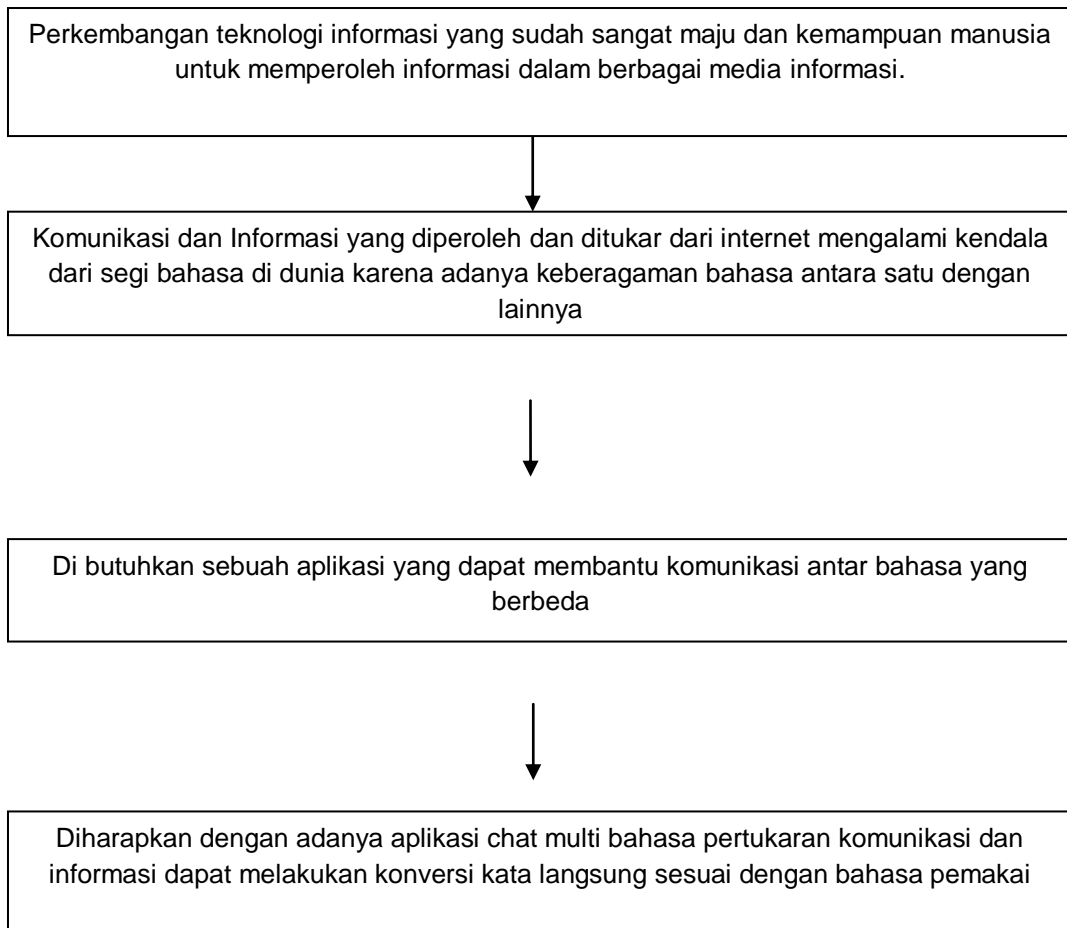
Pengujian *black box* adalah pengujian yang mengabaikan mekanisme internal sistem atau komponen dan fokus semata-mata pada output yang dihasilkan yang merespon input yang dipilih dan kondisi eksekusi, (Janner Simarmata, 2010:30).

Dengan menggunakan metode *black box testing* ukuran yang dapat kita peroleh adalah:

- 1) Digunakan untuk menguji fungsi-fungsi khusus dari perangkat lunak yang dirancang.
- 2) Kebenaran pengujian dilihat dari keluaran yang dihasilkan dari data atau kondisi masukan yang diberikan untuk fungsi yang ada tanpa melihat bagaimana proses untuk mendapatkan keluaran tersebut.
- 3) Dari keluaran yang dihasilkan, kemampuan program dalam memenuhi kebutuhan pemakai dapat diukur sekaligus dapat diketahui kesalahannya.
- 4) Tujuan dari pengujian *black box* adalah menemukan :
- 5) Fungsi yang tidak benar atau hilang
- 6) Kesalahan interface
- 7) Error pada struktur data atau akses database external
- 8) Error pada kinerja
- 9) Error pada saat inisialisasi dan terminasi
- 10) Kesensitifan sistem terhadap nilai input tertentu

11) Batasan dari suatu data

## F. KERANGKA KONSEPTUAL



**Gambar 2.14** Kerangka Konseptual