BAB 2
TINJAUAN PUSTAKA

2.1. Profil Tempat TA

Tinjauan umum instansi tugas akhir pada bab ini penulis menjelaskan secara singkat profil instansi SMA IT ALIA TANGERANG.

2.1.1 Sejarah Instansi

Dengan segala daya dan kemampuan Yayasan Pendidikan ALIA, Insya Allah selalu berupaya agar dapat memenuhi kebutuhan dan harapan masyarakat di bidang pendidikan. Alhamdulillah sambutan dan kepercayaan masyarakat terhadap pendidikan yang kami selenggarakan terus meningkat, dapat dilihat dari grafik pertumbuhan anak didik baik dari segi kuantitas maupun kualitas.

SMA IT ALIA TANGERANG adalah instansi Pendidikan satu satunya yang ada di wilayah kabupaten tangerang didirikan pada tahun 2011 yang insha Allah bermanhaj salaf, sekolahan ini memiliki gelar IT, karena melihat kebanyakan sekolahan negri atau swasta lainnya yang lebih banyak pelajaran umumnya, dan hanya sedikit sekali tentang porsi agamanya, tapi di sekolahan Alia ini tidak hanya ilmu umum saja yang di berikan kepada siswanya, akan tetapi ilmu agama islam yang berdasarkan al-quran dan As sunnah dengan berusaha mengikuti pemahaman shalaffushalih.

Sebagai tanggung jawab penyelenggara pendidikan tingkat dasar dan menengah, kami berusaha menghasilkan lulusan berkualitas yang mampu bersaing di tingkat pendidikan selanjutnya dengan prestasi memuaskan, memiliki aqidah, akhlak mulia serta pemahaman agama yang lurus …Insya Allah… di sekitarnya.

2.1.2 Logo Instansi

Adapun logo yang dimiliki oleh SMA IT ALIA TANGERANG dapat dilihat sebagai berikut:

Sumber Gambar : http://www.aliaislamicschool.sch.id/

Gambar 2.1 Logo Instansi

2.1.3 Visi dan Misi Instansi

Visi nya adalah sebagai berikut:

"UNGGUL DALAM IMAN DAN TAQWA, SERTA BERKARAKTER ISLAMI BERDASARKAN ALQUR'AN DAN ASSUNNAH"

Indikatornya:

1. Suskes Ujian Nasional, Olimpiade (OSN, OOSN) dan seleksi PTN
2. Santun dalam berperilaku, ber-Imtaq , dan berkarakter Islami
3. Berjiwa Kompetitif, trampil, dan kreatif.

Misi nya adalah sebagai berikut :

1. Mendidik Siswa dengan pengetahuan akademik dan cara berfikir yang benar sesuai kebijakan kurikulum yang sistemik dan terpadu.
2. Membekali siswa dengan aqidah dan pemahaman yang benar serta akhlaq yang mulia.
3. Melaksanakan pembelajaran dan bimbingan yang aktif, sehingga peserta didik dapat berkembang secara optimal sesuai dengan potensi yang dimiliki.
4. Menciptakan nuansa belajar yang menyenangkan dan komunikatif.
5. Mengembangkan hubungan yang baik antara keluarga dan Sekolah untuk memaksimalkan proses dan hasil pendidikan

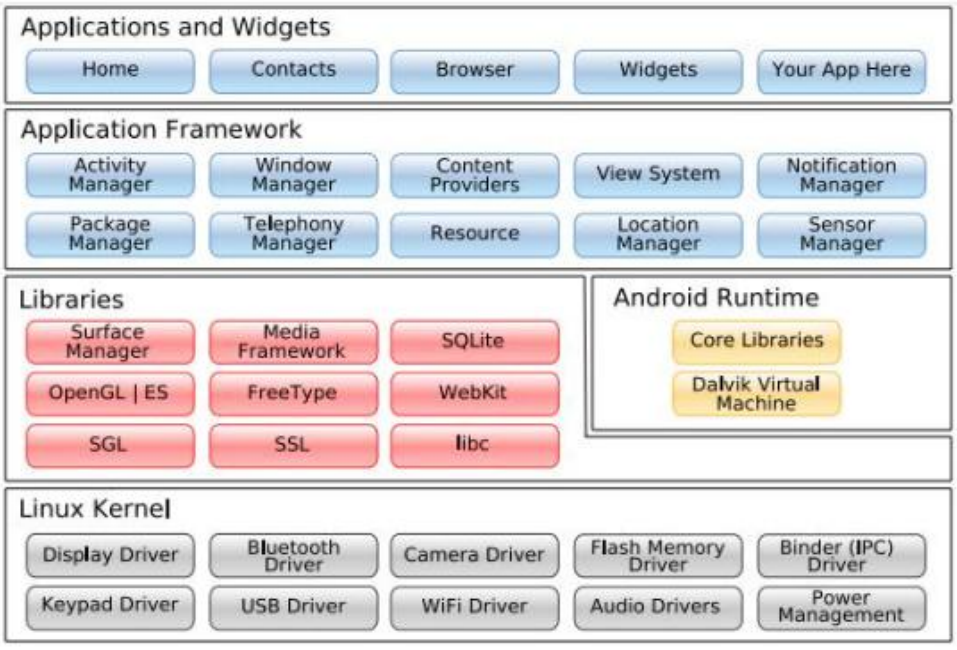
6. Membangun kerjasama produktif antara sekolah, institusi pemerintah dan lingkungan masyarakat.
7. Menjadikan Al Qur`an dan As Sunnah sebagai pedoman dalam beramal dan belajar
8. Menyelenggarakan kegiatan ekstra kurikuler pilihan yang dapat mengembangkan potensi siswa.

2.2 Landasan Teori

Subbab ini berisikan teori - teori pendukung yang digunakan dalam proses analisis dan implementasi pada pembangunan aplikasi.

2.2.1 Arsitektur Android

Arsitektur Android dapat dilihat pada Gambar 2.2. Arsitektur Android memiliki beberapa layer yaitu sebagai berikut :



Sumber Gambar : J. Simarmata **[5]**

Gambar 2.2 Arsitektur Android

1. *Layer Applicatoins and Widgets*

Pada layer Application dan *Widget* inilah berjalan aplikasi-aplikasi ini. Seperti Email, SMS, Kalender, *Browser*, Peta, Kontak dan lain-lain. Bahasa

Java digunakan untuk membuat aplikasi-aplikasi tersebut. Aplikasi yang kita buat akan berada di layer ini.

2. *Layer Applications Framework*

   Application Framework adalah dimana beradanya komponen-komponen yang digunakan para pembuat aplikasi, untuk mengembangkan aplikai mereka. Berikut contoh-conoh komponen yang masuk di dalam *Applications Framework*:

   1. *Views*
   2. *Content Provider*
   3. *Resource Manager*
   4. *Notification Manager*
   5. *Activity Manager*

3. *Layer Libraries*

   Pada *layer Libraries* inilah kita bisa temukan fitur-fitur dari Android. Untuk mengimplentasikan aplikasi biasanya mengakses libraries ini. Libraries ada dua, yaitu libraries media dimana ini memutar video dan audio, dan *libraries* untuk menjalankan tampilan, seperti *libraries graphic*, *libraries SQLite* untuk *support* data base dan masih banyak library lainnya.

4. *Android RunTime*

   Dilayer inilah aplikasi android dapat berjalan. *Android RunTime* dibagi jadi 2 bagian yaitu:

   1. *Core Libraries* : fungsinya untuk mentermahkan bahasa Java dan C.
   2. *Dalvik Virtual Machine* : berfungsi sebagai virtual mesin berbasisi register yang bertugas mengoptimalkan jalannya fungsi-fungsi di Android agar lebih efisien.

5. *Linux Kernel*

   Pada *Linux Kernel* inilah ini dari system operasi *Android* berada. Isinya adalah file-file system yang tugasnya mengelola *system processing*, *memory*, *resource*, *drivers* dan fungsi-fungsi system Android lain. Disini kita dapat melihat adanya kemiripan *file system* pada *Android* dan *system* operasi berbasis *Linux*. *Kernel* disini berbasis monolithic. Pada versi linux yang digunakan versi

2.6, versi 3.x dan pada *Android* versi 4.0 keatas.Mungkin itu kurang lebih mengenai gambaran mengenai arsitektur *Android*. Apa maksud penggunaan kembali object dari elemen-elemen penyusun system operasi, ini maksudnya komponen-komponen yang terdapat pada aplikasi *Android* kita bisa gunakan ketika membutuhkannya [5].

Pengetahuan dan subbab tentang arsitektur android diperlukan karena pada penelitian ini aplikasi yang dibangun berbasiskan platform android. Tentunya dengan adanya pembahasan ini dapat menjadi landasan pemahaman dari apa itu elemen-elemen dari arsitektur android. Pembahasan tentang arsitektur android ini juga akan diperdalam lagi dengan lifecycle atau siklus hidup dari android.

2.2.2 *Life Cycle Android*

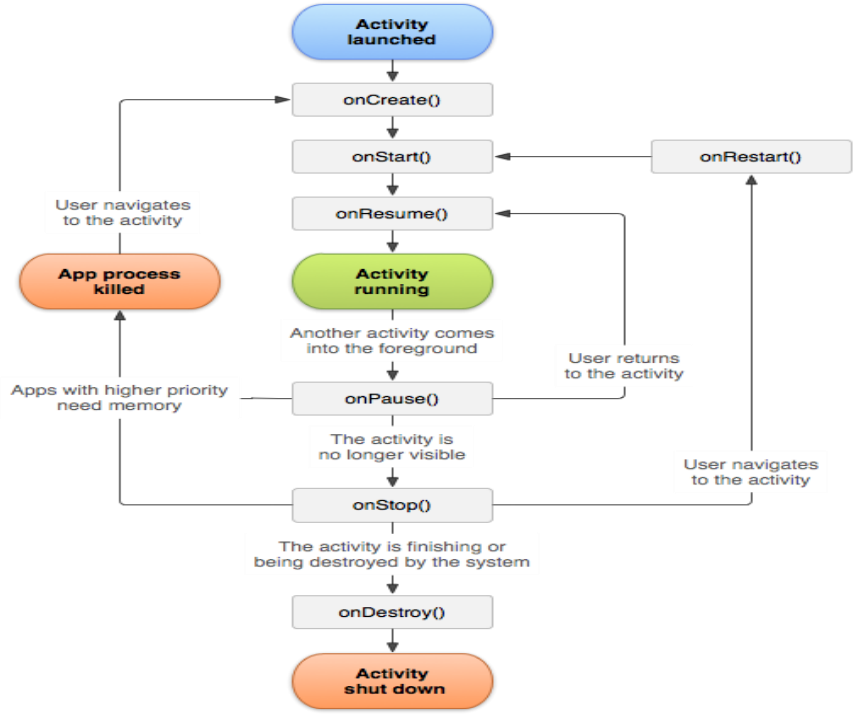
Aplikasi *android* terdiri dari beberapa fungsi dasar seperti mengedit catatan, memutar file musik, membunyikan alarm, atau membuka kontak telepon. Fungsi-fungsi tersebut dapat diklasifikasikan ke dalam empat komponen android yang berbeda. Komponen Aplikasi Android, klasifikasi tersebut berdasarkan kelas-kelas dasar java yang digunakan [6].

Tabel 2.1 Komponen Aplikasi *Android*

Functionality	*Java Base Class*	*Example*
Focused thing a user can do	*Activity*	*Edit a note, play a game*
Background process	*Service*	*Play music, update weather icon*
Receive messages	*BroadcastReceiver*	*Trigger alarm upon event*
Store and retrive data	*ContentProvider*	*Open a phone contact*

Pada bahasa pemrograman c, c++, dan java, program dimulai dengan function main(). Hal ini sangat mirip, pada aplikasi android, program dimulai dengan method callback onCreate(). Urutan method callback dari mulai activiy sampai berakhirnya activity dapat dilihat pada diagram activity lifecycle [6].

Terdapat beberapa *method* dalam *Android Life Cycle* yang dapat dilihat pada Gambar 2.3 Android Life Cycle yaitu:



Gambar 2.3 *Android Life Cycle*

Tabel 2.2 *Method* dalam *Android Life Cycle* [6]

Method	***Description***
onCreate()	*Method ini pertama kali dipanggil ketika activity pertama dimulai.*
onStart()	*Method ini dipanggil ketika activity sudah terlihat pada user.*
onResume()	*Method ini dipanggil ketika activity mulai berinteraksi dengan user.*
onPause()	*Method ini Dipanggil ketika activity berhenti sementara tidak menerima inputan user dan tidak mengeksekusi kode apapun.*
onStop()	*Method ini dipanggil ketika activity sudah tidak terlihat pada user.*
onDestroy()	*Method ini dipanggil sebelum sebuah activity di matikan.*
onRestart()	*Method ini dipanggil setelah activity berhenti dan ditampilkan ulang oleh user.*

Namun, hanya beberapa dari *state* atau *method* tersebut yang menjadi statis diantaranya:

1. Resumed

   Resumed terjadi ketika aplikasi berjalan setelah state paused. State ini akan menjalankan perintah program yang ditulis pada method onResume() [6].

2. Paused

   Dalam keadaan ini aktivitas yang terjadi dihentikan secara sementara tetapi masih terlihat oleh pengguna karena terdapat proses yang memiliki prioritas lebih tinggi seperti panggilan telepon. Aplikasi tidak dapat menjalankan perintah apapun ataupun menampilkan apapun dalan state ini [6].

3. *Stopped*

   Dalam keadaan ini, aplikasi benar-benar tidak ditampilkan dan tidak terlihat oleh pengguna tetapi masih meninggalkan *service* di *background* [6].

State lain seperti *Create and Started* bersifat sementara dan sistem dengan cepat menjalankan state berikutnya dengan memanggil metode *life cycle callback* berikutnya. Artinya, setelah sistem *onCreate()* dipanggil, dengan cepat sistem akan memanggil *method onStart(),* kemudian diikuti oleh *onResume*() [6].

Pengetahuan dan subbab tentang android *life cycle* diperlukan karena dalammembangun aplikasi berbasis android peneliti harus mengetahui siklus hidup dariaplikasi android tersebut. Siklus hidup dari aplikasi android berbeda dengan siklushidup aplikasi desktop karena aplikasi android memiliki tingkat interupsi yangtinggi. Tentunya dengan adanya pembahasan ini dapat menjadi landasanpemahaman dari bagaimana siklus hidup aplikasi android dan apa saja komponennya.

2.2.3 *JavaScript Object Notation* (JSON)

JSON (*JavaScript Object Notation*) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer. Format ini dibuat berdasarkan bagian dari *Bahasa Pemograman JavaScript*, *Standar ECMA-262 Edisi ke-3 - Desember 1999*. JSON merupakan format teks yang tidak bergantung pada bahasa pemprograman apapun

karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, *Java, JavaScript, Perl, Python* dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran-data [7].

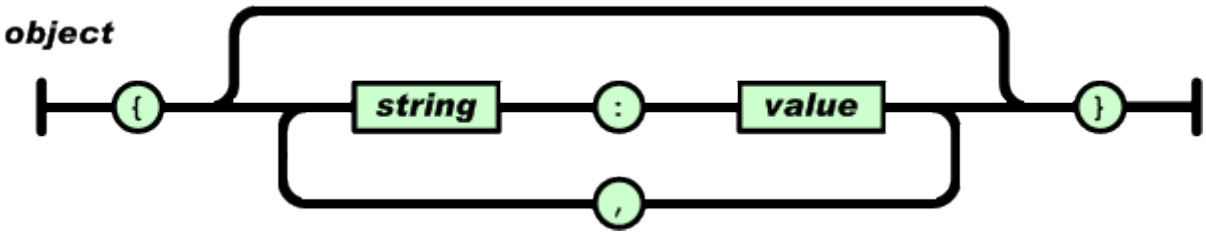
JSON terbuat dari dua struktur:

1. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), struktur (*struct*), kamus (*dictionary*), tabel hash (*hash table*), daftar berkunci (*keyed list*), atau *associative array*.
2. Daftar nilai terurutkan (*an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (*array*), vektor (*vector*), daftar (*list*), atau urutan (*sequence*) [7].

Struktur-struktur data ini disebut sebagai struktur data universal. Pada dasarnya, semua bahasa pemprograman moderen mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemprograman yang juga berdasarkan pada struktur data ini. JSON menggunakan bentuk sebagai berikut:

1. Objek

   Objek adalah sepasang nama/nilai yang tidak terurutkan. Objek dimulai dengan **{** (kurung kurawal buka) dan diakhiri dengan **}** (kurung kurawal tutup). Setiap nama diikuti dengan **:** (titik dua) dan setiap pasangan nama/nilai dipisahkan oleh **,** (koma) [7].

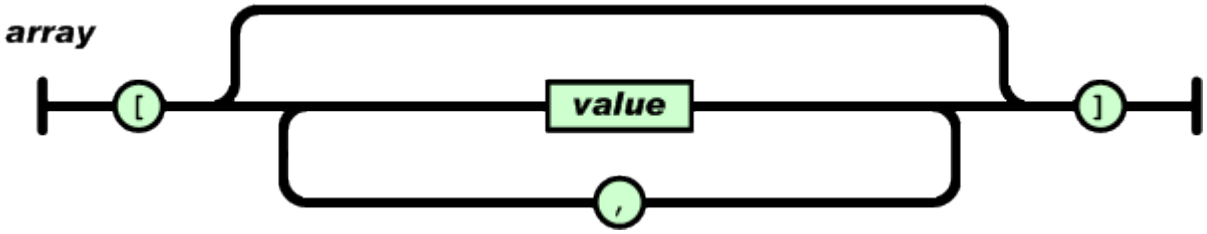


Sumber Gambar : A. Nugroho [7]

Gambar 2.4 Objek JSON

2. Larik

   Larik adalah kumpulan nilai yang terurutkan. Larik dimulai dengan **[** (kurung kotak buka) dan diakhiri dengan **]** (kurung kotak tutup). Setiap nilai dipisahkan oleh **,** (koma) [7].



Sumber Gambar : A. Nugroho [7]

Gambar 2.5 *Array* JSON

Dikarenakan aplikasi pembelajaran untuk chatbot yang menggunakan API dan pertukaran datanya menggunakan JSON maka subbab tentang pembahasan JSON perlu digunakan pada landasan teori ini. JSON berfungsi sebagai format pertukaran data antara aplikasi pembelajaran untuk chatbot dengan API. Tentunya dengan adanya pembahasan JSON ini dapat lebih meningkatkan pemahaman apa itu JSON dan bentuk-bentuk struktur datanya.

2.2.4 Java

Bahasa pemrograman java dibuat di Sun Microsystems dibawah arahan dari Net luminaries James Gosling dan Bill joy. Java di desain untuk menjadi bahasan pemrograman independen yang cukup aman pada jaringan dan cukup kuat untuk mengantikan kode eksekusi asli [8].

Java merupakan bahasa pemrograman yang bersifat umum (general purpose). Karena fungsionalitasnya aplikasi java bisa berjalan di beberapa platform sistem operasi yang berbeda. Java terkenal dengan librarynya yang lengkap serta (kumpulan program program yang disertakan dalam pemrograman java) yang sangat mempermudah digunakan oleh para pemrogram untuk membangun aplikasinya. Kelengkapan library ini ditambah dengan keberadaan banyaknya komunitas java yang besar yang terus menerus membuat library baru untuk memenuhi seluruh kebutuhan pembangunan aplikasi [8].

Pengetahuan dan subbab tentang Java diperlukan karena Bahasa pemrogramman yang digunakan dalam membangun aplikasi pembelajaran untuk chatbot adalah Java. Bahasa pemrogramman dengan menggunakan Java cocok diimplementasikan pada aplikasi yang dibangun diatas *platform* Android. Tentunya dari pembahasan Java ini dapat menjadi landasan pemahaman dari apa itu Java *programming language* dan fungsionalitasnya.

2.2.5 *Object Oriented Analysis And Design* (OOAD)

Object Oriented Analysis And Design atau lebih sering disebut OOAD adalah metode pengembangan sistem yang lebih menekankan objek dibandingkan dengan data atau proses. Ada beberapa ciri khas dari pendekatan ini yaitu *Object, Inhetitance,* dan *Object Class* [7].

1. *Object* adalah struktur yang mengenkapsulasi atribut dan metode yang beroperasi berdasarkan atribut-atribut tadi.
2. *Object Class* adalah sekumpulan objek yang berbagi sktruktur yang sama dan perilaku yang sama.
3. *Inheritance* merupakan properti yang muncul ketika tipe entitas atau kelas objek disusun secara hierarki dan setiap tipe entitas atau kelas objek menerima atau mewarisi atribut dan metode dari pendahulunya [7].

Sebagaimana tercantum pada batasan masalah mengenai desain model aplikasi yang akan digunakan adalah OOAD (Object Oriented Analysis And Design), maka subbab tentang pembahasan OOAD perlu digunakan pada landasan teori ini. Tentunya dengan adanya pembahasan OOAD ini dapat lebih meningkatkan pemahaman definisi dan konsep dasar dasar dari OOAD. Dan dengan adanya pembahasan ini juga pembangunan aplikasi pembelajaran untuk chatbot dapat diterapkan dengan cara *Object Oriented Analysis And Design*.

2.2.5.1 *Unified Modeling Language* (UML)

Unified Modeling Language (UML) adalah keluarga notasi grafis yang didukung oleh model-model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek (OO). UML merupakan standar yang relatif

terbuka yang dikontrol oleh Object Management Group (OMG) *Unified Modeling Language* (UML) adalah keluarga notasi grafis yang didukung oleh model-model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek (OO). UML merupakan standar yang relatif terbuka yang dikontrol oleh Object Management Group (OMG) [9].

Dikarenakan desain aplikasi pembelajaran untuk chatbot yang akan dibangun berorientasi objek *(Object Oriented Analysis And Design)* maka subbab tentang pembahasan UML (*Unified Modeling Language*) perlu digunakan pada landasan teori ini. Tentunya dengan adanya pembahasan UML ini dapat lebih meningkatkan pemahaman definisi dan bentuk-bentuk diagram pemodelan pada UML. Dan dengan adanya pembahasan tentang UML ini dapat menjadi dasar atau landasan dalam membuat perancangan aplikasi pembelajaran untuk chatbot.

2.2.5.1.1*Diagram Use Case*

Use Case Diagram adalah gambaran *graphical* dari beberapa atau semua *actor, use case*, dan interaksi diantaranya yang memperkenalkan suatu sistem. Use case diagram tidak menjelaskan secara detil tentang penggunaan *use case*, tetapi hanya memberi gambaran singkat hubungan antara Use case, aktor, dan sistem.Didalam use case ini akan diketahui fungsi - fungsi apa saja yang berada pada sistem yang dibuat [10].

Element - elemen pada Use Case Diagram :

1. Actor

   Aktor mempresentasikan seseorang atau sesuatu (seperti perangkat,sistem lain) yang berinteraksi dengan sistem. Actor hanya berinteraksi dengan use case tetapi tidak memiliki kontrol atas use case.

2. *Use Case*

   Use Case adalah gambaran fungsionalitas dari suatu sistem, sehingga customer atau pengguna sistem paham dan mengerti mengenai kegunaan sistem yang akan dibangun.

3. *System*

Sistem menyatakan batasan sistem dalam relasi dengan actor-actor yang menggunakannya (di luar sistem) dan fitur-fitur yang harus disediakan (dalam sistem). Digambarkan dengan segi empat yang membatasi semua use case dalam sistem terhadap pihak mana sistem akan berinteraksi. Sistem disertai label yang menyebutkan nama dari sistem, tapi umumnya tidak digambarkan karena tidak terlalu memberi arti tambahan pada diagram.

4. *Association*

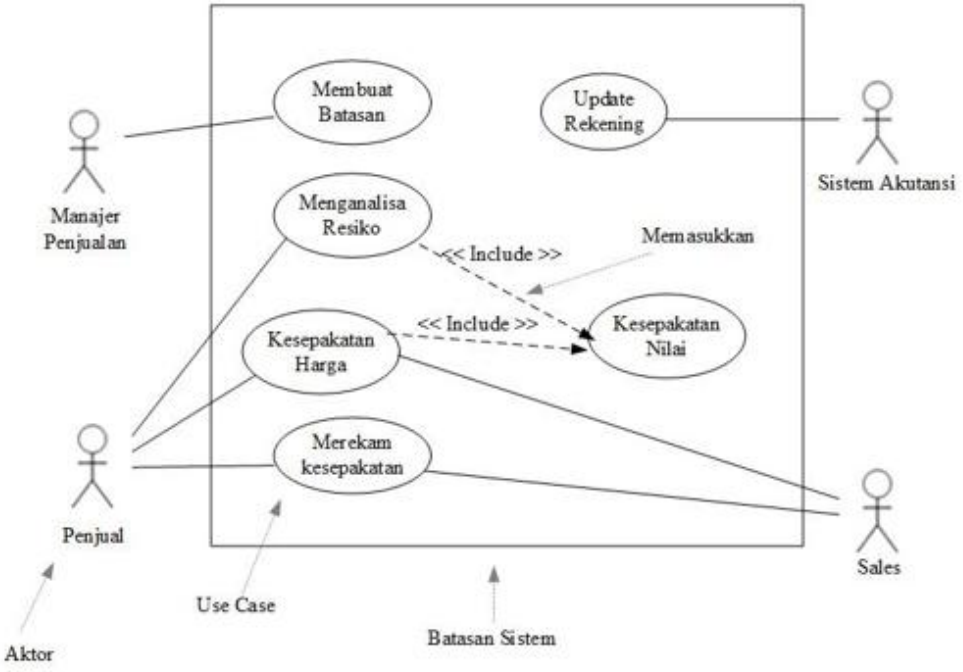
   Mengidentifikasikan interaksi antara setiap actor tertentu dengan setiap use case tertentu. Digambarkan sebagai garis antara actor terhadap use case yang bersangkutan. Asosiasi bisa berarah (garis dengan anak panah) jika komunikasi satu arah, namun umumnya terjadi kedua arah (tanpa anak panah) karena selalu diperlukan demikian, atau dengan kata lain menghubungkan link antar element.

5. *Dependency*

   Include adalah kelakuan yang harus terpenuhi agar sebuah *event* dapat terjadi, dimana pada kondisi ini sebuah *use case* adalah bagian dari *use case* lainnya [10]. E*xtend* relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan dapat berdiri sendiri walau tanpa use case tambahan itu.

6. *Generalization*

   Generalization mendefinisikan relasi antara dua actor atau dua use case yang mana salah satunya meng-inherit dan menambahkan atau override sifat dari yang lainnya. Penggambaran menggunakan garis bermata panah kosong dari yang meng-inherit mengarah ke yang di-inherit.



Sumber Gambar : J. Hermawan [10]

Gambar 2.6 Contoh Diagram Use Case

Use Case merupakan sebuah teknik yang digunakan dalam pengembangan sebuah aplikasi atau sistem informasi untuk menangkap kebutuhan fungsional dari sistem yang bersangkutan, *Use Case* menjelaskan interaksi yang terjadi antara 'aktor'—inisiator dari interaksi sistem itu sendiri dengan sistem yang ada, sebuah Use Case direpresentasikan dengan urutan langkah yang sederhana.

2.2.5.1.2 *Use Case Scenario*

Sebuah diagram yang menunjukkan use case dan aktor mungkin menjadi titik awal yang bagus, tetapi tidak memberikan detail yang cukup untuk desainer sistem untuk benar-benar memahami persis bagaimana sistem dapat terpenuhi. Cara terbaik untuk mengungkapkan informasi penting ini adalah dalam bentuk

penggunaan *use case scenario* berbasis teks per *use casenya*. Berikut adalah dasar format penulisan *use case scenario* [10]. Dasar pembangunan use case scenario dapat dilihat pada Tabel 2.3 Dasar Pembangunan Use Case Scenario.

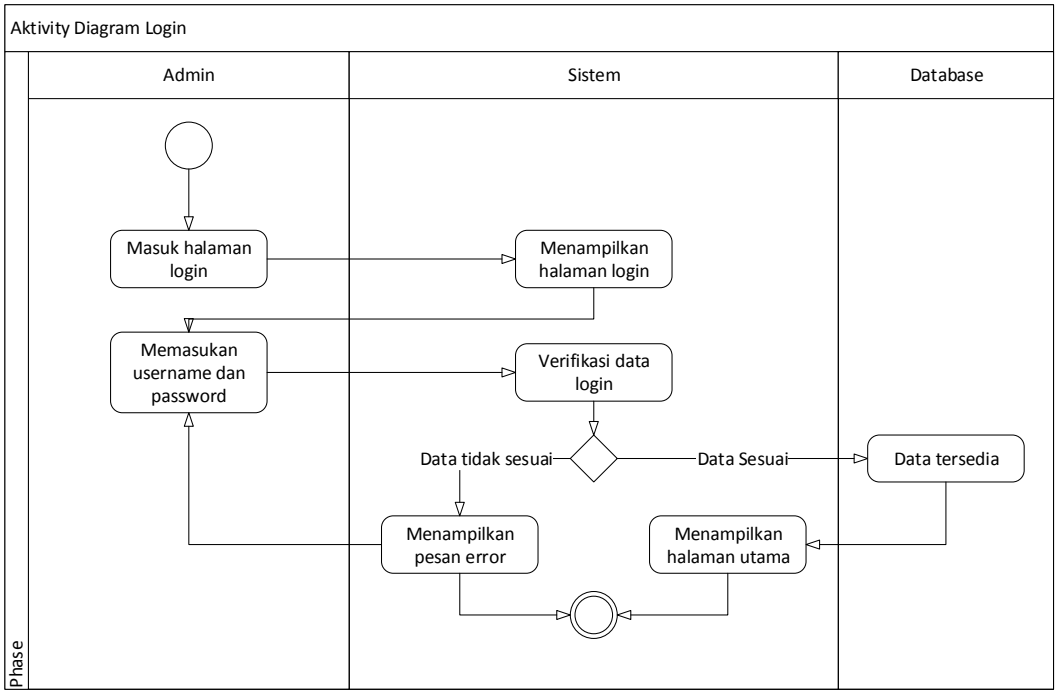
Tabel 2.3 Dasar Pembangunan *Use Case Scenario.*

Use Case Name	Berisi nama dari Use case yang akan digunakan	
Goal In Context	Menjelaskan apa yang aktor coba untuk dapatkan dari Use case	
Description	Menjelaskan gambaran dari Use case	
Related Use Case	Daftar Use case yang berhubungan dengan Use case tersebut	
Successful End Condition	Kondisi Use case jika berhasil	
Failed End Condition	Kondisi Use case jika gagal	
Actors	Daftar aktor yang dapat mengakses Use case	
Trigger	Aktifitas yang dilakukan untuk mengawali Use case	
Main Flow	Step	Action
	1	Deskripsi urutan aksi dari aktifitas Use case
	2	
	3	
Extension	Step	*Branching Action*
	2.1	Deskripsi urutan aksi lain selain urutan aksi utama
	2.1	

Use Case Scenario merupakan sebuah teknik yang digunakan dalam melihat alur jalannya suatu proses use case dari sisi aktor dan sistem. Alur skenario inilah yang nantinya menjadi dasar pembuatan diagram *sequence*.

2.2.5.1.3 *Activity Diagram*

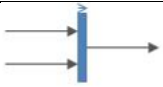
Activity diagram memodelkan alur kerja (*workflow*) sebuah proses bisnis dan urutan aktivitas dalam suatu proses. Diagram ini sangat mirip dengan sebuah flowchart karena dapat dimodelkan sebuah alur kerja dari satu aktivitas ke aktivitas lainnya atau dari satu aktivitas ke dalam keadaan sesaat (*state*). Seringkali bermanfaat bila dibuat sebuah activity terlebih dahulu dalam memodelkan sebuah proses untuk membantu memahami proses secara keseluruhan [11].



Gambar 2.7 Contoh Activity Diagram

Activity diagram juga sangat berguna ketika ingin menggambarkan perilaku paralel atau menjelaskan bagaimana perilaku dalam berbagai use case berinteraksi. Dapat digunakan statechart diagram untuk memodelkan perilaku dinamis satu kelas atau objek [11].

Tabel 2.4 *Elemen Activity Diagram*

Simbol	Keterangan
	*Start Point*
	*End Point*
	*Activities*
	Join (penggabungan)
	Fork(percabangan)
Swimline	Sebuah cara mengelompokkan aktivitas berdasarkan aktor (mengelompokkan

	aktivitas dalam sebuah urutan yang sama)

Activity Diagram merupakan sebuah teknik yang digunakan dalam rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.

2.2.5.1.4 *Class Diagram*

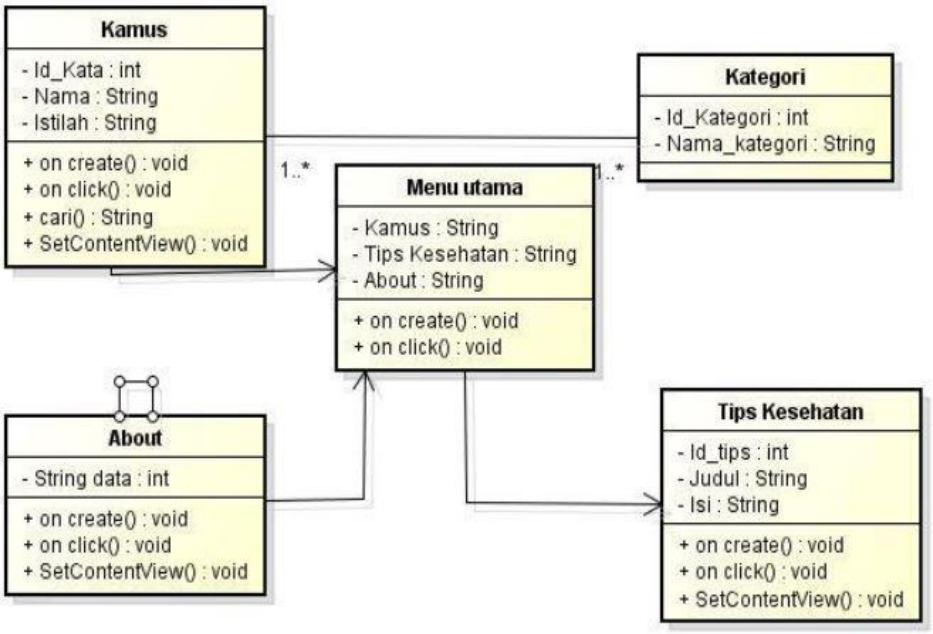
Sebuah Class Diagram menunjukkan struktur yang statis dari beberapa class dalam suatu sistem. Class-class merepresentasikan suatu keadaan (atribut/properti) dan yang akan dikerjakan oleh sistem (metode/fungsi) [12]. Class memiliki tiga area pokok:

1. Nama (*stereotype*)
2. Atribut
3. Metode

Atribut dan metode dalam class diagram dapat memiliki salah satu sifat seperti berikut di bawah ini:

1. Private, hanya dapat diakses oleh class itu sendiri.
2. Protected, hanya dapat diakses oleh class itu sendiri dan turunan dari class tersebut.
3. Public, dapat diakses oleh class selain dari class yang bersangkutan.

Class dapat direpresentasikan dalam sebuah interface atau sebaliknya merupakan implementasi dari sebuah interface yang berupa class abstrak yang hanya tidak memiliki attribute dan hanya memiliki metode. Berikut merupakan bentuk class diagram secara umum:



Sumber Gambar : W. Prayitno [12]

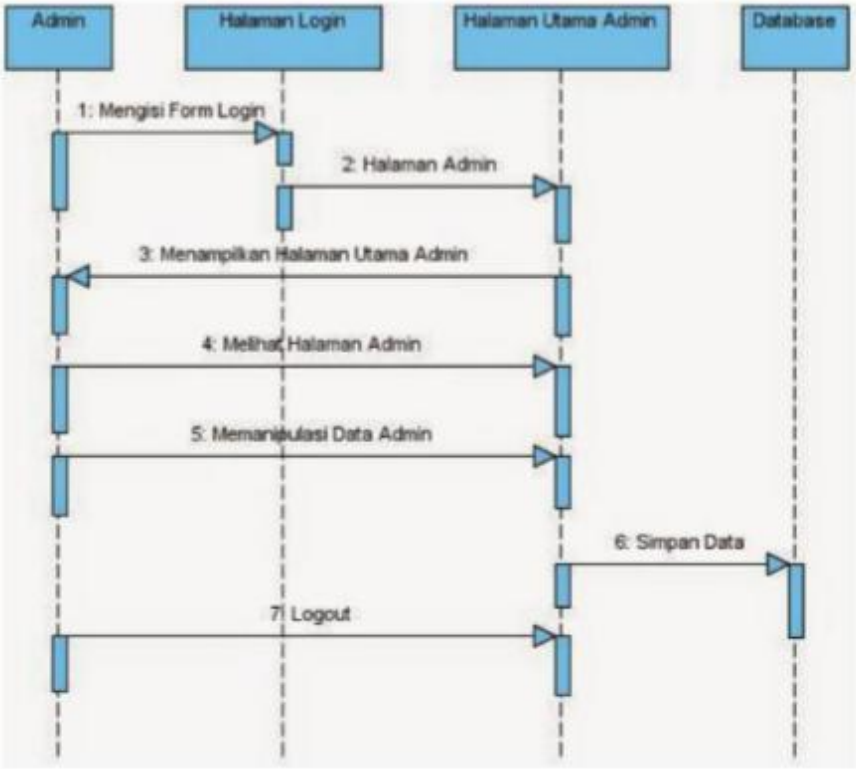
Gambar 2.8 Contoh Class Diagram

Class Diagram merupakan sebuah teknik yang digunakan dalam menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Diagram kelas mendeskripsikan jenis-jenis objek dalam sistem dan berbagai hubungan statis yang terdapat di antara mereka.

2.2.5.1.5 *Sequence Diagram*

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem berupa message yang digambarkan terhadap waktu. Sequence diagram terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah event untuk menghasilkan Output tertentu. Diagram sequence menjelaskan interaksi objek yang disusun dalam suatu urutan waktu. Diagram ini secara khusus berasosiasi dengan use case. Sequence diagram memperlihatkan tahap demi tahap apa yang seharusnya terjadi untuk menghasilkan sesuatu di dalam use case. Diagram sequen sebaiknya digunakan diawal tahap desain atau analisis karena kesederhanaannya dan mudah untuk dimengerti [11] untuk objek-objek yang

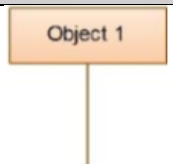
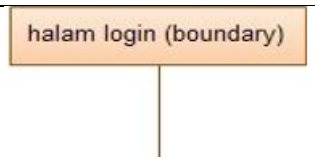
memiliki sifat khusus, standar UML mendefinisikan icon khusus untuk objek boundary, controller dan persistent entity.

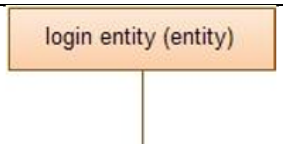
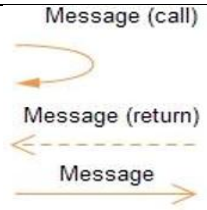


Gambar 2.9 Contoh Sequence Diagram

Sequence Diagram menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek [13].

Tabel 2.5 Elemen-elemen *Sequence Diagram*

No	Nama	Penjelasan	Gambar
1	*Objek Lifeline*	Menggambarkan batasan objek	
2	*Boundary*	Berhubungan dengan proses inputoutput interface	

3	*Controller*	Berhubungan dengan proses	
4	*Entity*	Berhubungan dengan inpu-output data	
5	*Message Arrow*	Menggambarkan alir proses, perintah atau pengiriman data	
6	*Aktivasi*	Menggambarkan aktivitas objek	
7	*Aktor*	Menggambarkan aktor suatu objek	

Dalam penelitian ini Sequence Diagram digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai sebuah respon dari suatu kejadian/even untuk menghasilkan output tertentu. Sequence Diagram diawali dari apa yang me-trigger aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan output apa yang dihasilkan.

2.2.6 *E-Learning*

E-learning adalah model pembelajaran yang menggunakan koneksi internet dengan jangkauannya yang luas serta memiliki kemampuan untuk menyimpan, memodifikasi, mendistribusi, dan membagikan suatu materi ajar melalui computer dan teknologi Internet yang standar. Dengan demikian, model pembelajaran ini secara praktis dapat menyampaikan suatu bahan ajar kepada peserta didik hanya dengan memanfaatkan jaringan Internet dan komputer. Artinya bahwa proses pembelajaran tidak harus dilakukan dengan cara tatap muka langsung. Studi empiris

menyatakan bahwa motivasi dan kemandirian peserta didik mengalami peningkatan dengan digunakannya pembelajaran e-learning tersebut, meskipun mungkin prestasi belajarnya tidak mengalami peningkatan ketika dibandingkan dengan pembelajaran konvensional [14]. Terlepas dari bentuk dan jenis tempat dimana *e-learning* diterapkan (sekolah, universitas, perusahaan, atau organisasi), *e-learning* harus selalu terdiri dari dua elemen dasar yaitu :

1. Pengajaran

   Pengajaran berarti mendapatkan pengetahuan baru atau meningkatkan pengetahuan yang ada, keterampilan atau kompetensi professional.

2. Teknologi

   Penggunaan teknologi informasi khususnya komputer dan interne mempermudah dalam mengatur belajar jarak jauh/ *distance learning* yang tersedia untuk siswa di tempat dan waktu pilihan mereka sendiri. Dikarenakan aplikasi pembelajaran untuk chatbot yang akan dibangun menggunakan *E-learning* maka subbab tentang pembahasan *E-learning* perlu digunakan pada landasan teori ini. Tentunya dengan adanya pembahasan *E-learning* ini dapat lebih meningkatkan pemahaman definisi dan keuntungan menggunakan *E-learning*. Dan dengan adanya pembahasan tentang *E-learning* ini dapat menjadi dasar atau landasan dalam membuat perancangan aplikasi pembelajaran untuk chatbot. Untuk detail dari *E-learning* yang digunakan dalam membangun aplikasi pembelajaran untuk chatbot ini adalah sebagai berikut:

2.2.6.1 *Moodle*

Moodle merupakan sebuah program aplikasi yang dapat merubah sebuah media pembelajaran ke dalam bentuk web dan dapat berfungsi sebagai media informasi dalam bentuk teks, grafik, simulasi, animasi, latihan-latihan, analisis kuantitatif, dan umpan balik langsung. Moodle dapat diakses oleh siswa dimanapun dan kapanpun tanpa batasan ruang dan waktu selama mereka masih terhubung dengan jaringan internet. Sehingga siswa dapat mengakses materi dan

menggunkannya sebagai bahan untuk belajar tanpa menginstal media ke komputer terlebih dahulu [15].

MOODLE merupakan salah satu *LMS open source* yang sangat fleksibel untuk kursus dan managemen pembelajaran. Kata MOODLE (http://moodle.org/) adalah singkatan dari modular *object-oriented dynamic learning environment* yang berarti tempat belajar dinamis dengan menggunakan model berorientasi objek. Dalam penyediaannya MOODLE memberikan paket software yang lengkap (MOODLE + Apache + MySQL + PHP). Pengembangan MOODLE awalnya merupakan penelitian doctoral Martin Dougiamas yang yang berasal dari Australia. Saat ini, MOODLE telah menarik banyak pengembang khususnya dalam penyempurnaan MOODLE.

Sumber Gambar : http://www.moodle.org

Gambar 2.10 Contoh MOODLE Learning Management System

Pengetahuan dan subbab tentang *Moodle* diperlukan karena *Moodle* ini dibutuhkan untuk membangun aplikasi pembelajaran untuk chatbot. Dengan adanya *Moodle* memungkinkan guru untuk masuk kedalam "ruang kelas digital" untuk mengakses materi-materi pembelajaran. Dengan menggunakan *Moodle*, pengguna dapat membuat materi pembelajaran, kuis, jurnal elektronik dan lain-lain serta menguploadnya dalam bentuk materi pembelajaran yang digunakan untuk menambah pengetahuan chatbotnya.. Tentunya dari pembahasan *Moodle* ini dapat menjadi landasan pemahaman dari definisi dan kegunaan *Moodle*.

2.2.7 *Application Programming Interface* (API)

Application Programming Interface (API) adalah kumpulan fungsi-fungsi atau perintah-perintah untuk menggantikan bahasa yang digunakan dalam system call dengan bahasa yang lebih terstruktur dan mudah dimengerti.Keuntungan memprogram dengan menggunakan API adalah:

1. Portabilitas. Programmer yang menggunakan API dapat menjalankan
2. programnya dalam sistem operasi mana saja asalkan sudah terinstall API tersebut. Sedangkan system call berbeda antar sistem operasi, dengan catatan dalam implementasinya mungkin saja berbeda.
3. Lebih Mudah Dimengerti. API menggunakan bahasa yang lebih terstruktur dan mudah dimengerti daripada bahasa system call. Hal ini sangat penting dalam hal editing dan pengembangan.

System call interface ini berfungsi sebagai penghubung antara API dan *system call* yang dimengerti oleh sistem operasi. System call interface ini akan menerjemahkan perintah dalam API dan kemudian akan memanggil system calls yang diperlukan.

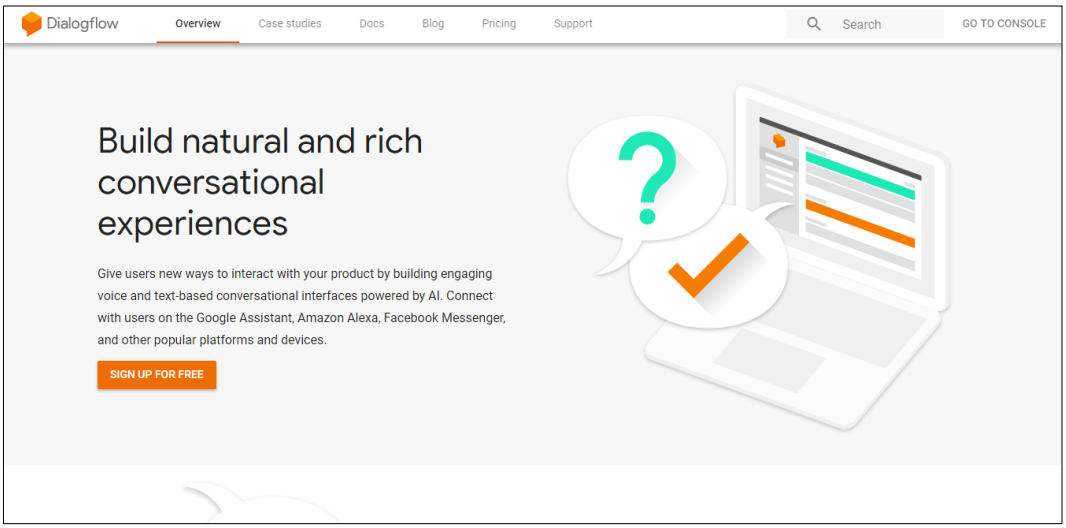
API adalah antarmuka yang digunakan untuk mengakses aplikasi atau layanan dari sebuah program. API memungkinkan pengembang untuk memakai fungsi yang sudah ada dari aplikasi lain sehingga tidak perlu membuat ulang dari awal. Pada konteks web, API merupakan pemanggilan fungsi lewat *Hyper Text Transfer Protocol* (HTTP) dan mendapatkan respon berupa *Extensible Markup Language* (XML) atau *JavaScript Object Notation* (JSON). Pemanggilan fungsi ke suatu situs tertentu akan menghasilkan respon yang berbeda kepada pengguna untuk membangun aplikasi enterprise di dalam websitenya [16].

Dikarenakan aplikasi pembelajaran untuk chatbot yang akan dibangun menggunakan berbagai macam API maka subbab tentang pembahasan API perlu digunakan pada landasan teori ini. Tentunya dengan adanya pembahasan API ini dapat lebih meningkatkan pemahaman definisi dan keuntungan menggunakan API. Dan dengan adanya pembahasan tentang API ini dapat menjadi dasar atau landasan dalam membuat perancangan aplikasi pembelajaran untuk chatbot. Untuk

detail dari API yang digunakan dalam membangun aplikasi pembelajaran untuk chatbot ini adalah sebagai berikut:

2.2.7.1 *API Dialogflow*

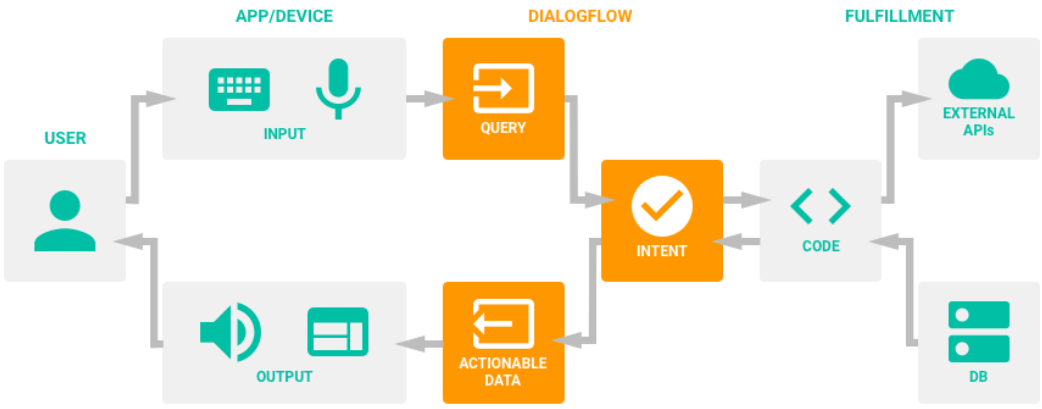
API Dialogflow Berikan cara baru kepada pengguna untuk berinteraksi dengan produk Anda dengan membangun antarmuka percakapan suara dan percakapan berbasis teks yang didukung oleh AI. Terhubung dengan pengguna di *Asisten Google, Amazon Alexa, Facebook Messenge*r, dan platform dan perangkat populer lainnya.



Sumber Gambar : http://www.dialogflow.com/

Gambar 2.11 Contoh Gambar API Dialogflow

Baik pengguna sedang bepergian atau di rumah, libatkan mereka melalui Perangkat yang dapat dikenakan, telepon, mobil, speaker, dan perangkat pintar lainnya.



Sumber Gambar : D. Dutta [17]

Gambar 2.12 Contoh Cara Kerja API *Dialogflow*

Dialogflow.com adalah platform untuk mengembangkan Chat-bots berdasarkan percakapan bahasa alami. Konsep penting seperti Intents and Contexts digunakan untuk memodelkan perilaku chat-bot. Maksudnya adalah pemetaan antara apa yang dimasukkan pengguna dan respons atau tindakan apa yang harus dilakukan oleh bot. Konteks digunakan untuk membedakan input pengguna yang mungkin memiliki maksud yang berbeda tergantung pada input pengguna sebelumnya. Ketika pengguna memasukkan data dalam platform Dialogflow.com, itu pertama kali diperiksa jika cocok dengan maksud yang ditentukan sebelumnya. Dialogflow.com memiliki fitur bernama "*Default Fallback*
intent "untuk menangani input pengguna yang tidak sesuai dengan maksud yang telah ditentukan. Kasus kecocokan suatu maksud dapat dibatasi dengan menyatakan daftar konteks yang seharusnya berfungsi. Kasus kecocokan suatu maksud dapat membuat dan menghapus konteks. Sistem maksud dan konteks ini membuat ketentuan untuk mengembangkan chat-bot yang dapat memiliki arus besar dan kompleks. Keterbatasan dari Dialogflow.com adalah: chat-bot tidak dapat dirancang dengan cara yang maksudnya hanya dapat dicocokkan jika konteks tertentu tidak hadir [17].

Pengetahuan dan subbab tentang *API Dialogflow* diperlukan karena API ini dibutuhkan untuk membangun aplikasi pembelajaran untuk chatbot. Dengan adanya *API Dialogflow* untuk pembuatan *chatbot* pengguna dapat tanya jawab seputar materi pembelajaran di chatbot tersebut, agar pengguna dapat lebih memahami materi pelajaran berdasarkan materi yang telah diupload. Tentunya dari pembahasan *API Dialogflow* ini dapat menjadi landasan pemahaman dari definisi dan kegunaan *API Dialogflow*.

2.2.8 *Chatbot*

Chatbot memiliki kemampuan untuk meniru percakapan manusia dan dapat menawarkan layanan pribadi. Ada dua jenis aplikasi chatbot: Jenis chatbot pertama adalah chatbot berbasis-web yang berjalan di cloud dan dapat diakses melalui antarmuka web. Jenis chatbot kedua adalah bot obrolan mandiri aplikasi yang dapat

diakses pada satu komputer. Aplikasi chatbot diminta dengan bahasa alami input teks, dan aplikasi chatbot membalas dengan yang terbaik respons cerdas terhadap teks masukan pengguna. Percakapan dilanjutkan dengan mengulangi proses ini [17]. Chatbot merupakan program computer yang berinteraksi dengan user menggunakan bahasa natural. Teknologi *chatbot* pertama dimulai tahun 1960an. Tujuan pembuatan chatbot ini adalah pengujian apakah *chatbot* dapat menipu *user* agar mereka mengira sedang berkomunikasi dengan manusia [18]. Pengujian ini dikenal dengan nama "*Turning Test*".

Chatbot adalah sebuah program yang dapat melakukan obrolan dalam bahasa alami mengenai sebuah topik yang ada dalam model pengetahuan chatbot tersebut. Artinya, *chatbot* harus bisa mengenali dan merespon kata-kata yang diberikan oleh user. Kemampuan chatbot dalam mengenali dan memberikan respon di tentukan oleh ruang lingkup dari pengetahuan *chatbot*.

Pengetahuan dan subbab tentang chatbot diperlukan karena pada penelitian ini aplikasi yang dibangun menggunakan chatbot. Tentunya dengan adanya pembahasan ini dapat menjadi landasan pemahaman dari apa itu isi dari chatbot tersebut. Pembahasan tentang chatbot ini juga akan diperdalam lagi seperti pengetahuan chatbotnya seperti yang telah ditulis diatas.

2.2.9 PHP

PHP atau kependekan dari Hypertext Preprocessor adalah salah satu bahasa pemrograman open source yang sangat cocok atau dikhususkan untuk pengembangan web dan dapat ditanamkan pada sebuah skripsi HTML. Bahasa PHP dapat dikatakan menggambarkan beberapa bahasa pemrograman seperti C, Java, dan Perl serta mudah untuk dipelajari. PHP merupakan bahasa scripting server – side, dimana pemrosesan datanya dilakukan pada sisi server. Sederhananya, serverlah yang akan menerjemahkan skrip program, baru kemudian hasilnya akan dikirim kepada client yang melakukan permintaan. Adapun pengertian lain PHP adalah akronim dari Hypertext Preprocessor, yaitu suatu bahasa pemrograman berbasiskan kode – kode (script) yang digunakan untuk mengolah suatu data dan mengirimkannya kembali ke web browser menjadi kode HTML" [19]. Adapun

contoh script dalam bahasa pemrograman PHP (*Hypertext Preprocessor*) sebagai berikut :

```
<?php
     echo "Hello World…!!!";
     echo "<br />";
     print "Hello Again World…!!!";
     print "<br /><br />";
```

Pengetahuan dan subbab tentang PHP diperlukan karena Bahasa pemrogramman yang digunakan dalam membangun aplikasi pembelajaran untuk chatbot pada Moodle adalah PHP. Bahasa pemrogramman dengan menggunakan PHP cocok diimplementasikan pada aplikasi yang dibangun diatas *platform* Web. Tentunya dari pembahasan PHP ini dapat menjadi landasan pemahaman dari apa itu PHP *programming language* dan fungsionalitasnya.

2.2.10 MySQL

MySQL adalah database server open source yang cukup popular keberadaannya. Dengan berbagai keunggulan yang dimiliki, membuat software database ini banyak digunakan oleh praktisi untuk membangun suatu project. Adanya fasilitas API (Application Programming Interface yang dimiliki oleh MySQL, memungkinkan bermacam–macam aplikasi komputer yang ditulis dengan berbagai bahasa pemrograman dapat mengakses basis data MySQL. "Tipe data MySQL adalah data yang terdapat dalam sebuah tabel berupa field – field yang berisi nilai dari data tersebut. Nilai data dalam field memiliki tipe sendiri – sendiri" [19].

Berikut contoh bahasa pemograman pada MySQL :

1. Membuat database

```
CREATE DATABASE admin;
```

2. Melihat database

```
SHOW DATABASES;
```

3. Membuat tabel

```
CREATE TABLE login(
     id VARCHAR (20) NOT NULL,
     nama VARCHAR (50) NOT NULL,
     password VARCHAR(100) NOT NULL,
     created_at DATETIME NOT NULL,
     updated_at TIMESTAMP,
```

4. Melihat tabel

```
SHOW TABLES;
```

5. Menghapus tabel

```
DROP TABLE login;
```

6. Menghapus database

```
DROP DATABASE admin;
```

Pengetahuan dan subbab tentang MySQL diperlukan karena Bahasa pemrogramman yang digunakan dalam menyimpan data aplikasi pembelajaran untuk chatbot adalah MySQL. Bahasa pemrogramman dengan menggunakan MySQL cocok digunakan sebagai tempat penyimpanan data. Tentunya dari pembahasan MySQL ini dapat menjadi landasan pemahaman dari apa itu *Structure Query Language* dan fungsionalitasnya.

2.2.11 Web Server

Web Server adalah sebuah perangkat lunak yang bertugas menerima permintaan client melalui port HTTP maupun HTTPS dan merubahnya ke dalam format HTML. Terdapat beberapa format selain HTML yaitu PHP atau ASP,

tetapi format – format tersebut hanyalah berfungsi untuk menghubungkan HTML dengan *database*.

Web server yang umum diguanakan adalah *Apache*. Tugas utama dari Web server adalah menerjemahkan permintaan ke dalam respon yang cocok untuk keadaan pada saat itu, ketika klien membuka komunikasi dengan *Apache*, *Apache* mengirimkan permintaan untuk sumber daya. *Apache* menyediakan sumber daya yang baik atau memberikan respon alternatif untuk menjelaskan mengapa permintaan tidak dapat terpenuhi. Setiap komunikasi HTTP dimulai dengan permintaan dan berakhir dengan jawaban. *Executable Apache* mengambil nama dari protokol, dan pada sistem Unix umumnya disebut httpd, kependekan daemon HTTP [20].

Pengetahuan dan subbab tentang web server diperlukan karena pada penelitian ini aplikasi yang dibangun menggunakan web server. Tentunya dengan adanya pembahasan ini dapat menjadi landasan pemahaman dari apa itu isi dari web server tersebut. Pembahasan tentang web server ini juga akan diperdalam lagi seperti web server yang umum digunakan seperti diatas.

2.2.12 Metode Pengujian

Metode pengujian adalah cara atau teknik untuk menguji perangkat lunak, mempunyai mekanisme untuk menentukan data uji yang dapat menguji perangkat lunak secara lengkap dan mempunyai kemungkinan tinggi untuk menemukan kesalahan. Pengujian perangkat lunak adalah proses untuk mencari kesalahan pada setiap *item* perangkat lunak, mencatat hasilnya, mengevaluasi setiap aspek pada setiap komponen (sistem) dan mengevaluasi aktivitas-aktivitas dari perangkat lunak yang akan dikembangkan [10].

Dikarenakan aplikasi pembelajaran untuk chatbot yang akan dibangun akan diuji maka subbab tentang pembahasan Metode Pengujian perlu digunakan pada landasan teori ini. Tentunya dengan adanya pembahasan Metode Pengujian ini dapat lebih meningkatkan pemahaman definisi dan keuntungan menggunakan Metode Pengujian. Dan dengan adanya pembahasan tentang Metode Pengujian ini dapat menjadi dasar atau landasan dalam melakukan pengujian aplikasi

pembelajaran untuk chatbot. Untuk detail dari Metode Pengujian yang digunakan dalam melakukan pengujian aplikasi pembelajaran untuk chatbot ini adalah sebagai berikut:

Berikut tahapan pengujian perangkat lunak yaitu :

1. Pengujian Unit / Unit Testing

   Unit Testing adalah metode verifikasi perangkat lunak di mana programmer menguji suatu unit program layak untuk tidaknya dipakai. Unit testing ini fokusnya pada verifikasi pada unit yang terkecil pada desain perangkat lunak (komponen atau modul perangkat lunak). Karena dalam sebuah perangkat lunak banyak memiliki unit-unit kecil maka untuk mengujinya biasanya dibuat program kecil atau main program) untuk menguji unit-unit perangkat lunak. Unit-unit kecil ini dapat berupa prosedur atau fungsi, sekumpulan prosedur atau fungsi yang ada dalam satu file jika dalam pemrograman terstruktur, atau kelas, bisa juga kumpulan kelas dalam satu package dalam PBO. Pengujian unit biasanya dilakukan saat kode program dibuat.

2. Pengujian Integrasi

   Pengujian integrasi lebih pada pengujian penggabungan dari dua atau lebih unit pada perangkat lunak. Pengujian integrasi sebaiknya dilakukan ssecara bertahap untuk menghindari kesulitan penelusuran jika terjadi kesalahan *error* / *bug* [21].

3. Pengujian Sistem

   Unit-unit proses yang telah diintegrasikan diuji dengan antarmuka yang sudah dibuat sehingga pengujian ini dimaksud untuk menguji sistem perangkat lunak. Perlu diingat bahwa pengujian sistem harus dilakukan secara bertahap sejak awal pengembangan, jika pengujian hanya diakhir maka dapat dipastikan kualitas sistemnya kurang bagus [21].

4. Pengujian Penerimaan

   Pengujian penerimaan perangkat lunak dilakukan oleh pengguna yang telah bekerja sama dengan pembuat program guna untuk mengetahui secara langsung bagaimana perangkat lunak yang telah dibuat dapat bekerja sebelum

perangkat lunak yang dibuat disebar luaskan. Pengujian penerimaan ini bertujuan untuk mengetahui kepuasan pengguna atau *user* [21].

2.2.12.1 *Black-Box Testing*

Black box testing adalah pengujian yang dilakukan hanya mengamati hasil eksekusi melalui data uji dan memeriksa fungsional dari perangkat lunak. Jadi dianalogikan seperti kita melihat suatu kotak hitam, kita hanya bisa melihat penampilan luarnya saja, tanpa tau ada apa dibalik bungkus hitam nya. Sama seperti pengujian black box, mengevaluasi hanya dari tampilan luarnya(interface nya) , fungsionalitasnya.tanpa mengetahui apa sesungguhnya yang terjadi dalam proses detilnya (hanya mengetahui input dan output) [21].

Kelebihan *Black Box:*

a. Dapat memilih subset test secara efektif dan efisien
b. Dapat menemukan cacat
c. Memaksimalkan testing Investment

Kelemahan *Black Box*

Tester tidak pernah yakin apakah PL tersebut benar – benar lulus uji.

Perbedaan White Box & Black Box.

1. *White box (Struktural)*
   1) Dilakukan oleh penguji yang mengetahui tentang QA.
   2) Melakukan testing pada software/program aplikasi menyangkut security dan performance program tersebut (meliputi tes code, desain implementasi, security, data flow, software failure).
   3) Dilakukan seiring dengan tahapan pengembangan software atau pada tahap testing.
2. *Metode BlackBox (Fungsional)*
   1) Dilakukan oleh penguji Independent.
   2) Melakukan pengujian berdasarkan apa yang dilihat, hanya fokus terhadap fungsionalitas dan output. Pengujian lebih ditujukan pada desain software sesuai standar dan reaksi apabila terdapat celah-celah bug/vulnerabilitas pada program aplikasi tersebut setelah dilakukan white box testing.

3) Dilakukan setelah white box testing.

Langkah-langkah pengujian software ada 4 yaitu:

1. *Unit testing*-testing per unit yaitu mencoba alur yang spesifik pada struktur modul kontrol untuk memastikan pelengkapan secara penuh dan pendeteksian error secara maksimum.
2. *Integration testing*-testing per penggabungan unit yaitu pengalamatan dari isu-isu yang diasosiasikan dengan masalah ganda pada verifikasi dan konstruksi program.
3. *High-order test* yaitu terjadi ketika software telah selesai diintegrasikan atau dibangun menjadi satu –tidak terpisah-pisah.
4. *Validation test* yaitu menyediakan jaminan akhir bahwa software memenuhi semua kebutuhan fungsional, kepribadian dan performa.

Saat ini terdapat banyak metoda atau teknik untuk melaksanakan Black Box Testing, antara lain:

1. *Equivalence Partitioning*
2. *Boundary Value Analysis/Limit Testing*
3. *Comparison Testing*
4. *Sample Testing*
5. *Robustness Testing*
6. *Behavior Testing*
7. *Requirement Testing*
8. *Performance Testing*
9. Uji Ketahanan (*Endurance Te*sting)
10. Uji Sebab-Akibat (*Cause-Effect Relationship Testing*).

Dikarenakan aplikasi pembelajaran untuk chatbot yang menggunakan Pengujian *Black-Box Testing* maka subbab tentang pembahasan *Black-Box Testing* perlu digunakan pada landasan teori ini. *Black-Box Testing* berfungsi untuk pengujian yang dilakukan hanya mengamati hasil eksekusi melalui data uji dan memeriksa fungsional dari perangkat lunak. Tentunya dengan adanya pembahasan *Black-Box Testing* ini dapat lebih meningkatkan pemahaman apa itu *Black-Box Testing* dan bagaimana cara menguji suatu perangkat lunaknya.