BAB 2
TINJAUAN PUSTAKA

2.1. Profil Perusahaan

Profil CV. Mandiri Expres meliputi logo perusahaan, sejarah berdirinya, serta struktur organisasi.

2.1.1. Logo Perusahaan

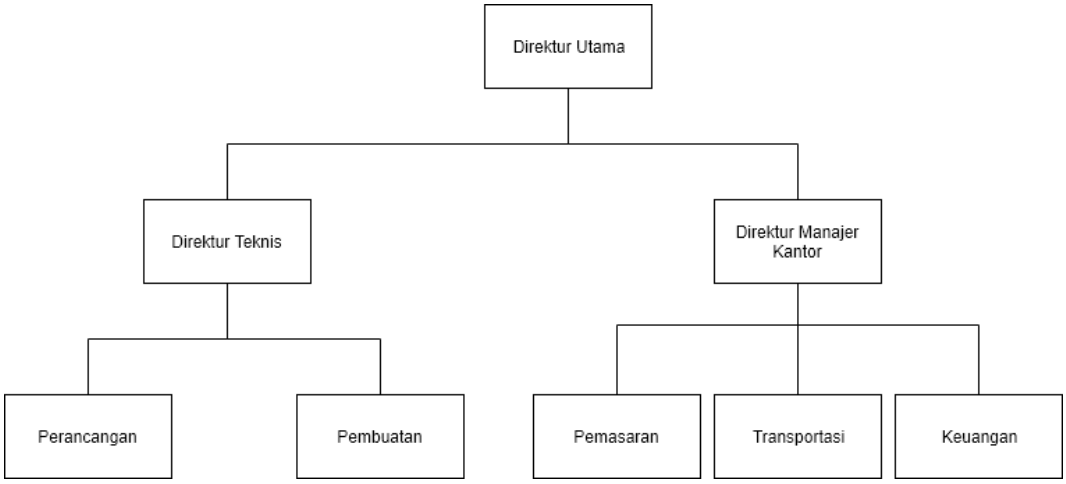
Gambar 2.1 Logo Perusahaan

2.1.2. Sejarah Perusahaan

CV. Mandiri Expres merupakan perusahaan yang bergerak di bidang Jasa Pembuatan dan Desain Maket Miniatur, bertempat di Bandung tepatnya di Jalan Saluyu B VII No. 14 RT.002 RW.009, Kelurahan Cipamokolan, Kecamatan Rancasari, Kota Bandung. CV. Mandiri Expres membuat suatu maket konstruksi bangunan seperti kantor dan perusahaan secara detail. Perusahaan ini mulai didirikan oleh Haedar Ambari pada tahun 2009.

2.1.3. Struktur Organisasi

Struktur organisasi merupakan suatu kerangka dalam manajemen perusahaan agar suatu perusahaan dapat berjalan sesuai dengan tujuan berdirinya perusahaan. Stuktur organisasi memuat tatanan kerja yang ada di perusahaan yang menyangkut tugas, tanggung jawab, dan wewenang. Struktur organisasi di perusahaan CV. Mandiri Expres dapat dilihat pada Gambar 2.2.



Gambar 2.2 Struktur Organisasi CV. Mandiri Expres

Adapun uraian tugas pada struktur organisasi CV. Mandiri Expres sebagai berikut :

1. Direktur Utama

   Mewakili dan membuat perjanjian-perjanjian dan kontrak kerja sama dengan pihak luar, seperti perusahaan atau pemerintah.
2. Direktur Teknis

   Mengawasi kinerja dan memutuskan kebijakan pada bagian perancangan dan pembuatan maket.
3. Direktur Manajer Kantor

   Mengawasi kinerja dan memutuskan kebijakan pada bagian pemasaran, transportasi, dan keuangan.
4. Perancangan

   Merancang model maket yang akan dibangun.
5. Pembuatan

   Membuat maket sesuai dengan rancangan.
6. Pemasaran

   Mengenalkan rancangan maket kepada perusahaan sesuai dengan tema yang diminta.

7. Transportasi

   Mengatur penggunaan transportasi untuk pengiriman maket ke perusahaan lain.

8. Keuangan

   Mengatur dan mengawasi setiap pengeluaran bagi penyedia bahan baku dan pemasukan hasil penjualan produk.

2.2. Landasan Teori

Landasan teori yang berkaitan dengan materi atau teori yang digunakan sebagai acuan dalam melakukan penelitian. Landasan teori yang diuraikan merupakan hasil dari literature dan buku-buku.

2.2.1. Maket

Maket adalah bentuk tiruan dari suatu objek yang telah diubah menjadi kecil dengan skala tertentu. Dalam bahasa Indonesia, maket disebut juga dengan istilah "miniatur". Memang tidak ada sesuatu yang bagus dan indah dalam mengilustrasikan suatu karya desain selain dalam bentuk gambar, akan tetapi hal ini masih dalam bentuk dua dimensi, padahal di sisi lain sebuah maket dapat menampilkan dalam bentuk tiga dimensi, dan ini sangat menarik untuk ditampilkan atau dipresentaikan dalam suatu pameran [4].

Menurut Mills, maket-maket studi dapat dibagi menjadi dua, yaitu maket primer dan maket sekunder. Maket primer berkaitan dengan tahap evolusi desain, sedangkan maket sekunder lebih berkaitan dengan unit bagian atau aspek-aspek proyek yang sedang diberi fokus [5].

2.2.2. Augmented Reality

a. Definisi

Augmented Reality adalah variasi lingkungan *virtual*, atau *virtual reality* seperti yang lebih umum disebut. Teknologi VR benar-benar membenamkan pengguna di dalam lingkungan sintetis. saat terbenam, pengguna tidak dapat melihat dunia nyata di sekitarnya. Sebaliknya, AR memungkinkan pengguna

untuk melihat dunia nyata, dengan benda-benda *virtual* yang ditumpangkan atau digabungkan dengan dunia nyata. Oleh karena itu, AR melengkapi kenyataan, daripada sepenuhnya menggantikannya. Idealnya, akan tampak bagi pengguna bahwa benda-benda *virtual* dan nyata hidup berdampingan dalam ruang yang sama. AR dapat dianggap sebagai "jalan tengah" antara VE dan *Telepresence* (Milgram dan Kishino, 1994a; Milgram et al., 1994b).

Beberapa peneliti mendefinisikan AR dengan cara yang membutuhkan penggunaan head-mount display (HMDs). Untuk menghindari pembatasan AR pada teknologi tertentu, survei ini mendefinisikan AR sebagai sistem apa pun yang memiliki tiga karakteristik berikut:

1. Menggabungkan nyata dan *virtual*
2. Interaktif secara *real time*
3. Terdaftar dalam tiga dimensi

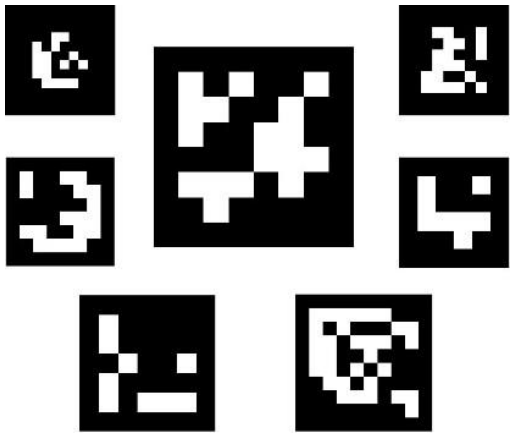
b. Tipe *Augmented Reality*

1) Marker Based

Berbagai jenis penanda *Augmented Reality* adalah gambar yang dapat dideteksi oleh kamera dan digunakan dengan perangkat lunak sebagai lokasi untuk aset virtual yang ditempatkan dalam sebuah *scene*. Sebagian besar hitam dan putih, meskipun warna dapat digunakan selama kontras di antara mereka dapat dikenali dengan baik oleh kamera. Penanda *augmented reality* sederhana dapat terdiri dari satu atau lebih bentuk dasar yang terdiri dari kotak hitam dengan garis belakang putih penanda yang lebih rumit dapat dibuat menggunakan gambar sederhana yang masih dibaca dengan baik oleh kamera, dan kode ini bahkan dapat berbentuk tato.

Kamera digunakan dengan perangkat lunak AR untuk mendeteksi penanda *augmented reality* sebagai lokasi untuk objek *virtual*. Hasilnya adalah gambar dapat dilihat, bahkan langsung, di layar dan aset digital ditempatkan di *scene* di lokasi penanda. Keterbatasan pada jenis penanda *augmented reality* yang dapat digunakan didasarkan pada perangkat lunak yang mengenalinya. Meskipun mereka harus tetap sederhana untuk koreksi kesalahan, mereka dapat menyertakan

berbagai gambar yang berbeda. Jenis penanda *augmented reality* yang paling sederhana adalah gambar hitam dan putih yang terdiri dari *barcode* dua dimensi (2D). Contoh *Marker Based* dapat dilihat pada Gambar 2.3.



Gambar 2.3 *Marker Based Tracking*

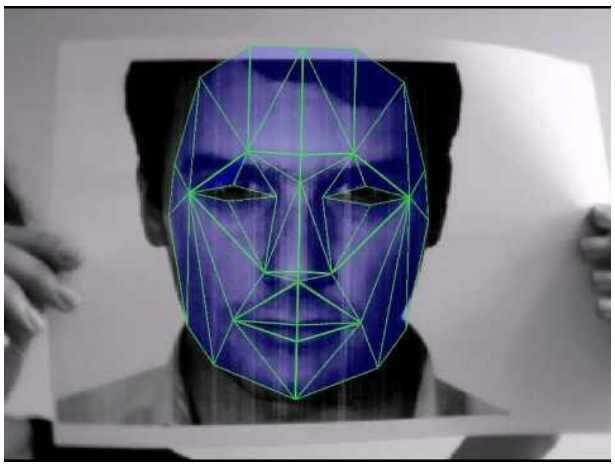
2) Markerless

Salah satu metode *Augmented Reality* adalah menggunakan metode *Markerless Augmented Reality*, dengan metode ini pengguna tidak perlu menggunakan sebuah *marker* untuk menampilkan elemen-elemen digital. Teknologi *Marker*les *Augmented Reality* yang dikembangkan dalam perangkat android diharapkan dapat membuat implementasi *Augmented Reality* jauh lebih efisien, praktis, menarik, dan bisa digunakan dimanapun, kapanpun, oleh siapapun tanpa perlu mencetak *marker* [6].

Dalam perancangannya, *markerless* seolah-olah menggabungkan antara objek 2D atau 3D yang merupakan objek *virtual* dan objek nyata beruba gambar dengan pola tertentu. *Markerless* tetap melakukan pemindaian terhadap objek untuk menjalankannya, namun ruang lingkup yang dipindai lebih luas dibandingkan dengan pemindaian menggunakan metode *based tracking*. Berbagai macam teknik *Markerless Based Tracking* yakni *Face Tracking*, 3D *Object Tracking*, dan *Motion Tracking*.

a. Face Tracking

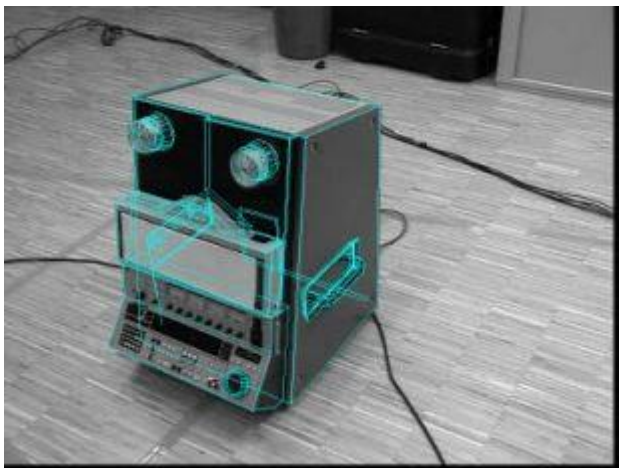
Perkembangan algoritma saat ini memungkinkan komputer dapat mengenali wajah manusia secara umum dengan cara mengenali posisi mata, hidung, dan mulut manusia, sedangkan objek-objek lain akan diabaikan. Contoh *face tracking* dapat dilihat pada Gambar 2.4.



Gambar 2.4 Face Tracking

b. 3D *Object Tracking*

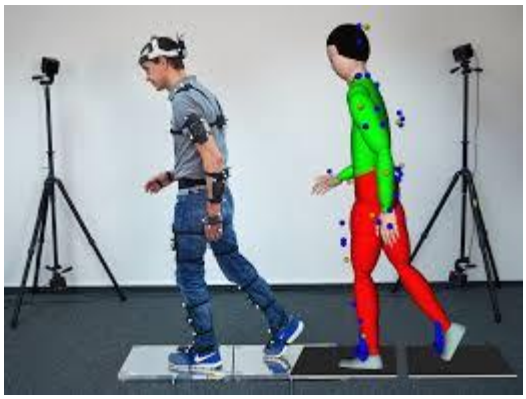
Teknik 3D *Object Tracking* dapat mengenali semua bentuk benda yang ada disekitar, seprti kendaraan, meja, televisi, gedung, dan lain-lain. Contoh 3D *Object Tracking* dapat dilihat pada Gambar 2.5.



Gambar 2.5 3D *Object Tracking*

c. Motion Tracking

Teknik ini memungkinkan komputer dapat menangkam gerakan. *Motion Tracking* biasanya digunakan untuk meproduksi film-film yang mencoba mensimulasikan gerakan. Contoh *Motion Tracking* dapat dilihat pada gambar 2.6.



Gambar 2.6 *Motion Tracking*

2.2.3. Android

Android merupakan sebuah sistem operasi pada ponsel berbasis Linux yang mencakup sistem operasi dan *middleware*. Fasilitas opensource atau sistem operasi yang dapat dikembangkan dengan bebas bagi penggunanya membuat banyak orang untuk mengembangkannya dengan inovasi – inovasi yang semakin berkembang terhadap sistem operasinya maupun pada pembangunan aplikasi *mobile* nya tersebut. Maka tak heran saat ini banyak pengembang yang membangun aplikasi *mobile* pada *platform* Android [7].

2.2.4. Denah

a. Definisi

Denah atau *Plan* merupakan penampang potongan horizontal dari suatu obyek/bangunan yang potongannya terletak pada ketinggian 1 meter dari atas lantai ruangan dalam bangunan. Denah mencerminkan skema organisasi kegiatan dalam bangunan dan merupakan unsur penentu bentuk bangunan. Denah berguna untuk mengunkapkan banyak hal seperti ruang sirkulasi dengan ruang untuk beraktivitas, dan hubungannya baik antar ruang di dalam bangunan maupun di

luar bangunan yang masih terletak didalam tapak, secara keseluruhan memberi makta bagi bangunan tersebut [8].

b. Tujuan Gambar Denah

Pembuatan gambar denah memiliki beberapa tujuan. Tujuan denah diantaranya :

1) Untuk menjelaskan ruang-ruang dua dimensi yang direncanakan
2) Hubungan ruang
3) Fungsi ruang
4) Ukuran ruang
5) Posisi/elevasi lantai/ruang
6) Hubungan ruang dalam (interior) dan ruang luar (eksterior)
7) Letak pintu dan jendela
8) Susunan *furniture*
9) Karakter obyek bangunan.

c. Kelengkapan dalam mengkomunikasikan gambar denah

Ada beberapa kelengkapan yang ada dalam mengkomunikasikan gambar denah, yaitu :

1) Nama gambar dan skala gambar
2) Ukuran ruang
3) Notasi dinding
4) Notasi bukaan pintu dan jendela
5) Notasi struktur kolom
6) Notasi area basah
7) Notasi teras
8) Piel muka tanah
9) Piel lantai
10) Garis arah pemotong
11) Garis rencana atap

d. Skala

Setiap jenis gambar memiliki ukuran yang berbeda-beda. Oleh karena itu, tidak memungkinkan mengambar suatu gambar dalam kertas gambar ukuran tertentu dalam ukuran sebenarnya. Untuk itu, ukuran gambar harus di perbesar jika objek aslinya kecil dan harus diperkecil jiga objek aslinya besar. Pengecilan atau pembesaran gambar dilakukan dengan skala tertentu. Skala adalah perbandingan ukuran linear pada gambar terhadap ukuran linera dari benda sebenarnya [9]. Ada tiga macam Skala gambar, yaitu :

1. Skala Pembesaran

Skala pembesaran digunakan jika gambar dibuat lebih besar daripada ukuran sebenarnya. Penunjukan untuk skala pembesaran adalah X:1.

2. Skala Penuh

Skala penuh digunakan bilamana gambar yang dibuat sama besar dengan ukuran sebenarnya. Penunjukan untuk skala penuh adalah 1:1.

3. Skala Pengecilan

Skala pengecilan digunakan jika gambar dibuat lebih kecil daripada ukuran sebenarnya. Penunjukan untuk skala pengecilan adalah 1:X

2.2.5. Unified Modeling Language (UML)

Berikut beberapa pendapat pengertian mengenai *Unified Modeling Languange* (UML) :

Unified Modeling Languange (UML) adalah keluarga notasi grafis yang didukung oleh meta-model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek (OO). Definisi ini merupakan definisi yang sederhana. Pada kenyataannya, pendapat orang-orang tentang UML berbeda satu sama lain. Hal ini dikarenakan oleh sejarahnya sendiri dan oleh perbedaan persepsi tentang apa yang membuat sebuah proses rancang-bangun perangkat lunak efektif [10].

Secara umum UML merupakan ‘bahasa’ untuk visualisasi, sepesifikasi, konstruksi, serta dokumentasi. Dalam kerangka visualisasi, para pengembang

menggunakan UML sebagai suatu cara untuk mengkomunikasikan idenya kepada para pemrogram serta calon pengguna sistem/perangkat lunak. Dengan adanya ‘bahasa’ yang bersifat standar, komunikasi perancang dengan pemrogram serta calon pengguna diharapkan menjadi mulus [11].

Dalam kerangka spesifikasi, UML menyediakan model-model yang tepat,tidak mendua-arti (ambigu), serta lengkap. Secara khusus, UML menspesifikasi langka-langka penting dalam pengambilan keputusan analisis, perancangan, serta implementasi dalam sistem yang sangat bernuansa perangkat lunak (*software intensive system*). Dalam hal itu, UML bukanlah merupakan Bahasa pemrograman tetapi model-model yang tercipta berhubungan langsung dengan berbagai macam Bahasa pemrograman perorientasi objek, katakanlah Java, Borland Delphi, Visual BASIC, C++, dan lain-lain [11].

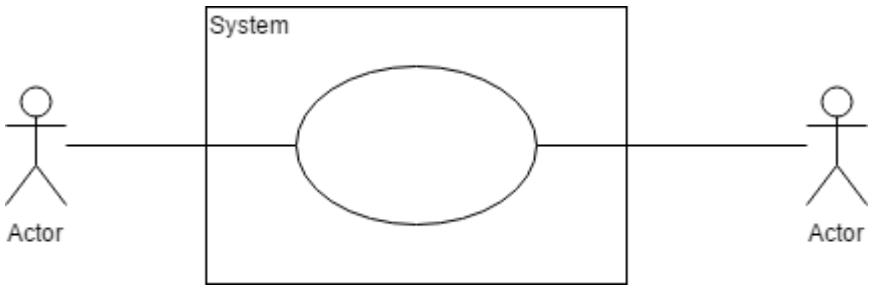
UML lahir dari penggabungan banyak Bahasa pemodelan grafis berorientasi objek yang berkembang pesat pada akhir 1980-an dan awal 1990-an. Sejak kehadirannya pada tahun 1997, UML menghancurkan Menara Babel tersebut menjadi sejarah. Pada intinya peran UML dalam pengembangan perangkat lunak, orang-orang memiliki cara-cara yang berbeda dalam penggunaanya, perbedaan-perbedaan yang masih dibawa dari Bahasa-bahasa pemodelan grafis lain. Perbedaan-perbedaan ini mengakibatkan perselisihan yang panjang dan keras tentang bagaimana UML seharusnya digunakan.

a. Use Case Diagram

Use Case merupakan deskripsi fungsi dari sebuah sistem. *Use case* bekerja dengan cara mendeskripsikan tipikal interaksi antara pengguna sebuah sistem dengan sistem itu sendiri [12].

Secara umum, tujuan dari *use case diagram* adalah sebagai berikut :

1. Digunakan untuk mengumpulkan kebutuhan dari sebuah sistem.
2. Untuk mendapatkan pandangan dari luar sistem
3. Untuk mengidentifikasi faktor yang mempengaruhi sistem
4. Untuk menunjukan interaksi dari para *actor* dari sistem ilustrasi dari *actor*, *use case* dan *boundary* dapat dilihat pada Gambar 2.7.



Gambar 2.7 *Use Case Model*

b. Activity Diagram

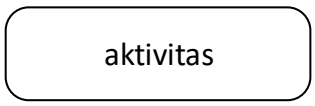
Activity Diagram merupakan bagian penting dari UML. *Activity Diagram* menggambarkan aspek dinamis dari sistem. Logika *procedural*, proses bisnis, dan aliran kerja suatu bisnis bisa dengan mudah dideskripsikan dalam *activity diagram*. *Activity diagram* memiliki peran seperti *flowchart*, akan tetapi *activity diagram* bisa mendukung perilaku paralel.

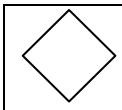
Tujuan dari *activity diagram* adalah menangkap tingkah laku sistem dengan cara menunjukan aliran pesan dari aktifitas satu ke aktifitas lainnya. Secara umum tujuannya adalah sebagai berikut :

1. Menggambarkan aliran aktivitas dari sistem
2. Menggambarkan urutan aktifitas dari suatu aktifitas ke aktifitas lainnya.
3. Menggambarkan paralelisme, percabangan, dan aliran konkuren dari sistem.

Berikut ini adalah simbol-simbol yang ada pada diagram *use case* [13]*:*

Tabel 2.1 Simbol Diagram Aktivitas Beserta Fungsinya

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
Percabangan / *decision*	Asosiasi percabangan dimana jika ada

	pilihan aktivitas lebih dari satu.
Penggabungan / *join* 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.

c. Class Diagram

Class diagram merupakan diagram statis yang mewakili pandangan statis dari suatu aplikasi. *Class diagram* digunakan untuk membangun kode eksekusi dari aplikasi perangkat lunak. *Class diagram* menunjukan koleksi kelas, antarmuka, asosiasi, kolaborasi, dan costranint. *Class diagram* biasa dikenal dengan diagram struktural.

Tujuan dari *class diagram* untuk memodelkan pandangan statis dari suatu aplikasi. Secara lebih rinci tujuannya adalah sebagai berikut :

1. Analisis dan desain pandangan statis dari suatu aplikasi.
2. Menjelaskan tanggung jawab suatu sistem.
3. Basis untuk diagram komponen dan penyebaran.
4. Forward and reverese engineering

d. Sequence Diagram

Sequence diagram diguakan untuk menggambarkan perilaku dari sebuah skenario. Diagram ini menunjukan sejumlah obyek dan pesan yang diletakan antara obyek-obyek di dalam *use case.*

Komponen utama *sequence diagram* terdiri atas obyek yang dituliskan dengan kotak segiempat. Pesan diwakili oleh garis dengan tanda panah dan waktu yang ditunjukan dengan *progress vertical*.

2.2.6. *Tools* yang digunakan

Dalam pengembangan aplikasi dibutuhkan beberapa *tools* yang digunakan. *Tools* yang digunakan yaitu :

2.2.6.1. Vuforia

Vuforia merupakan *software* untuk *augmented reality* yang dengan menggunakan sumber yang konsisten mengenai komputer vision yang fokus pada *image recognition*. Vuforia memiliki berbagai fitur dan kemampuan yang dapat membantu pengembang untuk mewujudkan pemikiran mereka tanpa adanya batasan secara teknikal.

Vuforia dapat digunakan pada iOS, Android, dan Unity3D. Hal itu mendukung para pengembang untuk membuat aplikasi yang dapat digunakan dihampir seluruh jenis *smartphone* dan tablet. Pengembang diberikan kebebasan untuk mendesain dan membuat aplikasi dengan kemampuan sebagai berikut (Fernando, 2013) :

1. Teknologi *computer vision* tingkat tinggi yang mengijinkan *developer* untuk membuat efek khusus pada *device*.
2. Terus menerus mengenali *multiple image*.
3. *Tracking* dan *detection* tingkat lanjut.
4. Solusi pengaturan database gambar yang *flexibel*.

Target pada vuforia merupakan objek pada dunia nyata yang dapat dideteksi oleh kamera untuk menampilkan objek *virtual*. Terdapat 4 target pada vuforia yaitu *image targets*, *frame markers*, *multi-target*, dan *virtual buttons*.

2.2.6.2. Unity3D

Unity merupakan *game engine*, yaitu *software* yang digunakan untuk memudahkan dalam membuat game. Unity telah menyediakan berbagai macam *tool* yang dapat membantu dalam membuat game dan di unity dapat menambahkan skrip untuk mengatur jalannya game [14]. Unity berbasis *cross-platform,* unity dapat digunakan untuk membuat sebuah *game* yang bisa digunakan pada perangkat komputer, *smartphone* Android, iPhone, dan bahkan X-BOX.

Unity juga memiliki IDE atau yang disebut juga *Integrated Development Environment* yaitu *MonoDevelop* yang bertujuan untuk mengintegrasikan semua script yang dibuat kedalam unity sehingga dapat langsung diproses. Unity biasanya digunakan untuk membuat *game mobile* atau *browser*, tetapi bisa digunakan untuk PC dan *Console. Game engine* unity di kembangkan dengan menggunakan dengan bahasa pemograman C/C++ dan dapat dengan baik mendukung berbagai macam bahasa pemograman yang lainnya seperti C#, BOO, JAVASCRIPT.

2.2.6.3. SketchUp

SketchUp merupakan program grafis 3D yang dikembangkan oleh Google yang mengkombinasikan seperangkat alat (*tools*) yang sederhana, namun sangat handal dalam desain grafis 3D di dalam layar komputer. SketchUp pertama kali dirilis oleh @Last Software pada tahun 2000. Sejak Google mengakuisisi @Last Software pada tahun 2006, SketchUp lebih dilekan sebagai Google SketchUp dan telah berhasil berkembang dengan sangat pesat [15].

Terdapat dua versi SketchUp yang tersedia dan dapati diunduh melalui alamat situs http://sketchup.google.com/intl/en/download/index.html. Versi Pertama adalah Google SketchUp yang tersedia secara gratis. Versi ini mendukung secara penuh seluruh fungsi yang dibutuhkan untuk menghidupkan ruang imajinasi penggunanya. Dengan versi ini, penguna dapat mendesain dan membuat objek-objek 3D dan mendisribusikannya kepada semua pengguna Google SketchUp di seluruh dunia [15].

Versi kedua adalah Google SketchUp Pro with LayOut yang didedikasikan bagi para professional yang bekerja di dunia grafis 3D. Versi kedua ini memiliki fitur yang sama dengan versi pertama. Perbedaanya terletak pada fasilitas untuk menukar *file* yang dibuat dengan Google SketchUp dengan *software-software* grafis lainnya. Selain itu, versi ini juga dilengkapi dengan Google SketchUp Layout yang dapat digunakan untuk membuat presentasi desain yang menarik, serta berbagai fungsi tambahan lainnya [15].

2.2.6.4. AutoCAD

AutoCAD adalah perangkat lunak CAD (*Computer Aided Design*) untuk menggambar teknik baik 2 dimensi dan 3 dimensi. *Software* ini dikembangkan oleh perusahaan bernama Autodesk sejak awal tahun 1980-an. AutoCAD dan turunannya merupakan software CAD yang paling banyak digunakan di dunia [16].

AutoCAD saat ini hanya berjalan di sistem opersai Microsoft. Versi untuk Unix dan Macintosh sempat dikeluarkan tahun 1980-an dan 1990-an, tetapi tidak dilanjutkan. Namun untuk kompatibilitas dengan sistem operasi lain, AutoCAD masih bisa berjalan di emulator dengan menggunakan alat seperti Virtual PC atau Wine. Tapi kemudian akhir-akhir ini dari tahun 2013, AutoCAD mulai lagi mengembangkan versi MAC-nya [16].

2.2.6.5. Bahasa Pemrogramman C#

Visual C-Sharp atau C# merupakan salah satu bahasa pemrograman berorientasi objek yang dikeluarkan Microsoft. Proyek pembuatannya ditangani oleh Anders Helsberg dan diperkenalkan unuk pertama kali pada bulan Juli 2000. Visual C# merupakan bahasa pemrograman modern berorientasi objek yang menjadi bahasa pemrograman utama dalam platform Microsoft.NET Framework. Visual C# dianggap sebagai kombinasi antara efisiensi pemrograman C++, kesederhanaan pemrograman java, dan penyederhanaan dari pemrograman Visual Basic [17].

C# pertama kali diperkenalkan pada bulan Juli 2000 sebagai sebuah bahasa pemrograman modern berorientasi objek yang menjadi sebuah bahasa pemrograman utama di dalam pengembangan di dalam platform Microsoft NET.Framework. C# didisain untuk memenuhi kebutuhan akan sintaksis C++ yang lebih ringkas dan Rapid Application Development yang 'tanpa batas' (dibandingkan dengan RAD yang 'terbatas' seperti yang terdapat pada Delphi dan VisualBasic.

Pada akhir dekade 1990-an, Microsoft membuat program Microsoft Visual J++ sebagai sebuah langkah percobaan untuk menggunakan Java di dalam sistem operasi Windows untuk meningkatkan antarmuka dari Microsoft Component

Object Model (COM). Akan tetapi, akibat masalah dengan pemegang hak cipta bahasa pemrograman Java, Sun Microsystem, Microsoft pun menghentikan pengembangan J++, dan beralih untuk membuat pengganti J++, kompilernya dan mesin virtual sendiri dengan menggunakan sebuah bahasa pemograman yang bersifat general-purpose. Untuk menangani proyek ini, Microsoft merekrut Anders Helsberg, yang merupakan mantan karyawan Borland yang membuat bahasa Turbo Pascal, dan Borland Delphi, yang juga mendesain *Windows Foundation Classes* (WFC) yang digunakan di dalam J++.

Kelemahan-kelemahan yang dikemukakannya itu yang menjadi basis CLR sebagai bentukan baru yang menutupi kelemahan-kelemahan tersebut, dan pada akhirnya memengaruhi desain pada bahasa C# itu sendiri. Ada kritik yang menyatakan C# sebagai bahasa yang berbagi akar dari bahasa-bahasa pemrograman lain. Fitur-fitur yang diambilnya dari bahasa C++ dan Java adalah desain berorientasi objek, seperti garbage collection, reflection, akar kelas (root class), dan juga penyederhanaan terhadap pewarisan jamak (multiple inheritance). Fitur-fitur tersebut di dalam C# kini telah diaplikasikan terhadap iterasi, propertis, kejadian (event), metadata, dan konversi antara tipe-tipe sederhana dan juga objek.

2.2.7. Pengujian Black Box

Black Box Testing merupakan pengujian yang berfokus pada spesifikasi fungsional dari perangkat lunak, tester dapat mendefinisikan kumpulan kondisi input dan melakukan pengetesan pada spesifikasi fungsional program [18]. *Black Box Testing* berfokus pada spesifikasi fungsional dari perangkat lunak. Tester dapat mendefinisikan kumpulan kondisi input dan melakukan pengetesan pada spesifikasi fungsional program.

Black Box Testing bukanlah solusi alternatif dari *White Box Testing* tapi lebih merupakan pelengkap untuk menguji hal-hal yang tidak dicakup oleh *White Box Testing*.

Black Box Testing cenderung untuk menemukan hal-hal berikut [18]:

1. Fungsi yang tidak benar atau tidak ada.
2. Kesalahan antarmuka (*interface errors*).

3. Kesalahan pada struktur data dan akses basis data.
4. Kesalahan performansi (*performance errors*).
5. Kesalahan inisialisasi dan terminasi.

Klasifikasi *black box* mencakup beberapa pengujian, yaitu [18] :

1. Pengujian fungsional (*alpha*)

Pada jenis pengujian ini perangkat lunak diuji untuk persyaratan fungsional. Pengujian dilakukan dalam bentuk tertulis untuk memeriksa apakah aplikasi berjalan seperti yang diharapkan. Walaupun pengujian fungsional sudah sering dilakukan di bagian akhir dari siklus pengembangan, masing-masing komponen dan proses dapat diuji pada awal pengembangan, bahkan sebelum sistem berfungsi, pengujian ini sudah dapat dilakukan pada seluruh sistem. Pengujian fungsional meliputi seberapa baik sistem melaksanakan fungsinya, termasuk perintah-perintah penggunan, manipulasi data, pencarian dan proses bisnis, pengguna layar dan integrasi. Pengujian fungsional juga meliputi permukaan yang jelas dari jenis fungsi-fungsi, serta operasi *backend* (seperti keamanan dan bagaimana meningkatkan sistem.

2. Penerimaan Pengguna (*User Acceptance*)

Pada jenis pengujian ini perangkat lunak akan diserahkan kepada pengguna untuk mengetahui apakah perangkat lunak memenuhi harapan pengguna dan bekerja seperti yang diharapkan. Pada pengembangan perangkat lunak, *user acceptance testing* (UAT), juga disebut pengujian beta (*beta testing*), pengujian aplikasi (*application testing*) dan pengujian pengguna akhir (*end user testing*) adalah tahapan pengembangan perangkat lunak ketika perangkat lunak diuji pada dunia nyata yang dimaksudkan oleh pengguna.

UAT dapat dilakukan dengan *in-house testing* dengan membayar relawan atau subjek pengujian menggunakan perangkat lunak atau biasanya mendistribusikan perangkat lunak secara luas dengan melakukan pengujian versi yang tersedia secara gratis untuk diunduh melalui web. Pengalaman awal pengguna akan diteruskan kembali kepada para pengembang yang membuat perubahan sebelum akhirnya melepaskan perangkat lunak komersial.

2.2.8. Skala *Likert*

Skala *likert* adalah skala pengukuran yang dikembangkan oleh *Likert*. Skala *likert* mempunyai empat atau lebih butir-butir pertanyan yang di kombinasikan sehingga membentuk sebuah skor/nilai yang merepresentasikan sifat individu, misalkan pengetahuan, sikap, dan perilaku. Dalam proses analisis data, komposit skor, biasanya jumlah atau rataan, dari semua butir pertanyaan dapat digunakan. Skala *Likert* adalah suatu skala psikometrik yang umum digunakan dalam kuesioner, dan skala yang paling banyak digunakan dalam riset berupa survei. Nama skala ini diambil dari nama *Rensis Likert*, yang menerbitkan suatu laporan yang menjelaskan penggunaannya. Sewaktu menanggapi pertanyaan dalam skala likert responden menentukan tingkat persetujuan mereka terhadap suatu pernyataan dengan memilih salah satu dari pilihan yang tersedia. Biasanya disediakan lima pilihan skala dengan format seperti [19]:

1. Sangat setuju
2. Setuju
3. Netral
4. Tidak Setuju
5. Sangat Tidak Setuju

Skala *Likert* kerap digunakan sebagai skala penilaian karena memberi nilai terhadap sesuatu. Untuk keperluan analisis kuantitatif, skala jawaban pada skala likert dapat diberi skor misalnya :

1. Sangat Setuju (SS) diberi skor 5
2. Setuju (ST) diberi skor 4
3. Ragu-ragu (RG) diberi skor 3
4. Tidak Setuju (TS) diberi skor 2
5. Sangat Tidak Setuju (STS) skor 1