PERANCANGAN GAME PLATFORMER PEMBURU KOIN MENGGUNAKAN GODOT ENGINE

SKRIPSI

Oleh:
Ferdi
171510069

PROGRAM STUDI SISTEM INFORMASI
FAKULTAS TEKNIK DAN KOMPUTER
UNIVERSITAS PUTERA BATAM
TAHUN 2022

PERANCANGAN GAME PLATFORMER PEMBURU KOIN MENGGUNAKAN GODOT ENGINE

SKRIPSI

Untuk memenuhi salah satu syarat
memperoleh gelar Sarjana

Oleh:
Ferdi
171510069

PROGRAM STUDI SISTEM INFORMASI
FAKULTAS TEKNIK DAN KOMPUTER
UNIVERSITAS PUTERA BATAM
TAHUN 2022

SURAT PERNYATAAN ORISINALITAS

Yang bertanda tangan di bawah ini saya:

Nama	: Ferdi
NPM	: 171510069
Fakultas	: Teknik dan Komputer
Program Studi	: Sistem Informasi

Menyatakan bahwa **"SKRIPSI"**. Yang saya buat dengan judul

PERANCANGAN GAME PLATFORMER PEMBURU KOIN MENGGUNAKAN GODOT ENGINE

Adalah hasil karya sendiri dan bukan "duplikasi" dari karya orang lain. Sepengetahuan saya, didalam naskah proposal ini tidak terdapat karya ilmiah atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis dikutip didalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila ternyata di dalam naskah proposal ini dapat dibuktikan terdapat unsur-unsur PLAGIASI, saya bersedia naskah proposal ini digugurkan dan proposal yang saya peroleh dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku.

Demikian pernyataan ini saya buat dengan sebenarnya tanpa ada paksaan dari siapapun.

Batam, 22 Januari 2022

Ferdi

171510069

PERANCANGAN GAME PLATFORMER PEMBURU KOIN MENGGUNAKAN GODOT ENGINE

SKRIPSI

Untuk memenuhi salah satu syarat
memperoleh gelar Sarjana

Oleh
Ferdi
171510069

Telah disetujui oleh Pembimbing pada tanggal
seperti tertera di bawah ini

Batam, 22 Januari 2022

Sasa Ani Arnomo, S.Kom., M.SI.
Pembimbing

ABSTRAK

Aplikasi *video game* sebagai perangkat lunak memerlukan proses pengembangan (*development*). Perbedaan yang signifikan dan mudah diketahui antara aplikasi biasa dengan aplikasi *video game* adalah aplikasi *video game* diperlukannya sebuah *game engine*. Perangkat *software* yang nantinya akan digunakan sebagai alat untuk merancang sebuah *video game*. Pada umumnya pengguna awam dan pemula memerlukan *game engine* untuk memudahkan perancangan sebuah *video game*. Jika pengembang memiliki kemampuan yang mumpuni, maka pengembang tersebut dapat merancang *game engine* sendiri. *Video game* merupakan salah satu sarana hiburan yang saat ini digunakan oleh banyak orang. Bukan hanya sarana hiburan, video game juga dapat digunakan untuk sarana pembelajaran dan mengasah kemampuan berpikir pemain.kebutuhan video game. Tujuan penelitian guna membangun dan merancang sebuah *video game* berjenis *platformer* menggunakan Godot Engine. *Software* pendukung yang digunakan dalam pengembangan *video game* dalam penelitian ini menggunakan Godot Engine. Hasil penelitian yang berjudul Pemburu Koin, dimana pemain mengendalikan karakter yang bernama blanc untuk mengumpulkan koin-koin yang bertebaran di hutan terpencil. Perancangan dan pengembangan dapat menggunakan Godot Engine sebagai sarana dari game engine yang lain. Proses perancangan meliputi penggunaan aset dan pemograman pada *game engine* Godot Engine. Godot Engine dipilih dengan alasan agar tidak dikenakan biaya royalty. Aset dapat dibuat menggunakan aplikasi gratis dan *open-source* agar perancangan tidak memerlukan biaya pembelian aplikasi yang terlalu tinggi.

Kata kunci: *Video Game; Platformer;* Godot Engine

ABSTRACT

Video game applications as software require a development process. A significant and easily recognizable difference between ordinary applications and video game applications is that video game applications require a game engine. Software that will be used as a tool to design a video game. In general novice users and beginners generally need a game engine to facilitate the design of a video game. If the developer has qualified capabilities, then the developer can design his own game engine. Video games are one of the entertainment facilities that are currently used by many people. Not only a means of entertainment, video games can also be used as a means of learning and sharpening players' thinking skills. Video game needs. The research objective is to build and design a platformer type video game using Godot Engine. The supporting software used in the development of video games in this study uses the Godot Engine. The results of the research entitled Coin Hunter, where players control a character named blanc to collect coins scattered in a remote forest. Design and development can use Godot Engine as a tool from other game engines. The design process includes the use of assets and programming on the Godot Engine game engine. Godot Engine was chosen for the reason that it is not subject to royalty fees. Assets can be created using free and open-source applications so that the design does not require exorbitant application purchases.

Keywords: *Video Game; Platformer;* Godot Engine

KATA PENGANTAR

Segala puji bagi Tuhan Yang Maha Esa yang telah melimpahkan segala rahmat dan karuniaNya, sehingga penulis dapat menyelesaikan laporan tugas akhir yang merupakan salah satu persyaratan untuk menyelesaikan program studi strata satu (S1) pada Program Studi Sistem Informasi Universitas Putera Batam.

Penulis menyadari bahwa skripsi ini masih jauh dari sempurna. Karena itu, kritik dan saran akan senantiasa penulis terima dengan senang hati. Dengan segala keterbatasan, penulis menyadari pula bahwa skripsi ini takkan terwujud tanpa bantuan, bimbingan, dan dorongan dari berbagai pihak. Untuk itu, dengan segala kerendahan hati, penulis menyampaikan ucapan terima kasih kepada:

1. Ibu Dr. Nur Elfi Husda, S.Kom., M.SI. selaku Rektor Universitas Putera batam;
2. Bapak Muhammat Rasid Ridho, S.Kom., M.SI. selaku Ketua Program Studi Sistem Informasi;
3. Bapak Welly Sugianto, S.T., M.M. selaku Dekan Fakultas Teknik dan Komputer;
4. Bapak Sasa Ani Arnomo, S.Kom., M.SI. selaku pembimbing Skripsi pada Program Studi Sistem Informasi Universitas Putera Batam;
5. Bapak Amrizal, S.Kom., M.SI. selaku dosen pembimbing akademik;
6. Dosen Sistem Informasi dan Staff Universitas Putera Batam, yang telah memberikan ilmu dan pengetahuan kepada penulis;
7. Kedua Orang Tua saya yang setia memberikan dukungan semangat, dorongan serta doa;
8. Rekan seperjuangan: Ricky dan alumni yang memberikan semangat kepada penulis untuk menyelesaikan tugas akhir ini;

Semoga Tuhan Yang Maha Esa membalas kebaikan dan karma baik mereka dapat berbuah tepat pada waktunya, Sadhu.

Batam, 22 Januari 2022

Ferdi

DAFTAR ISI

Halaman

HALAMAN SAMPUL DEPAN ..... **i**
HALAMAN JUDUL ..... **ii**
SURAT PERNYATAAN ORISINALITAS ..... **iii**
HALAMAN PENGESAHAN ..... **iv**
ABSTRAK ..... **v**
ABSTRACT ..... **vi**
KATA PENGANTAR ..... **vii**
DAFTAR ISI ..... **viii**
DAFTAR GAMBAR ..... **x**
DAFTAR TABEL ..... **xi**
BAB I PENDAHULUAN
1.1 Latar Belakang 1
1.2 Identifikasi Masalah 6
1.3 Batasan Masalah 7
1.4 Rumusan Masalah 7
1.5 Tujuan Penelitian 8
1.6 Manfaat Penelitian 8
1.6.1 Manfaat Teoritis 8
1.6.2 Manfaat Praktis 8
BAB II TINJAUAN PUSTAKA
2.1 Tinjauan Teori Umum 9
2.1.1 Sistem Informasi 9
2.1.2 Perancangan dan Pembangunan Perangkat Lunak 9
2.1.3 Aliran Sistem Informasi 13
2.1.4 *Unified Model Languange* 15
2.2 Teori Khusus 26
2.2.1 *Video Game* 26
2.2.2 *Game Engine* 28
2.2.3 Genre *Platformer* 29
2.2.4 Godot Engine 30
BAB III METODE PENELITIAN
3.1 Desain Penelitian 32
3.2 Objek Penelitian 36
3.3 Analisis SWOT 36
3.4 Analisis Sistem yang Sedang Berjalan 38
3.5 Aliran Sistem Informasi yang Sedang Berjalan 38
3.6 Permasalahan yang sedang terjadi 39
3.7 Usulan Pemecahan Masalah 40
BAB IV PEMBAHASAN
4.1 Analisis Sistem yang Baru 41
4.1.1 Aliran Sistem Informasi yang Baru 42
4.1.2 *Use Case Diagram* 42
4.1.3 *Activity Diagram* 44
4.1.4 *Sequence Diagram* 47
4.1.5 *Class Diagram* 48

4.2 Desain Rinci 49
4.2.1 Desain Latar Belakang Cerita *Video Game* Pemburu Koin 49
4.2.2 Desain Karakter Protagonis: Pemburu Koin 50
4.2.3 Desain Lingkungan *Level* 51
4.2.4 Desain Latar Belakang Layar 52
4.2.5 Desain Objek yang Dapat Dikoleksi: Koin Emas 52
4.2.6 Desain *Graphical User Interface* dan *Heads-Up Display* 53
4.3 Rencana Implementasi 57
4.3.1 Jadwal Implementasi 58
4.3.2 Perkiraan Biaya Implementasi 59
4.1 Perbandingan Sistem 60
4.2 Analisis Produktivitas 61
4.2.1 Segi Efisiensi 62
4.2.2 Segi Efektivitas 62
BAB V SIMPULAN DAN SARAN
5.1 Simpulan 66
5.2 Saran 66
DAFTAR PUSTAKA
LAMPIRAN
Lampiran 1. Pendukung Penelitian
Lampiran 2. Daftar Riwayat Hidup
Lampiran 3. Surat Keterangan Penelitian

DAFTAR GAMBAR

Gambar 1.1 Contoh tampilan layar kerja pada Unreal 5
Gambar 1.2 game platformer "Cat Mario" 7
Gambar 2.1 Diagram pada UML 19
Gambar 2.2 Contoh video game genre Battle Royale 28
Gambar 2.3 Contoh video game genre simulasi 29
Gambar 2.4 Contoh game engine untuk pengembangan video game 31
Gambar 2.5 Tampilan awal Godot Engine 33
Gambar 3.1 Alur penelitian 34
Gambar 3.2 Alur model pengembangan waterfall 36
Gambar 3.3 Diagram alur permainan Cat Mario 41
Gambar 4.1 Diagram alur permainan "Pemburu Koin" 45
Gambar 4.2 Diagram use case untuk video game "Pemburu Koin" 46
Gambar 4.3 Activity Diagram untuk Use Case About 47
Gambar 4.4 Activity Diagram untuk Use Case New Game 48
Gambar 4.5 Activity Diagram untuk Use Case Quit 49
Gambar 4.6 Activity Diagram untuk Use Case Level 50
Gambar 4.7 Sequence Diagram untuk Use Case About 51
Gambar 4.8 Sequence Diagram untuk Use Case New Game 51
Gambar 4.9 Sequence Diagram untuk Level 52
Gambar 4.10 Sequence Diagram untuk Quit 52
Gambar 4.11 Class Diagram untuk video game Pemburu Koin 53
Gambar 4.12 Desain rintangan Pemburu Koin dengan latar belakang yang berada di hutan terpencil 54
Gambar 4.13 Karakter "Blanc" 55
Gambar 4.14 Blok tanah dan batu hutan 56
Gambar 4.15 Latar belakang Layar 57
Gambar 4.16 Koin yang dapat dikoleksi sebagai nilai tambahan 57
Gambar 4.17 Layar pada menu utama 59
Gambar 4.18 Menu pemilihan level yang muncul setelah pemain mengklik tombol new game 59
Gambar 4.19 Tampilan layar about setelah pemain mengklik tombol about 60
Gambar 4.20 Heads-up display yang merekam jumlah nilai pada level yang sedang dimainkan 61
Gambar 4.21 Pop-up layar game paused 62
Gambar 4.22 Tampilan layar game over 62
Gambar 4.23 Tampilan layar untuk penyelesaian level 63

DAFTAR TABEL

Tabel 2.1 Simbol-simbol yang akan digunakan dalam aliran sistem informasi ... 16
Tabel 2.2 Simbol yang terdapat pada use case diagram ... 20
Tabel 2.3 Simbol-simbol yang terdapat pada activity diagram ... 22
Tabel 2.4 Simbol-simbol yang terdapat pada sequence diagram ... 24
Tabel 2.5 Simbol class diagram ... 26
Tabel 3.1 Analisis SWOT ... 39
Tabel 4.1 Jadwal implementasi ... 65

BAB I

PENDAHULUAN

1.1 Latar Belakang

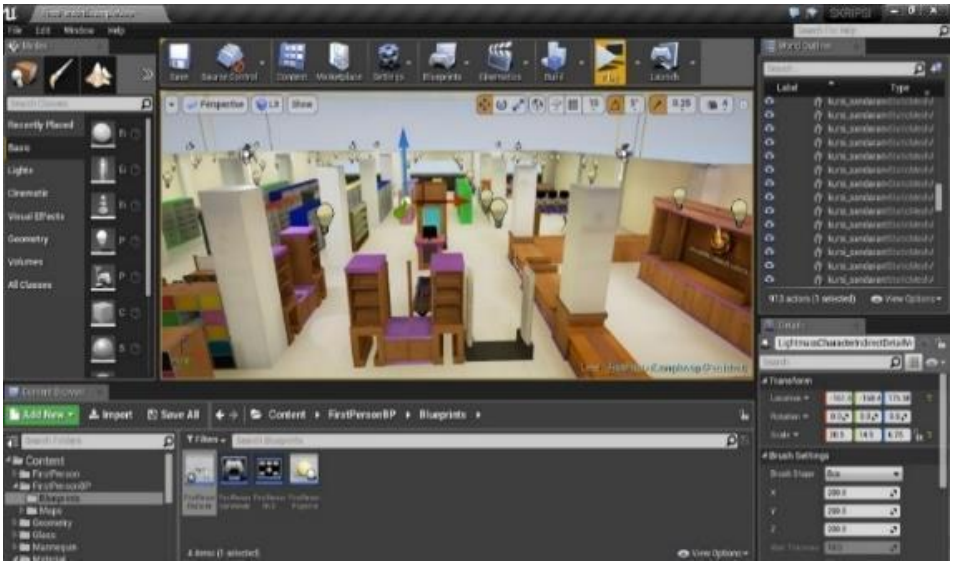
Dari kalangan masyarakat, telah tidak asing lagi mengenai rancang bangun serta pengembangan perangkat lunak. Banyak mas framework yang jumlah pemakai komputer atau *smartphone* untuk dapat menjalankan dan mengembangkan perangkat lunak (*software*) sendiri. Ada juga berbagai macam perangkat lunak selain Visual Studio, yang bisa dikembangkan menggunakan aplikasi perancangan itu. Misalnya seperti aplikasi *video game*, aplikasi *social media*, aplikasi manajemen, dan aplikasi yang lainnya.

Dalam sebuah perangkat lunak pada aplikasi *video game* membutuhkan prose pengambangan (*development*). Yang membedakan secara signifikan dan mudah diketahui antara aplikasi biasa dengan aplikasi *video game* adalah aplikasi *video game* membutuhkan *game engine*. Dimana *game engine* ini merupakan sebuah perangkat *software* yang nantinya akan digunakan sebagai alat untuk merancang sebuah *video game*. Biasanya bagi pemula umum dan pengguna awal membutuhkan *game engine* supaya lebih memudahkan dalam *video game*. Jika Pengembang (*developer*) memiliki kemampuan yang mumpuni, maka pengembang tersebut dapat merancang *game engine* sendiri. *Video game* termasuk salah satu alat hiburan yang banyak digunakan orang pada saat ini. *Video game* ini tidak hanya sebagai alat hiburan, akan tetapi bisa juga dijadikan sebagai pembelajaran dalam meningkatkan kemampuan berpikir bagi para penggunanya (Shihab et al.,113 :2019).

Sehingga munculah istilah Game Development Life Cycle (GDLC) yang merupakan penyesuain System Development Life Cycle (SDLC) agar lebih mudah diterapkan dalam pengembangan video game. (Mustofa et al., 2021)

Unreal Engine juga memiliki permasalahan yang berhubungan dengan lisensi. Sesuai pada situs resmi *Unreal Engine* dalam bagian *frequently asked question*, apabila seorang pengembang ingin menjual *video game* yang telah dibuatnya itu, maka pengembang tersebut perlu membayar sebagian persen dari hasil penjualan ke pemilik dari *Unreal Engine* ini. Apabila pengembang tidak *video game* hasil buatannya dijualkan, dengan begitu tidak perlu membayar keuntungan ke pemilik *Unreal Engine* ini (*Frequently Asked Questions - Unreal Engine*, n.d.). *Unreal Engine* termasuk perangkat lunak pengembangan game dalam membangun dan merancang *game*, visualisasi dan simulasi (Narwan et al.,743 :2021).

Berdasarkan beberapa artikel, terdapat kelebihan dan kelemahan Unity Engine. Salah satu kelemahan paling jelas menurut artikel *NewGenApps* adalah harus membeli lisensi versi *Pro* untuk mendapatkan semua fitur. Kelebihan yang disebutkan dalam artikel ini sejalan bagi pemula dan pengguna awam (*13 Pros & Cons to Know Before Choosing Unity 3D* -, n.d.).



Gambar 1.1 Contoh tampilan layar kerja pada Unreal

Selain aplikasi yang disebutkan sebelumnya dalam mengembangkan *video game* pada gambar 1.1, terdapat *game engine* selain Unreal Engine yang bersifat *open source*. Artinya, jika suatu saat pengembang (*developer*) berencana menjual hasil *video game* buatan sendiri, tidak perlu lagi membayar royalti. Aset serta fitur yang ada pada aplikasi tersebut tersedia secara gratis tanpa perlu lisensi dahulu. Selain itu, aplikasi tersebut tidak hanya mendukung pembuatan *video game* 2 (dua) dimensi, tetapi juga mendukung pembuatan *video game* 3 (tiga) dimensi, aplikasi yang peneliti sebutkan yaitu Godot Engine. Godot mempunyai bahasa tersendiri dalam *scripting*, yang dinamakan (GDScript) dan termasuk pengembangan dari Bahasa pemograman Python. Kemungkinan besar Godot telah cukup membantu dalam mewujudkannya. Lebih lagi komunitas *online*-nya bisa dibilang di golongkan aktif. Sesuai pada situs resmi Godot *engine*, aplikasi itu termasuk lisensi dari universitas Institut Teknologi Massachusetts (*Massachusetts*

Institute of Technology atau MIT) yang mana produk yang memiliki ikatan pada lisensi itu ialah produk yang gratis (Saltzer & Walden,94 :2020).

Disini peneliti menyimpulkan bahwa masih sedikit kalangan yang menggunakan Godot jika dibandingkan dengan *game engine* lainnya seperti Unity atau Unreal. Godot seharusnya mulai banyak digunakan karena Godot tidak meminta biaya royalti dibandingkan dengan *game engine* seperti Unity atau Unreal. Jika suatu saat pengembang berencana menjual hasil pengembangan *video game* tersebut, maka pilihan terbaiknya adalah dengan menggunakan Godot *Engine*, yang mana sebagian besar royaltinya tidak perlu dibagikan/membayar royalti kepada pemilik dari Godot Engine ini.

Berdasarkan hasil data statistik dari Steam terjadinya pergeseran minat dari game dua (2) dimensi ke game 3 dimensi, penulis menemukan salah satu video game 2d yang populer berjenis platformer yaitu Cat Mario. *Game* tersebut bisa dibilang cukup popular karena sudah pernah dimainkan oleh sekian banyak youtuber dan kalangan pecinta *game platformer*. Tetapi dari segi tingkat popularitasnya, jenis permainan yang bersifat *platformer* ini masih kalah jika dibandingkan dengan jenis *game* lainnya berdasarkan data statistik Steam seperti *genre Battle Royale*, *Multiplayer Online Battle Arena* (MOBA), *First Person Shooter* (FPS) dan sebagainya.

Gambar 1.2 *game platformer* **“**Cat Mario”

Video game pada gambar 1.2 termasuk salah satu contoh dari *game* yang memiliki jenis platformer serta bisa dikembangkan pada kualitas yang tinggi. Akan tetapi hal itu tidak bisa menjami *video game* tersebut bisa sukses. Ketenaran pada *video game* tergantung dari keberihakan dari pemain itu sendiri. Terdapat beberapa orang yang bermain *video game* pada jenis khusus, seperti *video game* yang memiliki tema pertualanga, perang, simulasi ataupun tema khusus yang lain. Kualitas pada *video game* tidak berpengaruh pada *game engine*. *Game engine* sendiri juga memiliki kelemahan dan kelebihan dengan tersendiri. Akan tetapi, terdapat faktor yang paling berpengaruh pada kualitas *video game* yaitu dari pengembang itu sendirinya. Dengan perkembangan zaman yang semakin maju, perangkat elektronik memiliki kemampuan dalam menampilkan objek yang bersifat tiga dimensi, dan juga mendorong munculnya *video game* yang bersifat tiga dimensi (Tambunan & Zetli,76 :2020).

Berawal dari ketertarikan serta berbekal ilmu dari berbagai sumber, peneliti mulai merancang sebuah *video game* berjenis *platformer* yang berwujud 2 (dua) dimensi. Permainan ini mengangkat *genre* yang sama dengan video game

pembanding yaitu *platformer*. Kumpulan ide serta referensi yang bakal diterapkan dalam *video game* itu ialah sebuah sifat petualang yang memiliki misi untuk mengumpulkan koin emas yang tersebar di berbagai posisi dalam sebuah hutan terpencil, selain itu juga karakter dapat menghindari ataupun melawan rintangan yang dihadapi. *Video game* yang dikembangkan penulis, untuk sementara hanya bisa dimainkan pada *platform* PC (*Personal Computer*) dan untuk pengendalian karakter menggunakan *Keyboard*. Pengembang menggunakan Godot *engine* sebagai aplikasi pendukung bagi pengguna yang ingin mengembangkan sebuah *video game*. Sesuai pada penjelasan latar belakang, maka dengan ini penulis berminat dalam membangun dan merancang serta mencatat prosesnya melalui skripsi ini yang berjudul "PERANCANGAN GAME PLATFORMER PEMBURU KOIN MENGGUNAKAN GODOT ENGINE".

1.2 Identifikasi Masalah

Sesuai dari penjelasan yang tertulis pada latar belakang, dengan begitu permasalahan bisa diidentifikasikan paad penelitian ini, diantaranya:

1. Penggunaan Godot engine, sebagai pilihan dari *game engine* yang digunakan pada *video game* Cat Mario yaitu, Open-syobon. Open-syobon memiliki keterbatasan teknologi dalam ketersediaan aset, fitur, pemilihan animasi karakter, serta logika dalam game tersebut saat dirilis.
2. Pada saat itu *video game* Cat Mario hanya dirilis pada konsol permainan Nintendo Wii, yang membatasi jangkauan kepada pemain yang tertarik, tetapi tidak memiliki konsol permainan tersebut.

3. Pengembangan *video game* yang memiliki jenis *platformer*, masih terlihat kalah dalam persaingan terhadap jenis *video game* yang lain.

1.3 Batasan Masalah

Untuk memperjelas suatu penelitian, dengan ini peneliti membatasi masalah dalam batasan masalahnya, diantaranya yaitu:

1. Pengembangan dalam *video game* membuat *assets* sendiri atau menggunakan *assets* yang gratis.
2. *Assets* berupa gambar menggunakan *pixel art*.
3. *Video game* yang berjenis *platformer* dalam 2 (dua) dimensi.
4. Permainan hanya dapat dijalankan pada konsol Nintendo Wii. Tidak mendukung perangkat selain konsol, seperti *Smartphone* dan perangkat yang lain.
5. Permainan bisa menggunakan perangkat keras *keyboard* atau *Joystick* sebagai pengendalian karakter.

1.4 Rumusan Masalah

Sesuai pada uraian permasalahan diatas, dengan begitu penelitin menyusun rumusan masalah, diantaranya:

1. Bagaimana merancang dan membangun *video game* yang berjenis *platformer* menggunakan Godot Engine?
2. *Assets* apa saja yang digunakan untuk *video game* berjenis *platformer* dengan Godot Engine?

1.5 Tujuan Penelitian

Adapun tujuan dari tugas akhir ini adalah pengembangan sebuah *video game* dan diuraikan seperti berikut:

1. Untuk merancang sebuah *video game* berjenis *platformer* menggunakan Godot Engine.
2. Untuk membangun sebuah *video game* berjenis *platformer* menggunakan Godot Engine.

1.6 Manfaat Penelitian

Terdapat beberapa manfaat teoritis serta manfaat praktis yang penulis sampaikan dari penelitian ini ialah:

1.6.1 Manfaat Teoritis

Yang menjadi manfaat teoritis pada penelitian ini yaitu:

1. Godot *engine* merupakan perangkat lunak pendukung yang dapat dijadikan sebagai sumber pengetahuan tentang pengembangan pada *video game*.
2. Godot *engine* dapat dijadikan sebagai acuan kepada penelitian selanjutnya bagi pembaca yang memiliki rencana menggunakannya.

1.6.2 Manfaat Praktis

Yang menjadi manfaat peaktis pada penelitian ini yaitu:

1. Sebagai materi inspirasi mengenai Godot *engine*, serta meningkatkan minat pembaca dalam mencoba mengembangkan *video game* atau sejenisnya.
2. Meningkatkan perkembangan pada *video game* buatan lokal di kalangan masyarakat.

BAB II

TINJAUAN PUSTAKA

2.1 Tinjauan Teori Umum

2.1.1 Sistem Informasi

Dapat dikatakan sistem ialah perkumpulan dari bagian yang saling terhubungan dengan maksud supaya dapat mencapai suatu tujuan. Apabila dalam suatu sistem memiliki bagian yang tidak dapat memberikan manfaat, dengan begitu bagian itu sudah pasti bukanlah bagian dari suatu sistem. Informasi termasuk pemrosesan suatu data yang dapat meningkatkan wawasan pada pengguna infrmasi itu. Dan informasi juga merupakan suatu data yang telah diolah menjadi suatu bentuk yang mempunyai makna dan manfaat untuk orang yang menerimanya pada saat melakukan pengambilan keputusan pada masa sekarang ini maupun masa yang akan datang (Aventinus, 2020).

Kemudian yang dimaksud dari sistem informasi yaitu penggabungan dari empat bagian utama. Dari keempat bagian utama itu diantaranya termasuk *hardware*, *software*, SDM yang terlatih dan juga infrastruktur (Ultariani et al., 2020).

2.1.2 Perancangan dan Pembangunan Perangkat Lunak

Suatu *software* dapat dikatakan baik dan efisien, apabila *software* tersebut bisa mempersiapkan seluruh kebutuhan yang dibutuhkan dalam memudahkan pengguna pada saat menyelesaikan permasalahan, atau berdasarkan pada perjanjian yang terdapat pada kontrak (Setyowardani et al., n.d.).

Sebuah *software* dapat dikatakan berkualitas apabila *software* tersebut bisa memuaskan sebagian besar bagi orang yang menggunakannya. Suatu produk dapat dikatakan memiliki kualitas apabila produk tersebut bisa memberikan kepastian terhadap produknya memenuhi mutu yang telah ditentukan. Atau dapat dikatakan menguji kualitas pada perangkat lunak termasuk bagian dari jaminan pada kualitas *software* yang menjelaskan inti dari desain, pengkodean dan spesifikasi (Andriyani et al., 2021).

Kemudian pegujian ini memiliki tujuan dalam memperlihatkan kesamaan pada fungsi *software* pada spesifikasinya. Sebuah *software* dapat dikatakan gagal, apabila *software* tersebut tidak memenuhi dari spesifikasi yang telah ditentukan. Sebelum sebuah *software* digunakan, maka perlu dilakukan uji coba supaya menjumpai kesalahan-kesalahan yang tidak diinginkan. Pengujian ini merupakan tahapan yang paling mahal dalam mengembangkan sebuah perangkat lunak supaya *software* tersebut bebas dari kesalahan umunya (Setiawan et al., 2020).

Model dalam mengembangkan *software* yang sering digunakan yaitu dengan model *waterfall*. Pada model ini memiliki sifat linear dimulai dari tahapan awal dalam merencanakan pngembangan sistem sampai dengan tahapan akhir yaitu tahapan pemeliharaan (Wahid, 2020). Tahapan-tahapan pada umumnya yaitu:

1. Analisis, dimana komunikasi diperlukan dalam mengembangkan sistem dengan tujuan supaya dapat mengerti *software* yang pengguna harapkan dan

pembatasan pada *software* itu. Proses analisis diperlukan keterlibatan pengguna untuk mendapatkan data yang dibutuhkan.

2. Desain, dimana desin sistem dibuat oleh pengembang supaya dapat membantu dalam menetaplam *hardware* dan kriteria dalam suatu sistem serta juga dalam membantu merancang *hardware* yang didalamnya terdapat gambaran tampilan aplikasi seperti tampilan aplikasi yang secara umum dan tampilan antarmuka.
3. Implementasi, dimana pertama kali pengembang memulai sistem dala program yang kecil atau biasanya disebut sebagai unit, yang terpadu pada tahapan berikutnya. Setiap unit diuji serta dikembangkan memiliki fungsionalitas pada *software*. Aktivitas yang termasuk pemograman, pembuatan aset seperti musik, suara, gambar dan aset lain yang nanti akan digabungkan bersamaan.
4. Pengujian, dimana pengujian pada sistem apakah sistem dengan sebagian atau sepenuhnya memenuhi syarat-syarat pada sistem, pengujian bisa di golongkan dalam unit testing, pengujian pada sistem aakn beraksi saat semua modul menyatu dan penerimaan dalam pengujian dalam melihat bagaimana keberhasilan suatu sistem dalam aplikasi bisa memenuhi kebutuhan bagi penggunanya, kemampuan aplikasi meminimalkan kesalahan seperti (*bug dan glitch*), serta kemampuan aplikasi dalam menjalankan fungsinya. Jika ditemukan kesalahan, maka akan dilakukan peerbaikan dan pengujian lagi sampai masalah tersebut selesai teratasi.

5. Pemeliharaan, yang merupakan metode yang mana *software* yang telah selesai dirancang, kemudian dijalankan serta dapat diperbaharui dengan fitur tambahan. Hal ini merupakan tahapan akhir dalam metode *waterfall*. *Software* yang telah selesai dirancang, akan dijalankan dan dilakukan pemeliharaan. Di dalam pemeliharaan ini sudah termasuk perbaikan atas kesalahan yang tidak didapati di langkah sebelumnya.

Model *waterfall* merupakan model *System Development Life Cycle* (SDLC) yang paling sederhana. Dalam model ini sangat serasi dalam mengembangkan *software* pada spesifikasi yang tidak beraturan. Pada model ini juga memiliki beberapa kelemahan dan model ini juga sebagai dasar dari model yang lain pada saat memperbaiki pengembangan *software*. Model ini sangat mudah dipahami oleh pengguna dan memungkinkan terjadinya perubahan pada saat mengembangkan perangkat *software*.

Seiring berkembangnya dalam dunia industri *game*, munculnya *game* lebih berkualitas dan menarik dari segi visualisasi ataupun dari segi cerita juga semakin banyak. Supaya pada era globalisasi ini membuat setiap warga negara mempunyai perasaan cinta terhadap tanah air pada seluruh budaya serta perkembangan pada teknologi informasi termasuk *game* yang disebarluaskan pada kehidupan ini. Sekarang ini *game* merupakan perangkat lunak yang sedang mengalami perkembangan. Berbagai macam *game* yang memiliki tampilan yang lebih menarih dan dijadikan sebagai alat *refreshing* (Armanda & Rizqi, 2020).

Untuk saat ini, *video game* belum hendak memperlihatkan pada dunia publik. Pengembang *video game* mengambil keputusan seperti ini supaya bisa memberikan kejutan ke calon pembeli yang memiliki minat untuk melakukan pembelian pada *video game* itu. Apabila keputusan itu diambil, dengan begitu pihak lain akan dijadikan sebagai calon pembeli. Pihak penguji biasanya disebut sebagai *beta tester*. Setelah *video game* selesai dikembangkan, bukan berarti *video game* itu langsung akan diterima oleh khalayak ramai. *Beta testing* atau yang disebut sebagai *eksternal tersting* dilakuakn pada saat melakukan pengujian pada *video game* dalam menemukan berbagai *bug* atau *error* atau permasalahan yang lain yang diperoleh dari pihak bermain. *Beta* berada pada luar *production cycle*, akan tetapi hasil dari *testing* mempunyai kemampuan yang membuat tim melakukan pengulangan pada *production cycle* (Prasetyo et al., 2021). Pihak penguji umumnya sudah terikat kontrak dengan pengembang sebagai tujuan untuk tidak membocorkan komponen yang ada yang terdapat pada *video game* yang sedang dirancang itu.

2.1.3 Aliran Sistem Informasi

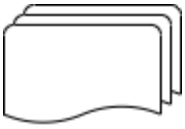
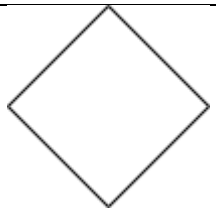
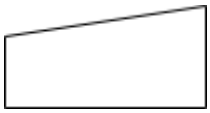
Supaya melihat suatu sistem yang ada dan mengetahui sistem informasi yang sedang berjalan itu, diperlukan memberikan penjelasan pada aliran sistem informasi mengenai elemen penting yang terdapat di luar ataupun di dalam sistem. Dari hal tersebut lebih memudahkan pemahaman tentang informasi yang sudah ditemukan dan ditekan pada sistem itu dengan sendirinya (Tukino & Arnomo, 2021).

Apabila sistem informasi tersebut tidak sebanding, dengan begitu perlu dilakukan perubahan pada olahan datanya sehingga dapat menghasilkan informasi dengan akurat dan cepat serta pengambilan keputusan menjadi lebih baik. Aliran sistem informasi dapat menjelaskan proses berjalannya pada sistem informasi (Yulisman, 2019). Hal ini serupa dengan *flowchart*, dimana aliran pada sistem informasi dibuat supaya memperlihatkan kepada pihak yang ikut serta pada proses berjalannya suatu sistem. Sistem yang baru saja dikembangkan juga dapat menggunakan aliran pada sistem informasi ini. Adapun simbol dari Aliran Sistem Informasi (ASI), diantaranya:

Tabel 2.1 Simbol-simbol yang akan digunakan dalam aliran sistem informasi

No	Simbol	Nama	Penjelasan
1		*Start/End*	Melambangkan alur awal ataupun akhir
2		*Process*	Melambangkan aktivitas yang bisa dilakukan pada sistem

Tabel 2.1 Lanjutan

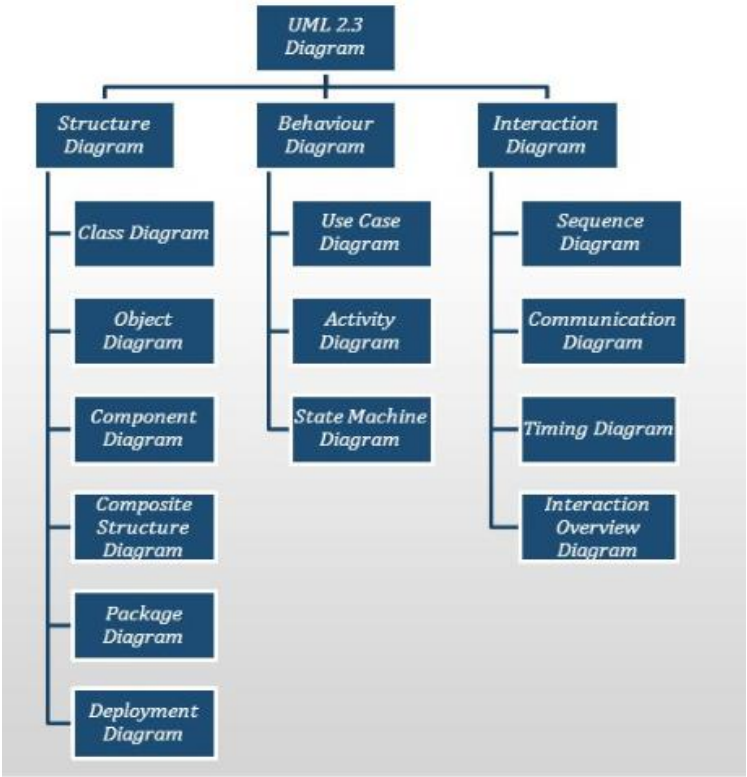
3		*Document*	Melambangkan keterlibatan dokumen terhadap sistem
4		*Multiple Document*	Melambangkan sejumlah dokumen yang memiliki keterlibatan terhadap sistem
5		*Decision*	Simbol yang memperlihatkan pada kejadian percabangan
6		*Input/Output*	Melambangkan data atau informasi yang masuk ataupun keluar
7		*Manual Input*	Memperlihatkan data atau informasi pada data yang sudah di *input* oleh pihak yang menggunakannya

2.1.4 *Unified Model Languange*

Pada dunia industri, bahasa yang banyak digunakan untuk membuat analisis, menjelaskan analisis, menggambarkan arsitektur dan desain dalam pemrograman

yang berfokus pada objek ialah UML. Yang dimaksud dengan UML yaitu suatu sarana pada bidang pengembangan sistem yang berfokus pada objek. Hal tersebut dikarenakan UML mempersiapkan bahasan pemodelan visal yang dapat memungkinkan pengembang sistem dalam membuat bentuk yang baku dan cetak baru dari visi mereka. UML juga memiliki fungsi sebagai jembatan pada saat melakukan komunikasi pada beberapa sudut pandang dari sistem (Gunawan et al., 2019).

UML memiliki tiga konsep abstraksi yang bisa dilakukan pada berbagai jenis diagram, diantaranya *use case diagram* yang biasanya digunakan dalam menggambatkan kelakukuan pada sistem yang akan dibuat, *activity diagram* yang biasanya digunakan dalam menggambarkan alur bekerja pada sistem, *sequence diagram* yang biasanya digunakan menggambarkan kerja yang terdapat pada objek, *class diagram* yang biasanya digunakan menggambarkan bagaimana suatu operasi dilaksanakan, dan yang terakhir *deployment diagram* yang biasanya digunakan menggambarkan elemen sistem yang digabungkan (Ependi, 2018). UML dibagi menjadi tiga kelompok, diantaranya: *structure diagram*, *behavior diagram*, dan *interaction diagram* yang setiap diagram bisa diperlihatkan pada gambar berikut ini.

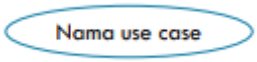


Gambar 2.1 Diagram pada UML

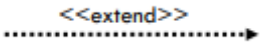
Sesuai pada gambar 2.1 merupakan langkah dalam merancang sebuah *video game*, digunakan diagram *use case*, *class*, *activity*, dan *sequence diagram.* Diagram-diagram itu digunakan sebagai perwakilan pada diagram yang lainnya.

1. Diagram *use case* termasuk model untuk menunjukkan *behavior* atau kelakukan pada sistem informasi yang akan dibuatnya (Syarif & Nugraha, 2020). *Use case* menerangkan sebuah hubungan yang terjadi antara sistem yang akan dibuat dengan aktor. *Use case* pada tugas akhir ini digunakan untuk mamberikan gambaran fungsi atau fungsional yang terdapat didalam sistam dan untuk mengetahui fungsi-fungsi tersebut dapat digunakan oleh siapa saja.

Tabel 2.2 Simbol yang terdapat pada *use case diagram*

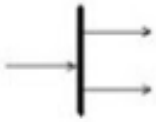
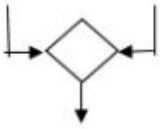
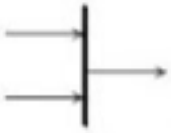
No	Simbol	Nama	Penjelasan
1		*Actor*	Melambangkan pihak yang mengharapkan *output* dari sistem dan juga dapat men-*input* data.
2		*Use Case*	Memperliharkan aktivitas yang bisa dilakukan oleh sistem.
3		*Include*	Melambangkan interaksi antar *use case* yang akan ikut terjadi.

Tabel 2.2 Lanjutan

4		*Extend*	Memperlihatkan relasi antar *use case* yang bisa terjadi tergantung dari campur tangan dari sistem atau *input.*
5		*Generalization*	Memperlihatkan relasi antar *use case* yang kemudian disederhanakan.
6		*Communication*	Memperlihatkan relasi antara *use case* dengan akhir yang bisa dilakukan pada aktor.

2. *Activity diagram* biasanya digunakan dalam menggambarkan berbagai jenis alur dari kegiatan pada sistem yang akan dikembangkan, kemudian bagaimana dimulainya alur itu, pengambilan keputusan yang mungkin terjadi serta bagaimana alur tersebut berakhir. *Activity diagram* menunjukkan aktivitas sistem berbentuk perkumpulan aksi, kemudian bagaimana aksi itu mulai dilakukan, dan keputusan yang kemungkinan terjadi sampai aksi itu berakhir (Asmasari & Informatika, 2020).

Tabel 2.3 Simbol-simbol yang terdapat pada *activity diagram*

No	Simbol	Nama	Penjelasan
1		*InitialState/Start*	Melambangkan status awal dari diagram
2		*Decision*	Melambangkan peristiwa pada keadaan pilihan
3		*Fork*	Melambangkan permulaan dari kegiatan yang secara bersamaan dilakukan
4		*Merge*	Melambangkan peristiwa akhir pada keadaan pilihan
5		*Join*	Melambangkan berbagai aktivitas terkhir yang pada sebelumnya telah berjalan dengan bersamaan

Tabel 2.3 Lanjutan

6		*FinalState/End*	Melambangkan berakhirnya diagram
7		*Action*	Melambangkan aksi
8		*Object*	Melambangkan objek

3. *Sequence diagram* biasanya digunakan pada saat memberikan penjelasan pada hubungan antara objek yang terdapat dalam sistem dengan memberikan keterangan penambahan urutan waktu (Maylawati et al., 2018). Masing-masing interaksi akan disusun sesuai dengan waktu interaksi tersebut terjadi, dan akan dimulai dari atas sampai dengan bawah. Diagram ini tidak akan menjelaskan tentang apa yang akan terjadi pada interaksi ini. Tujuan dari diagram ini yaitu untuk memberikan penjelasan secara berurutan proses terjadinya interaksi anta objek yang terdapat pada sistem.

Tabel 2.4 Simbol-simbol yang terdapat pada *sequence diagram*

No	Simbol	Nama	Penjelasan
1		*Actor*	Melambangkan pihak yang mengharapkan *output* dan akan melakukan *input* pada data.
2		*Object*	Melambangkan objek terdapat dalam sistem yang memiliki keterlibatan pada saat terjadinya interaksi.
3		*Destroy*	Melambangkan objek akan berhenti melakukan interaksinya.
4		*Lifeline*	Melambnagkan jangka waktu dalam keterlibatan aktor atau objek yang berlangsung selama interaksi.
5		*Activation*	Melambangkan aktor atau objek sedang berinteraksi ataupun sedang beroperasi.

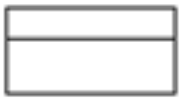
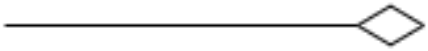
Tabel 2.4 Lanjutan

6		*Call message*	Melambangkan jenis operasi atau interaksi yang nantinya akan dikirim ke aktor atau objek lain.
7		*Return message*	Melambangkan jenis operasi atau interaksi yang nantinya akan dibalas ke aktor atau objek pengirim.
8		*Self message*	Melambangkan jenis operasi atau interaksi yang nantinya akan diterima dan dikirim ke *lifeline* aktor atau objek yang serupa.

4. *Class diagram* biasanya digunakan dalam memberikan penjelasan hubungan antar kelas. Setiap kelas memiliki bagian operasi dan atribut. Simbol relasi dapat digunakan dalam menghubungkan antar kelas. Diagram ini juga memudahkan para pembaca untuk memperoleh gambaran umum tentang

software yang akan dikembangkan (*UML Class Diagram Tutorial | Lucidchart*, n.d.).

Tabel 2.5 Simbol *class diagram*

No	Simbol	Nama	Penjelasan
1		*Class*	Melambangkan objek yang terdapat dalam sistem dan mempunyai kemampuan operasi dan atribut.
2		*Association*	Melambangkan hubungan antar kelas, tanpa memiliki ketergantungan yang khusus.
3		*Aggregation*	Melambangkan relasi yang khusus antar kelas, seperti *child class* termasuk pada elemen dari *parent class*. Akan tetapi *child class* juga bisa berdiri dengan sendirinya.

Tabel 2.5 Lanjutan

4		*Composition*	Melambangkan relasi yang khusus antar kelas, seperti bergantungan hidup terhadap *parent class*. *child class* akan menjadi tidak berlaku apabila *parent class* tidak ada.
5		*Inheritance*	Melambangkan relasi khusus antar kelas, seperti perlengkapan yang terdapat pada *parent class* dan juga yang terdapat dalam *child class*.
6		*Dependency*	Melambangkan relasi yang khusus antar kelas, seperti respon pada perubahan yang akan terjadi dalam *parent class*.

2.2 Teori Khusus

2.2.1 *Video Game*

Yang dimaksud dengan *video game* yaitu permainan yang bersifat elektronik dan yang melibatkan interaksi antar pengguna untuk mendapatkan umpan balik secara visual terhadap perangkat video. *Video game* berasal dari kata tradisional yaitu perangkat layar *raster*. Akan tetapi sampai sekarang ini digunakan sebagai sebutan *video game*, dan nama *video game* digunakan untuk menyebutkan permainan yang terdapat perangkat layar apapun. *Video game* yang khusus seperti *game arcade*, telah umum dan sering digunakan. *Video game* juga dikenal sebagai suatu bentuk seni (Acep, 2018).

Game dalam kamus bahasa indonesia memiliki arti yaitu permainan. G*ame* adalah suatu aktifitas baik itu terstruktur maupun semi terstruktur yang bertujuan sebagai sarana hiburan dan kadang untuk Pendidikan (Pradana & Nita, 2019).

Gambar 2.2 Contoh video game genre Battle Royale

Misalnya pada jenis *game* yang banyak dikenal oleh masyarakat, terutama untuk pengguna perangkat *smartphone* ialah *battle royale*, dengan *sub-genre* tembak-tembakan (*shooter*). Hal penting yang didapatkan dari jenis *game* ini yaitu memiliki satu sesi dengan jumlah pemain lebih dari 50 yang saling bertarung antara yang saru dengan yang lainnya, dan akan tersisa satu pemain yang masih bertahan hidup atau dikatakan sebagai pemenang dari *game* tersebut (GyuHyeok Choi, 2018). *Game* yang memiliki *sub-genre shooter*, akan menggunakan perangkap dan senjata api sebagai sarana dalam membuat musuh kalah (Karavolos et al., 2018). Contoh lain dari *game* jenis ini yaitu *Free Fire, Rules of Survival, Fortnite, Apex Legend, PlayerUnknown's BattleGround* dan game lainnya.

Gambar 2.3 Contoh *video game genre* simulasi

Pada permainan yang bersifat tradisional atau *game* yang telah ada pada dunia nyata juga bisa dikembangkan menjadi bentuk versi digital. Misalnya seperti catur, ular tangga, monopoli dan permaian lainnya. Selain itu pekerjaan dan kegiatan juga bisa dijadikan sebagai ide dalam mengembangkan sebuah *video*

game atau biasanya disebut sebagai *simulation*. Misalnya jenis *video game* simulasi ialah *Overcooked*, *Monopoly*, *Chess*, *The Sims*, dan lain sebagainya.

2.2.2 *Game Engine*

Dalam menjalankan *video game* yang pada umumnya atau aplikasi simulasi tiga dimensi akan dijalankan menggunakan aplikasi yang khusus. Dalam mengolah objek yang berbentuk tiga dimensi, biasanya akan menggunakan aplikasi yang khusus seperti Autodesk Maya atau AutoCAS. Selain itu, sebagai tempat *video game* tersebut dijalankan juga membutuhkan aplikasi yang khusus. Sebutan yang biasanya digunakan yakni *game engine* (Fachroni et al., 2018). *Game Engine* ini dibutukan supaya memudahkan dalam melakukan pengembangan, produksi dan pengujian pada *video game*. Apabila pengembang atau *developer* dari pengembang *video game* mempunyai kompetensi, bisa juga dengan membuat *game engine* dengan sendirinya. Misalnya seperti Godot Engine, Cry Engine, Unreal Engine, Unity, dan berbagai jenis *software* lainnya (*Features - Unreal Engine*, n.d.; *Godot Engine - Free and open source 2D and 3D game engine*, n.d.; *Solutions | Unity*, n.d.).

Gambar 2.4 Contoh *game engine* untuk pengembangan *video game*

2.2.3 Genre *Platformer*

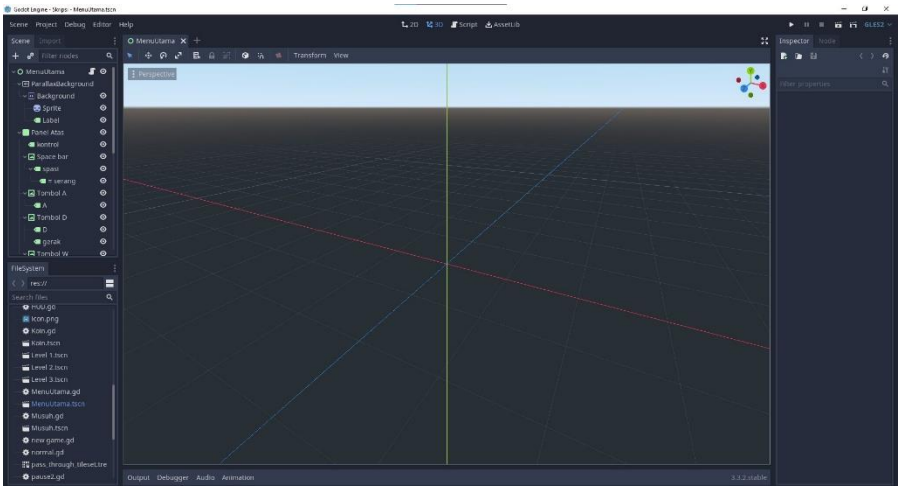
Video game yang memiliki jenis seperti *platformer* termasuk *game* yang mempersiapkan lingkungan yang baik dan bisa berbentuk dua dimensi ataupun tiga dimensi. *Video game* jenis ini memiliki tujuan untuk mengedalikan karakter utama sampai pada titik tujuan. Dalam *game genre* ini memiliki unsur yang paling penting, yaitu melompat. Permaianan ini memiliki mekanik dalam membimbing karakter pada pemain untuk melompati diantara rintangan dan plastform yang terdapat pada *genre* lain atau biasanya disebut sebagai *platforming* atau *platform* (Sulaeman & Permana, 2020).

Video game yang memiki jeni *platformer* ini memiliki aspek penting yaitu obejek yang berupa *platform*, seperti papan atau bidang yang mana objek lain atau karakter bisa berdiri atau dijadikan sebagai tempat untuk berjalan (Mustofa et al., 2018). Pemainan ini memiliki alur yaitu mengendalikan karakter utama untuk melompati atau melewati bidang yang pertama sampai dengan bidang yang terakhir. Selain itu juga terdapat firut lainnya yang telah disediakan, contohnya

objek yang dikumpulkan untuk menambah jumlah pada *score* (nilai), dan karakter yang lain seperti musuh akan menjadi hambatan dalam mengalahkan karakter utama.

2.2.4 Godot Engine

Yang dimaksud dengan Godot Engine yaitu suatu aplikasi dalam mengembangkan dan mempersiapkan berbagai *tools* pada *video game* yang nantinya akan dirilis secara *open source* dan gratis (*Godot Engine - Free and open source 2D and 3D game engine*, n.d.). Sistem *node* dan *scene* didukung dalam sistem ini. Setiap objek yang dikembangkan dalam *game* ini melambangkan sebagai *node*. *Node* bisa diletakkan dan digunakan kembali pada *scene* lainnya (Linden et al., 2020). *Engine* yang merancang *video game* ini akan sepenuhnya menjadi pemiliki dari pengembang itu. Pihak pemilik *engine* ini tidak akan meminta membayarkan keuntungan apabila pengembang *game* ini memiliki rencana untuk menjual *game* hasil buatannya sendiri. Apabila Godot Engine ini dibantu oleh pengguna *engine* dalam melakukan perkembangannya, dengan begitu pengguna bisa langsung mengujungi web dan memberikan donasi pada sejumlah yang sebagai sumbangan dalam membantu pengembangan dan perbaikan pada Godot Engine ini.



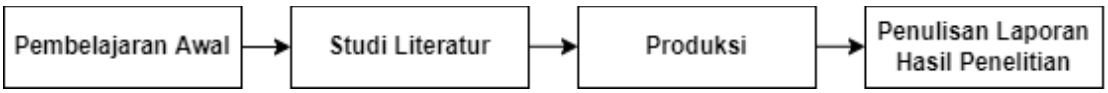
Gambar 2.5 Tampilan awal Godot Engine

BAB III

METODE PENELITIAN

3.1 Desain Penelitian

Adapun desin penelitian yang dilakukan peneliti saat penyelesaian skripsini ini. Berikut ini aktivitas yang dilakukan ialah:



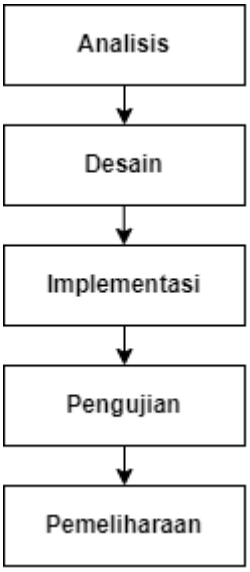
Gambar 3.1 Alur penelitian

1. Dalam tahap pembelajaran awal, peneliti akan mengumpulkan informasi tentang *game engine* yang saat ini dikenal dalam pengembangan *video games*, kemudian dari beberapa pilihan tersebut diambil pilihan yang terbaik. Dan peneliti juga mengumpulkan informasi tentang jenis atau *genre* pada *video game* yang sedang kalah bersaing pada dunia industri *video game* yang ada di Indonesia.
2. Dalam tahap studi literatur, peneliti akan mengumpulkan informasi tentang proses umum dalam perancangan *video game* dan apa yang menjadi perbedaan dari pengembangan aplikasi yang lainnya. Dan peneliti juga mencari informasi tentang perkembangan *video game* yang sedang terjadi di Indonesia melalui jurnal penelitian yang berhubungan dengan *video game* ataupun melalui beeita. Pada tahapan ini, peneliti juga memperlajari metode dan proses perancangan pada *video game*.
3. Dalam tahap produksi, peneliti memulai mengembangkan *video game*. Metode yang peneliti pilih yakni menggunakan metode *waterfall*. Perubahan

dalam metode akan terjadi dikarenakan dalam proses pengembangan *video game* memiliki perbedaan dari *software* pengembangan *video game* lainnya. Pada tahapan ini, peneliti juga akan melaksanakan pengujian terhadap hasil dari pengembangan *video game* tersebut. Pengujian ini bisa meminta teman, keluarga, atau orang yang tidak dikenal bertindak sebagai pihak *beta tester*.

4. Dalam tahap pencatatan hasil penelitian, peneliti akan menuliskan keseluruhan proses dan hasil saat dimulainya, berlangsung dan penyelesaian selama hasil penelitian ini . Pada tahapan ini juga dilakukan pemberian saran dan penarikan kesimpulan pada akhir naskah.

Metode *waterfall* merupakan metode dalam pengembangan aplikasi ini. Model pada pengembangan ini sangat sesuai untuk mengembangkan *software* yang masih bagi orang awam. Model *waterfall* disesuaikan pada penelitian, dikarenakan *software* yang dikembangkan ialah *video game* yang belum diketahui apakah *requirement* tidak akan terjadi perubahan saat masuk ke tahapan pengujian. Setiap tahapan, peneliti akan melakukan aktivitas antara lain:



Gambar 3.2 Alur model pengembangan *waterfall*

1. Dalam tahap analisis, peneliti menentukan tema, tujuan dan cerita singkat tentang *video game*. Pada tahap ini menyangkutkan calon pengguna. Tahapan ini tidak akan berlaku apabila aplikasi yang dirancang termasuk *video game*. Apabila calon pemain memiliki ketertarikan pada permainannya, dengan begitu calon pembeli akan membeli atau mengunduh permainan itu. Tema pada cerita yang akan dibuatkan tidak diwajibkan harus masuk akan, dikarenakan *video game* mendukung terjadinya kegiatan dan kondisi berdasarkan dengan imajinasi dari perancang *video game* tersebut.
2. Dalam tahapan desain, peneliti memulai membuat alur dan sketsa pada permainan yang berhubungan dengan tema dan lingkungan yang nanti akan digunakan pada *video game* tersebut. Sketsa bisa berupa gambar tentang latar belakang singkat, lokasi dan karakter. Alur permaianan dan sketsa

permainan bisa dijadikan secagai acuan dalam proses menggambar aset pada tahapan implementasi.

3. Dalam tahap implementasi, peneliti menggunakan beberapa aset yang bisa berupa efek suara dan gambarm pemrograman serta menyusun semua aset menjadi sebuah *video game*. Pengaplikasian aset, pemrograman serta pembuatan *video game* secara penuh akan dilaksanakan pada aplikasi Godot versi 3.3.2 dengan bahasan pengrograman Gdscript.
4. Dalam tahap pengujian, peneliti akan meminta beberapa orang untuk dijadikan sebagai *beta tester* dalam hal *black-box testing*. Pada proses ini akan diminta penilaian yaitu prndapat tentang perasaan pemain saat bermain, seseruan bermain, perasaan pemain saat menyelesaikan permainan, motivasi yang didapatkan pemain dalam bermain, keberhasilan pemakaian kontrol dan fitur serta saran yang berhubungan dengan *game* ini. Tahapan ini akan terjadi kesalahan yang nantinya akan diperbaiki ataupun diubah. Misalnya perubahan ynag tidak dimasukkan pada kesalahan *software* yaitu karakter melompat terlalu rendah atau tinggi, karakter berjalan terlalu lambat atau cepat, dan aspek yang lain.
5. Dalam tahap pemeliharaan, peneliti menambahkan fitur pada permainan, contohnya layar *credits*, pengaturan bahasa, dan aplikasi yang terlibat pada saat mengembangkan *video game* dan fitur tambahan lainnya.

3.2 Objek Penelitian

Objek pada penelitian yaitu peneliti sebagai *playtester* dan juga sebagai pengembang. Apabila dibutuhkan, keluarga ataupun teman juga bisa diminta sebagai *beta tester*.

3.3 Analisis SWOT

Peneliti akan menjadikan *game* Cat Mario untuk melakukan analisis SWOT, yang mana *game* ini termasuk salah satu *video game* yang memiliki jenis platformer dan karakter dari Cat Mario juga dikenal oleh banyak pemain. Analisis SWOT yang terdapat pada *game* Cat Mario dibuat berdasarkan pertimbangan keadaan dalam waktu mengembangkannya yaitu pada tahun 2007 dengan aplikasi pengembangan *video game* yaitu Open-Syobon.

Tabel 3.1 Analisis SWOT

	Strength	***Weakness***
	Penggunaan *game engine* oleh semua kalangan (publik)	*Video game* Cat Mario dikembangkan pertama kali khusus untuk platform *Windows*.
Opportunity	***Strength-Opportunity Strategy***	***Weakness-Opportunity Strategy***
Video game Cat Mario menyediakan versi 3 dimensi yang bernama Cat Mario 3D	Pengembangan *video game* ini tanpa campur tangan dari orang lain, oleh sebab itu kualitasnya bisa memenuhi visi pribadi	Memiliki aspek lain seperti musik, efek suara, kontrol, dan tampilan visual bisa dimaksimalkan dalam menutup keterbatasan pada permainannya
Threat	***Strength-Threat Strategy***	***Weakness-Threat Strategy***
Video game Cat Mario tersedia *opensource* agar bisa dimodifikasi atau dipindahkan ke konsol lain dengan nama Open Syobon Action dan lisensi GPLv2 di Sourceforge	*Video game* bisa dibangun sesuai standar dengan kelayakan yang lebih tinggi, oleh sebab itu tidak kalah bersaing pada *video game* yang menggunakan *game engine* lainnya	Pemasaran tentang fitur *video game* bisa meningkatkan total peminat/pemain yang dapat menyaingi *video game* lainnya pada komputer pribadi

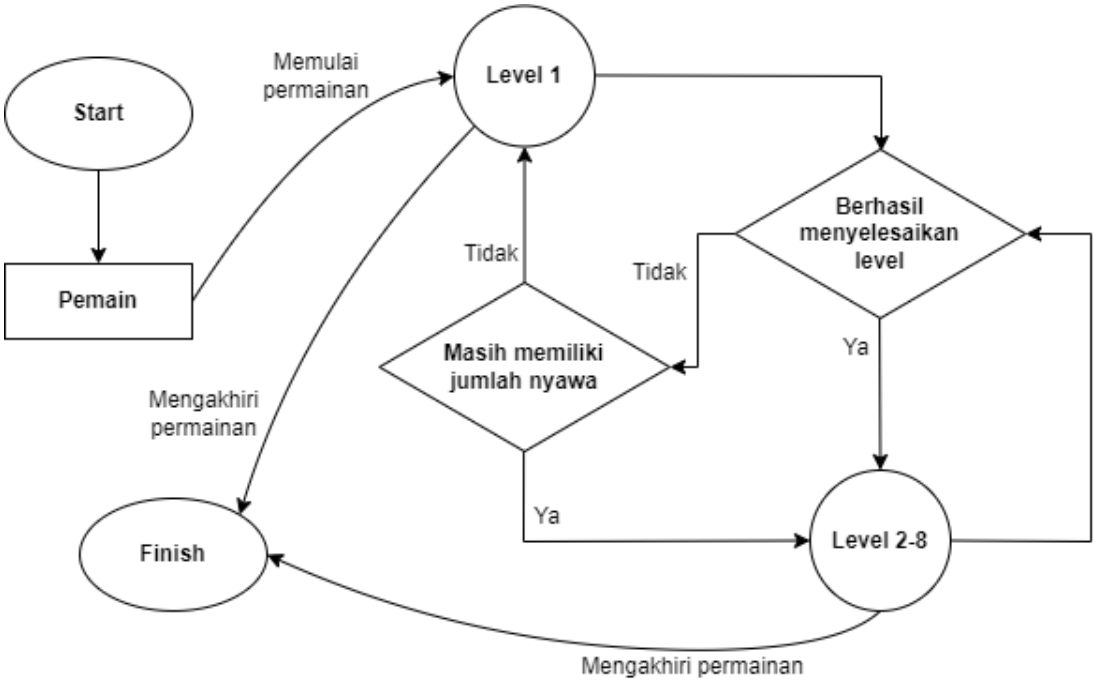
3.4 Analisis Sistem yang Sedang Berjalan

Sebelum merilis *video game*, akan digunakan sebagai pengganti dalam melakukan perbantingan terhadap sistem yang sedang dijalankan. *Game* yang akan dipilih yaitu Cat Mario yang dirilis pada bulan Desember Tahun 2007. *Video game* itu kemudian dipublikasi serta dikembangkan oleh Chiku (ちく chiku).

Cat Mario atau disebut “Mario dari Neraka", termasuk *game* yang terdapat pada komputer dari Super Mario Bros. Pada saat memainkan permainan ini, harus melompat ataupun berjalan dari kanan ke kiri, menginjak dan menghindari musuh serta merlewati berbagai rintangan dan jebakan. Sifat dari *video game* ini yaitu *open source* dan bisa dimainkan pada konsol Nintendo Wii. Adapun tersedianya Cat Mario versi tiga dimensi dari permainan ini yang dinamakan Cat Mario 3D.

3.5 Aliran Sistem Informasi yang Sedang Berjalan

Pada awalnya *video game* ini memiliki empat tingkatan, kemudian dikembangkan kembali menjadi enam sampai dengan delapan. Setiap tingkatan tersebut mempunyai tujuan untuk sampai ke bendera yang terakhir. Akan tetapi terkadang memiliki jebakan yang berbentuk seperti bendera atau penjaga yang bisa membuat pemaint tersebut gagal dalam memenangkan permainan tersebut. Pemain diberikan 3 nyawa, akan tetapi pemain tidak akan memiliki batasan untuk mati, oleh sebab itu harus berjuang sampai nyawanya menjadi minus.



Gambar 3.3 Diagram alur permainan Cat Mario

3.6 Permasalahan yang sedang terjadi

Dari penjelasan yang terdapat pada bab sebelumnya, peneliti menjumpai masalah yang sedang terjadi, diantaranya:

1. *Genre platformer* pada permainan *video game* salah satunya ialah Cat Mario yang pada pertama kali dirilis hanya pada *platform* Nintendo Wii, oleh sebab itu membuat calon pemain yang tidak mempunyai konsol Nintendo atau menggunakan sistem operasi selain Nintendo menyebabkan pemain tidak bisa memaikan *game* tersebut.
2. *Genre platformer* pada *video game* Cat Mario saat dirilis pada *platform* Nintendo Wii berbasis Nintendo hanya bisa dimainkan dengan alat pengendali permainan seperti *Joystick*. *Keybaord* tidak bisa digunakan

pemain untuk bermain *game*, dikarenakan saat itu *video game* Cat Mario masih belum memiliki dukungan dari perangkat pengendali lainnya.

3. *Genre platformer* pada *video game* Cat Mario, tidak memiliki fitur pilihan *level*. Apabila pemain telah kehabisan nyawa atau gagal pada *level*, dengan begitu permainan akan berakhir dan harus diulang dari *level* paling awal.

3.7 Usulan Pemecahan Masalah

Sesuai pada penjelasan bab masalah yang sedang terjadi, adapun pemecahan masalah yang peneliti temukan, yaitu

1. *Genre platformer* pada *video game* yang berjudul Pemburu Koin dirancang dan dibangun menggunakan Godot Engine, yang mana hasil pengembangan *video game* bisa dijalankan pada laptop atau komputer dengan sistem operasi berbasis Windows, oleh sebab itu diharapkan *video game genre platformer* berjudul Pemburu Koin bisa dijadikan sebagai pilihan dari Cat Mario yang bisa dimainkan oleh seluruh pemain yang menggunakan laptop atau komputer.
2. Pemain menggunakan alat pada pengendali permainan yakni menggunakan *keyboard* yang tersambung pada komputer yang sedang menjalankan *video game* Pemburu Koin.
3. Penerapan fitur pilihan *level* dikategorikan (*easy*, *normal*, dan *hard*), oleh sebab itu pemain tidak perlu merasa frustasi untuk mulai dari *level* sebelumnya dan mencoba memainkan lagi *level* yang gagal itu.