BAB 2
LANDASAN TEORI

2.1 Pengertian Properti

Properti merupakan aset yang dimana termasuk sarana dan prasarana yang ada dalam tanah dan bangunan dan merupakan bagian yang tidak terpisahkan. Adapun dari properti itu sendiri dibagi menjadi 2 yaitu :

1. Non Komersial

   Non komersial adalah properti yang hanya dimiliki untuk kebutuhan sehari-hari seperti rumah yang digunakan untuk tempat tinggal.
2. Komersial

   Komersial adalah properti yang hanya disediakan untuk kebutuhan dalam mencari keuntungan, dengan tujuan untuk penyewaan atau jual-beli perumahan.

2.2 Pengertian Rumah

Rumah adalah sebuah bangunan yang disusun dan dibangun dengan berbagai macam bahan seperti semen, pasir dan batu bata merah, rumah juga digunakan sebagai tempat untuk berlindung ataupun bernaung dari pengaruh keadaan alam di sekitarnya.

2.3 Syarat Rumah Tinggal

1. Aksesibilitas
   a. Kebutuhan akan transportasi terpenuhi dengan mudah dan murah, dan jarak tempat ke fasilitas mudah dan cepat.
   b. Kualitas jalan menuju suatu lokasi yang cukup baik, aman dan nyaman hendaknya lancar.
2. Lingkungan
   a. Sarana dan prasarana harus memadai, misalnya : jalan, lingkungan, tempat-tempat ibadah, olahraga, taman, sekolah, bank atau kantor polisi.
   b. Keamanan lingkungan jauh dari lingkungan yang negatif, tidak rawan kriminalitas.

c. Penataan lingkungan cukup asri dan alami, tidak terjadi banjir ataupun rawan bencana alam.

2.4 Sistem Operasi Android

Android adalah sebuah sistem operasi untuk smartphone berbasis linux. Android sendiri menyediakan platform terbuka bagi para pengembang untuk menciptakan perangkat lunak sendiri untuk digunakan oleh berbagai macam perangkat bergerak (*mobile devices*). Android mulai keluar di tahun 2003 dan dibeli oleh google pada tahun 2005 sebagai jalan masuknya pasar, versi pertama dari android yang rilis adalah G1 atau yang dikenal juga dengan HTC Dream di tahun 2008 [4]. Selain *Google*, beberapa nama-nama besar juga ikut serta dalam Open Handset Alliance, antara lain *Motorola, Samsung, LG, Sony Ericsson, T-Mobile, Vodafone, Toshiba, dan Intel*. Dalam penelitian ini versi android yang digunakan untuk dapat menjalankan aplikasi yang sedang dibangun ini sebagai berikut [4]:

1. Android versi 4.1-4.3 (Jelly Bean)

Pada 27 Juni 2012, Android versi 4.1 (*Jelly Bean*) diluncurkan. Pada versi ini merupakan rilis minor yang membawa perbaikan fungsionalitas dan performa rendering User Interface (UI). Versi 4.2 mengusung fitur baru seperti gesture typing dan dukungan multiuser di piranti tablet. Versi 4.3 membawa perbaikan. berupa dukungan *Bluetooth Low Energy* dan OpenGL ES 3.0. beberapa fitur baru yaitu dukungan terhadap bahasa internasional dan penulisan teks dua arah.

2. Android versi 4.4 (Kitkat)

Android versi 4.4 (Kitkat) diluncurkan pada 3 September 2013. Pada versi ini mengusung sejumlah perbaikan dan fitur baru yaitu dukungan Near Field Communication melalui host card emulation, pencetakan printer nirkabel, WebView dengan *rendering engine Chromium*, dan dukungan yang lebih baik bagi piranti yang menggunakan Random Access Memory (RAM) rendah.

3. Android versi 5.0 (Lollipop)

Android versi lollipop telah menerapkan processor 64 bit dan mampu melakukan pengoptimalan penggunaan baterai, sehingga smartphone yang support

dengan versi android ini akan memiliki daya tahab baterai yang lebih tinggi dibandingkan dengan versi sebelumnya. Perubahan internal pada platform juga dilakukan, yaitu menggantikan dalvik dengan android runtime (ART) untuk meningkatkan performa aplikasi.

4. Android 6.0 (Marshmallow)

Android 6.0 dengan kode M (Marshmellow) telah diperkenalkan oleh google pada bulan oktober 2015. Sistem operasi android marshmellow merupakan lanjutan dari pengembangan android lolipop. Pada versi ini, diperkenalkan aplikasi API untuk menyempurnakan asisten kontekstual dalam Google Now On Tap, sebagai kemampuan baru dari aplikasi google search [5].

5. Android 7.0 (Nougat)

Android 7.0 dengan kode N (Nougat) diresmikan pada bulan maret 2016. Nama "Nougat" diambil dari nama permen kacang yang popular di perancis. Android nougat memungkinkan para penggunanya untuk dapat membuka dua aplikasi sekaligus dalam mode split screen. Pengguna juga dapat memiliki sampai Sembilan toggles dalam satu layar dengan fitur pengaturan cepat [5].

2.5 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 seperti ditunjukkan pada klasifikasi tersebut berdasarkan kelas-kelas dasar java yang digunakan [6].

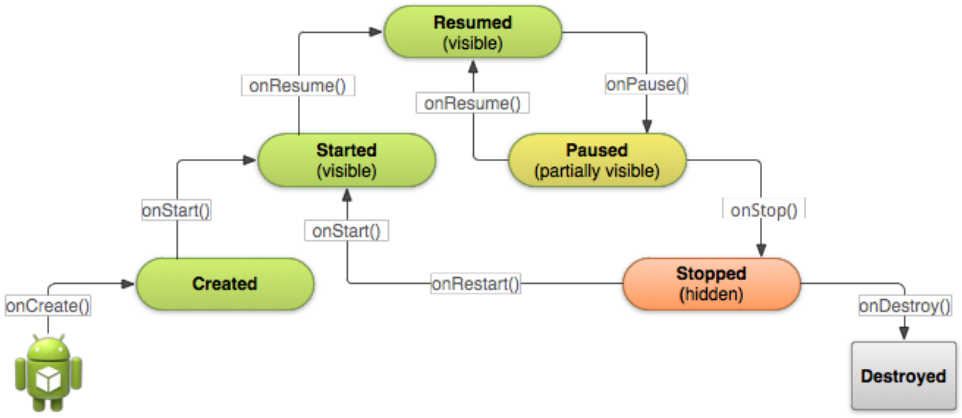
Tabel 2.1 Komponen Aplikasi Android

Functionality	Java Base Class	Examples
Focused thing a user can do	Activity	Edit a note, play a game
Background process	Service	Play music, update weather icon
Receive messages	Broadcast Receiver	Trigger alarm upon event

Store and retrieve data	Content Provider	Open a phone contact

Setiap aplikasi pasti menggunakan minimal satu dari komponen tersebut, akan tetapi terdapat beberapa komponen yang mengharuskan mencantumkan specified permission sebelum digunakan seperti komponen Service, Broadcast Receiver. Content Provider.

Android memiliki paradigma pemrograman lain tidak seperti paradigma pemrograman biasa di mana aplikasi yang dijalankan pada fungsi main(), sistem android menjalankan kode dalam method Activity dengan menerapkan metode callback tertentu yang sesuai dengan tahap tertentu dari siklus hidup. Setiap aplikasi yang berjalan dalam sistem operasi android memiliki siklus hidup yang berbeda dengan aplikasi desktop atau web, Hal ini dikarenakan aplikasi mobile memiliki tingkat interupsi proses yang lumayan tinggi seperti ketika handling panggilan masuk aplikasi diharuskan menghentikan proses sementara, Penerapan siklus hidup juga berguna untuk memastikan aplikasi tidak menghabiskan sumber daya baterai pengguna [6].



Gambar 2.1 Siklus Hidup Android

Terdapat beberapa state dalam siklus hidup android yang terjadi seperti diilustrasikan pada Gambar II.2, akan tetapi hanya beberapa dari state 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().

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 dalam state ini.

3. Stopped

Dalam keadaan ini, aplikasi benar-benar tidak ditampilkan dan tidak terlihat oleh pengguna tetapi masih meninggalkan service di background.State lain seperti *Created* dan *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*().

2.6 Android SDK (Software Development Kit)

Android SDK merupakan kumpulan-kumpulan tools yang digunakan untuk mengembangkan aplikasi android. Seperti pada java yang dapat dijalankan pada multiplatform, SDK android pun tersedia untuk beberapa platform seperti Windows, mac OS dan linux [7]. Adapun aplikasi SDK yang ada didalamnya terdapat beberapa folder sebagai berikut :

1. Tools

   Berisikan tools-tools yang digunakan dalam membangun aplikasi android seperti emulator, Android Virtual Devices (AVD), SDK Manager, ddms dan masih banyak tools lain untuk memudahkan dalam membangun aplikasi android. Semua tools dapat di update mengikuti setiap perkembangan pada platform android.

2. Platforms

   Direktori ini ditujukkan untuk meletakkan versi platform-platform android yang telah kita download dari SDK Manager.

3. Add-ons

Direktori ini nantinya berisikan tools tambahan untuk Android SDK, yang fungsinya untuk menambahkan library yang akan digunakan untuk pengembangan aplikasi. Seperti jika ingin mengembangkan aplikasi menggunakan map dari google maka harus memerlukan platform android yang sudah terdapat library map. Semua dapat di download melalui SDK Manager. Selain folder-folder diatas, akan terdapat file tambahan jika akan mendownload beberapa platform dan tool dalam folder android SDK tersebut.

2.7 Java

Bahasa komputer dimotivasi oleh dua faktor: perbaikan dalam seni pemrograman dan perubahan dalam lingkungan komputasi, tidak terkecuali Java. Dibangun di atas warisan yang kaya dariC dan C++, Java menambahkan perbaikan dan fitur yang mencerminkan keadaan seni dalam pemrograman saat ini. Menanggapi munculnya lingkungan online, Java menawarkan fitur yang merampingkan pemrograman untuk arsitektur yang sangat terdistribusi.

Java adalah bahasa pemrograman yang dapat dijalankan di berbagai komputer termasuk telepon genggam. Bahasa ini awalnya dibuat oleh James Gosling saat masih bergabung di Sun Microsystems saat ini merupakan bagian dari Oracle dan dirilis tahun 1995. Bahasa ini banyak mengadopsi sintaksis yang terdapat pada C dan C++ namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin-rutin aras bawah yang minimal. Aplikasi-aplikasi berbasis java umumnya dikompilasi ke dalam p-code (bytecode) dan dapat dijalankan pada berbagai Mesin Virtual Java (JVM).

Banyak jenis komputer dan sistem operasi yang terhubung ke Internet. Untuk program-program untuk secara dinamis didownload ke semua berbagai jenis platform, beberapa sarana untuk menghasilkan kode dieksekusi portabel diperlukan. Mekanisme yang sama yang membantu menjamin keamanan juga membantu menciptakan portabilitas karena fungsionalitasnya yang memungkinkan aplikasi java mampu berjalan di beberapa platform sistem operasi yang berbeda. Saat ini java merupakan bahasa pemrograman yang populer digunakan dan secara

luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi ataupun aplikasi berbasis web [8].

Java memiliki beberapa kelebihan dibandingkan dengan Bahasa pemrograman lain, diantaranya :

a. Multiplatform
b. OOP (*Object Oriented Programming*)
c. Library class yang lengkap
d. Mewarisi kekayaan C/C++
e. Pengumpulan sampah otomatis
f. Mudah didekompilasi

   Dekompilasi adalah proses membalikkan hasil kompilasi menjadi kode sumber. Ini dimungkinkan karena kode pada java merupakan bytecode yang menyimpan banyak atribut bahasa tingkat tinggi. Dengan demikian, algortima yang digunakan program akan lebih sulit disembunyikan dan mudah dibajak/direverse-engineer.
g. Penggunaan memori yang banyak

   Penggunaan memori untuk program berbasis java jauh lebih besar daripada Bahasa tingkat tinggi pada generasi sebelumnya seperti C/C++ dan pascal [8].

2.8 Application Programming Interface (API)

API merupakan sebuah teknologi untuk memfasilitasi pertukaran informasi atau data antara dua atau lebih aplikasi perangkat lunak. API adalah antarmuka virtual antara dua fungsi perangkat lunak yang saling bekerja sama.

Perkembangan teknologi API berawal dari penciptaan sebuah subrutin sederhana yang menyediakan kemampuan interoeperability dan system modifiability untuk mendukung pertukaran data antar beberapa aplikasi. Subrutin tersebut hanya mampu melaksanakan perhitungan matematis sederhana hingga terbentuk sebuah API pustaka penghitungan yang hamper selalu ada pada setiap bahasa pemrograman. Dari sebuah subrutin sederhana tersebut mulai bermunculan gagasan-gagasan bagaimana API harus dikembangkan khususnya sejalan dengan

berkembangnya paradigma pemrograman berorientasi objek yang mengakibatkan sekumpulan subrutin sejenis dapat dikumpulkan menjadi sebuah pembungkus subrutin-subrutin tersebut [9].

Dengan berkembangnya sebuah perangkat lunak menjadi sebuah system yang terdiri dari beberapa perangkat lunak lain (subsistem-subsistem) maka API juga berkembang untuk tetap mewujudkan tujuannya yaitu sebagau jembatan penghubung antar perangkat lunak. Perkembangan API diwujudkan dengan bertambahnya sifat API yaitu mampu mendukung *interoperability* antar perangkat lunak. API tidak hanya bertugas mempertukarkan data dan informasi antar subrutin dalam suatu perangkat lunak namun juga mempertukarkan data dan informasi antar perangkat lunak. Dalam kasus ini API harus memiliki kemampuan komunikasi antar proses melalui perantara file, socket, atau layanan IPC lainnya [9].

Pada pengembangan sebuah system yang lebih luas, kadang sebuah API dapat menjadi sebuah middleware, yaitu sebuah subsistem tersendiri yang memiliki fungsi-fungsi yang berguna oleh subsistem lain dan untuk mengakses fungsi-fungsi tersebut diperlukan koneksi tersendiri ke middleware tersebut. Koneksi ke middleware umumnya dilakukan dengan menggunakan socket. Ada beberapa standar pengaksesan middleware tersebut [9]. Berikut adalah protokol pengaksesan middleware antara lain adalah :

1. Remote Procedure Calls (RPC)

   Sebuah perangkat lunak pengguna memanggil prosedur atau subrutin yang berjalan pada sebuah middleware jarak jauh, pemanggilan prosedur daoat berupa sinkron maupun asinkron.

2. Message Oriented Middleware (MOM)

   System yang melakukan pooling data dan informasi ke middleware, data yang menunggu untuk diproses oleh subsistem-subsistem yang terdapat di system keseluruhan disimpan pada suatu middleware tersebut.

3. Structured Query Language (SQL)

   Protokol dan juga Bahasa pembacaan dan penulisan data yang tersimpan pada middleware basis data.

4. Object Request Broker (ORB)

Protocol ini memungkinkan suatu perangkat lunak mengirim dan menerima objek dan meminta layanan pada sebuah middleware berbasis objek.

API dapat diklasifikasikan menjadi bebrapa kategori, hal ini dilihat dari abstraksi apa yang dideskripsikan di dalam system. Kategori-kategoru ini diantaranya :

Tabel 2.2 Kategori API

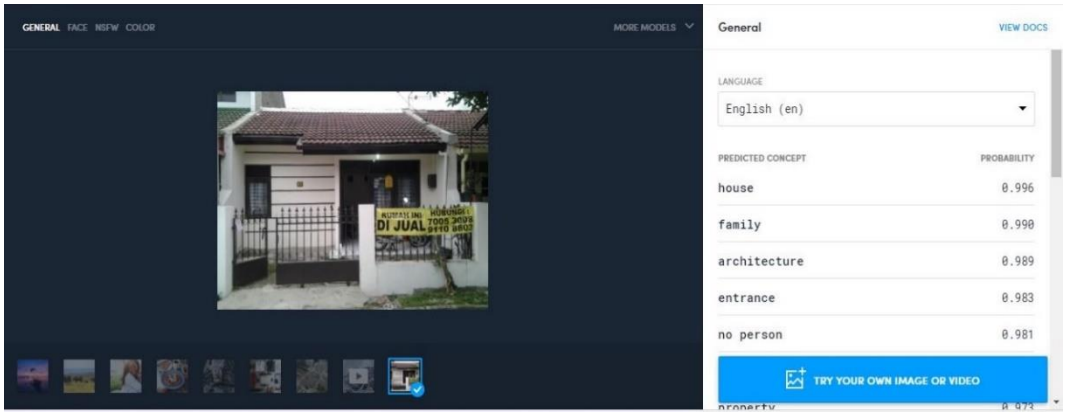
Kategori API	Deskripsi	Contoh
Operating System	API yang digunakan untuk fungsi dasar yang dapat dilakukan oleh computer seperti proses I/O, eksekusi program.	API for MS Windows
Prgoramming Languages	API yang digunakan untuk memperluas kapabilitas dalam melakukan eksekusi terhadap suatu Bahasa pemrograman.	Java API
Application Services	API yang digunakan untuk mengakses data dan layanan yang disediakan dari suatu aplikasi.	API for mySAP (BAPI/Business Application Programming Interface)
Infrastructure Services	Digunakan untuk mengakses infrastruktur dari suatu computer. Infrastruktur disini adalah computer beserta peripheral seperti storage, aplikasi, dan lain-lain.	Amazon EC2 (Elastic Compute Cloud) untuk akses pada virtual computing dan amazon S3 (Simple Storage Service) untuk menyimpan data dalam jumlah yang besar.
Web Services	API yang digunakan untuk mengakses content dan layanan yang disediakan oleh suatu web application.	Facebook Graph API yang digunakan untuk mengakses informasi yang dapat dibagikan.

2.9 Clarifai API

Clarifai adalah alat pengenalan gambar dan video yang secara otomatis memberikan tag ke objek dan kategori mengambil sebuah piksel sebagai input, menggunakan library semantik dan visual untuk kecerdasan buatan. Pada sistem ini didasarkan pada jaringan syaraf tiruan, teknik pembelajaran mesin scalable yang dapat menangani skala besar konten visual yang mengalir melalui API. Clarifai juga menggunakan kesamaan semantic dan visual untuk membandingkan gambar yang diunggah dengan gambar lainnya pada library untuk menampilkan tingkat kesamaan pada objek [10].

2.9.1 Model Clarifai

Gambar yang diambil terlebih dahulu akan ditandai menggunakan model. Sebuah model dari classifier yang terlatih yang dapat mengenali apa yang ada di dalam sebuah gambar. Model rumah menganalisa gambar dan skor probabilitas pengembalian kemungkinan bahwa gambar mengandung tag rumah dan arsitektur lainnya yang berhubungan dengan gambar [10]. Berikut adalah gambar 2.3 contoh tag yang didapatkan oleh clarifai menggunakan model general.



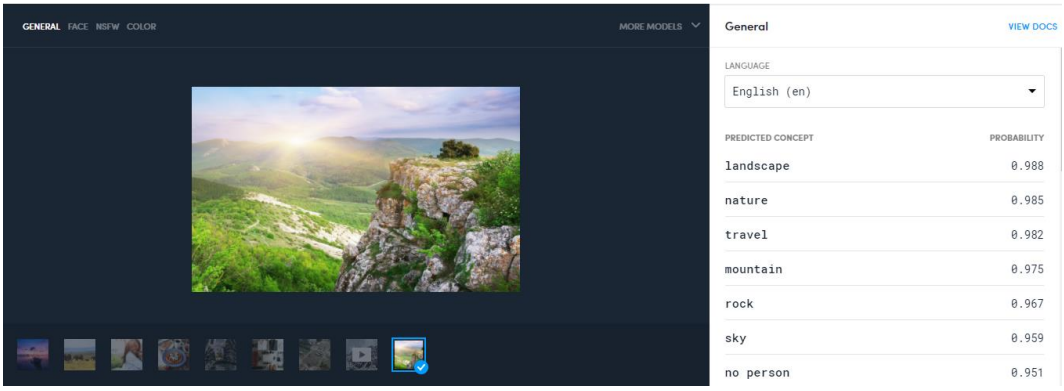
Gambar 2.2 Tag Clarifai

Ketika gambar atau video dijalankan mereka ditandai menggunakan model. Sebuah model adalah classifier yang terlatih yang dapat mengenali apa yang ada di dalam gambar atau video sesuai dengan apa yang diketahui . model yang berbeda

digunakan untuk mengetahui hal-hal yang berbeda. Menjalankan gambar atau video melalui model yang berbeda dapat menghasilkan hasil yang berbeda secara drastis [10]. Model clarifai yang sudah dikembangkan sebanyak 16 model yaitu, *general, face, nsfw, color, wedding, travel, celebrity, demographics, logo, food, apparel, focus, moderation, portrait quality, landscape quality, textures* dan *pattern.*

1. General Model

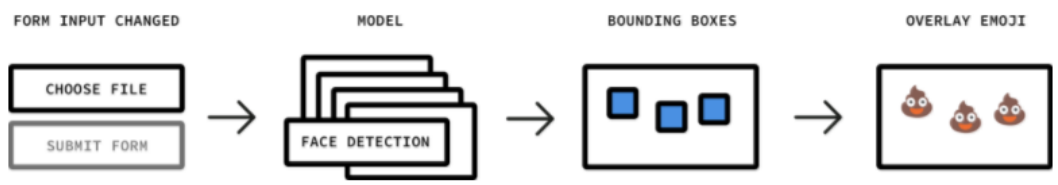
General Model berisi berbagai macam tag di berbagai topik yang berbeda. Dalam kebanyakan kasus, tag kembali dari model umum cukup akan mengenali apa yang ada di dalam sebuah objek. Berikut adalah contoh model general pada gambar 2.3.



Gambar 2.3 Model Clarifai General

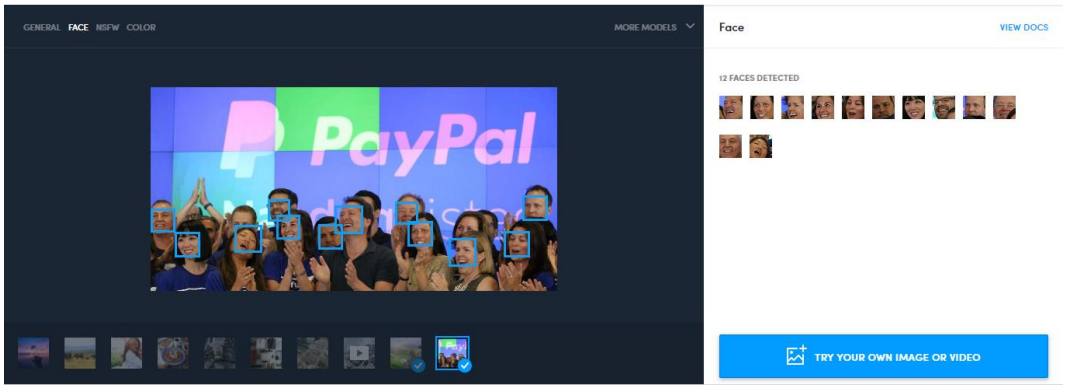
2. Face Detection Model

Face Detection pada clarifai akan mendeteksi wajah pada suatu gambar, clarifai akan menunggu gambar yang akan dipilih, dan kemudian gunakan klien clarifai JavaScript untuk mendapatkan posisi wajah yang ditemukannya. Akhirnya, clarifai akan mengambil nilai-nilai itu, menghubungkannya langsung dengan gambar. Berikut adalah proses dari face detection pada gambar 2.4 [11].



Gambar 2.4 Proses Face Detection Clarifai

Face Detction Model berisi berbagai macam tag di pada sebuah gambar. Clarifai akan mendeteksi dan mengenali sebuah objek wajah. Berikut adalah contoh model face detection pada gambar 2.5.

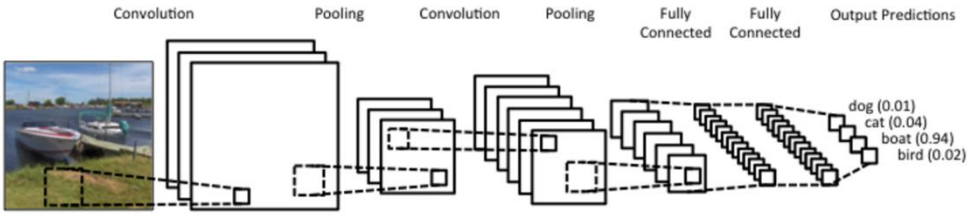


Gambar 2.5 Model Clarifai Face Detection

2.9.2 Cara kerja Clarifai API

Teknologi Clarifai bergantung pada penggunaan convolutional neural network, untuk memproses gambar, dan kemudian menghasilkan daftar tag gambar. Didefinisikan sebagai model pembelajaran mesin hirarkis, yang belajar pada gambar yang kompleks representasi dari volume data yang besar.

Clarifai bekerja melalui analisis gambar untuk menghasilkan daftar deskripsi tag dari gambar yang diberikan. Untuk setiap tag dalam daftar ini, system juga menyediakan nilai probabilitas. Nilai probabilitas ini merupakan kemungkinan menggambarkan gambar menggunakan tag yang spesifik. The clarifai API dapat diakses sebagai layanan web jarak jauh [11]. Skema kerja teknologi clarifai ditunjukan pada gambar 2.6.



Gambar 2 6 Skema Proses Clarifai API

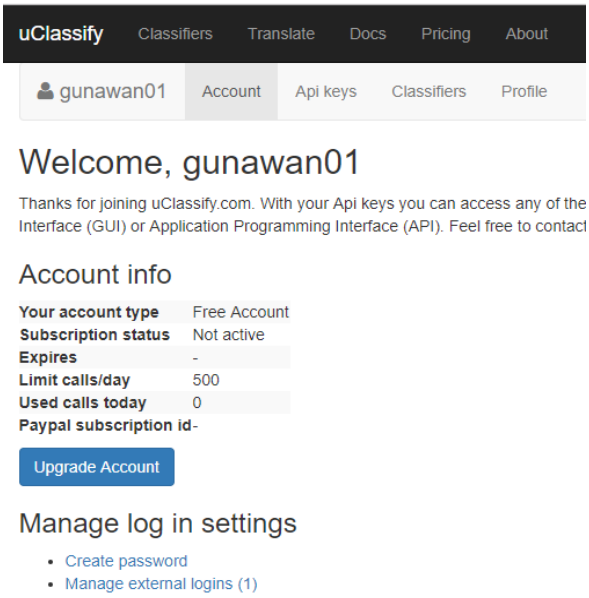
Clarifai API dapat diakses menggunakan pilihan trial atau digunakan di bawah pembayaran mode. Dalam karya ini, mode percobaan telah digunakan untuk mendapatkan tag gambar. Mode ini memungkinkan untuk mendapatkan tag untuk sejumlah terbatas gambar (10000), dan membatasi jumlah panggilan per jam untuk 1000. Menggunakan mode percobaan, kita hanya bisa memperoleh 20 tag per gambar dengan probabilitas yang terkait [11].

2.10 Clarifai API

Uclassify adalah layanan API yang melakukan banyak fungsi seperti deteksi Bahasa, teks gender dan mendeteksi usia, spam filter, analisis sentimen, dokumen tagging dan dapat mendeteksi emosi. Berikut adalah demo dari uclassify disini bisa membuat classifier baru. Akan disediakan sebuah class yaitu untuk menentukan kata atau kalimat bersifat negatif atau positif. Dengan cara melakukan pelatihan pada data atau kata yang akan digunakan dalam classifier baru [12]. API Uclassify dapat diakses dengan pilihan gratis dan berbayar. Uclassify telah banyak dikembangkan dalam analisis sentimen terutama pada media sosial. Berikut adalah proses pembuatan classifier baru pada WEB API Uclassify [12].

1. Pembuatan Clasifier Baru

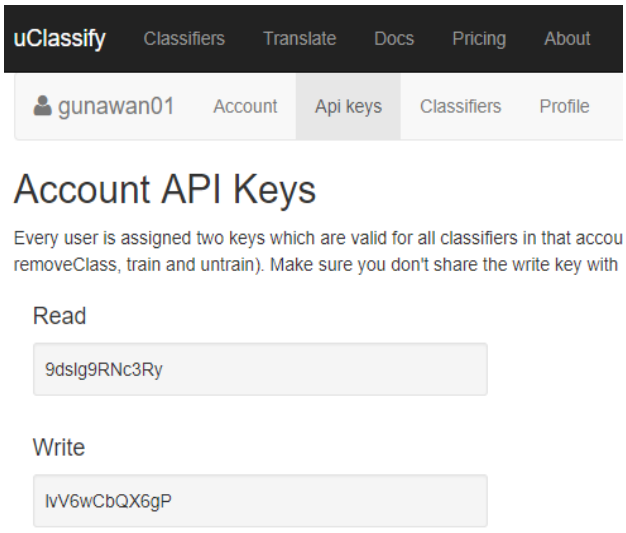
   Berikut adalah halaman utama Uclassify.



Gambar 2.7 Halaman Utama Uclassify

2. API Key Uclassify

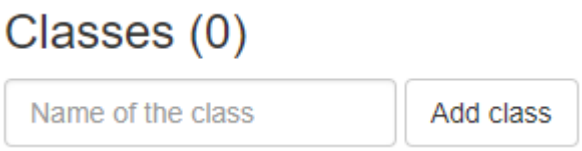
Berikut adalah API key yang ada pada uclassify, api key ini dapat digunakan untuk memanggil api dari uclassify.



Gambar 2.8 API key Uclassify

3. Tentukan Kelas Yang Relevan

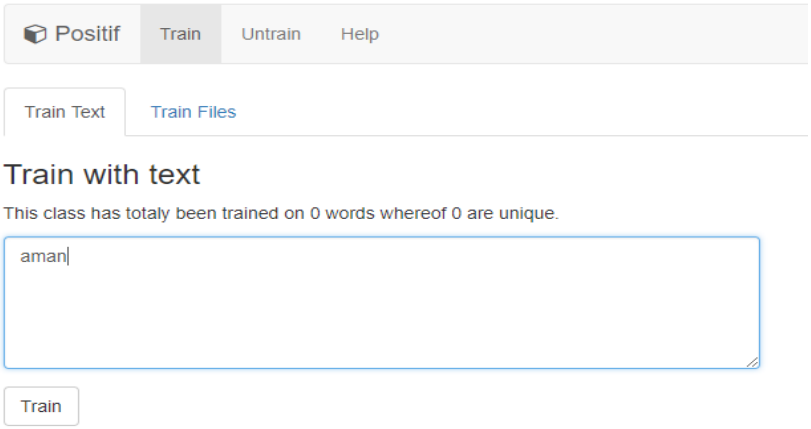
Untuk membuat classifier baru, harus menentukan terlebih dahulu kelas untuk classifier. Hal ini digunakan agar klasifikasi bisa membedakan antara dua atau lebih kelas untuk membantu proses klasifikasi. Berikut adalah interface dari pembuatan kelas pada WEB API Uclassify.



Gambar 2.9 Pembuatan Kelas Baru

4. Buat Data Latih

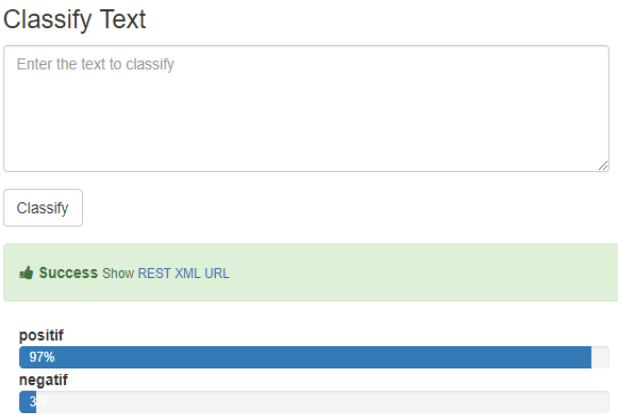
   Sebelum pengklasifikasian bisa dimulai mengkategorikan teks ke dalam kelas, selenjutnya perlu membuat data latih yang akan digunakan pada proses klasifikasi.



Gambar 2.10 Membuat Data Latih

5. Pengklasifikasian

   Setelah membuat data latih, maka setelah itu classifier dapat digunakan untuk melakukan klasifikasi teks.



Gambar 2 11 Pengklasifikasian

2.10.1 Mengirim Permintaan Uclassify API

Setelah membuat classifier, selanjutnya classifier tersebut bisa dipanggil menggunakan permintaan HTTP dengan bentuk sebagai berikut.

```
https://api.uclassify.com/v1/{username}/{classifierName}/classify
```

2.10.2 Data Latih

Proses klasifikasi pada Uclassify adalah dengan menggunakan data latih berupa kata yang memiliki kelas negatif dan positif. Kata-kata tersebut nantinya akan digunakan untuk melakukan analisis sentimen terhadap kondisi lingkungan suatu kawasan melalui tweet publik pada twitter. Berikut adalah data latih yang digunakan dalam penelitian ini.

Tabel 2 3 Data Latih

Negatif	Positif
Begal	Aman
Awas	Lancar
Rawan	Ramai
Pencuri	Rekomended
Jambret	Bagus
Bgal	Indah
Begl	Nyaman
Hati-hati	Nyman

Jmbret	Bgus
Macet	Sesuai
Mcet	Tidak ada
Parah	Polisi
Preman	Meringkus
Geng motor	Menangkap
Curanmor	Menahan
Pencurian	Pelaku
Pembegalan	Akhirnya
Penjambretan	Alhamdulillah
Pembunuhan	Selamat
Kriminalitas	Asik
Kriminal	Sudah aman
Takut	Aman dari begal
Waspada	Tidak macet
Bahaya	Diringkus
Bhaya	Ditahan
Banjir	Ditangkap
Korban	Sejuk
Sudah tidak aman	Pembersihan
Pembegal	Dekat
Longsor	Pengamanan
Kejahatan	Teratur

2.10.3 Respon Klasifikasi

Respon klasifikasi terdiri dari serangkaian klasifikasi. Setiap klasifikasi sesuai dengan teks dalam permintaan. Klasifikasi tercantum dalam urutan yang sama seperti teks, yang berarti klasifikasi untuk teks pertama adalah elemen pertama dalam array [12].

Tabel 2 4 Respon Klasifikasi

Nama	Tipe	Deskripsi
textCoverage	Double	Proposisi teks rahasia yang ditemukan dalam data pelatihan. 1 berarti setiap kata digunakan, 0 bahwa tidak ada kata-kata yang

		digunakan. Berguna untuk mengetahui relevansi klasifikasi.
Classification	Array	Array kelas dan probabilitas masing-masing.
className	RestrictedString	Nama kelas
P	double	Probabilitas bahwa teks itu termasuk dalam kelas ini [0-1].

2.11 GPS (Global Positioning System)

Global Positioning System (GPS) merupakan sistem navigasi yang menggunakan satelit dalam penggunaannya. GPS terdiri dari tiga bagian yaitu system control, satelit dan pengguna. Sistem kontrol adalah bagian yang mengontrol pergerakan pada satelit yang ada dan saling berinteraksi satu sam lain kemudian pengguna adalah alat navigasi yang digunakan seperti perangkat mobile yang kini sudah memiliki fitur GPS di dalamnya. GPS biasanya digunakan untuk menunjukan suatu lokasi yang berada di permukaan bumi dengan tingkat akurasi yang cukup baik yaitu kurang dari 10 meter selama tidak ada benda padat yang dapat menghambat sinyal untuk mendapatkan lokasi pengguna [13].

2.12 Google Direction API

Google Direction API adalah layanan yang diperuntukan untuk menghitung arah antara lokasi menggunakan permintaan HTTP. Pengguna dapat mencari petunjuk arah untuk beberapa mode transportasi seperti transit, mengemudi, berjalan atau bersepeda [14]. Layanan ini dapat menentukan asal-usul, tujuan dan waypoints baik sebagai string teks (misalnya Chicago, IL atau Darwin, NT, Australia) atau sebagai lintang atau bujur. Layanan ini juga dapat mengembalikan banyak bagian petunjuk arah menggunakan serangkaian titik. Layanan ini umumnya dirancang untuk mencari petunjuk arah alamat statis yang digunakan untuk penempatan konten aplikasi pada peta dan tidak dirancang untuk merespon secara real-time untuk input pengguna. Perhitungan arah adalah tugas intensif yang menghabiskan banyak waktu dan sumber daya. Bila memungkinkan, sangat disarankan untuk menggunakan perhitungan alamat yang dikenal sebelumnya dan menyimpan hasil perhitungan tersebut di cache sementara [14].

2.13 Google Direction API

Google Places API memungkinkan pengguna untuk mengirmkan query untuk informasi tempat pada berbagai kategori, seperti: perusahaan, poin penting yang menarik, lokasi geografis, dan banyak lagi. Anda dapat mencari tempat baik oleh kedekatan atau string teks. Tempat Cari mengembalikan daftar tempat bersama dengan ringkasan informasi tentang masing-masing tempat; Informasi tambahan tersedia melalui query tempat detail [15].

2.13.1 Permintaan Pencarian Tempat Terdekat

Pencarian terdekat memungkinkan pengguna mencari tempat dalam area tertentu. Pengguna dapat menyesuaikan permintaan pencarian dengan kata kunci atau menentukan jenis tempat yang dicari. Permintaan pencarian terdekat merupakan URL HTTP dengan bentuk berikut [15].

```
https://maps.googleapis.com/maps/api/place/nearbysearch/output?parameters
```

Keterangan dari bentuk tersebut pada output adalah salah satu dari nilai json (dianjurkan) menunjukkan output dalam JavaScript Object Notation (JSON) atau xml yang menunjukkan output XML.

2.13.2 Parameter Permintaan Google Place API

Parameter tertentu yang diperlukan sementara yang lain adalah opsional. Seperti standar di URL, semua parameter dipisahkan menggunakan ampersand (&) karakter.

Tabel 2.5 Parameter Permintaan Google Place API

Nama Parameter	Keterangan	Jenis
Key	API Key ini mengidentifikasi aplikasi untuk keperluan manajemen kuota, sehingga tempat yang ditambahkan	utama

	pada aplikasi yang dibuat segera tersedia untuk aplikasi.	
Location	Lintang atau bujur sekitar yang untuk mengambil informasi tempat. Parameter ini harus ditetapkan sebagai lintang, bujur.	Utama
Radius	Mendefinisikan jarak (dalam meter) di mana untuk menghasilkan tempat. Jarak maksimum yang diizinkan adalah 50.000 meter.	Utama
Keyword	Sebuah istilah yang akan dicocokkan terhadap semua konten yang terindeks oleh Google untuk tempat, tidak terbatas pada nama, jenis, dan alamat, serta ulasan pelanggan dan konten pihak ketiga lainnya.	Tambahan
Language	Kode bahasa, menunjukkan di mana bahasa adalah nilai yang harus dikembalikan, jika memungkinkan.	Tambahan
minprice dan max price	Membatasi hasil hanya tempat-tempat dalam kisaran tertentu. Nilai valid berkisar antara 0 (paling terjangkau) ke 4 (paling mahal), inklusif. Jumlah Tambahan 39 yang tepat ditunjukkan oleh nilai tertentu akan bervariasi dari daerah ke daerah.	Tambahan
Name	Satu atau lebih hal yang akan dicocokkan dengan nama-nama tempat, dipisahkan dengan karakter spasi. Hasil akan dibatasi untuk yang mengandung nilai-nilai nama. Perhatikan bahwa tempat	tambahan

	mungkin memiliki nama tambahan yang terkait di luar nama yang tercantum. API akan mencoba untuk mencocokkan nilai nama yang cocok terhadap semua nama-nama tempat.	
Opennow	Pengembalian hanya tempat-tempat yang terbuka untuk bisnis pada saat permintaan dikirim. Tempat yang tidak menentukan jam buka dalam database Google Places tidak akan dikembalikan.	Tambahan
Types	Membatasi hasil ke tempat-tempat yang cocok setidaknya salah satu jenis tertentu. Jenis harus dipisahkan dengan simbol pipa (type1 | type2 | …).	Tambahan
Pagetoken	Mengembalikan 20 hasil berikutnya dari pencarian yang dijalankan sebelumnya. Mengatur pagetoken parameter akan mengeksekusi Tambahan 40 pencarian dengan parameter yang sama yang digunakan sebelumnya. Semua parameter selain pagetoken lainnya akan diabaikan.	Tambahan

2.14 Google Maps API

Google Maps API adalah sebuah layanan pada peta virtual gratis dan online yang disediakan oleh Google. Layanan ini dapat diakses di *http://maps.google.com.* Google Maps menawarkan peta diseluruh dunia, dan juga menawarkan fitur perencanaan rute dan pencari lokasi bisnis. API (Apllication Programing Interface) adalah antarmuka yang digunakan komponen software untuk saling berhubungan,

sehingga bisa dikatakan Google Maps API itu berisi code atau script library untuk memanfaatkan fungsi Google Maps terutama pada halaman web yang kita buat apalagi berfungsi sebagai web komersil dan Google Maps sebagai fitur tambahan. Tujuan dari penggunaan Google Maps API adalah untuk melihat lokasi, mencari alamat, dan mendapatkan petunjuk mengemudi [16].

2.15 Twitter

Twitter merupakan jejaring sosial berkategori microblog yang memungkinkan penggunanya untuk mengirim dan membaca pesan berbasis teks dengan maksimum 140 karakter.

2.16 Geotagging

Geotagging adalah proses penambahan identifikasi metadata geografis terhadap berbagai media contohnya, geotagged photografi, atau video, websites, pesan SMS, QR code, atau RSS feed dan merupakan bentuk dari metada geospatial. Data tersebut biasanya terdiri dari koordinat garis lintang (latitude) dan garis bujur (longitude), dan mungkin juga termasuk ketinggian (altitude), bantalan poros (bearing), jarak, data akurasi, dan nama tempat.

Geotagging dapat membantu pengguna untuk menemukan informasi posisi keberadaan spesifik dari lingkungan yang luas. Contohnya, seseorang dapat menemukan gambar terdekat dengan memberikan informasi koordinat latitude dan longitude ke dalam mesin pencari gambar yang sesuai. Geotagging dapat memberikan informasi layanan yang potensial digunakan untuk menemukan lokasi berdasarkan berita, website, dan sumber lainnya. Geotagging dapat memberitahukan kepada pengguna lokasi dari konten dari gambar atau media lainnya atau titik pandangan (point of view), dan sebaliknya pada beberapa platform media menunjukkan media yang relevan ke lokasi tertentu[16].

2.17 JSON

JSON adalah bagian dari bahasa pemrograman JavaScript (Standard ECMA-262 3rd Edition – December 1999). JSON merupakan format teks yang

sepenuhnya independen tetapi menggunakan konvensi yang familiar dengan bahasa pemrograman dari keluarga-C, termasuk C, C++, C#, Java, JavaScript, Perl, Python, dan sebagainya. Kelebihan inilah yang membuat JSON menjadi sebuah bahasa data-interchange yang ideal [17]. JSON dibangun kedalam dua struktur yaitu :

1. Beberapa pasangan dari nama/nilai. Dalam beberapa bahasa pemrograman biasa disebut dengan istilah object, record, struct, tabel hash, key list atau associative array.
2. Nilai-nilai yang terusun secara ordered list. Biasa disebut dengan array, vector, list atau daftar dalam bahasa pemrograman.

   JSON adalah struktur data yang universal, dalam artian bisa digunakan dalam berbagai bahasa pemrograman. Hampir semua bahasa pemrograman mendukung penuh JSON dalam berbagai format. Hal ini memungkinkan format data yang dapat dipertukarkan menggunakan bahasa pemrograman juga menggunakan dasar dari struktur JSON.

Struktur-struktur data ini disebut sebagai struktur data universal. Pada dasarnya, semua bahasa pemprograman modern mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemrograman yang juga berdasarkan pada struktur data ini. JSON di gunakan sebagai format data dalam pertukaran informasi dari aplikasi menuju web service dan juga sebaliknya dalam aplikasi yang akan di bangun.

2.18 Bentuk 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 Pemrograman *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 [17].

JSON terbuat dari dua struktur, 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*. Daftar nilai terurutkan (*an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (*array*), vektor (*vector*), daftar (*list*), atau urutan (*sequence*).

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 [17].

JSON menggunakan bentuk sebagai berikut:

1. **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).
2. **Larik** adalah kumpulan nilai yang terurutkan. Larik dimulai dengan [(kurung kotak buka) dan diakhiri dengan] (kurung kotak tutup). Setiap nilai dipisahkan oleh , (koma).
3. **Nilai** (*value*) dapat berupa sebuah **string** dalam tanda kutip ganda, atau *angka*, atau true atau false atau null, atau sebuah *objek* atau sebuah *larik*. Struktur-struktur tersebut dapat disusun bertingkat.
4. **String** adalah kumpulan dari nol atau lebih karakter Unicode, yang dibungkus dengan tanda kutip ganda. Di dalam string dapat digunakan *backslash escapes* "\" untuk membentuk karakter khusus. Sebuah karakter mewakili karakter tunggal pada string. String sangat mirip dengan string C atau Java.
5. **Angka** adalah sangat mirip dengan angka di C atau Java, kecuali format oktal dan heksadesimal tidak digunakan.

2.19 Android Studio

Android Studio adalah Integrated Development Environment resmi untuk pengembangan aplikasi Android dan bersigat open source atau gratis. Peluncuran Android Studio ini diumumkan oleh Google pada 16 Mei 2013 pada event Google I/O Conference untuk tahun 2013. Sejak saat itu, Android Studio menggantikan Eclipse sebagai IDE resmi untuk mengembangkan aplikasi Android. Android Studio digunakan pada pembangunan aplikasi ini di karenakan IDE resmi untuk pembangunan aplikasi android [6].

2.20 MySQL

MySQL adalah sebuah perangkat lunak system manajemen basis data SQL (DBMS) yang multithread, dan multi-user. MySQL adalah implementasi dari system manajemen basisdata relasional (RDBMS). MySQL dibuah oleh TcX dan telah dipercaya mengelola system dengan 40 buah database berisi 10.000 tabel dan 500 di antaranya memiliki 7 juta baris[19].

2.21 UML (Unified Modelling Language)

Pemodelan yang diterapkan pada penelitian ini yaitu *Unified Modeling Languange* (UML). Unified Modeling Language (UML) merupakan bahasa yang telah menjadi standar dalam industri untuk visualisasi, merancang, dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. UML adalah bahsa grafis untuk mendokumentasi, menspesifikasikan, dan membangun sistem perangkat lunak. UML berorientasi objek, menerapkan banyak lebel abstraksi, tidak bergantung proses pengembangan, tidak bergantung bahasa dan teknologi. Dalam pembangunan perangkat lunak perlu merancang, menggambarkan sistem perangkat lunak sehingga UML di perlukan dalam pembangunan perangkat lunak. Diagram UML yang akan di gunakan dalam memvisualisasikan perangkat lunak yang akan di bangun antara lain : *use case diagram, class diagram, activity diagram, sequence diagram* [18].

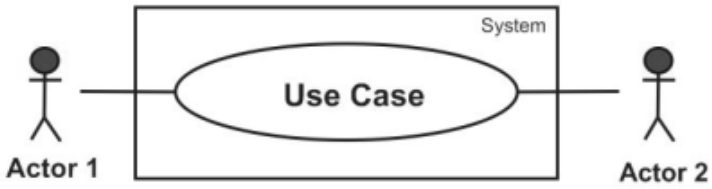
2.21.1 Object Oriented Programming

Object Oriented Programming merupakan paradigma pemrograman yang memandang perangkat lunak sebagai kumpulan objek yang saling berinteraksi di dalam suatu sistem[21]. Tujuan dari OOP sendiri adalah untuk mempermudah pengembangan perangkat lunak dengan cara membangun gambaran sistem dari perangkat lunak yang akan dibangun.

1. Use Case Diagram

Use Case Diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *Use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. Sebuah *use case* dapat meng-*include* fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa *use case* yang di-*include* akan dipanggil setiap kali *use case* yang meng-*include* dieksekusi secara normal. Sebuah *use case* dapat di- include oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang serupa. Sebuah *use case* juga dapat meng-*extend usecase* lain dengan *behaviour*-nya sendiri. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain.

Dasar menentukan sebuah *use case* adalah *use case* merupakan sesuatu yang menyediakan beberapa hasil terukur kepada pengguna atau sistem eksternal. *Use case* harus memiliki sangat jelas kriteria lulus / gagal. Pengembang, *tester*, penulis teknis, dan pengguna harus secara eksplisit tahu apakah sistem memenuhi kasus penggunaan atau tidak. Setiap bagian dari *use case* yang memenuhi tes sederhana ini mungkin menjadi kandidat yang baik untuk *use case* . Berikut adalah contoh dari *use case diagram* dapat dilihat pada 2.12 [21].



Gambar 2.1 Use Case Diagram

1. Aktor

   Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem yang akan dibuat di luar sistem yang akan dibuat itu sendiri, walaupin dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.

2. *Use Case*

   Use case merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang yang saling bertukar pesan antar unit ataupun aktor. Berikut adalah simbol-simbol yang ada pada diagram use case

   Dalam sistem yang dibangun use case digunakan untuk mendekripsikan interaksi antara aktor dan mendeskripsikan fungsi-fungsi yang ada dalam sistem.

2. Use Case Skenario

Use Case Scenario adalah 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 case* nya. Berikut adalah dasar format penulisan use case scenario [21]. Dasar pembangunan use case scenario dapat dilihat pada tabel 2.2.

Tabel 2.6 Dasar Pengembangan Use Case Skenario

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	
Extension	Step	Branching Action
	2.1	Deskripsi urutan aksi lain selain urutan aksi utama

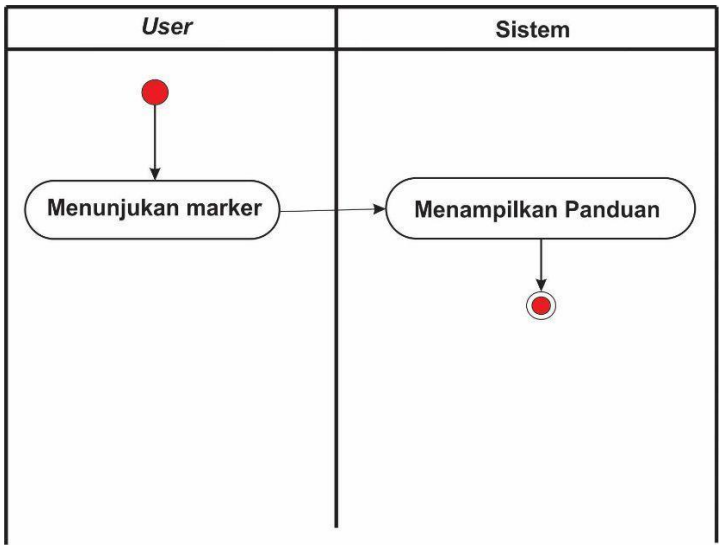
3. Activity Diagram

Activity diagram menggambarkan berbagai alur aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing aliran berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity* diagram juga dapat menggambarkan proses paralel yang mungkin terjdi pada beberapa eksekusi. Sebuah aktivitas dapat direalisasikan oleh satu *usecase* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *usecase* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas. Standar UML menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas.

Decision digunakan untuk menggambarkan *behaviour* pada kondisi tertentu. untuk mengilustrasikan proses-proses paralel (*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal [21]. *Activity* diagram dapat dibagi menjadi beberapa *object swimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu.Diagram aktivitas banyak digunakan untuk mendefinisikan hal-hal:

1. Rancangan proses bisnis dimana setiap utrutan aktivitas yang digambarkan merupakan proses bisnis sistem yang akan didefinisikan

2. Urutan atau pengelompokan tampilan dari sistem dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan [21].



Gambar 2.2 *Activity Diagram*

4. Class Diagram

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. Class menggambarkan keadaan (atribut/properti) satu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi) [21]. Class diagram menggambarkan struktur dan deskripsi *class*, *package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain.

Class memiliki tiga area pokok:

1. Nama dan Stereotype
2. Atribut
3. Metode

Atribut dan metode dapat memiliki salah satu sifat berikut:

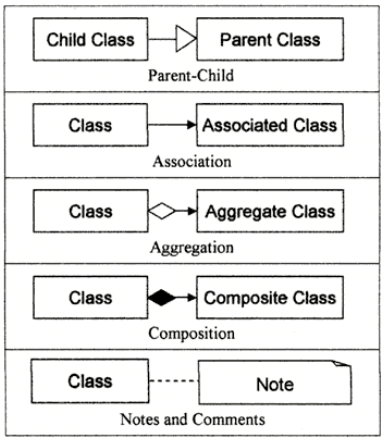
1. *Private*, tidak dapat dipanggil dari luar *class* yang bersangkutan.
2. *Protected*, hanya dapat dipanggil oleh *class* yang bersangkutan dan anak-anak yang mewarisinya.

3. *Public*, dapat dipanggil oleh siapa saja.

Class dapat merupakan implementasi dari sebuah *interface*, yaitu *class* abstrak yang hanya memiliki metode. *Interface* tidak dapat langsung diinstansiasikan, tetapi harus diimplementasikan dahulu menjadi sebuah *class*. Sesuai dengan perkembangan *class* model, *class* dapat dikelompokkan menjadi *package*. Kita juga dapat membuat diagram yang terdiri atas *package* [21].

Class memiliki tipe-tipe *relationship*, diantaranya:

1. Asosiasi, yaitu hubungan statis antar *class*. Umumnya menggambarkan *class* yang memiliki atribut berupa *class* lain, atau *class* yang harus mengetahui eksistensi *class* lain.
2. Agregasi, yaitu hubungan yang menyatakan bagian terdiri atas dimana ketika satu *class* di-*share* atau direferensikan kepada objek yang ada di *class* lain.
3. Pewarisan, yaitu hubungan hirarkis antar *class*. Class dapat diturunkan dari *class* lain dan mewarisi semua atribut dan metoda *class* asalnya dan menambahkan fungsionalitas baru, sehingga ia disebut anak dari *class* yang diwarisinya. Kebalikan dari pewarisan adalah generalisasi.
4. Komposisi, yaitu jenis relasi *class diagram* yang kuat dimana jika sebuah *class* tidak bisa berdiri sendiri dan harus merupakan bagian dari *class* yang lain, maka *class* tersebut memiliki relasi *Composition* terhadap *class* tempat dia bergantung tersebut. Sebuah *relationship composition* digambarkan sebagai garis dengan ujung berbentuk jajaran genjang berisi/solid.
5. Depedensi, salah satu jenis relasi *class* diagram yang lemah dimana objek dalam suatu *class* akan bekerja sangat singkat dengan objek yang ada pada *class* lain.

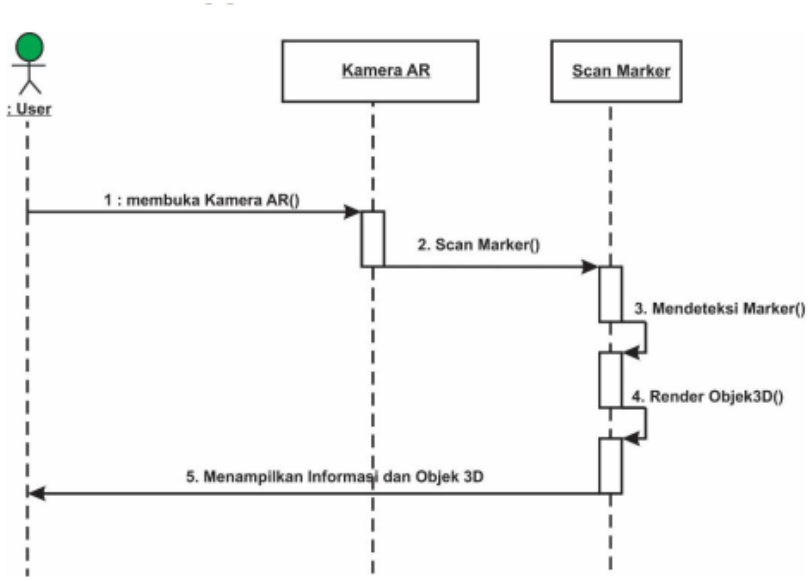


Gambar 2.12 Dasar Pembangunan Class Diagram

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. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan.

Masing-masing objek, termasuk aktor, memiliki *lifeline* vertikal. *Message* digambarkan sebagai garis berpanah dari satu objek ke objek lainnya. Pada fase desain berikutnya, *message* akan dipetakan menjadi operasi/metoda dari *class*. *Activation bar* menunjukkan lamanya eksekusi sebuah proses, biasanya diawali dengan diterimanya sebuah *message* [21]. Untuk objek-objek yang memiliki sifat khusus, standar UML mendefinisikan icon khusus untuk objek *boundary*, *controller* dan *persistent entity*. Dasar pembangunan *sequence diagram* dapat dilihat pada pada gambar 2.3.



Gambar 2.13 Contoh Sequence Diagram

2.11 Metode Pengujian Sistem

Metode pengujian sistem bertujuan untuk mengetahui efektifitas dari aplikasi yang dibangun dengan memberikan kesempatan kepada pengguna untuk mengoperasikan dan melakukan pengecekan terhadap aplikasi. Metode pengujian sistem terdiri dari Pengujian *White-box* dan Pengujian *black-box.*

2.11.1 Pengujian *Black-Box*

Black-box testing atau pengujian kotak hitam yaitu menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian dimaksudkan untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan. Pengujian *Black-box* dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak apakah sesuai dengan spesifikasi yang dibutuhkan. Kasus uji yang dibuat untuk melakukan pengujian *Black-box* harus dibuat dengan kasus benar dan kasus salah [22].

Pengujian *black-box* berusaha untuk menemukan kesalahan dalam beberapa kategori, diantaranya:

1. fungsi-fungsi yang salah atau hilang,

2. kesalahan *interface*,
3. kesalahan dalam struktur data atau akses *database* eksternal,
4. kesalahan performa,
5. kesalahan inisialisasi dan terminasi.

2.11.2 Pengujian Beta

Pengujian beta adalah pengujian aplikasi terhadap pengguna untuk memvalidasi kegunaan, fungsi, kompatibilitas, dan uji reliabilitas dari aplikasi yang dibuat dilakukan secara objektif. Pengujian beta yang dilakukan yaitu untuk mengetahui kepuasan pengguna dengan kandungan *point* syarat *user friendly* untuk selanjutnya disebarkan kepada pengguna. Tujuan dari pengujian beta adalah untuk menempatkan aplikasi di tangan pengguna yang sebenarnya untuk menemukan setiap kekurangan dari perspektif pengguna akhir. Keuntungan pengujian beta adalah sebagai berikut :

1. Pengguna dapat menguji aplikasi dan mengirim umpan balik kepada pengembang selama periode pengujian *beta*.
2. Penguji beta dapat menemukan masalah aplikasi yang tidak disadari, seperti aliran aplikasi membingungkan, dan bahkan *crash*.
3. Dapat memperbaiki masalah menggunakan umpan balik yang didapatkan dari pengguna.

2.11.3 Skala Data Yang Digunakan

Skala data yang digunakan untuk mengukur variabel yang didapat, menggunakan skala likert. Skala likert adalah skala respon psikometri terutama digunakan dalam kuesioner untuk mendapatkan preferensi responden atas sebuah pernyataan atau serangkaian laporan[20]. Setelah menyelesaikan definisi operasional variabel maka langkah selanjutnya menyusun item-item. Sebuah skala menjadi penting untuk mengukur derajat pendapat dan data kuantitatif berarti analisis relatif mudah dilakukan. Prinsip pengukuran sikap yaitu meminta orang untuk menanggapi serangkaian pernyataan tentang suatu topik. Sejauh mana mereka setuju dengan memasuki komponen kognitif dan afektif. Data yang telah

terkumpul melalui angket, kemudian diolah kedalam bentuk kuantitatif, yaitu dengan sebuah kontinum dari sangat setuju sampai sangat tidak setuju yang dapat dilihat pada Tabel 2.7.

Tabel 2.7 Penilaian Skala Likert

Jawaban	Skor
Sangat Setuju	5
Setuju	4
Ragu-Ragu	3
Tidak Setuju	2
Sangat Tidak Setuju	1

Kemudian dengan teknik pengumpulan data kuisioner, maka instrument tersebut misalnya diberikan kepada 30 orang yang diambil secara random. Dari 30 orang tersebut setelah dilakukan analisis didapat hasil sebagai berikut:

1. Sebanyak 15 orang menjawab Sangat Setuju (SS)
2. Sebanyak 10 orang menjawab Setuju (S)
3. Sebanyak 2 orang menjawab Ragu-Ragu (RR)
4. Sebanyak 2 orang menjawab Tidak Setuju (TS)
5. Sebanyak 1 orang menjawab Sangat Tidak Setuju (STS)

Berdasarkan data tersebut 25 orang (15 + 10) atau 83.3% *stakeholder* menjawab setuju dan sangat setuju.

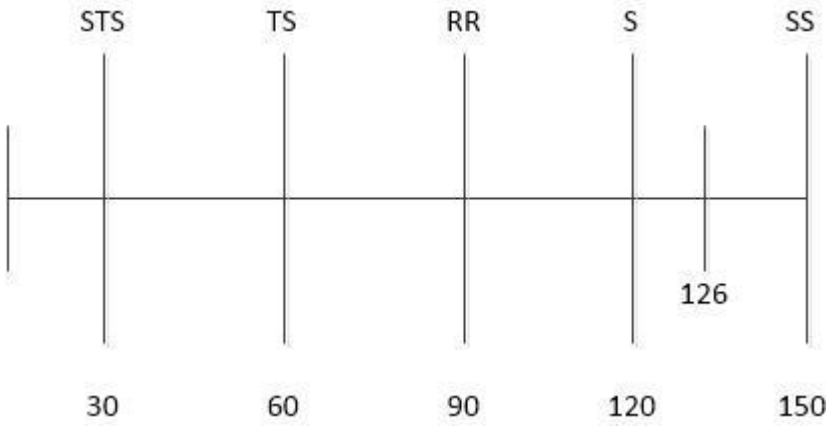
Data Interval tersebut akan dianalisis dengan cara menghitung rata-rata jawaban berdasarkan *skor* setiap jawaban dari responden. Berdasarkan skor yang telah didapat dari analisis sebelumnya dapat dihitung sebagai berikut:

Jumlah skor untuk 15 orang yang menjawab SS = 15x5 = 75
Jumlah skor untuk 10 orang yang menjawab S = 10x4 = 40
Jumlah skor untuk 2 orang yang menjawab RR = 2x3 = 6
Jumlah skor untuk 2 orang yang menjawab TS = 2x2 = 4
Jumlah skor untuk 1 orang yang menjawab STS = 1x1 = 1

Jumlah Total Nilai = 126

Jumlah ideal (kriterium untuk seluruh item = 5x30 = 150 (seandainya semua menjawab SS). Jumlah skor yang diperoleh dari penelitian = 126. Jadi berdasarkan data itu maka tingkat persetujuan stakeholder terhadap penggunaan media pembelajaran interaktif = (126:150) x 100% = 84% dari yang diharapkan (100%).

Secara kontinum dapat dilihat seperti pada Gambar 2.3.



Gambar 2.3 Pengolahan Hasil Kuesioner Secara Kontinum

Jadi berdasarkan data yang diperoleh dari 30 responden maka rata-rata 126 terletak pada daerah mendekati setuju.

Alasan mengapa menggunakan skala likert untuk mengukur variabel adalah sebagai berikut:

1. Skala likert memudahkan responden untuk menjawab kuesioner apakah setuju atau tidak setuju.
2. Skala likert mudah digunakan dan mudah dipahami oleh responden.
3. Skala likert secara visual lebih menarik dan mudah diisi oleh responden