

BAB 2

TINJAUAN PUSTAKA

2.1 Bencana Indonesia

Indonesia secara geografis dan geologis terletak di daerah yang rawan terhadap bencana alam. Berbagai bencana, seperti: gempa bumi, tsunami, banjir, tanah longsor, topan, dan angin puting beliung melanda hampir di seluruh pelosok negeri sehingga timbul anggapan bahwa Indonesia merupakan "*supermarket*" bencana. Serangkaian kejadian bencana alam ini telah mengakibatkan banyak korban jiwa, kerugian harta benda, dan kerusakan lingkungan [8].



Gambar 2.1 Potensi dan Ancaman Bencana

2.1.1 Definisi Bencana

Undang-undang Nomor 24 Tahun 2007 Tentang Penanggulangan Bencana menyebutkan definisi bencana sebagai berikut [9]:

Bencana adalah peristiwa atau rangkaian peristiwa yang mengancam dan mengganggu kehidupan dan penghidupan masyarakat yang disebabkan, baik oleh faktor alam dan atau faktor nonalam maupun faktor manusia sehingga mengakibatkan timbulnya korban jiwa manusia, kerusakan lingkungan, kerugian harta benda, dan dampak psikologis. Oleh karena itu, Undang-undang Nomor 24 Tahun 2007 tersebut juga mendefinisikan mengenai bencana alam, bencana nonalam, dan bencana sosial.

- a. Bencana alam adalah bencana yang diakibatkan oleh peristiwa atau serangkaian peristiwa yang disebabkan oleh alam antara lain berupa gempa bumi, tsunami, gunung meletus, banjir, kekeringan, angin topan, dan tanah longsor.
- b. Bencana nonalam adalah bencana yang diakibatkan oleh peristiwa atau rangkaian peristiwa nonalam yang antara lain berupa gagal teknologi, gagal modernisasi, epidemi, dan wabah penyakit.
- c. Bencana sosial adalah bencana yang diakibatkan oleh peristiwa atau serangkaian peristiwa yang diakibatkan oleh manusia yang meliputi konflik sosial antar kelompok atau antar komunitas masyarakat dan terror.

2.1.2 Kejadian Bencana

Kejadian Bencana adalah peristiwa bencana yang terjadi dan dicatat berdasarkan tanggal kejadian, lokasi, jenis bencana, korban dan/ataupun kerusakan. Jika terjadi bencana pada tanggal yang sama dan melanda lebih dari satu wilayah maka dihitung sebagai suatu kejadian. Berikut adalah definisi dari bencana alam gempa dan banjir yang sering terjadi di Indonesia [9] :



Gambar 2.2 Bencana Alam Sentani

1. Gempa bumi adalah getaran atau guncangan yang terjadi di permukaan bumi yang disebabkan oleh tumbukan antar lempeng bumi, patahan aktif, aktivitas gunung api atau runtuhannya batuan.
2. Banjir adalah peristiwa atau keadaan dimana terendamnya suatu daerah atau daratan karena volume air yang meningkat. Sedangkan banjir bandang adalah

banjir yang datang secara tiba-tiba dengan debit air yang besar yang disebabkan terbendungnya aliran sungai pada alur sungai.

3. Letusan gunung api merupakan bagian dari aktivitas vulkanik yang dikenal dengan istilah "erupsi". Bahaya letusan gunung api dapat berupa awan panas, lontaran material (pijar), hujan abu lebat, lava, gas racun, tsunami dan banjir lahar.
4. Tsunami berasal dari bahasa Jepang yang berarti gelombang ombak lautan ("tsu" berarti lautan, "nami" berarti gelombang ombak). Tsunami adalah serangkaian gelombang ombak laut raksasa yang timbul karena adanya pergeseran di dasar laut akibat gempa bumi.
5. Tanah longsor merupakan salah satu jenis gerakan massa tanah atau batuan, ataupun percampuran keduanya, menuruni atau keluar lereng akibat terganggunya kestabilan tanah atau batuan penyusun lereng.
6. Banjir adalah peristiwa atau keadaan dimana terendamnya suatu daerah atau daratan karena volume air yang meningkat.
7. Kekeringan adalah ketersediaan air yang jauh di bawah kebutuhan air untuk kebutuhan hidup, pertanian, kegiatan ekonomi dan lingkungan. Adapun yang dimaksud kekeringan di bidang pertanian adalah kekeringan yang terjadi di lahan pertanian yang ada tanaman (padi, jagung, kedelai dan lain-lain) yang sedang dibudidayakan .
8. Kebakaran adalah situasi dimana bangunan pada suatu tempat seperti rumah/pemukiman, pabrik, pasar, gedung dan lain-lain dilanda api yang menimbulkan korban dan/atau kerugian.
9. Kebakaran hutan dan lahan adalah suatu keadaan di mana hutan dan lahan dilanda api, sehingga mengakibatkan kerusakan hutan dan lahan yang menimbulkan kerugian ekonomis dan atau nilai lingkungan. Kebakaran hutan dan lahan seringkali menyebabkan bencana asap yang dapat mengganggu aktivitas dan kesehatan masyarakat sekitar.
10. Angin puting beliung adalah angin kencang yang datang secara tiba-tiba, mempunyai pusat, bergerak melingkar menyerupai spiral dengan kecepatan

40-50 km/jam hingga menyentuh permukaan bumi dan akan hilang dalam waktu singkat (3-5 menit).

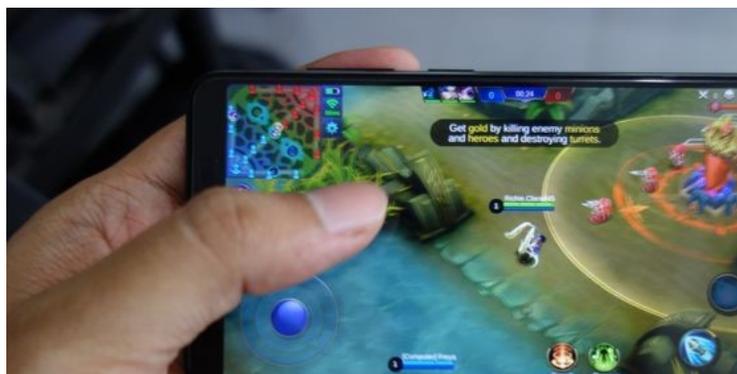
11. Gelombang pasang atau badai adalah gelombang tinggi yang ditimbulkan karena efek terjadinya siklon tropis di sekitar wilayah Indonesia dan berpotensi kuat menimbulkan bencana alam. Indonesia bukan daerah lintasan siklon tropis tetapi keberadaan siklon tropis akan memberikan pengaruh kuat terjadinya angin kencang, gelombang tinggi disertai hujan deras.
12. Abrasi adalah proses pengikisan pantai oleh tenaga gelombang laut dan arus laut yang bersifat merusak. Abrasi biasanya disebut juga erosi pantai. Kerusakan garis pantai akibat abrasi ini dipicu oleh terganggunya keseimbangan alam daerah pantai tersebut. Walaupun abrasi bisa disebabkan oleh gejala alami, namun manusia sering disebut sebagai penyebab utama abrasi.
13. Kecelakaan transportasi adalah kecelakaan moda transportasi yang terjadi di darat, laut dan udara.
14. Kecelakaan industri adalah kecelakaan yang disebabkan oleh dua faktor, yaitu perilaku kerja yang berbahaya (unsafe human act) dan kondisi yang berbahaya (unsafe conditions). Adapun jenis kecelakaan yang terjadi sangat bergantung pada macam industrinya, misalnya bahan dan peralatan kerja yang dipergunakan, proses kerja, kondisi tempat kerja, bahkan pekerja yang terlibat di dalamnya.
15. Kejadian Luar Biasa (KLB) adalah timbulnya atau meningkatnya kejadian kesakitan atau kematian yang bermakna secara epidemiologis pada suatu daerah dalam kurun waktu tertentu. Status Kejadian Luar Biasa diatur oleh Peraturan Menteri Kesehatan RI No. 949/MENKES/SK/VII/2004.
16. Konflik Sosial atau kerusuhan sosial atau huru hara adalah suatu gerakan massal yang bersifat merusak tatanan dan tata tertib sosial yang ada, yang dipicu oleh kecemburuan sosial, budaya dan ekonomi yang biasanya dikemas sebagai pertentangan antar suku, agama, ras (SARA).
17. Aksi Teror adalah aksi yang dilakukan oleh setiap orang yang dengan sengaja menggunakan kekerasan atau ancaman kekerasan sehingga menimbulkan

suasana teror atau rasa takut terhadap orang secara meluas atau menimbulkan korban yang bersifat massal, dengan cara merampas kemerdekaan sehingga mengakibatkan hilangnya nyawa dan harta benda, mengakibatkan kerusakan atau kehancuran terhadap obyek-obyek vital yang strategis atau lingkungan hidup atau fasilitas publik internasional.

18. Sabotase adalah tindakan yang dilakukan untuk melemahkan musuh melalui subversi, penghambatan, pengacauan dan/ atau penghancuran. Dalam perang, istilah ini digunakan untuk mendiskripsikan aktivitas individu atau grup yang tidak berhubungan dengan militer, tetapi dengan spionase. Sabotase dapat dilakukan terhadap beberapa struktur penting, seperti infrastruktur, struktur ekonomi, dan lain-lain.

2.2 Permainan (*Game*)

Game dalam bahasa Indonesia yang berarti permainan. Permainan adalah sesuatu yang dapat dimainkan dengan aturan tertentu sehingga ada yang menang dan ada yang kalah, biasanya dalam konteks tidak serius atau dengan tujuan *refreshing*. Suatu cara belajar yang digunakan dalam menganalisa interaksi antara sejumlah pemain maupun perorangan yang menunjukkan strategi-strategi yang rasional [10].



Gambar 2.3 Game *Mobile*

2.2.1 Pengertian *Game*

Pengertian *game* menurut beberapa ahli [10] :

1. Menurut Clark C. Abt, *game* adalah kegiatan yang melibatkan keputusan pemain, berupaya mencapai tujuan dengan dibatasi oleh konteks tertentu (misalnya, dibatasi oleh peraturan).
2. Menurut David Parlett, *game* adalah sesuatu yang memiliki “akhir dan cara mencapainya” artinya ada tujuan, hasil dan serangkaian peraturan untuk mencapai keduanya.
3. Menurut Greg Costikyan, *game* adalah sebarang karya seni dimana peserta, yang disebut pemain membuat keputusan untuk mengelola sumberdaya yang dimilikinya melalui benda di dalam game demi mencapai tujuan.
4. Menurut Chris Crawford, seorang komputer *game designer* mengemukakan bahwa *game* pada intinya adalah sebuah interaktif, aktivitas yang berpusat pada sebuah pencapaian, ada pelaku aktif (*player*), ada pelaku pasif (NPC).

Maka dari itu *Game* merupakan salah satu media hiburan yang sangat ramai digunakan oleh setiap orang baik tua mau pun muda. Teknologi *game* saat ini telah mengalami berbagai kemajuan yang sangat pesat. Terbukti hal ini di tandai dengan banyak berkembangnya jenis, produk dan berbagai alat yang beragam untuk berinteraksi dengan *game* dan saat ini untuk membuat dan mengembangkan *video game* banyak *software* yang memiliki *tools* yang dapat mendukung pembuatan *game* salah satunya adalah unity.

2.2.2 Sudut Pandang *Game (Game View Point)*

Sebuah *game* biasanya mempunyai sudut pandang permainan tersendiri disesuaikan berdasarkan genre *game* yang diambil. Berikut beberapa macam sudut pandang permainan yang biasa digunakan [11] :

1. Side Scrolling Adalah sudut pandang permainan yang terlihat dari samping dan memungkinkan karakter utama untuk bergerak dari kiri ke kanan serta

memungkinkan background pada game seolah-olah bergeser mengikuti pergerakan karakter utama.

2. Top Down Adalah sudut pandang permainan yang memungkinkan karakter utama bermanuver ke empat arah namun cara permainannya sendiri bergeser dari bawah ke atas, dan biasanya game yang menggunakan sudut pandang permainan jenis ini adalah shooter game.
3. Isometric Adalah sudut pandang permainan yang memungkinkan permainan terlihat diantara sisi Side Scrolling dan juga Top Down, dan biasanya diterapkan pada game dengan genre RTS (Real Time Strategy).

2.2.3 Genre Game

Permainan *video game* saat ini banyak mengalami perkembangan yang sangat pesat, sehingga memiliki beragam jenis *genre game*. *Video game* dapat dibagi ke beberapa *genre* diantaranya [7]:

1. Action



Gambar 2.4 Contoh Game Action

Permainan video bergenre action memberikan tantangan fisik yang membutuhkan koordinasi tangan dan mata, serta kemampuan motorik yang baik untuk dimainkan. Pemain mengendalikan karakter untuk mencapai misi tertentu.

Genre action ini terbagi lagi menjadi beberapa jenis seperti :

- a. *shooter*, yang memberikan tujuan bagi pemain untuk menembak sasaran/musuh,
- b. *fighting*, permainan pertarungan dalam arena, stealth, permainan yang mengharuskan pemain untuk menyelesaikan misi tanpa terdeteksi musuh, dan
- c. *survival*, pemain diharuskan untuk bertahan hidup untuk menyelesaikan permainan.

Permainan *video* bergenre *action* biasanya menjadi bagian dari permainan *video* bergenre lain, seperti *action-strategy*, *action-shooter*, dan lain-lain.

2. *Adventure*



Gambar 2.5 Contoh *Game Adventure*

Permainan bergenre petualangan mulai ada pada tahun 1970, yaitu permainan video berjudul *Colossal Cave Adventure*. Permainan petualangan memberikan sedikit tekanan pada pemainnya untuk menyelesaikan misi, dengan tantangan tertentu seperti batas waktu. Permainan video dengan genre petualangan memberikan daya tarik yaitu kemampuan unik dari karakter yang dimainkan oleh pengguna.

Game petualangan bukan merupakan sebuah kompetisi atau simulasi seperti game yang lainnya. *Game* jenis ini tidak menawarkan proses untuk dikelola atau

mengalahkan musuh melalui strategi dan taktik. *Game* petualangan merupakan cerita interaktif mengenai karakter yang dikontrol oleh pemain.

3. *Role Playing*



Gambar 2.6 Contoh *Game Role Playing*

Permainan dengan genre *role playing* memungkinkan pemain untuk mengendalikan karakter sesuai dengan perannya. Peran yang dimaksud adalah fungsionalitas karakter, mulai dari penyihir, penembak jitu, ksatria, atau penyembuh.

Permainan bergenre *role playing* biasanya memiliki elemen *action* dan petualangan sehingga lebih menarik. Permainan *role playing* yang pertama kali dirilis adalah *Dragon Series* yang diproduksi oleh Nihon Falcom pada tahun 1980. Permainan video *role playing* biasanya memiliki fitur pertarungan antar karakter.

4. Simulasi



Gambar 2.7 Contoh *Game Simulasi*

Permainan video dengan genre simulasi membutuhkan sangat banyak detail yang benar-benar menyerupai dunia nyata. Melibatkan banyak peneliti dalam pembangunannya. Contoh permainan video dengan genre simulasi adalah simulasi kendaraan. Permainan ini menyediakan fitur yang memungkinkan pemain untuk mengendalikan berbagai macam kendaraan dengan operasi interpretasi yang semirip mungkin dengan aslinya.

Permainan simulasi juga dapat berupa simulasi kehidupan, melibatkan pengendalian manusia dalam suatu kota virtual. Dari suatu kota tersebut dapat membentuk ekosistem yang memiliki relasi tertentu. Contoh permainan video yang sangat terkenal adalah The Sims.

5. Strategi



Gambar 2.8 Contoh *Game* Strategi

Permainan video dengan genre strategi memfokuskan cara bermain yang membutuhkan kemampuan berpikir dan perencanaan untuk mencapai suatu kemenangan. Menurut Rollings permainan video strategi memiliki karakteristik, “pemain diberikan pandangan sebagai tuhan, secara tidak langsung mengendalikan unit di bawahnya melalui perintah”. Permainan video dengan genre strategi berakar dari permainan yang menggunakan papan.

6. Olahraga



Gambar 2.9 Contoh Game Olahraga

Permainan video game dengan *genre* olahraga adalah simulasi dari olahraga tersebut. Pemain dapat memilih lawan yaitu pemain lain ataupun kecerdasan buatan (AI). Berbagai contohnya adalah permainan olahraga sepakbola, basket, balapan, dan lain-lain.

Game olahraga mempunyai tantangan yang tidak seperti game-game yang lain, dimana pemain memiliki sedikit pengetahuan mengenai dunianya, tetapi game olahraga meniru aturan pada olahraga di kehidupan nyata.

7. Arcade



Gambar 2.10 Contoh Game Arcade

Permainan *arcade* adalah permainan berupa hiburan yang dioperasikan menggunakan koin. Permainan video komputer arcade biasanya hanya dimainkan untuk jangka waktu yang pendek. Permainan *endless running* termasuk permainan

video genre ini. Permainan endless running adalah suatu permainan yang dirancang tanpa akhir, tanpa tingkatan, dan tanpa level pemainnya.

Permainan video ini dimulai dengan pelan dan sangat mudah. Pemain memainkan karakter permainan yang akan terus berlari untuk mendapatkan suatu koin/skor pada jalan/jalur lainnya. Permainan ini populer karena dianggap sebagai “cemilan”, karena dapat dimainkan antara waktu tiga hingga lima menit.

2.2.4 Game Simulasi

Simulasi ialah suatu metodologi untuk melaksanakan percobaan dengan menggunakan model dari satu sistem nyata. Simulasi adalah suatu proses peniruan dari sesuatu yang nyata beserta keadaan sekelilingnya (state of affairs). Aksi melakukan simulasi ini secara umum menggambarkan sifat-sifat karakteristik kunci dari kelakuan sistem fisik yang di abstraksikan. Menurut Hasan, simulasi merupakan suatu model pengambilan keputusan dengan mencontoh atau mempergunakan gambaran sebenarnya dari suatu sistem kehidupan dunia nyata tanpa harus mengalaminya pada keadaan yang sesungguhnya.

Simulasi adalah suatu teknik yang dapat digunakan untuk memformulasikan dan memecahkan model - model dari golongan yang luas. Simulasi merupakan suatu teknik meniru operasi-operasi atau proses-proses yang terjadi dalam suatu sistem dengan bantuan perangkat komputer dan dilandasi oleh beberapa asumsi tertentu sehingga sistem tersebut bisa dipelajari secara ilmiah. Dalam simulasi digunakan komputer untuk mempelajari sistem secara numerik, dimana dilakukan pengumpulan data untuk melakukan estimasi statistik untuk mendapatkan karakteristik asli dari sistem.

Game simulasi dibuat dengan tujuan menempatkan pemain sebagai pengatur dari aktivitas tertentu dan mencoba membuatnya serealistik mungkin. Dalam artian yang sempit, semua game elektronik merupakan simulasi, karena umumnya game tidak bisa ada tanpa simulasi aspek kehidupan nyata. Namun, genre simulasi hanya mencakup permainan yang berfokus (seluruhnya atau sebagian) pada meniru kegiatan kehidupan nyata. Ini tidak berarti bahwa game

simulasi harus benar-benar realistis, misalnya, game simulasi pertempuran ruang angkasa yang mensimulasikan suatu kegiatan yang tidak (belum) diketahui oleh manusia. Fitur yang membedakan game simulasi adalah penekanan pada realisme dan detail sebanding dengan yang ditemui dalam kehidupan nyata.

Kebanyakan game simulasi tidak memiliki alur cerita, karena game simulasi berkonsentrasi pada penjelasan kegiatan umum, bukan situasi yang konkret. sebagai pengecualian adalah game Wing Comander, yang menggabungkan ruang simulasi tempur dengan narasi berkesinambungan.

2.2.4.1 Jenis-jenis *Game* Simulasi

Game Simulasi terdiri dari beberapa jenis varisasi dan mereka hanya dibatasi oleh jumlah atau tingkatan kegiatan yang dikenal oleh manusia. Tema game simulasi meliputi :

1. **Sosiodrama.** Sosiodrama adalah metode pembelajaran bermain peran untuk memecahkan masalah-masalah yang berkaitan dengan fenomena sosial, permasalahan yang menyangkut hubungan antara manusia seperti masalah kenakalan remaja, narkoba, gambaran keluarga otoriter, dan lain sebagainya. Sosiodrama digunakan untuk memberikan pemahaman dan penghayatan akan masalah-masalah sosial serta mengembangkan kemampuan siswa untuk memecahkannya.
2. **Psikodrama.** Psikodrama adalah metode pembelajaran dengan bermain peran yang bertitik tolak dari permasalahan-permasalahan psikologis. Psikodrama biasanya digunakan untuk terapi, yaitu agar siswa memperoleh pemahaman yang lebih baik tentang dirinya, menemukan konsep diri, menyatakan reaksi terhadap tekanan-tekanan yang dialaminya.
3. ***Role Playing.*** *Role playing* atau bermain peran adalah metode pembelajaran sebagai bagian dari simulasi yang diarahkan untuk mengkreasi peristiwa-peristiwa actual, atau kejadian-kejadian yang mungkin muncul pada masa mendatang. Topik yang dapat diangkat untuk *role playing* misalnya

memainkan peran sebagai juru kampanye suatu partai atau gambaran keadaan yang mungkin muncul pada abad teknologi informasi.

4. *Peer Teaching*. *Peer teaching* merupakan latihan mengajar yang dilakukan oleh siswa kepada teman-teman calon guru. Selain itu *peer teaching* merupakan kegiatan pembelajaran yang dilakukan seorang siswa kepada siswa lainnya dan salah satu siswa itu lebih memahami materi pembelajaran.
5. Simulasi Game. Simulasi game merupakan bermain peran, para siswa berkompetisi untuk mencapai tujuan tertentu melalui permainan dengan mematuhi peraturan yang ditentukan.

2.2.5 Pengertian *Game* Edukasi

Game edukasi adalah permainan yang dirancang atau dibuat untuk merangsang daya pikir termasuk meningkatkan konsentrasi dan memecahkan masalah [12]. *Game* Edukasi adalah salah satu jenis media yang digunakan untuk memberikan pengajaran, menambah pengetahuan penggunanya melalui suatu media unik dan menarik. Jenis ini biasanya ditujukan untuk anak-anak, maka permainan warna sangat diperlukan disini bukan tingkat kesulitan yang dipentingkan [12]. Berdasarkan uraian tersebut maka dapat disimpulkan *game* edukasi merupakan salah satu bentuk *game* yang berguna untuk dapat menunjang proses belajar-mengajar secara lebih menyenangkan dan kreatif, dan digunakan untuk memberikan pengajaran atau menambah pengetahuan penggunanya melalui suatu media yang menarik.

2.2.5.1 Kriteria *Game* Edukasi

Menurut Hurd dan Jenuings, perancang yang baik haruslah memenuhi kriteria dari *education game* itu sendiri. Berikut ini adalah beberapa kriteria dari sebuah *education game*, yaitu [13] :

1. Nilai Keseluruhan (*Overall Value*)

Nilai keseluruhan dari suatu game terpusat pada desain dan panjang durasi game. Aplikasi ini dibangun dengan desain yang menarik dan interaktif. Untuk penentuan panjang durasi, aplikasi ini menggunakan fitur *timer*.

2. Dapat Digunakan (*Usability*)

Mudah digunakan dan diakses adalah poin penting bagi pembuat *game*. Aplikasi ini merancang sistem dengan *interface* yang *user friendly* sehingga user dengan mudah dapat mengakses aplikasi.

3. Keakuratan (*Accuracy*)

Keakuratan diartikan sebagai bagaimana kesuksesan model/gambaran sebuah *game* dapat dituangkan ke dalam percobaan atau perancangannya. Perancangan aplikasi ini harus sesuai dengan model *game* pada tahap perencanaan.

4. Kesesuaian (*Appropriateness*)

Kesesuaian dapat diartikan bagaimana isi dan desain *game* dapat diadaptasikan terhadap keperluan user dengan baik. Aplikasi ini menyediakan menu dan fitur yang diperlukan user untuk membantu pemahaman user dalam menggunakan aplikasi.

5. Relevan (*Relevance*)

Relevan artinya dapat mengaplikasikan isi *game* ke target user. Agar dapat relevan terhadap user, sistem harus membimbing mereka dalam pencapaian tujuan pembelajaran. Karena aplikasi ini ditujukan untuk anak-anak, maka desain antarmuka harus sesuai dengan nuansa anak-anak, yaitu menampilkan warna-warna yang ceria.

6. Objektifitas (*Objectives*)

Objektifitas menentukan tujuan user dan kriteria dari kesuksesan atau kegagalan. Dalam aplikasi ini objektivitas adalah usaha untuk mempelajari hasil dari permainan.

7. Umpan Balik (*Feedback*)

Untuk membantu pemahaman user bahwa permainan (*performance*) mereka sesuai dengan objek *game* atau tidak, *feedback* harus disediakan. Aplikasi ini menyajikan animasi dan efek suara yang mengindikasikan kesuksesan atau kegagalan permainan.

2.2.6 *Game Design*

Game design merupakan proses menciptakan konten dan aturan permainan. Design game yang baik adalah proses menciptakan tujuan dimana seorang pemain merasa termotivasi untuk mencapainya dan seorang pemain harus mengikuti aturan saat dia membuat keputusan yang berarti dalam mengejar tujuan-tujuan tersebut. Menurut Brenda Brathwaite dan Ian Schreiber dalam bukunya *Challenges For Game Designers*, merancang game mencakup sebagai berikut [14]:

- a. *World Design*, membuat desain keseluruhan latarbelakang cerita, pengaturan, dan tema permainan.
- b. *System Design*, membuat desain aturan dan pola matematis yang mendasari dalam sebuah permainan memiliki aturan.
- c. *Content Design*, membuat desain karakter, item, teka-teki dan misi.
- d. *Game Writing*, membuat penulisan dialog, *text* dan cerita dalam dunia permainan.
- e. *Level Design*, menciptakan level dalam permainan berdasarkan tampilan permainan dan penempatan objek hingga tantangan dalam tampilan permainan.
- f. *User Interface*, membuat desain yang terdiri dari dua hal, yaitu bagaimana pemain dapat berinteraksi dengan permainan dan bagaimana pemain menerima informasi dan mengambil hasil dari permainan.

2.2.7 *Game Engine*

Game Engine adalah sebuah perangkat lunak yang dirancang untuk membuat sebuah *game*. Sebuah *game engine* biasanya dibangun dengan mengenkapsulasi beberapa fungsi standar yang umum digunakan dalam pembuatan sebuah *game*. Sebagian besar *game engine* umumnya berupa *library* atau sekumpulan fungsi-fungsi yang penggunaannya dipadukan dengan Bahasa pemrograman. Pada beberapa kasus *game engine* kadang pula disebut sebagai *middleware*. Hal ini dikarenakan *game engine* digunakan sebagai perantara antara

bahasa pemrograman dengan format data dari berbagai perangkat lunak pembuat *asset*. Sebuah *game engine* biasanya dilengkapi dengan fungsi-fungsi grafis atau komputasi dasar yang jarang diketahui oleh *developer* [15].

2.3 Scripting

Scripting adalah kode yang digunakan untuk proses diotomatisasi yang seharusnya dilakukan langkah demi langkah oleh seorang pengembang. Maka *scripting* merupakan pengkodean dalam membuat suatu program, di Unity pengkodean dapat menggunakan Mono yang merupakan IDE di dalam Unity, sehingga kita dapat langsung mengintegrasikan script yang kita buat kedalam Unity dengan cepat.

Sehingga dapat memudahkan para programmer untuk melakukan pengkodean terhadap apa yang akan ia bangun dengan menggunakan berbagai jenis *tools* yang telah disediakan oleh *software* penyediaanya yang dapat terintegrasi secara langsung.

2.4 Storyboard

Storyboard adalah sketsa gambar yang disusun berurutan sesuai naskah, storyboard dapat menyampaikan ide cerita kepada orang lain dengan lebih mudah, karena dapat menggiring khayalan seseorang mengikuti gambar-gambar yang tersaji, sehingga menghasilkan persepsi yang sama pada ide cerita [13].

Maka dari itu *storyboard* berfungsi untuk memudahkan dalam membuat dan memahami sebuah alur cerita dan *storyboard* juga sebagai suatu konsep dan ungkapan yang kreatif dalam menyampaikan suatu ide atau gagasan sehingga pesan yang akan disampaikan akan dapat tersampaikan dengan baik karena dibentuknya *storyboard*.

2.5 Unified Modeling Language (UML)

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

Unified Modeling Language (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 [16].

Unified Modeling Language (UML) adalah salah satu alat bantu yang sangat handal di dunia pengembangan sistem yang berorientasi objek. Hal ini disebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembang sistem untuk membuat cetak biru atas visi mereka dalam bentuk yang baku, mudah dimengerti serta dilengkapi dengan mekanisme yang efektif untuk berbagi (*sharing*) dan mengkomunikasikan rancangan mereka dengan yang lain [17].

Secara umum UML merupakan ‘bahasa’ untuk visualisasi, spesifikasi, 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 [18].

Dalam kerangka spesifikasi, UML menyediakan model-model yang tepat, tidak mendua-arti (ambigu), serta lengkap. Secara khusus, UML menspesifikasi langkah-langkah 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 [18].

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 penggunaannya, 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.

Berdasarkan pemaparan mengenai *Unified Modeling Language* (UML) dari beberapa sumber referensi, maka dapat disimpulkan UML merupakan alat bantu dalam melakukan pemodelan yang saling berhubungan secara langsung dalam pembangunan sebuah sistem agar lebih efektif.

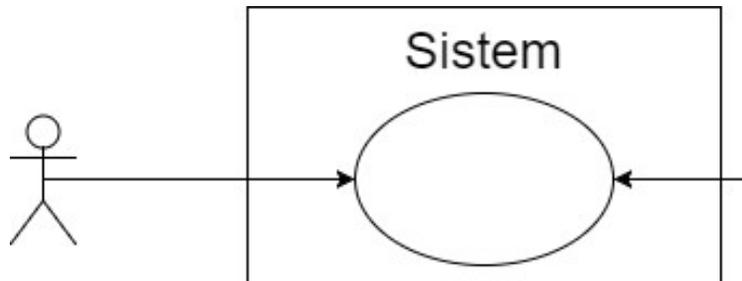
2.5.1 Use Case Diagram

Use Case adalah deskripsi fungsi dari sebuah sistem dari perspektif pengguna. Use Case bekerja dengan cara mendeskripsikan tipikal interaksi antara pengguna sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai [17].

Use Case diagram digunakan untuk menggambarkan analisis kebutuhan dari sistem dari level atas melalui fungsionalitas dari sistem dan interaksi diantara para aktor. Aktor adalah sesuatu yang berinteraksi dengan sistem.

Secara umum, tujuan dari use case diagram adalah sebagai berikut [17] :

1. Digunakan untuk mengumpulkan kebutuhan dari sebuah sistem
2. Untuk mendapatkan pandangan dari luar sistem
3. Untuk mengidentifikasi faktor yang mempengaruhi sistem baik internal maupun eksternal
4. Untuk menunjukkan interaksi dari para aktor dari sistem Ilustrasi dari aktor, use case dan *boundary* dapat dilihat pada gambar 2.11.



Gambar 2.11 Use Case Model

2.5.2 Activity Diagram

Activity Diagram adalah bagian penting dari UML yang menggambarkan aspek dinamis dari sistem. Logika procedural, proses bisnis, dan aliran kerja suatu bisnis bias dengan mudah dideskripsikan dalam *activity* diagram. *Activity* diagram memiliki peran seperti halnya *flowchart*, akan tetapi perbedaannya adalah *activity* diagram bisa mendukung perilaku paralel [17].

Tujuan dari *activity* diagram adalah untuk menangkap tingkah laku dinamis dari sistem dengan cara menunjukkan aliran pesan dari satu aktifitas ke aktifitas lainnya. Secara umum tujuan *activity* diagram adalah sebagai berikut [17]:

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

2.5.3 Class Diagram

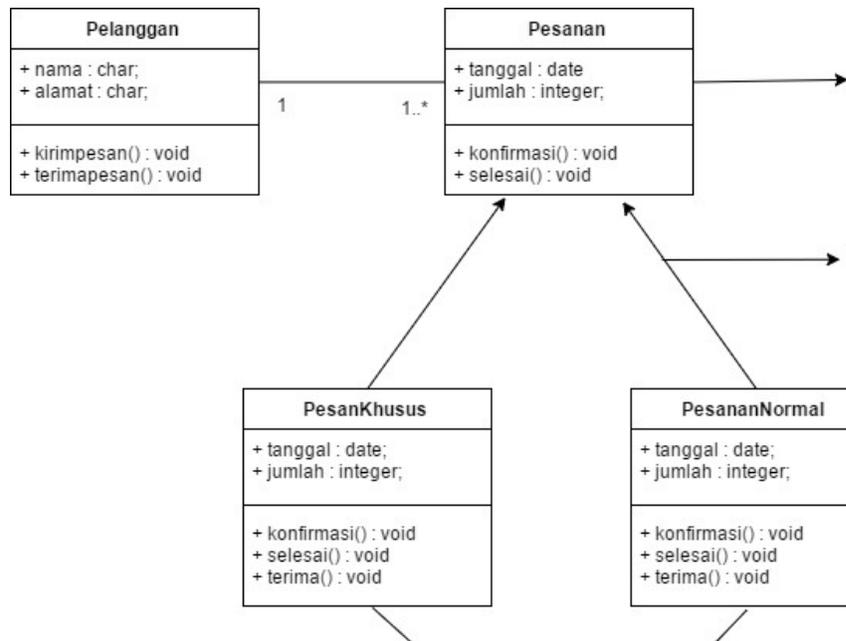
Class Diagram adalah diagram statis yang mewakili pandangan statis dari suatu aplikasi. Class diagram tidak hanya digunakan untuk memvisualisasikan, menggambarkan, dan mendokumentasikan berbagai aspek sistem tetapi juga untuk membangun kode eksekusi (*executable code*) dari aplikasi perangkat

lunak. Class diagram menunjukkan koleksi kelas, antarmuka, asosiasi, kolaborasi, dan constraint. Class diagram juga dikenal sebagai diagram struktural [17].

Tujuan dari class diagram adalah untuk memodelkan pandangan statis suatu aplikasi. Secara lebih rinci tujuannya adalah sebagai berikut [17] :

1. Analisis dan desain pandangan statis aplikasi.
2. Menjelaskan tanggung jawab suatu sistem.
3. Basis untuk diagram komponen dan penyebaran (deployment)
4. *Forward and reverse engineering*

Contoh class diagram dapat dilihat pada gambar 2.12.

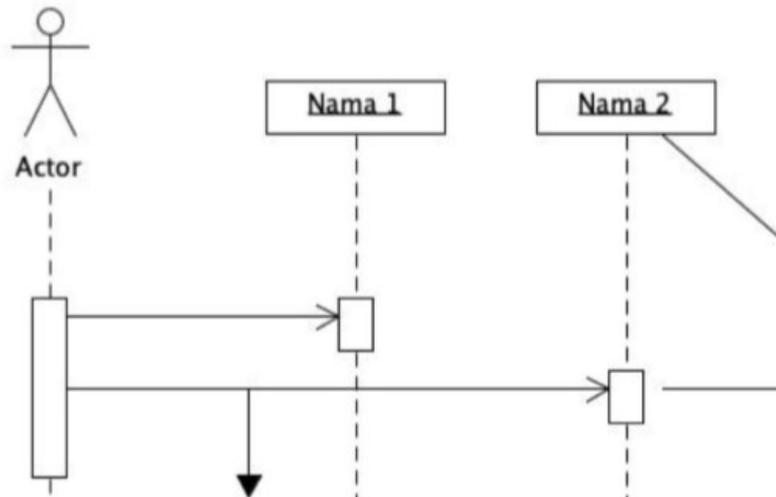


Gambar 2.12 Contoh Class Diagram Sistem Pemesanan

2.5.4 Sequence Diagram

Sequence Diagram digunakan untuk menggambarkan perilaku pada sebuah scenario. Diagram ini menunjukkan sejumlah contoh obyek dan message (pesan) yang diletakkan diantara obyek-obyek ini di dalam use case.

Komponen utama *sequence* diagram terdiri atas obyek yang dituliskan dengan kotak segiempat bernama. *Message* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan progress *vertical*. Simbol – simbol yang ada pada *sequence* diagram dapat dilihat pada gambar 2.13.



Gambar 2.13 Simbol-simbol yang ada pada *sequence diagram*

2.5.5 Bangunan Dasar UML

Metodologi UML menggunakan 3 bangunan dasar untuk mendeskripsikan sistem/perangkat lunak yang akan dikembangkan, yaitu [18] :

1. Sesuatu (*Things*).
Ada 4 macam '*things*' dalam UML, yaitu :
 - a. *Structural Things*.
 - b. *Behavioral Things*.
 - c. *Grouping Things*.
 - d. *Annotational Things*.
2. Relasi (*Relationship*).

Yang dimaksud *relationship* adalah hubungan-hubungan yang terjadi antarelemen dalam UML. Hubungan-hubungan ini penting sekali dalam UML.

Dapat dikatakan, tidak mungkin membuat model-model UML tanpa *relationship* ini.

3. *Diagrams*.

Setiap sistem yang kompleks seharusnya bisa dipandang dari sudut yang berbeda-beda sehingga kita bisa mendapatkan pemahaman secara menyeluruh.

Secara umum UML diterapkan dalam pengembangan sistem/perangkat lunak berorientasi objek sebab metodologi UML ini umumnya memiliki keunggulan-keunggulan, yaitu [18]:

1. *Uniformity*. Dengan metodologi UML (atau metodologi berorientasi objek pada umumnya), para pengembang cukup menggunakan 1 metodologi dari tahap analisis hingga perancangan. UML juga memungkinkan kita merancang komponen antarmuka pengguna (*User Interface*) secara terintegrasi bersama dengan perancangan perangkat lunak sekaligus dengan perancangan basis data.
2. *Understandability*. Dengan metodologi ini kode yang dihasilkan dapat diorganisasi kedalam kelas-kelas yang berhubungan dengan masalah sesungguhnya sehingga lebih mudah dipahami siapa pun juga.
3. *Stability*. Kode program yang dihasilkan relative stabil sepanjang waktu sebab sangat mendekati permasalahan sesungguhnya dilapangan.
4. *Reusability*. Dengan metodologi perorientasi objek, dimungkinkan penggunaan ulang kode, sehingga pada gilirannya akan sangat mempercepat waktu untuk pengembangan perangkat lunak (atau sistem informasi).

2.5 Pemrograman Berorientasi Objek

Object Oriented Programming (OOP) atau pemrograman berorientasi Objek adalah suatu cara baru dalam berfikir serta berlogika dalam menghadapi masalah-masalah yang akan dicoba-atasi dengan bantuan komputer. OOP, tidak seperti sebelumnya (pemrograman terstruktur), mencoba melihat permasalahan lewat pengamatan dunia nyata dimana setiap objek adalah entitas tunggal yang memiliki kombinasi struktur data dan fungsi tertentu. Ini kontras dengan pemrograman

terstruktur dimana struktur data dan fungsi didefinisikan secara terpisah dan tidak berhubungan secara erat [18].

Pada perkembangannya, filosofi OOP menciptakan sinergi yang luar biasa sepanjang siklus pengembangan perangkat lunak (perencanaan, analisis, perancangan, implementasi, serta pengujian) sehingga dapat diterapkan pada perancangan sistem secara umum; menyangkut perangkat lunak, perangkat keras serta sistem informasi secara keseluruhan [18].

Objek adalah konsep atau abstraksi tentang sesuatu yang memiliki arti bagi aplikasi yang akan kita kembangkan. Objek biasanya adalah kata benda, namun objek dalam konteks OOP bukan hanya objek nyata yang bisa diraba dan dilihat secara kasat mata seperti Anda, saya, mobil, pesawat terbang, sapi, kuda, gitar, buku, tape-recorder, komputer dan sebagainya, namun juga menyangkut entitas-entitas konseptual seperti rumusan persamaan kuadrat, liberalism, marxisme, dan sebagainya [18].

Secara spesifik, pengertian “berorientasi objek” berarti bahwa kita mengorganisasi perangkat lunak sebagai kumpulan dari objek tertentu yang memiliki struktur data dan perilakunya. Hal ini yang membedakan dengan pemrograman konvensional dimana struktur data dan perilakunya hanya berhubungan secara terpisah [19].

Setiap objek adalah nyata dan dapat dibedakan satu dari yang lainnya. Sekalipun 2 objek bernama sama, namun ia tetap dapat dipisahkan. Walaupun satu objek adalah kembar misalnya, kita tetap bisa membedakan satu individu terhadap individu yang lainnya. Misalnya, untuk objek kembar, sesuatu yang membedakannya adalah (misalnya) Namanya, sehingga objek yang satu memiliki identitas yang berbeda dengan yang lainnya.

Maka disimpulkan metodologi berorientasi objek merupakan suatu pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya dan metodologi berorientasi objek merupakan suatu cara bagaimana perangkat lunak

dibangun melalui pendekatan objek secara sistematis. Metode berorientasi objek didasarkan pada penerapan prinsip-prinsip pengelolaan kompleksitas. Metode berorientasi objek meliputi rangkaian aktivitas analisis berorientasi objek, perancangan berorientasi objek, pemrograman berorientasi objek dan pengujian berorientasi objek.

2.5.5 Pengembangan Berorientasi Objek

Pengembangan berorientasi objek merupakan cara berfikir baru tentang perangkat lunak berdasarkan abstraksi yang terdapat dalam dunia nyata. Dalam konteks pengembangan menunjuk pada bagian awal dari siklus hidup pengembangan sistem, yaitu survei, analisis, desain, implementasi dan pemeliharaan sistem. Hal yang lebih penting dalam pengembangan berorientasi objek adalah konsep mengidentifikasi dan mengorganisasi domain aplikasi dibandingkan dengan fokus penggunaan Bahasa pemrograman, berorientasi objek atau tidak [19].

2.5.6 Karakteristik Metodologi Berorientasi Objek

Metodologi pengembangan sistem berorientasi objek mempunyai tiga karakteristik utama, yaitu [19] :

a. *Encapsulation.*

Encapsulation (pengkapsulan) merupakan dasar untuk pembatasan ruang lingkup program terhadap data yang diproses. Data dan prosedur atau fungsi dikemas dalam Bersama-sama dalam suatu objek, sehingga prosedur atau fungsi lain dari luar tidak dapat mengaksesnya. Data terlindung dari prosedur atau objek lain kecuali prosedur yang berada dalam objek itu sendiri.

b. *Inheritance.*

Inheritance (pewarisan) adalah Teknik yang menyatakan bahwa anak dari obje akan mewarisi data/atribut dan metoda dari induknya langsung. Atribut dan metoda dari objek induk diturunkan kepada anak objek, demikian seterusnya. Pendefinisian objek dipergunakan untuk membangun suatu hirarki dari objek turunannya, sehingga tidak perlu membuat atribut dan

metoda lagi pada anaknya, karena telah mewarisi sifat induknya. *Inheritance* mempunyai arti bahwa atribut dan operasi yang dimiliki bersama di antara kelas yang mempunyai hubungan secara hirarki. Suatu kelas dapat ditentukan secara umum, kemudian ditentukan secara spesifik menjadi subkelas. Setiap subkelas mempunyai hubungan atau mewarisi semua sifat yang dimiliki oleh kelas induknya, dan ditambah dengan sifat unik yang dimilikinya. Sifat yang dimiliki oleh kelas induknya tidak perlu diulang dalam setiap subkelas.

c. *Polymorphism*.

Polymorphism (polimorfisme) yaitu konsep yang menyatakan bahwa sesuatu dapat mempunyai bentuk dan perilaku berbeda. Polimorfisme mempunyai arti bahwa operasi yang sama mungkin mempunyai perbedaan dalam kelas yang berbeda. Operasi *move* mungkin mempunyai perbedaan dalam kelas *windows* atau kendaraan bermotor. Suatu implementasi yang spesifik dari suatu operasi dari kelas tertentu disebut metoda. Karena operator perorientasi objek adalah bersifat polimorfisme, mungkin dapat mempunyai lebih dari satu metoda.

3.6.1 Paradigma Berorientasi Objek

Fokus utama ditemukannya pendekatan berorientasi objek adalah karena ditemukannya kekurangan-kekurangan pada pendekatan terstruktur. Biaya pengembangan perangkat lunak berkembang sesuai dengan berkembangnya, keinginan/kebutuhan pengguna, pemeliharaan yang sukar, lamanya penyelesaian suatu proyek, jangka waktu penyelesaian proyek yang hampir selalu terlambat, biaya pengembangan perangkat lunak yang sangat tinggi, dan sebagainya [18].

Pendekatan berorientasi objek membuat data terbungkus pada setiap fungsi/prosedur dan melindunginya terhadap perubahan tidak dikehendaki dari fungsi yang berada diluar.

Beberapa karakteristik yang menjadi ciri-ciri dari pendekatan berorientasi objek adalah [18]:

1. Pendekatan lebih pada data dan bukannya pada prosedur/fungsi.
2. Program besar dibagi pada apa yang dinamakan objek-objek.

3. Struktur data dirancang dan menjadi karakteristik dari objek-objek.
4. Fungsi-fungsi yang mengoperasikan data tergabung dalam suatu objek yang sama.
5. Data tersembunyi dan terlindung dari fungsi/prosedur yang ada di luar.
6. Objek-objek dapat saling berkomunikasi dengan saling mengirim *message* (pesan) satu sama lain.
7. Pendekatan adalah dari bawah ke atas (*bottom up approach*).

2.6 Metode *Collision Detector*

Metode collision detection merupakan salah satu metode yang bisa digunakan untuk memeriksa suatu kondisi objek apakah terjadi tubrukan objek atau tidak, ketika terjadi tubrukan objek maka beberapa proses bisa dieksekusi dan menghasilkan kembali berupa informasi yang diketahui oleh pengguna sistem. Setiap permainan menerapkan collision detection (deteksi tabrakan), baik itu dalam hal tabrakan antara sprite dengan spritemaupun antara sprite dengan peluru dan lain-lain. Proses collision dapat dibagi menjadi dua kategori dasar, yaitu collision detection dan collision response, dengan jarak respon yang telah diaplikasikan secara spesifik. Terdapat banyak sekali jenis dari collision detection itu sendiri [20].

Algoritma *collision detection* merupakan proses pengecekan apakah dua buah objek spasial saling bertumpuk atau tidak. Jika ternyata ada paling sedikit dua buah objek yang bertumpuk, maka kedua objek tersebut dikatakan saling bertumpukkan. Pada ruang spasial dua dimensi objek yang bertumpuk berarti objek spasialnya beririsan. *Collision detection* merupakan teknik deteksi tabrakan untuk mengetahui objek-objek apa saja yang bersentuhan dalam bidang koordinat tertentu [20].

Metode ini menggunakan sensor pemicu (*trigger*) untuk mendeteksi keberadaan objek ketika berada di lingkungan virtual. Mekanisme yang mendukung fitur ini adalah area pemicu (sensor) yang ditambahkan di sekeliling benda atau karakter. Sensor yang ditambahkan akan dihitung secara matematis

untuk menentukan apakah sensor mengenai objek lain seperti tembok (*unwalkable object*), kendaraan lain, atau objek-objek lain yang berada di lingkungan virtual.

2.7 Metode Pengujian

Metode pengujian merupakan metode – metode yang digunakan dalam pengujian perangkat lunak yang dibangun. Pengujian dilakukan untuk memberikan evaluasi atas perangkat lunak yang dibangun, apakah telah sesuai dengan kebutuhan yang menjadi masalah atau belum sesuai.

Berikut merupakan beberapa metode pengujian yang dilakukan pada penelitian ini.

2.7.1 Metode *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 [21]. *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 [21] :

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 [22] :

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 penggunaan, 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.

2.7.2 Skala Likert

Skala likert adalah skala pengukuran yang dikembangkan oleh Likert. Skala likert mempunyai empat atau lebih butir-butir pertanyaan 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 rata-rata, 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 [23]:

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

2.7.3. Skala Guttman

Skala Guttman merupakan skala kumulatif. Skala ini hanya mengukur suatu dimensi dari variable multidimensi. Skala guttman atau skala skalogram ini sangat

baik untuk meyakinkan peneliti tentang kesatuan dimensi dari sikap atau sifat yang diteliti, yang sering disebut dengan atribut universal. Pada skala guttman ada beberapa pertanyaan yang diurutkan secara hierarkis untuk melihat sikap tertentu seseorang [24].

Jadi skala guttman adalah skala yang digunakan untuk memberikan jawaban yang bersifat jelas, tegas, dan konsisten. Berikut ini adalah beberapa contoh dari skala guttman :

- a. Yakin – Tidak Yakin
- b. Benar – Salah
- c. Positif – negatif
- d. Pernah – belum pernah
- e. Setuju – tidak setuju

Berdasarkan itu maka data yang diperoleh berupa data interval atau rasio dikotomi (dua alternatif berbeda). Biasanya skala guttman diberikan dengan bentuk pilihan ganda, namun bisa juga dibuat dalam bentuk daftar cek. Jawaban responden dapat berupa skor tertinggi bernilai 1 dan skor terendah bernilai 0 sebagaimana yang dapat dilihat pada tabel 2.1.

Tabel 2.1. Skala Guttman

Keterangan	Skor
Yakin / benar / positif / pernah / setuju	1
Tidak yakin / salah / negatif / belum pernah / tidak setuju	0

2.8 Unity

Unity merupakan game engine, yaitu *software* yang digunakan untuk memudahkan dalam membuat game. Unity telah menyediakan berbagai macam *tools* yang dapat membantu dalam membuat game dan di unity dapat menambahkan skrip untuk mengatur jalannya game [25]. 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 mobil 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.

2.9 CorelDRAW

Coreldraw adalah suatu aplikasi desain grafis yang dipakai untuk membuat desain vektor, logo dan layout halaman. Untuk membuat suatu desain grafis, Coreldraw merupakan salah satu *software* yang sangat populer di Indonesia [26].

Coreldraw sering digunakan oleh para pembuat desainer baru maupun yang sudah berpengalaman karena yang penggunaannya cukup mudah di lakukan dan dipelajari. *Tools-tools* yang sering digunakan didalam coreldraw adalah *Freehand tools* merupakan sebuah *tools* yang digunakan untuk membuat sebuah garis yang dapat saling terhubung dan menciptakan suatu bentuk atau karya dan *Shape tools* dapat membantu pengguna untuk menata suatu garis atau gambar agar menjadi lebih rapi dan beberapa *tools* pembantu lain.

2.10 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 [27].

Sehingga android merupakan sebuah *platform* yang saat ini cukup populer digunakan oleh masyarakat umum karena android merupakan sebuah sistem operasi yang memiliki perkembangan yang pesat dan memiliki inovasi-inovasi baru karena banyak pengembang atau developer yang saling berlomba untuk mengembangkan sebuah software yang mereka buat dan dapat digunakan di android.

2.11 PNG (*Portable Network Graphics*)

PNG adalah salah satu format penyimpanan citra yang menggunakan metode pemadatan yang tidak menghilangkan bagian dari citra tersebut (*lossless compression*). Format PNG diperkenalkan untuk menggantikan format penyimpanan citra GIF. Citra dengan *format* PNG mempunyai faktor kompresi yang lebih baik dibandingkan citra dengan *format* GIF. Satu fasilitas dari GIF yang tidak terdapat pada *format* PNG adalah dukungan terhadap penyimpanan multi-citra untuk keperluan animasi. Untuk keperluan pengolahan citra, *format* PNG dapat dijadikan alternative selama proses pengolahan citra, karena *format* ini selain tidak menghasilkan bagian dari citra yang sedang diolah, sehingga penyimpanan berulang dari citra tidak akan menurunkan kualitas citra, juga PNG diciptakan untuk menggantikan keberadaan GIF karena masalah lisensi.

Format PNG dibuat sebagai alternative lain dari *format* GIF karena masalah lisensi. *Format* PNG dibuat sebagai alternative lain dari format GIF, dan digunakan untuk menyimpan berkas dengan kedalaman *24bit* serta memiliki kemampuan untuk menghasilkan *background* transparan dengan pinggiran yang halus. *Format* PNG menggunakan metode kompresi *lossless* untuk menampilkan citra *24 bit* atau warna-warna. *Format* ini mendukung transparansi di dalam *alpha channel*. *Format* PNG sangat baik digunakan pada dokumen *online*, dan mempunyai dukungan warna yang lebih baik saat dicetak [28].

2.12 Bahasa Pemrograman C#

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 Microsystems, Microsoft pun menghentikan pengembangan J++, dan beralih untuk membuat pengganti J++, kompilernya dan mesin virtual sendiri dengan menggunakan sebuah bahasa pemrograman 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, properti, kejadian (event), metadata, dan konversi antara tipe-tipe sederhana dan juga objek.