BAB IV

ANALISIS DAN PERANCANGAN SISTEM

4.1 Analisis Sistem

Pada bagian analisis system ini, akan dilakukan berbagai macam analisis yang berhubungan dengan *game* yang akan dibangun, analisis sistem dapat di definisikan sebagai penguraian dari suatu sistem yang utuh kedalam bagian – bagian komponennya dengan maksud untuk mengidentifikasi dan mengevaluasi kebutuhan – kebutuhan yang di harapkan sehingga dapat di usulkan perbaikan – perbaikannya. Sebagai analisis pada sistem yang sedang berjalan, akan di bahas bagaimana prosedur-prosedur yang sedang atau yang sudah berjalan pada sistem terdahulu. Analisis sistem non fungsional yang meliputi perangkat keras dan perangkat lunak yang di gunakan.

4.1.1 Analisis Masalah

Dalam merancang sebuah *game* ada beberapa permasalahan yang perlu diperhatikan. Seperti membuat AI (*Artificial Intelligence*) tanpa AI game yang di buat tidak akan menarik terlebih dalam pembuatan *game* bergenre RPG (*Role Playing Game*) dan tentunya dalam suatu game diperlukan *collision detection* yang berguna untuk mendeteksi setiap objek, tanpa adanya *collision detection*

disetiap objek maka *game* akan terlihat tidak berguna dan akan terlihat tidak masuk akal.

Game dengan bertemakan RPG (*Role Playing Game)* merupakan *game* yang sangat popular di kalangan pecinta game, kelebihan *game* RPG dengan *game* lain adalah pemain dapat mengeksplore tempat tanpa harus terpaku pada story line, berbeda dengan *game* yang bertemakan *adventure* dimana pemain harus menyelesaikan misi pada saat itu juga sehingga pemain tidak dapat mengeksplore ke tempat lain. Dan untuk merancang dan membuat game yang bertemakan RPG (*Role Playing Game*) sangatlah rumit terdapat beberapa komponen mulai dari pembuatan bahasa pemograman yang berfungsi sebagai AI, penerapan model untuk setting tempat dan *environment* pada game, dan pembuatan karakter utama maupun npc (*Non Playable Character*).

Dan untuk membangun game RPG (*Role Playing game*) yang begitu rumit, membutuhkan waktu dan sumber daya yang besar. Terdapat beberapa *game* yang bertemakan RPG (*Role Playing game*) yang ada saat ini adalah *Skyrim, Far Cry 3, FallOut 3, wFinal Fantasy, Raiderz, Age Of Empire.* Untuk itu penulis menganalisa beberapa game engine dengan *game engine* yang penulis pilih, karena setiap game engine memiliki kelebihan dan kekurangannya masing – masing. Dengan melihat kekurangan dari setiap game engine tersebut dapat terlihat game engine mana yang jauh lebih baik dalam pembuatan *game* yang bertemakan RPG (*Role Playing game*) dengan begitu game yang dibuat akan jauh lebih baik dari segi grafis dan pengembangan alur cerita pada game sehingga lebih realistis.

4.1.2 Analisis Game RPG

Game yang akan dibangun dengan *game engine* UNITY3D adalah *game* RPG (*Role Playing game*). Dalam game ini bercerita tentang kegiatan peperangan antara karakter utama dan musuh – musuh yang ada pada game. Game ini dibuat untuk memberitahukan bagaimana merancang dan membuat *game* RPG (*Role Playing game*) yang di buat dengan grafis 3D (3 *Dimensi)*. Cara permainannya menggunakan sistem *single player (Third Person Controller).* Dan AI (*Artificial Intelligence*) yang di buat adalah dengan menggunakan bahasa pemograman C#.

4.1.2.1 Storyline

Game ini menceritakan karakter utama yang merupakan seorang pejuang dari bangsa spartan yang terdampar di tempat yang tidak di ketahui. Dengan keahlian yang di miliki oleh karakter utama sebagai ksatria spartan, pemain harus bisa mencari jalan agar karakter dapat kembali ke tempat asal karakter utama berasal dengan mengalahkan monster yang mengganggu di setiap tempat yang di tempati oleh karakter utama. Setiap pemain dapat mengalahkan monster atau musuh yang ada maka level dari karakter utama akan naik dan pemain dapat menaikan level yang di inginkan sesuai dengan keinginan pemain.

Setiap kali pemain berhasil mengalahkan semua musuh maka pemain bisa pindah ke tempat lain dan pemain juga di tuntut agar bisa menaikan level karakter utama karena di setiap tempat yang berbeda musuh memiliki level yang jauh lebih tinggi dengan karakter utama. Pemain juga dapat berinteraksi dengan NPC *(Non Playable Character)* untuk mendapatkan misi tambahan.

4.1.2.2 Gameplay

Tugas utama pemain dalam memainkan *game* ini adalah mencari jalan kembali agar karakter utama dapat kembali ke tempat asal karakter utama itu berasal. Pemain harus bisa mengalahkan musuh dengan level yang berbeda-beda sehingga pemain di tuntut untuk bisa menaikan level karakter utama dengan baik. Tidak hanya menggunakan senjata, pemain dapat menggunakan serangan sihir yang ada pada karakter utama sehingga permainan akan jauh lebih menarik.

Dan untuk berpindah dari tempat satu ke tempat lain pemain membutuhkan *portal* yang menghubungkan antara dunia yang satu dengan dunia yang lainnya. Sebelum *portal* itu terbuka pemain harus bisa memusnahkan musuh dan misi yang ada. *Game* yang memiliki tipe RPG *(Role Playing Game)* pemain dapat menggerakan karakter utama secara bebas dan dapat mengeksplore tempat yang ada di game tersebut atau di sebut juga dengan *open world.*

4.1.2.3 Skenario

Skenario adalah urutan cerita yang di susun oleh seseorang agar suatu peristiwa terjadi sesuai dengan keinginan. Dan dalam *game* ini saya akan membuat sebuah skenario agar lebih mudah di mengerti. Dalam *game* ini saya membuat permainan untuk pemain yang berumur antara 15 tahun keatas dengan tujuan untuk menghabiskan waktu atau bersenang-senang dalam bermain *game* ini.

Di awal *game* ini pemain akan berada pada suatu tempat yang bernama *Embun* pos yang merupakan tempat para prajurit berkumpul untuk menjalankan tugas-tugasnya. Tempat ini berada di dataran tinggi yang di kelilingi oleh bukit-bukit sehingga sinar matahari sulit untuk mencapai tempat ini, ini yang menyebabkan tempat ini menjadi lembab dan tertutup oleh embun, sehingga tempat ini disebut *Embun* pos. *Embun* pos merupakan tempat yang tidak memiliki penduduk yang banyak, yang ada di sini hanya pepohonan dan bukit yang berbaris. Tetapi ada satu kejadian aneh yang melanda *Embun* pos, yaitu semua prajurit yang menjaga tempat tersebut tiba-tiba memiliki sikap yang aneh dan mulai melakukan tindakan kriminal. Maka dari itu, pemain harus mencari tahu apa yang menyebabkan prajurit di *Embun* pos bertindak criminal dan tidak menjaga tempar tersebut.

Selain itu di sisi *Embun* pos terdapat tempat yang begitu keramat yang merupakan tempat terakhir meninggalnya seorang prajurit yang memiliki kekuatan yang begitu besar. Konon tempat ini sudah ada sebelum *Embun* pos dibuat, dan terdapat patung yang menjadi legenda di tempat tersebut karena patung tersebut merupakan seorang prajurit yang berjuang untuk melawan penjahat yang di panggil dari kegelapan yang bernama *Geonidas*. Tetapi di saat *Geonidas* berperang melawan penjahat kegelapan kekuatan yang di miliki oleh *Geonidas* tidak bias mengalahkan penjahat dari kegelapan yang bernama *Skeleton.*

Kekuatan yang dimiliki oleh *Geonidas* terserap oleh *Skeleton*, karena *Geonidas* sudah tidak berdaya, maka dia mengeluarkan semua kekuatannya agar dapat mengalahkan *Skeleton.* Dan setelah itu *Geonidas* pun menghilang bersama

Skeleton, di percaya oleh penduduk setempat bahwa patung yang ada di kuburan tersebut adalah *Geonidas,* maka dari itu tempat tersebut disebut dengan *Kuburan Geonidas*.

Dalam *game* ini saya menerapkan beberapa model seperti : rumah, bukit, pohon, rumput, air, kuburan agar semua setting sesuai dengan yang di inginkan. Pada awal *game* pemain harus menggerakkan karakter utama ke tempat pertama yaitu : *Embun* pos karena di sini terdapat misi yang harus di selesaikan oleh pemain. Misi pertama yang harus di lakukan adalah sebagai berikut :

1. Membantu penduduk menemukan barang yang hilang.
2. Membantu prajurit *Xeno* mengalahkan prajurit yang terkena kutukan.

Setelah itu pemain akan pergi ke *Kuburan Geonidas* untuk mencari tahu apa sebenarnya yang terjadi pada *Embun* pos. Pemain akan berbicara pada sebuah patung yang merupakan arwah dari *Geonidas,* setelah itu pemain akan diberi tugas oleh arwah *Geonidas* untuk masuk ke dunia kegelapan yang di sebut *Dark Zone* yang merupakan tempat di mana semua kekacauan berasal. Di sini pemain akan memiliki misi yang harus di lakukan agar semua kekacauan bisa terselesaikan. Misi tersebut adalah sebagai berikut :

1. Mengalahkan semua musuh yang ada pada dunia *Dark Zone.*
2. Dan kembali ke *Kuburan Geonidas* untuk memberitahukan semua musuh telah di kalahkan.

Untuk menggerakan karakter pemain dapat menggunakan tombol yang ada pada *keyboard* yaitu, sebagai berikut :

1. Tombol W : Menggerakan karakter ke depan.
2. Tombol S : Menggerakan karakter ke belakang.
3. Tombol A : Menggerakan karakter ke kiri.
4. Tombol D : Menggerakan karakter ke kanan.
5. Tombol E : Untuk berinteraksi dengan objek lain.
6. Tombol SPACE : Melompat.
7. Tombol B : Membuka menu serangan.

Karakter yang di mainkan oleh pemain adalah seorang prajurit bernama *Genidas* dia di tugaskan oleh pimpinan divisi *Yellow Spartan* untuk memeriksa *Embun* pos dan *Kuburan Geonidas* yang di duga mengalami keanehan. Di tempat ini *Genidas* harus bisa mengatasi semua masalah yang ada pada *Embun* pos dan *Kuburan Geonidas* dan mencari tahu apa yang terjadi pada tempat tersebut.

4.1.2.4 Action Script

Action script merupakan kumpulan bahasa pemograman yang dibuat dalam *game* untuk memberikan perintah kepada objek yang terdapat pada *game*. Dan pada *game* ini terdapat beberapa *ActionScript* untuk memberikan perintah pada objek, *ActionScript* yang ada pada *game* ini adalah sebagai berikut :

1. **Action Script AdvancedMovement**

Action Script ini bertujuan untuk memberikan perintah kepada karakter pemain agar bisa di gerakan dengan inputan yang telah di sesuaikan. Dibawah ini adalah beberapa bagian dari *syntak* yang di gunakan pada Action Script AdvancedMovement :

```
using UnityEngine;
using System.Collections;

[RequireComponent (typeof(CharacterController))]
public class AdvancedMovement : MonoBehaviour {
    public enum State{
        Idle,
        Init,
        Setup,
        Run
    }
    public enum Turn{
        left = -1,
        none = 0,
        right = 1
    }
    public enum Forward{
        back = -1,
        none = 0,
        forward = 1
    }

    public string walkAnimName;
    public string runAnimName;
    public string jumpAnimName;
    public string idleAnimName;
    public string strafeAnimName;
    public string fallAnimName;
    public string swimAnimName;

    public AnimationClip meleeAttack;

    public float walkSpeed = 5;
    public float runMultiplier = 2;
    public float strafeSpeed = 2.5f;
    public float rotateSpeed = 250;
```

```
37      public float gravity = 20;
38      public float airTime = 1;
39      public float fallTime = .5f;
40      public float jumpHeight = 100;
41      public float jumpTime = 2.5f;
42
43      public CollisionFlags _collisionFlags;
44      private Vector3 _moveDirection;
45      private Transform _myTransform;
46      private CharacterController _controller;
47
48      private Turn _turn;
49      private Forward _forward;
50      private Turn _strafe;
51      private bool _run;
52      private bool _jump;
53      private bool _isSwimming;
54
55
56      private State _state;
57
58      public void Awake(){
59          _myTransform = transform;
60          _controller = GetComponent<CharacterController>();
61          _state = AdvancedMovement.State.Init;
62      }
63
64
65      IEnumerator Start () {
66          while(true){
67              switch(_state){
68              case State.Init:
69                  Init();
70                  break;
71              case State.Setup:
72                  SetUp();
```

Gambar 4.1 ActionScript AdvancedMovement

2. Action Script PlayerInput

   Action script ini berguna untuk mengatur inputan pada keyboard sehingga karakter dapat dimainkan. Action script ini berintegrasi dengan action script AdvancedMovement sehingga pada karakter harus memiliki kedua

actionscript ini. Di bawah ini merupakan beberapa bagian dari *action script* PlayerInput adalah sebagai berikut :

```
using UnityEngine;
using System.Collections;

[RequireComponent (typeof(AdvancedMovement))]
public class PlayerInput : MonoBehaviour {

    // Use this for initialization
    void Start () {

    }

    // Update is called once per frame
    void Update () {
        if(Input.GetButton("Move Forward")){
            if(Input.GetAxis("Move Forward") > 0){
                SendMessage("MoveMeForward",AdvancedMovement.Forward.forward);
            }
            else{
                SendMessage("MoveMeForward",AdvancedMovement.Forward.back);
            }
        }
        if(Input.GetButtonUp("Move Forward")){
            SendMessage("MoveMeForward",AdvancedMovement.Forward.none);
        }

        if(Input.GetButton("Rotate Player")){
            if(Input.GetAxis("Rotate Player") > 0){
                SendMessage("RotateMe",AdvancedMovement.Turn.right);
            }
            else{
                SendMessage("RotateMe",AdvancedMovement.Turn.left);
            }
        }
        if(Input.GetButtonUp("Rotate Player")){
            SendMessage("RotateMe",AdvancedMovement.Turn.none);
```

```
37
38          if(Input.GetButton("Strafe")){
39              if(Input.GetAxis("Strafe") > 0){
40                  SendMessage("Strafe",AdvancedMovement.Turn.right);
41              }
42              else{
43                  SendMessage("Strafe",AdvancedMovement.Turn.left);
44              }
45          }
46          if(Input.GetButtonUp("Strafe")){
47              SendMessage("Strafe",AdvancedMovement.Turn.none);
48          }
49
50          if(Input.GetButton("Jump")){
51              SendMessage ("JumpUp");
52          }
53
54          if(Input.GetButtonDown("Run")){
55              SendMessage("ToggleRun");
56          }
57      }
58  //Karakter agar bisa berenang
59      public void OnTriggerEnter(Collider other){
60          if(other.CompareTag("Water"))
61              SendMessage("IsSwimming",true);
62      }
63
64      public void OnTriggerExit(Collider other){
65          if(other.CompareTag("Water"))
66              SendMessage("IsSwimming",false);
67      }
68  }
```

Gambar 4.2 ActionScript Player Input

3. ActionScript LevelAndExperience

   Action script ini berguna untuk memberitahukan level pada pemain apabila telah berhasil menyelesaikan misi. Di bawah ini merupakan action script daro LevelAndExperience :

```
#pragma strict

var CurrentLevel : float = 1f;
var CurrentExp : float = 0f;
var MaxLevel : float;
var MaxExp : float;

function Start () {

}

function Update () {
        LevelExp();
}

function OnGUI () {

GUI.Box(Rect( 1, 578, Screen.width, 20), CurrentExp.ToString());

GUI.Label(Rect( 50, 50, 1000, 1000), "Level :");
GUI.Label(Rect( 100, 50, 1000, 1000), CurrentLevel.ToString());
}

function LevelExp(){
        if(CurrentExp == 100){
                CurrentLevel = 2f;
        }

        if(CurrentExp == 300){
                CurrentLevel = 3f;
        }

        if(CurrentExp == 900){
                CurrentLevel = 4f;
```

Gambar 4.3 ActionScript LevelAndExperience

4.2 Analisis Kebutuhan Non – Fungsional

Analisis kebutuhan non-fungsional adalah analisis yang dibutuhkan untuk mengetahui spesifikasi kebutuhan sistem yang di gunakan.Spesifikasi meliputi beberapa komponen apa saja yang dibutuhkan untuk sistem yang akan di bangun sampai sistem tersebut di implementasikan. Analisis kebutuhan non-fungsional

menjelaskan analisis kebutuhan perangkat lunak, analisis kebutuhan perangkat keras, dan analisis pengguna.

4.2.1 Analisis Kebutuhan Perangkat Lunak

Game rpg yang di buat oleh *game engine* Unity3D memiliki beberapa kebutuhan perangkat lunak sebagai berikut :

1. Sistem Operasi : Windows dan Mac.
2. Web Browser : Chrome, IE, Firefox.
3. WebPlayer : Unityweb player.
4. Flash : Flash Player.
5. IDE : MonoDevelop.

Dari hasil kebutuhan perangkat lunak, terdapat beberapa hal yang harus di perhatikan agar sistem yang akan di rancang dapat berjalan dengan baik. Pengguna di harapkan dapat memenuhi kebutuhan perangkat lunak yang telah di rekomendasikan agar aplikasi dapat berjalan dengan baik.

4.2.2 Analisis Kebutuhan Perangkat Keras

Agar *game* ini dapat berjalan dengan baik, maka di butuhkan spesifikasi perangkat keras yang sesuai dengan spesifikasi yang di butuhkan. Spesifikasi minimum perangkat keras yang dibutuhkan agar dapat menjalankan *game engine* Unity3D adalah sebagai berikut :

Gambar 4.1 Tabel Kebutuhan Perangkat Keras Game Engine Unity3D

Perangkat	**Spesifikasi**
Processor	Kecepatan 2.9 Ghz
RAM	2 GB
Harddisk	160 GB
Monitor	Resolusi 1024 x 768
VGA	Direct x 9.0 128 bit 1024 Gb

Dan untuk dapat menjalankan *game* yang telah di buat di butuhkan perangkat keras yang sesuai dan spesifikasi kebutuhan perangkat keras tersebut adalah sebagai berikut :

Gambar 4.2. Tabel Kebutuhan Perangkat Keras Game RPG

Perangkat	**Spesifikasi**
Processor	Kecepatan 1.0 Ghz
RAM	1 GB
VGA	Direct x 9.0

4.3 Analisis Kebutuhan User

Game yang di buat di tunjukan untuk usia remaja antara 15 sampai 18 keatas, alasannya adalah karena dengan usia antara 15 sampai dengan 18 remaja sudah memiliki kepekaan dan ketajaman dalam berpikir. Pada saat usia tersebut remaja dapat dengan cepat mengambil keputusan dan berpikir kritis, sehingga dapat menjalankan *game* yang di buat dengan baik. Dan dengan usia antara 15 sampai dengan 18 tahun keatas, setiap mental dan kemampuan psikologis sudah dapat di jaga dengan baik. Sehingga *game* yang mengandung kekerasan tidak dapat berpengaruh terhadap mental dan psikologisnya. Sehingga *game* dapat di mainkan dengan aman.

4.3 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional adalah analisis menggambarkan proses kegiatan yang akan di gunakan pada sebuah sistem dan menjelaskan kebutuhan yang di perlukan, agar system dapat berjalan dengan baik sesuai kebutuhan.

4.3.1 Perancangan Sistem

Perancangan sistem adalah merancang atau mendesain suatu sistemyang baik yang isinya adalah langkah-langkah operasi dalam prosespengolahan data dan proses prosedur-prosedur untuk mendukung operasisistem. Tujuan dari perancangan sistem adalah untuk memenuhi kebutuhanpara pemakai sistem serta

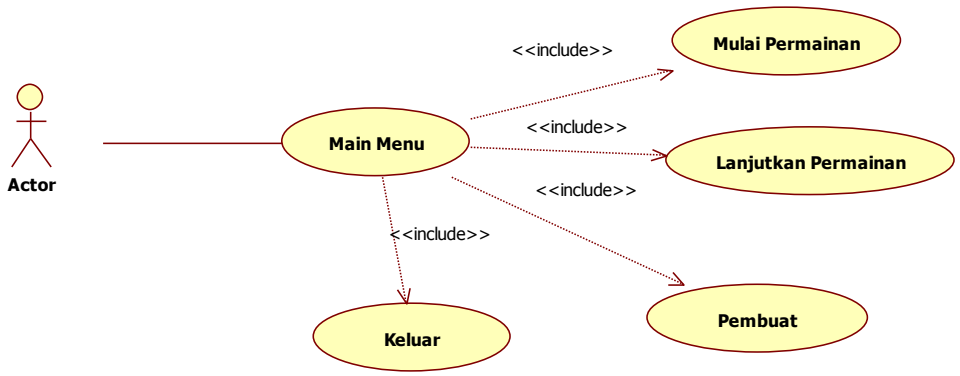
memberikan gambaran yang jelas dan rancangbangun yang lengkap kepada programmer dan ahli-ahli yang terlibat didalamnya.

Perancangan sistem di modelkan dengan menggunakan *UML (Unified Modeling Language).* Dimana tahap-tahap perancangan yang dilakukan dalam membangun *game* RPG *(Role Playing Game)* antara lain *Use Case Diagram, Class Diagram, Sequence Diagram, Activity Diagram.*

1.1 *Use Case Diagram*

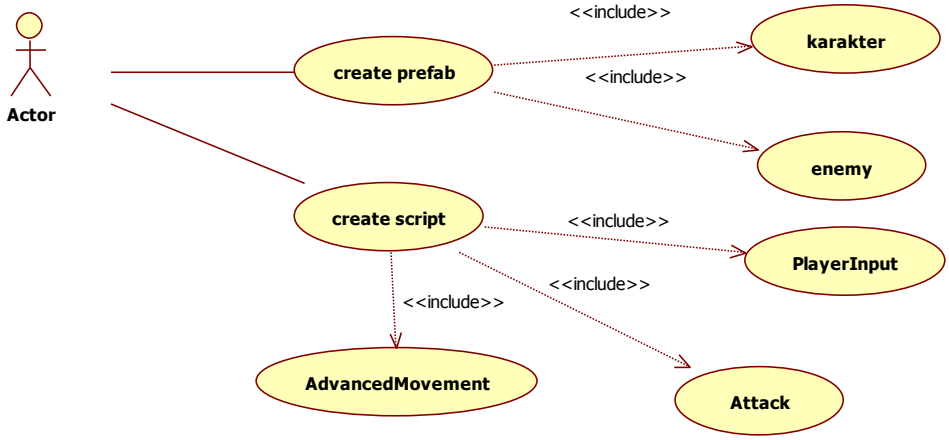
Use Case Diagram adalah rangkaiansekelompok yang saling terkait dan membentuk sistem secara teratur yang dilakukan atau diawasi oleh sebuah aktor. Use case digunakan untuk membentuk tingkah-laku benda dalam sebuah model serta di realisasikan oleh sebuah collaboration. Umumnya use case digambarkan dengan sebuah elips dengan garis yang solid, biasanya mengandung nama.Use case menggambarkan proses system (kebutuhan system dari sudut pandang user).

Berikut ini adalah perancangan proses-proses yang terdapat pada *game* RPG (*Role Playing game*) yang digambarkan dengan *Use Case Diagram* yang dapat di lihat pada Gambar 4.4.



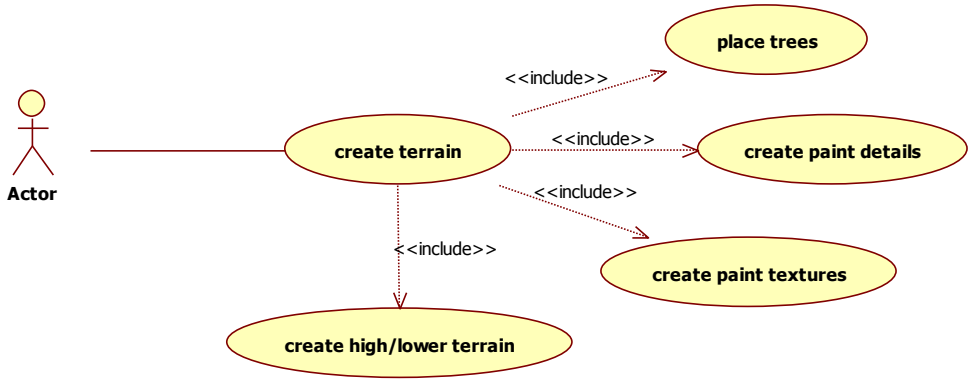
Gambar 4.4 Use Case Diagram *Main Menu.*

Use case terdiri dari 3 bagian yaitu definisi *actor*, definisi *Use Case* dan skenario *Use Case*. Dan di bawah ini merupakan *Use Case* proses-proses yang ada pada *game engine* UNITY3D, gambar dapat di lihat sebagai berikut :



Gambar 4.5 Use Case Diagram *Create Prefab* dan *Script.*

Di bawah ini dapat dilihat *Use Case Diagram* pembuatan *terrain*. Gambar dapat dilihat sebagai berikut :



Gambar 4.6 Use Case Diagram *Create Terrain.*

1.1.1 Definisi Aktor

Definisi actor berfungsi untuk menjelaskan *actor* yang terdapat pada *Use Case Diagram.*Definisi *actor* dijelaskan pada Tabel 4.2.

Tabel 4.3. Definisi Actor

No	Actor	Keterangan
1.	User	Orang yang memainkan *game*

1.1.2 Definisi *Use Case*

Definisi *Use Case* berfungsi untuk menjelaskan fungsi *Use Case* yang terdapat pada *Use Case Diagram.* Definisi *Use Case* diterangkan pada Tabel 4.4.

Tabel 4.4. Definisi *Use Case* Main Menu

NO	Use Case	Keterangan
1.	Main Menu	Menu untuk memulai permainan.
2.	Lanjutkan Permainan	Menu untuk memanggil karakter.
3.	Mulai Permainan	Permainan sudah bisa di mainkan.
4.	Pembuat	Menu untuk mengetahui pembuat *game*.
5.	Keluar	Menu utnuk keluar dari *game.*

Dan berikutnya adalah definisi *Use Case* pada *Use Case Diagram* Create Prefab dan *Script*. Definsi dapat dilihat di bawah ini.

Tabel 4.5. Definisi *Use Case Create Prefab* dan *Script*

NO	Use Case	Keterangan
1.	*Create Prefab*	Membuat cetakan untuk objek.
2.	*Create Script*	Membuat script.
3.	Karakter	Model 3D untuk karakter
4.	*Enemy*	Model 3D untuk enemy
5.	*AdvancedMovement*	Script untuk menggerakan karakter.
6.	*PlayerInput*	Script untuk memasukan kunci pada keyboard.
7.	*Attack*	Script animasi untuk menyerang pada model karakter

Dan berikutnya adalah definisi *Use Case* pada *Use Create Terrain*. Definsi dapat dilihat di bawah ini :

Tabel 4.6. Definisi *Use Case Create Terrain*

NO	Use Case	Keterangan
1.	*Create Terrain*	Membuat terrain atau tanah
2.	*Place Trees*	Memasukan pohon
3.	*Create Paint Details*	Memasukan rumput
4.	*Create Paint Textures*	Memasukan tipe tanah
5.	*Create high/lower terrain*	Membuat bukit atau gunung.

1.1.3 Skenario Use Case

Skenario *Use Case* menggambarkan alur penggunaan system dimana setiap scenario di gambarkan dari sudut pandang actor yang berinteraksi dengan perangkat lunak dalam berbagai cara.

1.1.4 Skenario Use Case Main Menu

Tabel 4.7 Skenario *Use Case* Main Menu

Identifikasi	
Nomor	1
Nama	Main Menu
Tujuan	Memilih Menu
Deskripsi	Proses untuk memulai permainan
Aktor	User
Skenario	
Kondisi Awal	Pemain berada pada main menu
Aksi Aktor	**Reaksi Sistem**
Memilih menu yang di inginkan	Menampilkan menu yang di pilih
Kondisi Akhir	Aplikasi *game* memulai permainan

1.1.5 Skenario *Use Case* Lanjutkan Permainan

Tabel 4.8. Skenario *Use Case* Lanjutkan Permainan

Identifikasi	
Nomor	2
Nama	Lanjutkan Permainan
Tujuan	Memanggil karakter.
Deskripsi	Proses untuk memulai *game.*
Aktor	User
Skenario	
Kondisi Awal	User berada pada main menu
Aksi Aktor	**Reaksi Sistem**
Memilih menu Load Karakter	Memproses menu yang di pilih
Kondisi Akhir	User dapat memulai permainan

1.1.6 Skenario *Use Case* Memulai Permainan

Tabel 4.9 Skenario *Use Case* Memulai Permainan

Identifikasi	
Nomor	3
Nama	Memulai Permainan
Tujuan	Memainkan *game*
Deskripsi	Proses *game*The Lost Of Spartan
Aktor	User
Skenario	
Kondisi Awal	User berada pada menu Load Karakter
Aksi Aktor	**Reaksi Sistem**
User memainkan *game*	Menjalankan *game*
Kondisi Akhir	Permainan berakhir dan kembali ke main menu.

1.1.7 Skenario *Use Case* Pembuat

Tabel 4.10 Skenario Use Case Pembuat

Identifikasi	
Nomor	4
Nama	Pembuat
Tujuan	Memberi informasi tentang pembuat *game*
Deskripsi	Memberi informasi pembuat *game*
Aktor	User
Skenario	
Kondisi Awal	User berada pada main menu
Aksi Aktor	**Reaksi Sistem**
Memilih menu Pembuat	Memproses menu yang di pilih
Kondisi Akhir	User akan berada pada tampilan menu pembuat

1.1.8 Skenario *Use Case* Keluar

Tabel 4.11 Skenario Use Case Menu Atribut

Identifikasi	
Nomor	5
Nama	Keluar
Tujuan	Keluar dari permainan
Deskripsi	Menutup permainan
Aktor	User
Skenario	
Kondisi Awal	User berada pada main menu
Aksi Aktor	**Reaksi Sistem**
Memilih menu Keluar	Memproses menu yang di pilih
Kondisi Akhir	Game akan memberhentikan dan menutup permainan

Di bawah ini adalah skenario yang ada pada *Use Case Diagram Create Prefab* dan *Script.*

1.1.9 Skenario *Use Case Create Prefab*

Tabel 4.12 Skenario *Use Case Create Prefab*

Identifikasi	
Nomor	**1**
Nama	*Create Prefab*
Tujuan	Membuat cetakan untuk objek
Deskripsi	Membuat cetakan agar objek dapat di simpan
Aktor	User
Skenario	
Kondisi Awal	User berada pada tampilan utama
Aksi Aktor	**Reaksi Sistem**
Memilih menu create prefab	Membuat prefab
Kondisi Akhir	Prefab siap digunakan

1.1.10 Skenario *Use Case Create Script*

Tabel 4.13 Skenario *Use Case Create Script*

Identifikasi	
Nomor	**2**
Nama	*Create Script*
Tujuan	Membuat script kosong
Deskripsi	Membuat script untuk memasukan koding
Aktor	User
Skenario	
Kondisi Awal	User berada pada tampilan utama
Aksi Aktor	**Reaksi Sistem**
Memilih menu create script	Membuat script
Kondisi Akhir	Script siap digunakan untuk di masukan koding

1.1.11 Skenario *Use Case* Karakter

Tabel 4.14 Skenario *Use Case* Karakter

Identifikasi	
Nomor	**3**
Nama	Karakter
Tujuan	Memasukan model 3D karakter
Deskripsi	Memasukan karakter agar terdapat model 3D yang di gunakan dalam permainan
Aktor	User
Skenario	
Kondisi Awal	User berada pada tampilan utama
Aksi Aktor	**Reaksi Sistem**
Memilih menu new asset	Menampilkan tampilan asset
Kondisi Akhir	Memasukan asset berupa model 3D karakter

1.1.12 Skenario *Use Case* Enemy

Tabel 4.15 Skenario *Use Case* Enemy

Identifikasi	
Nomor	**4**
Nama	Enemy
Tujuan	Memasukan model 3D enemy
Deskripsi	Memasukan enemy agar terdapat model 3D yang di gunakan untuk menyerang karakter
Aktor	User
Skenario	
Kondisi Awal	User berada pada tampilan utama
Aksi Aktor	**Reaksi Sistem**
Memilih menu new asset	Menampilkan tampilan asset
Kondisi Akhir	Memasukan asset berupa model 3D enemy

1.1.13 Skenario *Use Case* AdvancedMovement

Tabel 4.16 Skenario *Use Case* AdvancedMovement

Identifikasi	
Nomor	5
Nama	AdvancedMovement
Tujuan	Membuat action script Advanced Movement
Deskripsi	Untuk membuat karakter dapat di gerakan dengan action script AdvancedMovement
Aktor	User
Skenario	
Kondisi Awal	User berada pada tampilan utama
Aksi Aktor	**Reaksi Sistem**
Memilih menu create script	Menampilkan script
Kondisi Akhir	Menampilkan script untuk pembuatan koding Advanced Movement

1.1.14 Skenario *Use Case* PlayerInput

Tabel 4.17 Skenario *Use Case* PlayerInput

Identifikasi	
Nomor	6
Nama	PlayerInput
Tujuan	Membuat action script PlayerInput
Deskripsi	Untuk mengatur tombol-tombol pada keyboard untuk menggerakan karakter
Aktor	User
Skenario	
Kondisi Awal	User berada pada tampilan utama
Aksi Aktor	**Reaksi Sistem**
Memilih menu edit , project setting	Menampilkan menu input
Kondisi Akhir	Menampilkan menu input untuk pengaturan tombol

1.1.15 Skenario *Use Case* Attack

Tabel 4.18 Skenario *Use Case* Attack

Identifikasi	
Nomor	7
Nama	Attack
Tujuan	Membuat action script Attack
Deskripsi	Untuk mengatur animasi pada karakter
Aktor	User
Skenario	
Kondisi Awal	User berada pada tampilan utama
Aksi Aktor	**Reaksi Sistem**
Memilih menu create script	Menampilkan script
Kondisi Akhir	Menampilkan script untuk pembuatan koding Attack

Di bawah ini adalah skenario yang ada pada *Use Case Diagram Create Terrain.*

1.1.16 Skenario *Use Case Create Terrain*

Tabel 4.19 Skenario *Use Case Create Terrain*

Identifikasi	
Nomor	1
Nama	*Create Terrain*
Tujuan	Membuat terrain
Deskripsi	Untuk membuat latar pegunungan
Aktor	User
Skenario	
Kondisi Awal	User berada pada tampilan utama
Aksi Aktor	**Reaksi Sistem**
Memilih menu create terrain	Menampilkan menu terrain
Kondisi Akhir	Memilih menu terrain untuk membuat latar belakang pegunungan

1.1.17 Skenario Place Trees

Tabel 4.20 Skenario *Use Case* Place Trees

Identifikasi	
Nomor	2
Nama	Place Trees
Tujuan	Menerapkan Pohon
Deskripsi	Untuk menambahkan pepohonan
Aktor	User
Skenario	
Kondisi Awal	User berada pada tampilan utama
Aksi Aktor	**Reaksi Sistem**
Memilih menu Place Trees	Menampilkan menu trees
Kondisi Akhir	Memilih tree/pohon untuk di masukan ke dalam *game*

1.1.18 Skenario *Use Case Create Paint Details*

Tabel 4.21 Skenario *Use Case Create Paint Details*

Identifikasi	
Nomor	3
Nama	*Create Paint Details*
Tujuan	Menambah rumput
Deskripsi	Untuk menambahkan efek rumput pada *game*
Aktor	User
Skenario	
Kondisi Awal	User berada pada tampilan utama
Aksi Aktor	**Reaksi Sistem**
Memilih menu *Create Paint Details*	Menampilkan menu grass
Kondisi Akhir	Memilih rumput untuk di masukan ke dalam *game*

1.1.19 Skenario *Use Case Create Paint Textures*

Tabel 4.22 Skenario *Use Case Create Paint Textures*

Identifikasi	
Nomor	4
Nama	*Create Paint Textures*
Tujuan	Menambah jenis tanah
Deskripsi	Untuk menambahkan efek tanah yang berbeda
Aktor	User
Skenario	
Kondisi Awal	User berada pada tampilan utama
Aksi Aktor	**Reaksi Sistem**
Memilih menu *Create Paint Textures*	Menampilkan menu paint terrain
Kondisi Akhir	Memilih jenis tanah

1.1.20 Skenario *Use Case Create high/lower terrain*

Tabel 4.23 Skenario *Use Case Create high/lower terrain*

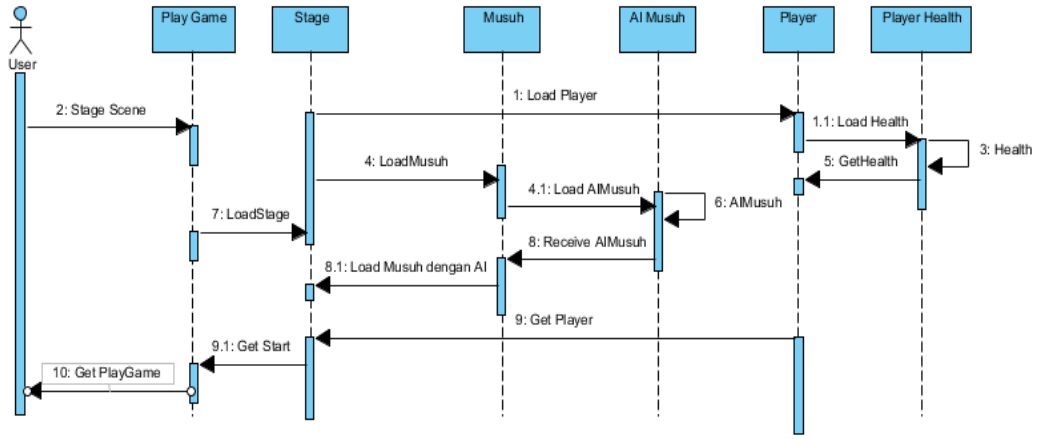
Identifikasi	
Nomor	5
Nama	*Create high/lower terrain*
Tujuan	Merubah tinggi atau rendah terrain
Deskripsi	Untuk menambahkan ukuran pada terrain
Aktor	User
Skenario	
Kondisi Awal	User berada pada tampilan utama
Aksi Aktor	**Reaksi Sistem**
Memilih menu *Create high/lower terrain*	Menampilkan menu paint terrain
Kondisi Akhir	Mengatur ketinggian atau kerendahan terrain

1.3 Sequence Diagram

Sequence diagram adalah suatu diagram yang menggambarkan interaksi antar obyek dan mengindikasikan komunikasi diantara obyek-obyek tersebut. Diagram ini juga menunjukkan serangkaian pesan yang dipertukarkan oleh obyek-obyek yang melakukan suatu tugas atau aksi tertentu. Obyek-obyek tersebut kemudian diurutkan dari kiri ke kanan, aktor yang menginisiasi interaksi biasanya ditaruh di paling kiri dari diagram. Pada diagram ini, dimensi vertikal merepresentasikan waktu. Bagian paling atas dari diagram menjadi titik awal dan waktu berjalan ke bawah sampai dengan bagian dasar dari diagram. Garis Vertical, disebut lifeline, dilekatkan pada setiap obyek atau aktor. Kemudian, lifeline tersebut digambarkan menjadi kotak ketika obyek melakukan suatu operasi,kotak tersebut disebut activation box. Obyek dikatakan mempunyai live activation pada saat tersebut.Pesan yang dipertukarkan antar obyek digambarkan sebagai sebuah anak panah antara activation box pengirim dan penerima.

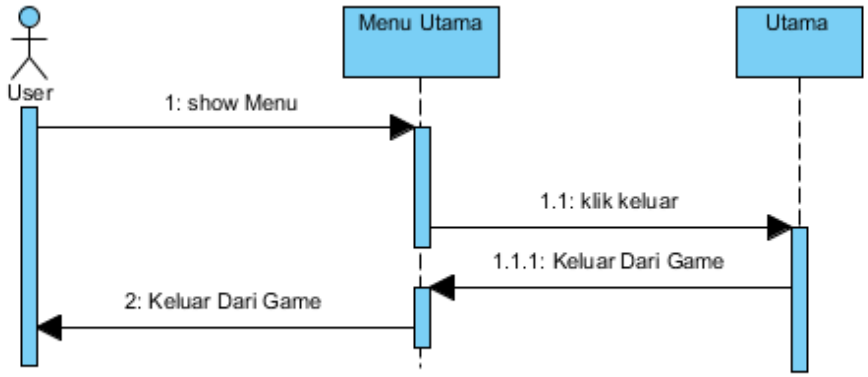
Bagian paling atas dari diagram menjadi titik awal dan waktu berjalan ke bawah sampai dengan bagian dasar dari diagram. Garis Vertical, disebut lifeline, dilekatkan pada setiap obyek. Kemudian, lifeline tersebut digambarkan menjadi kotak ketika obyek melakukan suatu operasi , kotak tersebut disebut activation box. Obyek dikatakan mempunyai live activation pada saat tersebut.Pesan yang dipertukarkan antar obyek digambarkan sebagai sebuah anak panah antara activation box pengirim dan penerima.

1.3.2 Sequence Diagram Mulai Permainan



Gambar 4.5 Sequence Diagram Mulai Permainan

1.3.3 Sequence Diagram Keluar

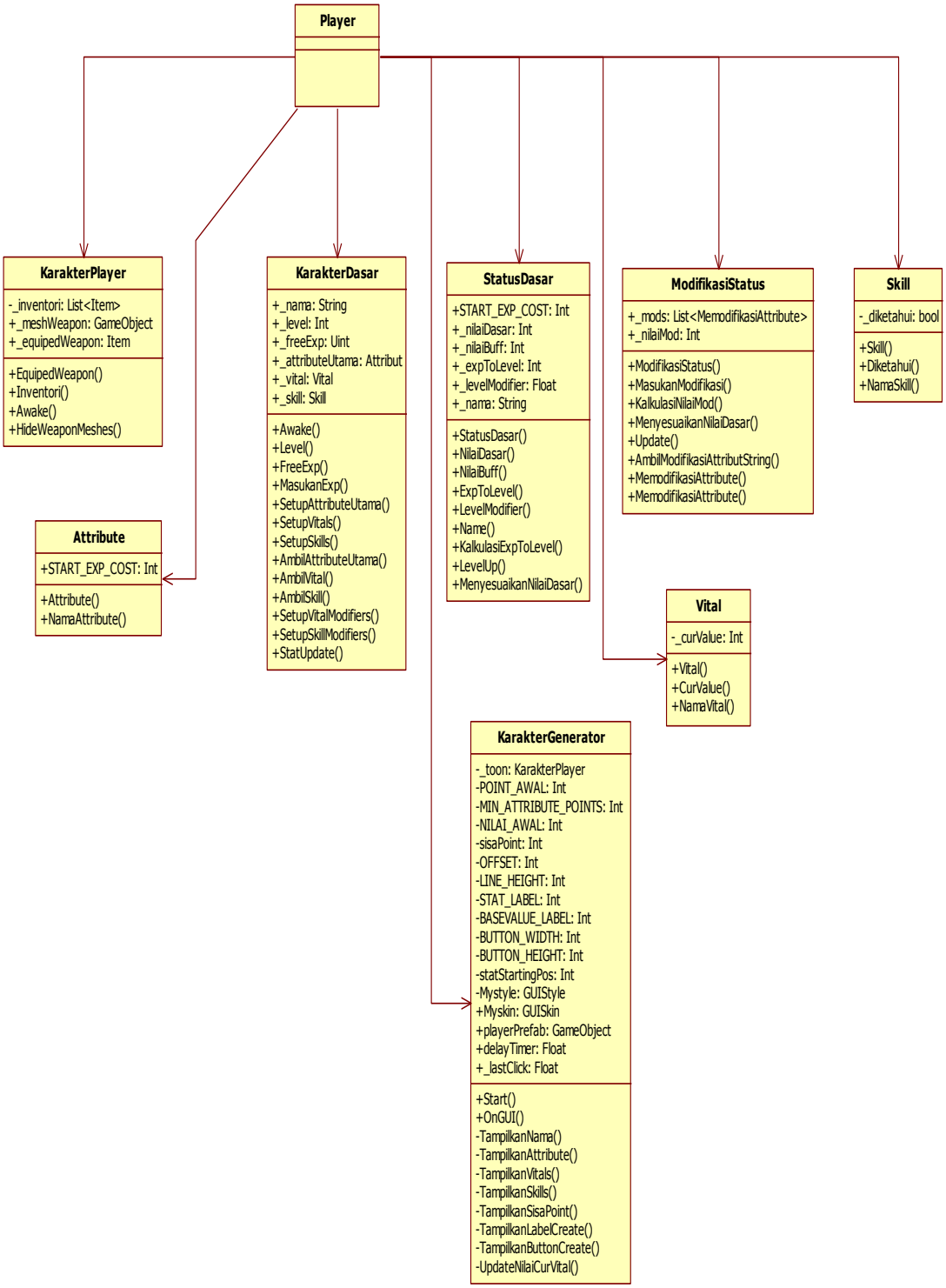


Gambar 4.6 Sequence Diagram Keluar

1.4 Class Diagram

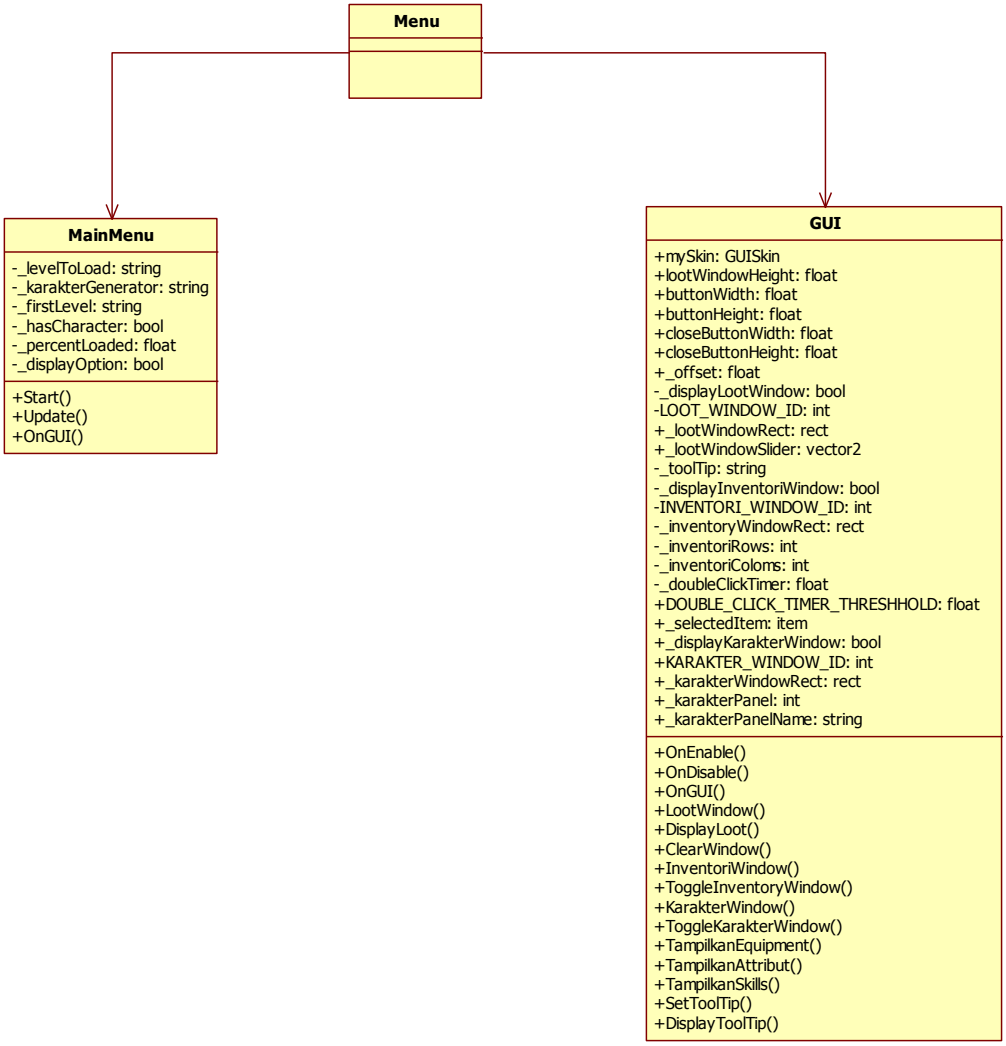
Class diagram digunakan untuk menampilkan kelas-kelas dan paket-paket di dalam system.Class diagram memberikan gambaran system secara statis dan relasi antar mereka.Biasanya, dibua beberapa class diagram untuk system tunggal. Beberapa diagram akan menampilkan subset dari kelas-kelas dan relasinya. Dapat dibuat beberapa diagram sesuai dengan yang diinginkan untuk mendapatkan gambaran lengkap terhadap system yang dibangun.

Class diagram adalah alat perancangan terbaik untuk tim pengembang. Diagram tersebut membantu pengembang mendapatkan struktur system sebelum kode ditulis, dan membantu untuk memastikan bahwa system adalah desain terbaik. Berikut ini merupakan gambar class diagram *palyer* pada *game* RPG (*Role Playing Game*) yang dapat di lihat pada gambar 4.7.



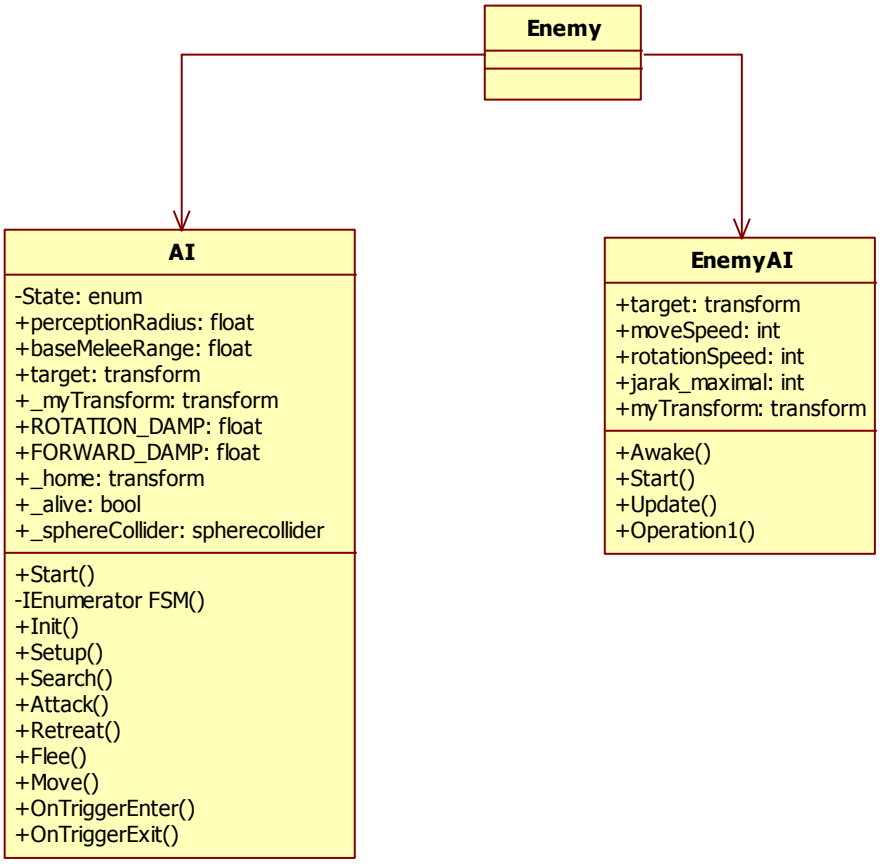
Gambar 4.7 Class Diagram Player

Di bawah ini merupakan *class diagram* Main Menu untuk memulai *game*



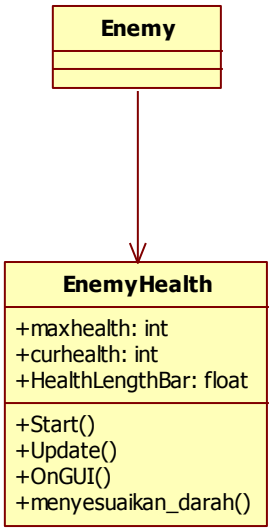
Gambar 4.8 Class Diagram *Menu*RPG (*Role Playing Game*)

Di bawah ini merupakan gambar *class diagram* pada *Enemy*, dapat dilihat di bawah ini.



Gambar 4.9 Gambar Class Diagram EnemyAI

Di bawah ini merupakan gambar *class diagram EnemyHealth*, dapat dilihat di bawah ini.



Gambar 4.10 Class Diagram EnemyHealth

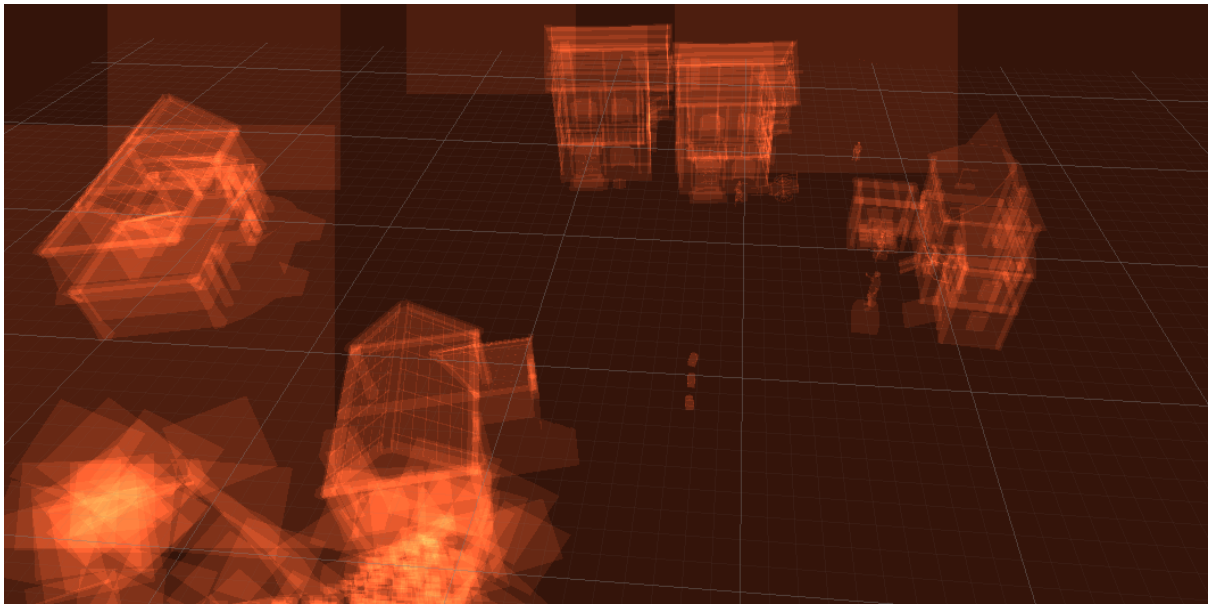
4.4 Storyboard

Storyboard adalah sketsa gambar yang disusun berurutan sesuai dengan naskah, dengan storyboard kita dapat menyampaikan ide cerita kita kepada orang lain dengan lebih mudah, karena kita dapat menggiring khayalan seseorang mengikuti gambar-gambar yang tersaji, sehingga menghasilkan persepsi yang sama pada ide cerita kita. Berikut ini merupakan storyboard dari *game* RPG (*Role Playing Game*).



Gambar 4.11 Storyboard Tempat Level 1.

Level 1 adalah tempat dimana perjalanan pemain dimulai, di stage ini terdapat beberapa musuh yang akan muncul dan mengejar pemain, sehingga pemain di tuntut untuk mewaspadai arah yang akan di tuju. Pada level ini musuh masih memiliki level yang rendah sehingga pemain tidak akan kesulitan dalam mengalahkan musuh yang ada pada level ini.



Gambar 4.12 Storyboard Desa Pada Level 1

Gambar di atas merupakan denah desa yang ada pada level 1, di sini pemain akan pertama muncul dan pemain dapat mengeksplore ke tempat lain untuk mencari musuh agar misi dapat di selesaikan dengan baik.

4.5 Perancangan Antar Muka

Perancangan antarmuka dibutuhkan untuk mewakili keadaan sebenarnyadari aplikasi yang akan dibangun, berikut akan disajikan perancangan antarmuka dari aplikasi yang akan dibangun:



Gambar 4.13 Antar Muka MainMenu

Untuk lebih jelas mengetahui keterangan dan navigasi pada perancangan antar muka permainan maka dibuatlah table khusus yang mendeskripsikan keterangan tersebut, keterangan detail perancangan antar muka permainan dapat di lihat pada Tabel 4.10.

Tabel 4.24 Keterangan Main Menu

No	Keterangan	Navigasi
1.	Mulai Permainan	Dengan memilih menu untuk memulai permainan
2.	Lanjutkan Permainan	Melanjutkan permainan yang telah di simpan
3.	Pembuat	Informasi pembuat *game*
4.	Keluar Game	Memilih menu untuk keluar dari *game*.

4.6 Perancangan Prosedural

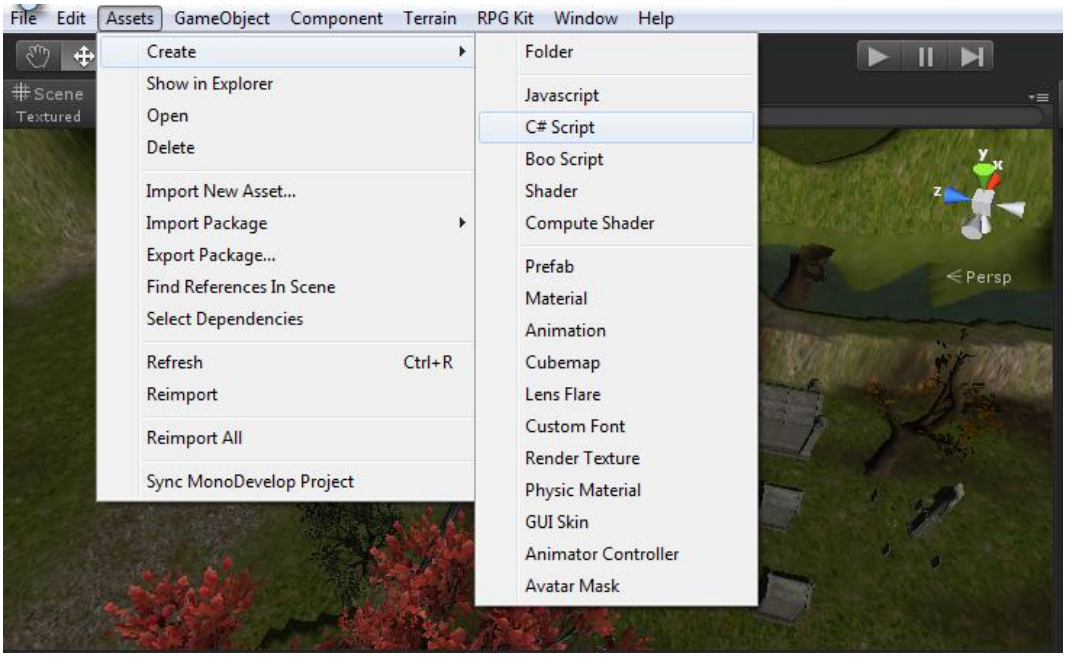
Perancangan prosedural merupakan perancangan yang dilakukan untuk menetapkan detail program algoritma yang akan dinyatakan kedalam suatu program. Adapun perancangan prosedural untuk *game* yang akan dibangun adalah:

1. Prosedur memilih main menu, merupakan prosedural untuk memulai permainan.
2. Prosedur memilih atribut pada karakter, merupakan prosedural yang terjadi ketika *user* akan bermain *game* di awal permainan untuk membuat karakter.

4.7 Perancangan Program

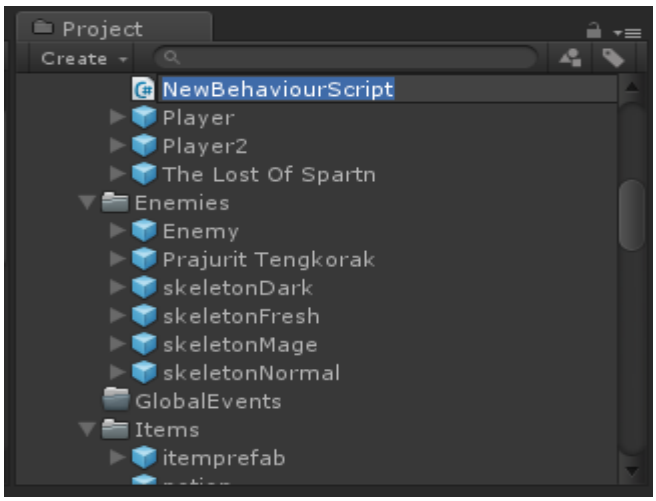
Perancangan program merupakan bagaimana kita membuat *AI (Artificial Intellegence)* yang berguna sebagai *action script* pada objek yang ada pada *game* yang akan di buat. Di bawah ini merupakan cara bagaimana kita membuat program dengan menggunakan IDE Monodevelop, dan tahapannya adalah sebagai berikut :

1. Membuka *game engine Unity3D,* lalu memilih menu : *Assets* lalu pilih submenu *Create* dan di situ terdapat beberapa pilihan bahasa pemograman yang dapat kita buat. Bahasa pemograman yang penulis buat adalah menggunakan C#, maka dari itu pilih *C# Script.* Untuk lebih jelas dapat melihat gambar di bawah ini.



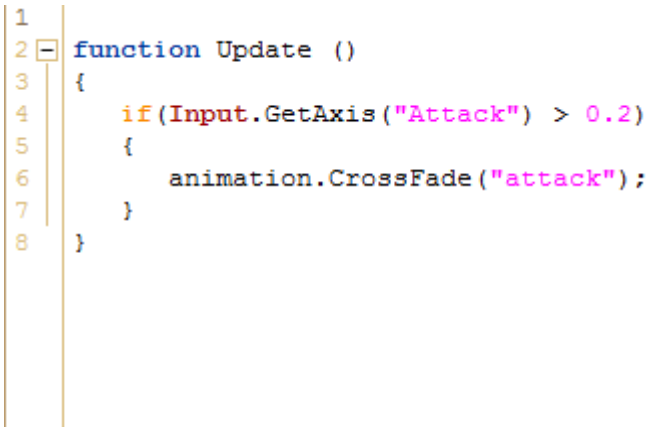
Gambar 4.14 Memilih C# Script.

Setelah itu akan ada file baru dengan nama *NewBehaviourScript* yang berguna untuk menampung semua method yang akan kita buat. Untuk membuka file tersebut dapat di klik 2 kali. Dan setelah di klik akan muncul IDE Monodevelop. Untuk lebih jelasnya dapat melihat gambar di bawah ini :



Gambar 4.15 NewBehaviourScript

Setelah di klik akan muncul IDE Monodevelop yang dapat di lihat di bawah ini.



```
1
2  function Update ()
3  {
4      if(Input.GetAxis("Attack") > 0.2)
5      {
6          animation.CrossFade("attack");
7      }
8  }
```

Gambar 4.16 Membuat program Animasi Attack

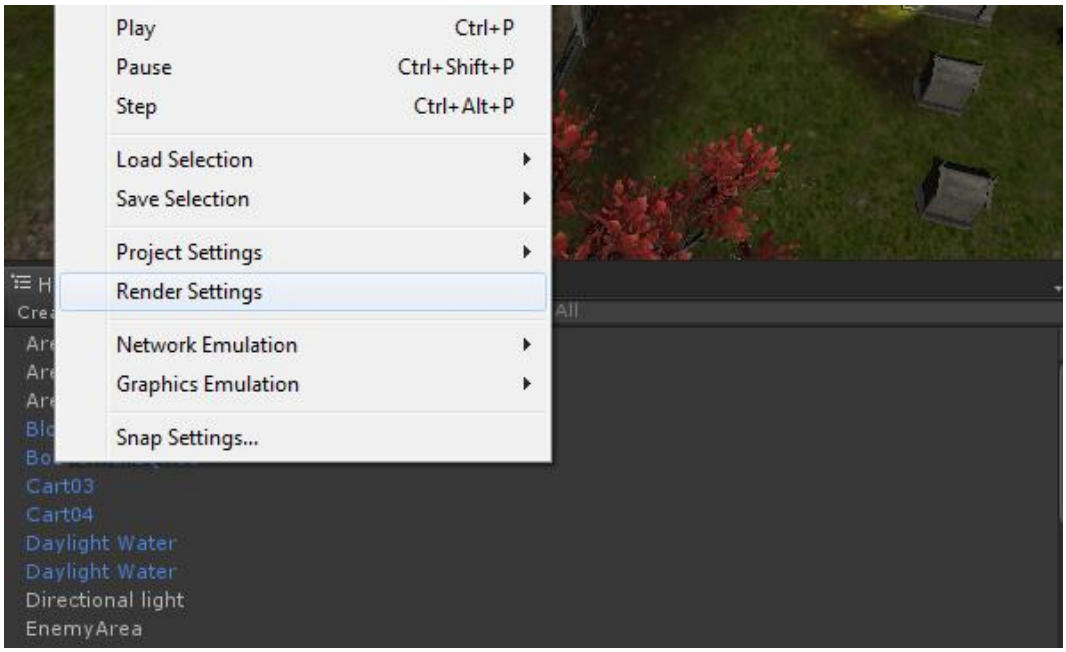
Di atas merupakan contoh program yang di buat oleh penulis untuk membuat animasi pada karakter, agar karakter dapat melakukan gerakan *Attack* atau penyerangan. Fungsi di atas menjelaskan bagaimana membuat animasi sederhana untuk karakter, fungsi ini akan berjalan apa bila di karakter sudah terdapat animasi yang di inginkan. Keterangan pada fungsi di atas adalah sebagai berikut :

Tabel 4.24 Keterangan Syntak

Syntaks	Keterangan
If(Input.GetAxis(“Attack”) > 0.2	Memanggil Input yang bernama Attack pada Input manager di game engine Unity3D
animation.CrossFade(“attack”);	Berfungsi untuk memanggil animasi yang bernama “attack” yang ada pada karakter

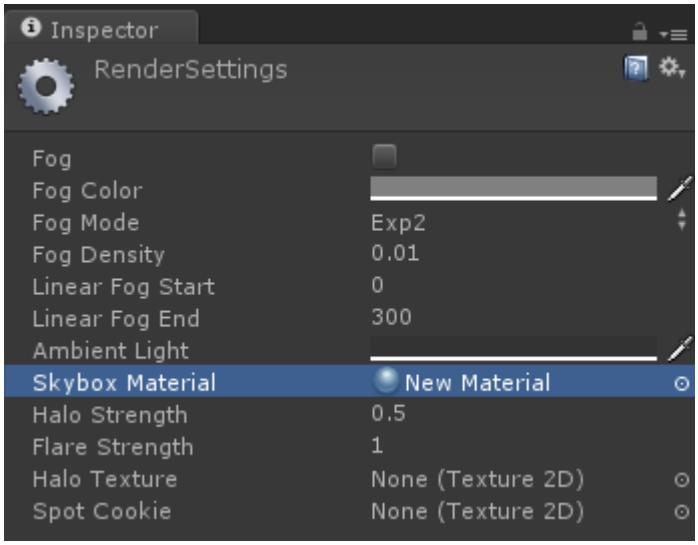
4.8 Perancangan Background

Pada perancangan background yang dibutuhkan adalah material yang bertemakan tentang langit. Pada *game engine* Unity3D terdapat menu *rendering settings* yang dapat kita pilih untuk memasukan background langit. Pada *game engine Unity3D* terdapat beberapa model background langit yang bisa kita pilih sesuai dengan keinginan kita. Untuk lebih jelasnya dapat di lihat sebagai berikut :



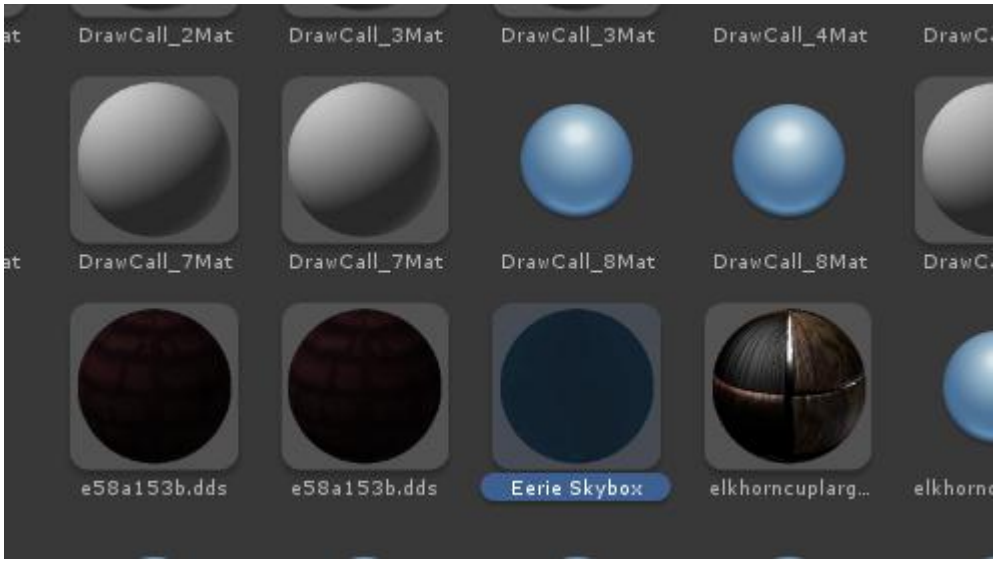
Gambar 4.17 Render Settings

Setelah itu akan melihat pengaturan di sebelah kanan layar pada *game engine Unity3D* dengan nama *Inspector* di situ kita dapat memilih background langit yang kita inginkan. Contohnya adalah sebagai berikut :



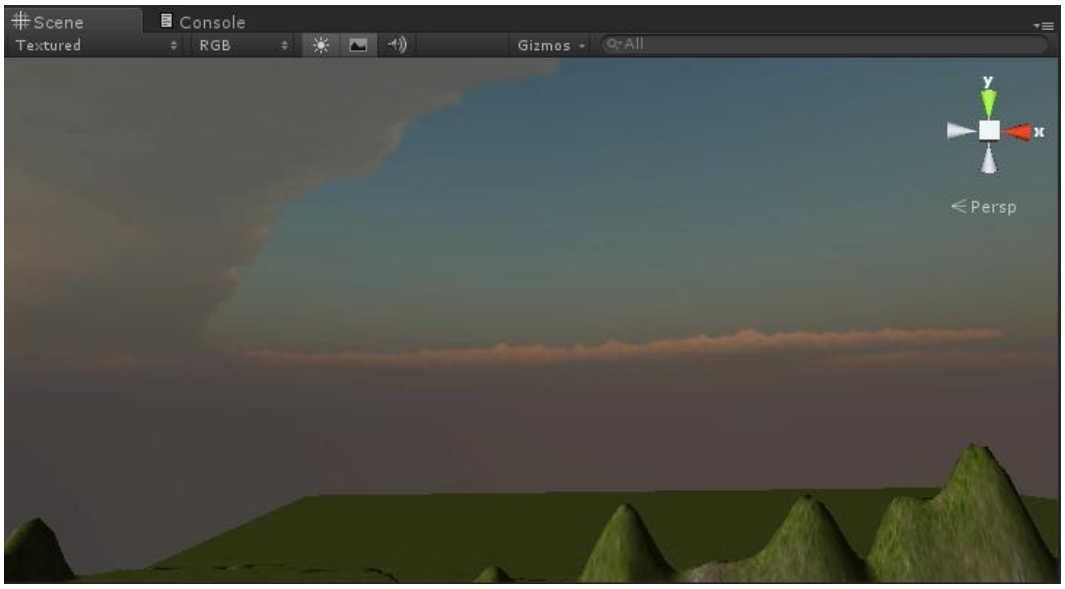
Gambar 4.18 Setting Background

Setelah pengaturan muncul pilihlah menu *Skybox Material* untuk mengimplementasikan material yang akan digunakan sebagai background. Setelah di klik maka akan muncul tampilan sebagai berikut :



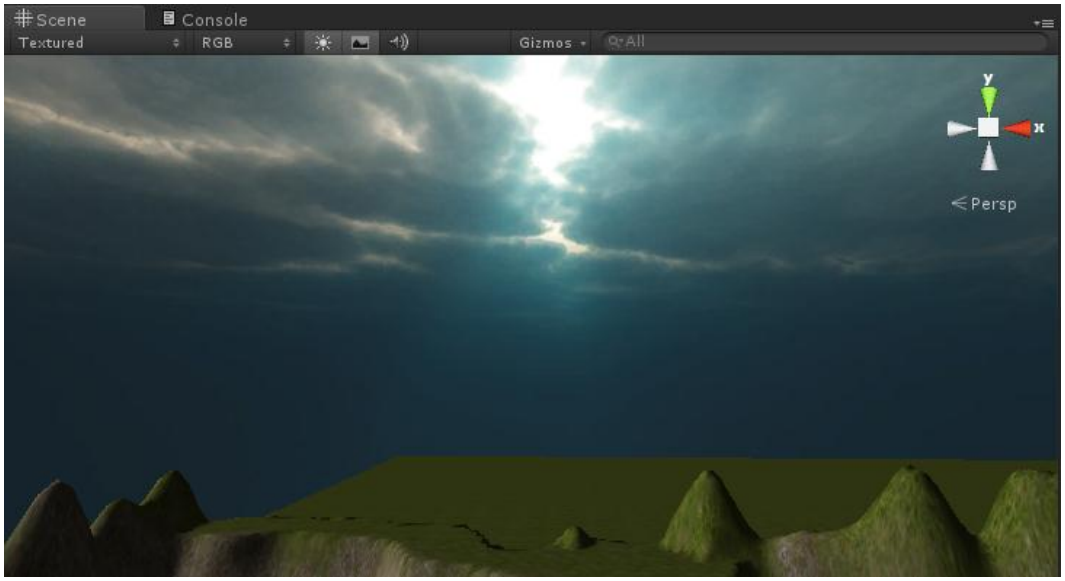
Gambar 4.19 Pilih Material.

Setelah material yang kita pilih secara otomatis background langit akan berubah sesuai dengan material yang di pilih. Gambar dapat di lihat sebagai berikut :



Gambar 4.20 Background langit sebelum di ubah.

Gambar 4.20 merupakan background langit menggunakan material *DawnDusk Skybox.* Dan untuk gambar 4.21 merupakan material yang baru menggunakan *Earie Skybox*

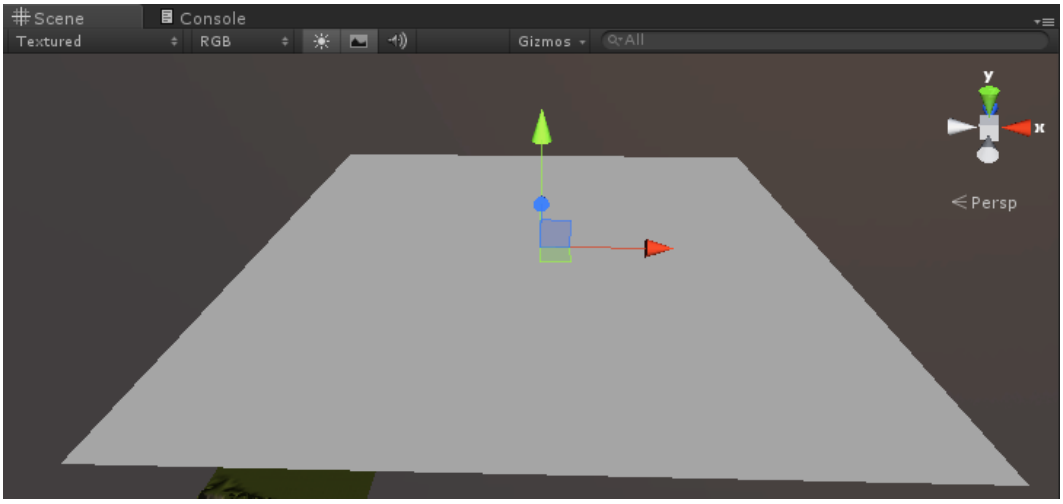


Gambar 4.21 Background langit setelah di ubah

4.9 Perancangan Environment

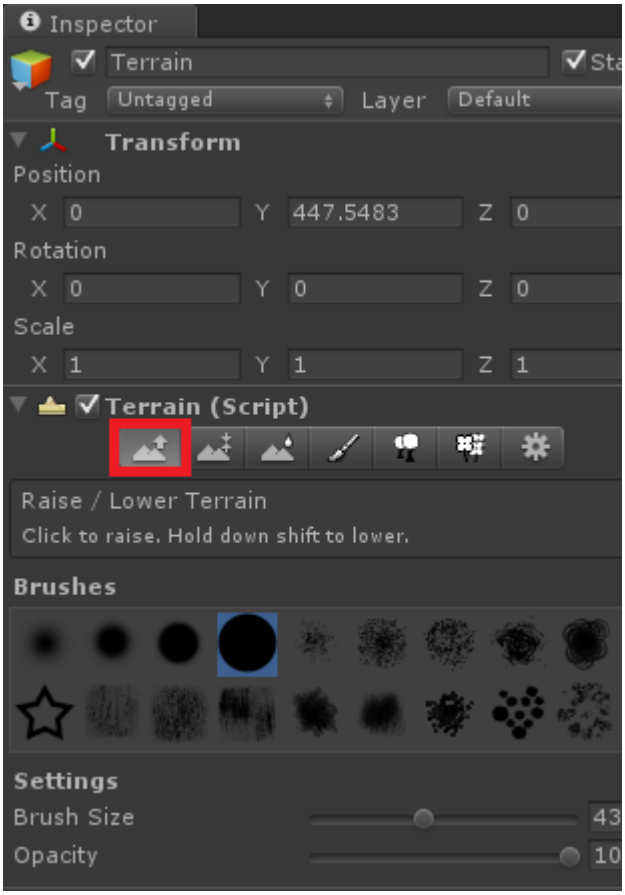
Environment atau lingkungan merupakan aspek yang penting dalam membuat *game* karena dengan adanya environment *game* yang di buat akan semakin hidup. Saya akan menjelaskan bagaimana membuat *environment* dengan menggunakan menu yang ada pada *game engine Unity3D*, tahapan dalam membuat *environment* adalah sebagai berikut :

1. Memilih menu *terrain* pada menubar setelah itu pilih *Create Terrain* akan muncul tampilan *terrain* seperti di bawah ini:



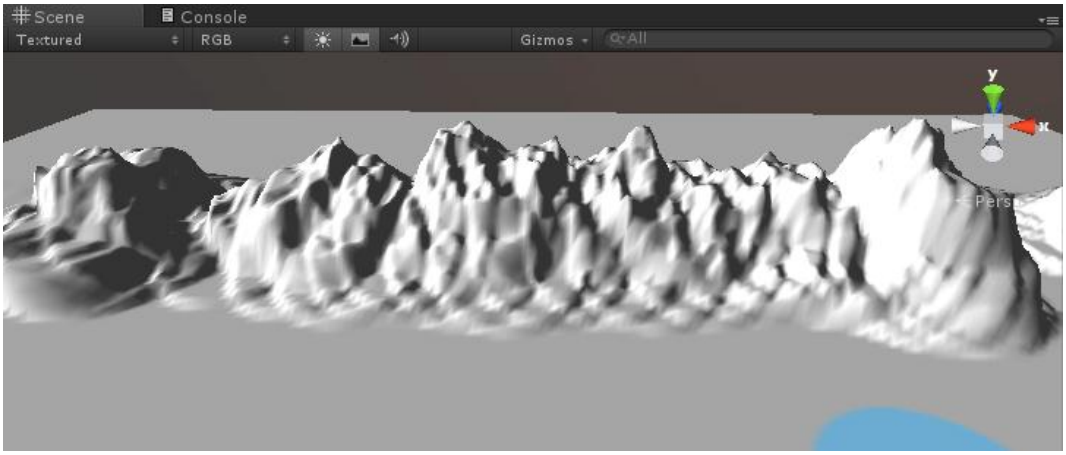
Gambar 4.22 Terrain baru

2. Membuat bukit menggunakan menu *terrain script* yang ada pada Unity3D, seperti gambar yang di beri kotak berwarna merah di bawah ini :



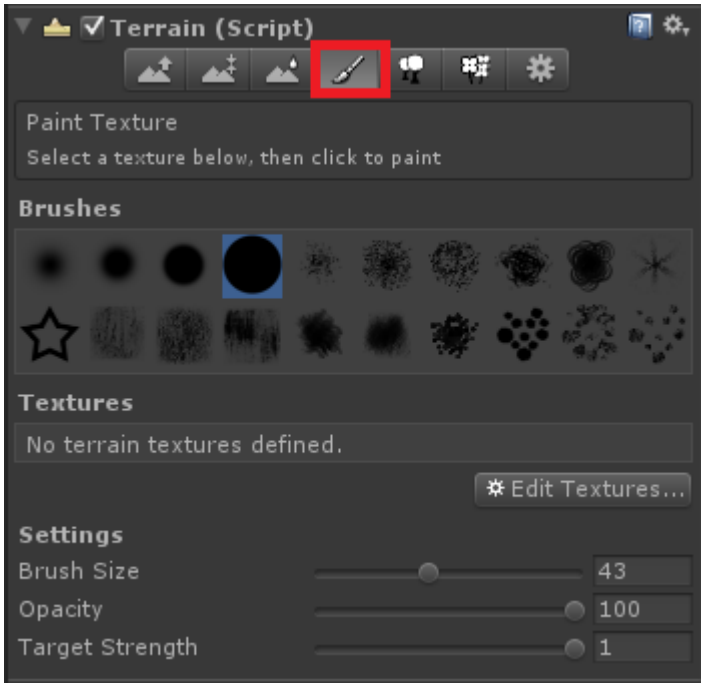
Gambar 4.23 Menu Terrain

Setelah pilihan di buat maka hasilnya akan terlihat lebih baik dengan bukit yang telah di hasilkan, seperti gambar berikut :



Gambar 4.24 Bukit yang telah di buat

Setelah bukit dibuat dapat di tambahkan *environment* yang lain seperti tanah, rumput dan pohon. Untuk membuat tanah pada terrain dapat memilih menu *paint texture* sepert gambar di bawah ini :



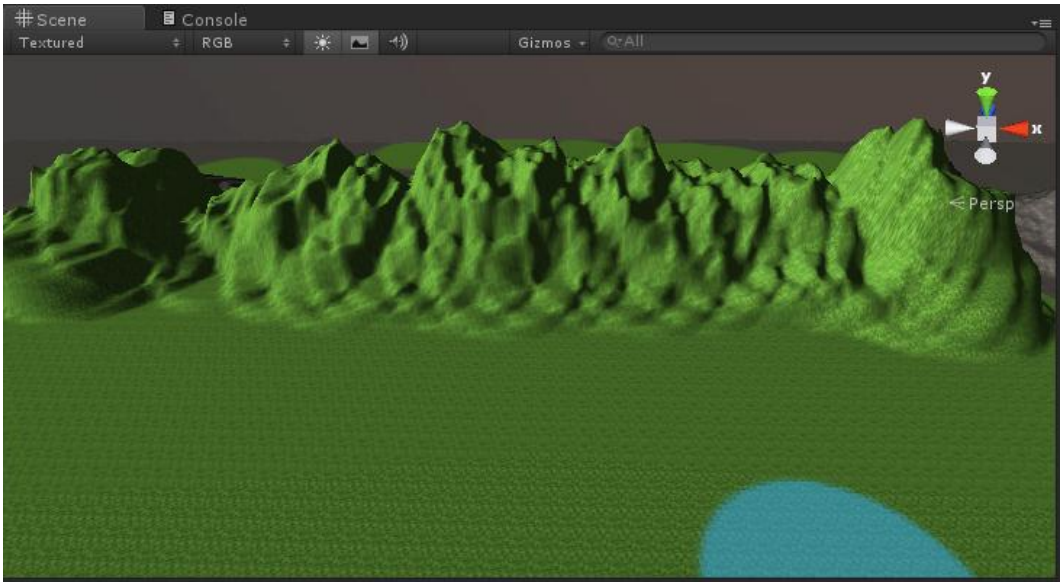
Gambar 4.25 Paint Texture

Setelah itu pilih *Edit Textures* lalu pilih *add textures*, akan muncul jendela *Add Terrain Texture* lalu pilih menu *select* untuk memilih jenis tanah yang di inginkan. Untuk lebih jelas dapat melihat gambar seperti di bawah ini :



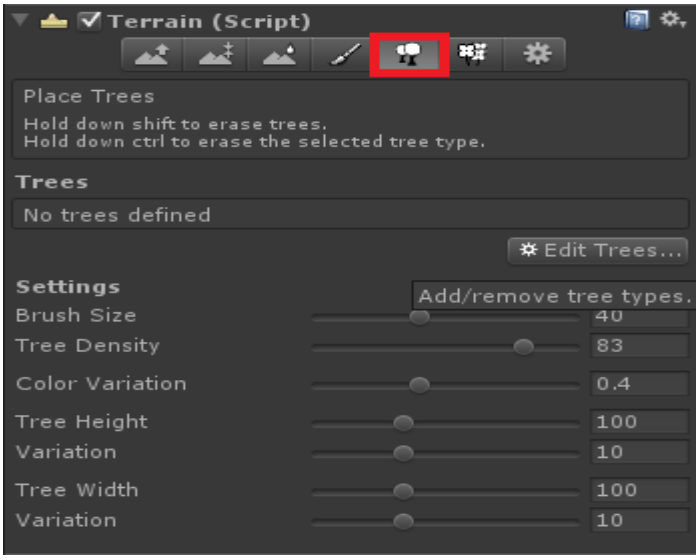
Gambar 4.26 Paint Terrain.

Setelah di pilih jenis tanah yang di inginkan, maka tekan tombol *Add* untuk mengimplementasikan jenis tanah pada *Terrain* yang telah di buat. Maka hasilnya akan seperti gambar di bawah ini :

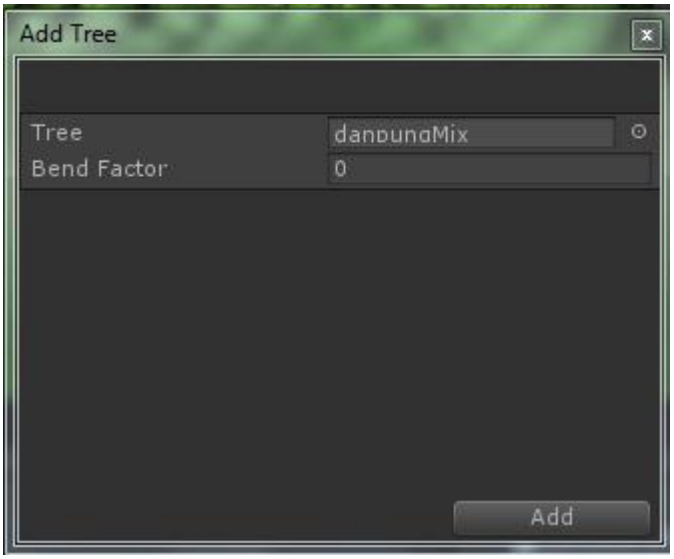


Gambar 4.27 Terrain dengan tanah.

Untuk menambahkan pohon pada terrain, penulis memilih menu *Place Trees* lalu pilih *Edit Trees* dan pilih menu *add tree*, setelah itu akan muncul jendela *Add Tree* untuk memilih jenis pohon yang sesuai dengan scenario. Untuk lebih jelas dapat melihat gambar di bawah ini :

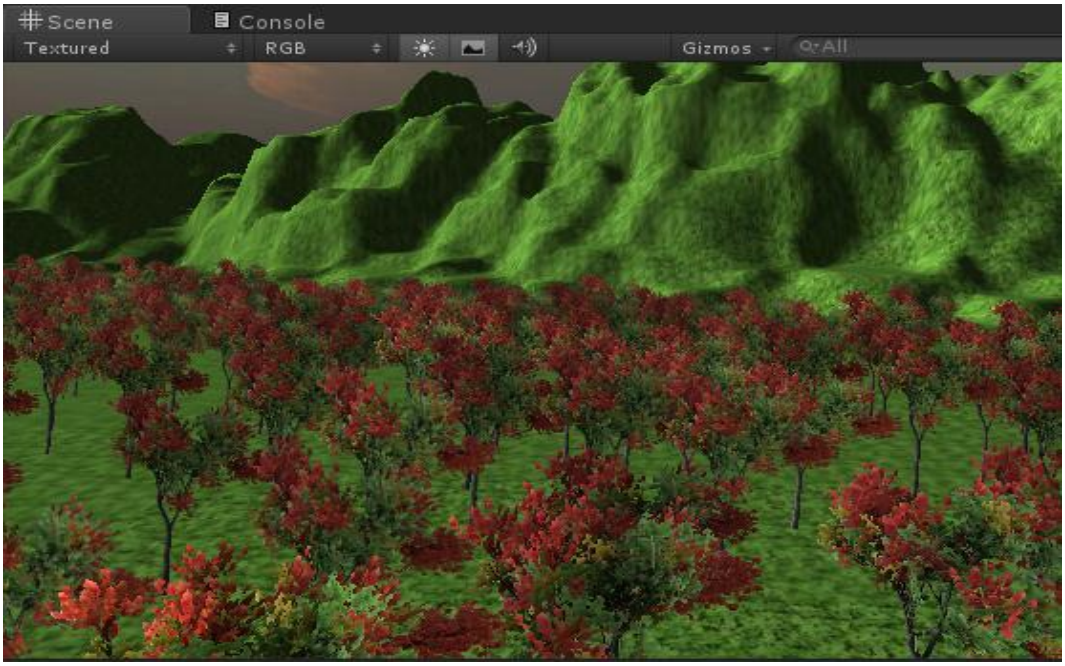


Gambar 4.28 Memasukan pohon



Gambar 4.29 Memilih pohon

Setelah memilih pohon maka pilih tombol *Add* untuk mengimplementasikan pohon pada terrain. Dan hasilnya adalah sebagai berikut :



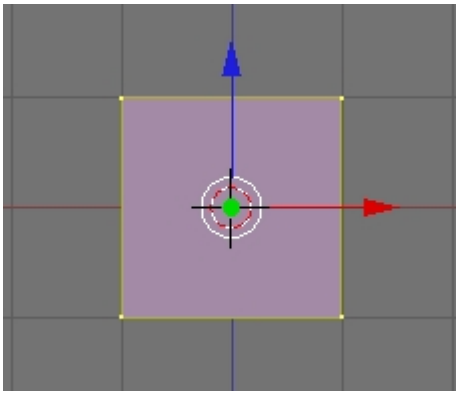
Gambar 4.30 Pohon yang telah di pilih

4.10 Perancangan Karakter

Karakter pada *game* merupakan hal yang sangat penting, tanpa karakter *game* tidak dapat di mainkan. Pada *game* yang penulis buat terdapat karakter yang di mainkan sebagai karakter utama. Untuk membuat karakter di perlukan *software* lain selain *Unity3D* yaitu *Blender*, di bawah ini adalah cara untuk membuat karakter dasar tetapi bukan karakter yang di gunakan oleh penulis, berikut adalah cara membuat karakter dasar sederhana :

1. Membuat Model Dasar

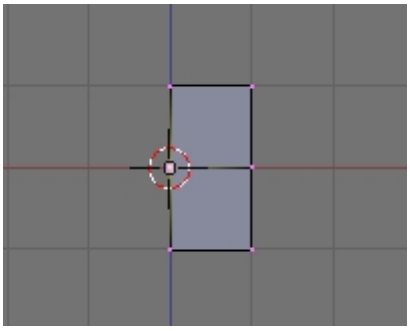
Untuk membuat suatu model, pertama kita buat model dasar atau disebut dengan Mesh. Model dasar yang kita gunakan adalah Plane, untuk menambahkannya tekan tombol SPACE >> Add >> Mesh >> Plane. Ubah tampilan objek ke depan dengan menekan tombol 1.



Gambar 4.31 Model Dasar

Ketika plane sudah ditambahkan, Blender akan berubah ke Edit Mode. Hapus setengah bagian dari model, yaitu bagian sebelah kiri, caranya: Tekan tombol A Tekan tombol B untuk merubah kursor ke mode Border

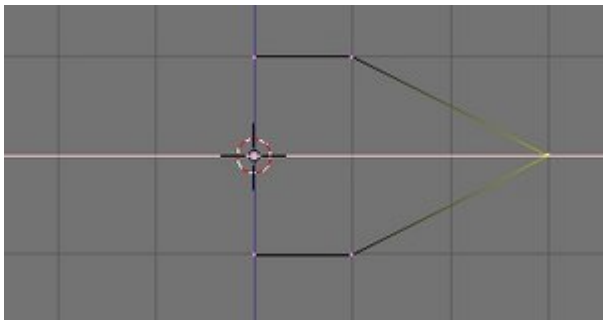
Select. Buat sebuah segi empat sampai setengah plane, lalu klik kiri dan tekan X untuk menghapusnya. Dan lakukan mirroring pada objek tersebut dengan menekan Tab lalu tekan F9.



Gambar 4.32 Model Dasar setengah plane

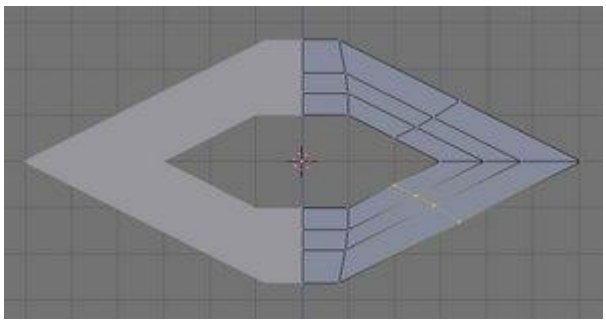
2. Membuat Mulut

Selanjutnya kita akan membuat mulut dari model dasar ini, caranya: Klik kanan pada titik pusat. Tekan tombol G, lalu tekan tombol X Tekan tombol 2 untuk memanjangkan ke kanan Klik kiri atau tekan Enter untuk konfirmasi.



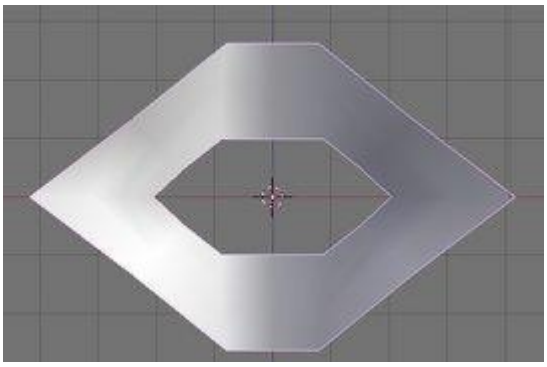
Gambar 4.33 Membuat mulut tahap I

Pilih semua vertex dengan menekan tombol A. Masuk ke mode Extrude dengan menekan tombol E >> Only Edges Lakukan pembesaran (skala) dengan menekan tombol S dan drag mouse sampai ukuran yang diinginkan, lalu klik kiri jika sudah selesai. Ulangi ekstrusi dan penskalaan sebanyak 2x lagi (sehingga total 3x) sehingga objek menjadi seperti ini:



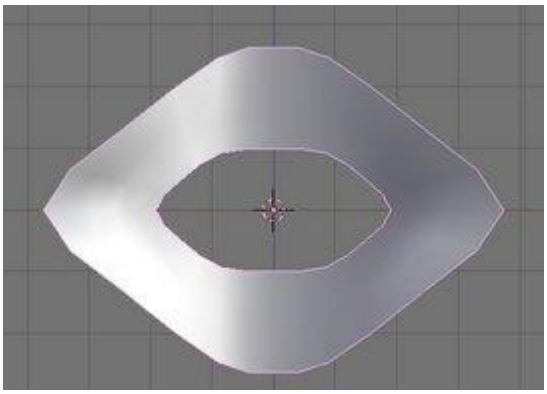
Gambar 4.34 Membuat mulut tahap II

Berikutnya bentuk mulut tersebut dengan cara: tekan tombol 1 tekan O (huruf) untuk mengaktifkan Proporsional Editing. Ubah tampilan ke bagian atas dengan tombol 7. Pindahkan vertex dengan menekan tombol G dan mouse wheel. Berikutnya tekan Tab untuk merubah ke Object Mode Tekan F9, lalu tekan CTRL+N hingga hasilnya seperti ini:



Gambar 4.35 Akhir membuat mulut

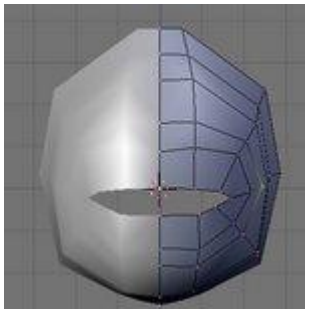
Selanjutnya tambahkan subsurf modifier untuk membuat bagian pojok mulut menjadi sedikit tumpul dan hasilnya akan seperti di bawah ini :



Gambar 4.36 Hasil Smooth

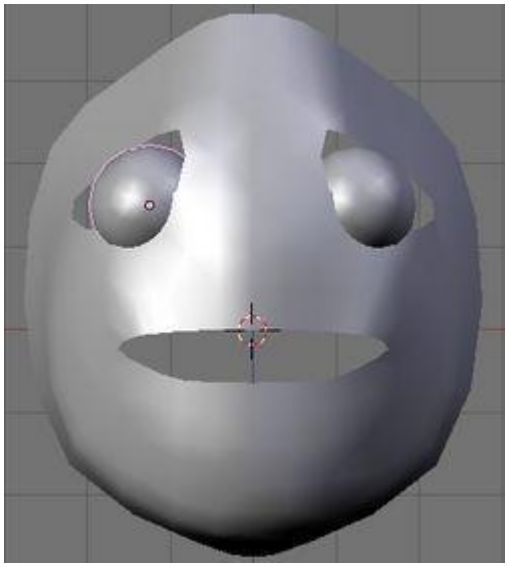
3. Membentuk Wajah Dan Mata

Tekan Alt + klik kanan untuk memilih semua vertex Tekan angka 3 untuk merubah tampilan Tekan O, lalu tekan E untuk Extrude. Pindahkan vertex kembali, lalu klik kiri untuk konfirmasi. Lakukan pembesaran dengan menekan tombol S dan mouse wheel. Berikan sedikit bentuk dengan mengaktifkan kembali menu Editing dengan menekan tombol O lalu bentuk hingga menyerupai bentuk wajah.



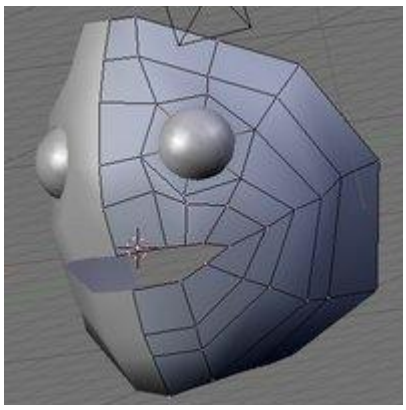
Gambar 4.37 Wajah awal

Setelah itu untuk membuat lubang mata tekan CTRL+R dan tombol mouse tengah. Pilih 2 vertex yang terlihat di tampilan Deleting Edge, lalu tekan X untuk menghapus. Menambahkan mata Untuk menambahkan mata kita buat objek baru yang terpisah, caranya: tekan Tab untuk ke mode Objek. Tambahkan model dasar UV Sphere dengan menekan tombol Space >> Add >> Mesh >> UV Sphere. Pada kotak notifikasi, rubah Segments and Rings ke 8. Tekan G untuk memindahkan mata ini ke posisi mata pada wajah yang kita buat sebelumnya. Lakukan penskalaan untuk mendapat ukuran yang sesuai. Lakukan Subsurf untuk menghaluskan objek mata yang kita buat. Berikutnya buat satu buah mata lagi dengan cara menduplikasikannya, tekan Shift + S >> Cursor to selection lalu pilih mata yang sebelumnya sudah dibuat. Tekan Shift + D untuk menduplikasikan mata tersebut Posisikan mata tersebut pada lubang mata satunya sehingga wajah yang sudah kita buat sekarang sudah memiliki 2 mata.



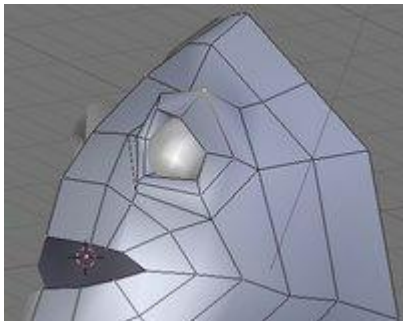
Gambar 4.38 Lubang mata

Setelah ditambahkan mata, kita lihat ada sedikit lubang di sekitar mata tersebut. Untuk menutupnya lakukan cara berikut: Ubah ke mode edit wajah, tekan TAB Pindahkan vertex mengelilingi mata Pilih soket mata dengan Alt + klik kanan, lalu skalakan dengan menekan S. Tekan E >> Edges Only untuk melakukan extrude.



Gambar 4.39 Menutup lubang mata

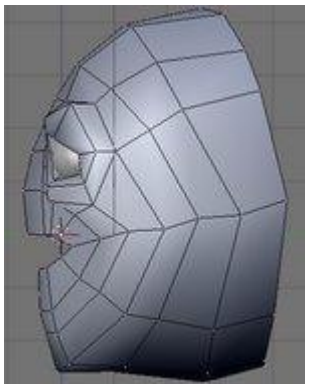
Terakhir untuk membuat kelopak mata, lakukan langkah berikut: Loop cut lingkaran di sekitar mata, tekan CTRL+R untuk membuat vertex baru. Tekan vertex baru tersebut ke depan dan bawah dengan menekan G Buat loop cut lain dengan Ctrl + R diantara mata lalu pindahkan ke sekitar mata untuk membuat kelopak matanya.



Gambar 4.40 Kelopak Mata

4. Memodelkan Seluruh Bentuk Kepala

Tekan Alt + mouse kanan pada bagian atas kepala Tekan E untuk extrude Geser vertex sehingga bentuknya memanjang. Lakukan hal yang sama untuk bagian tengah dan bawah sehingga hasilnya menyerupai bola kepala. Dan lakukan extrude lagi sehingga bagian belakang kepala agak melengkung ke dalam.



Gambar 4.41 Bentuk kepala tahap I

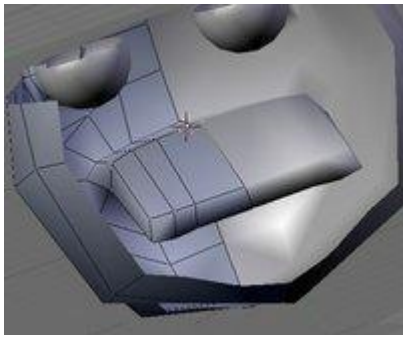
Tutup lubang di belakang kepala dengan cara memilih 4 vertex belakang kepala secara bersamaan, dan buat face-out dengan menekan tombol F.



Gambar 4.42 Bentuk kepala tahap II

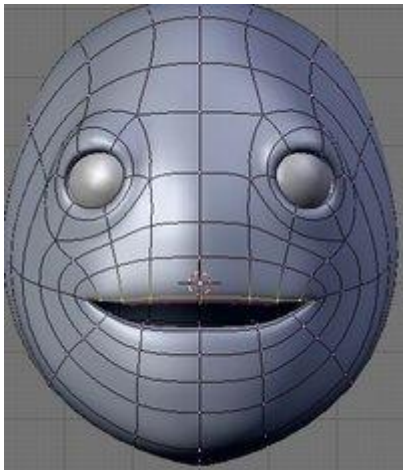
5. Membuat Bagian Dalam Mulut

Pindah ke tampilan samping dengan menekan tombol 3. Tekan Alt+B dan geser kotak disekitar bagian bawah kepala. Klik kiri untuk konfirmasi Putar tampilan sehingga kita bisa melihat bagian dalam kepala. Tekan Alt + mouse kanan untuk melakukan Loop-Select. Pilih 4 vertex di mulut lalu tekan F untuk membuat face-out Buat loop-cut disekeliling bagian dalam mulut dengan menekan Ctrl+R. Bentuk bagian dalam mulut tersebut hasilnya menjadi seperti ini:



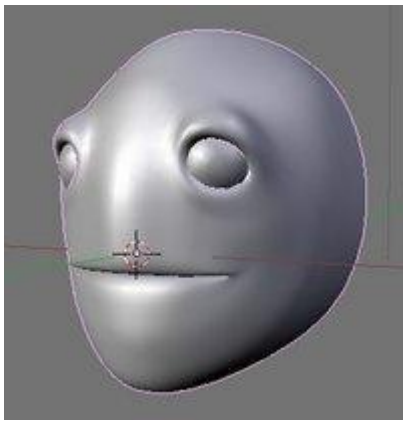
Gambar 4.43 Bentuk Mulut

Untuk menutup lubang mulut, lakukan langkah berikut: Buat 2 buah loop-cut di bagian depan wajah dan di bagian dalam mulut. Pilih menu subsurf, lalu pilih vertexnya dan mulai tutup mulutnya. Gambar-gambar di bawah ini menunjukkan proses penutupan mulut dengan status Subsurf menyala.



Gambar 4.44 Mulut dengan Subsurf on

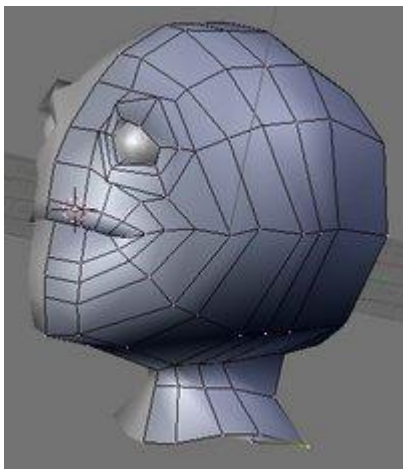
Jika sudah, matikan mode Subsurf, dan inilah hasil akhir wajah yang dibuat:



Gambar 4.45 Mulut dengan Subsurf off

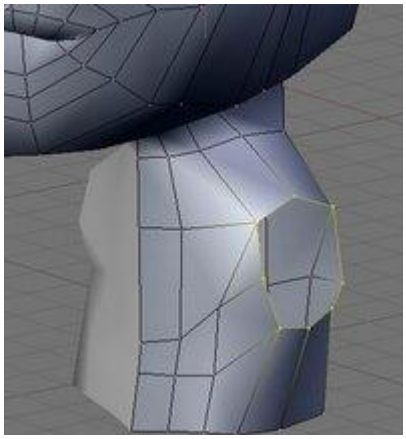
6. Membuat Leher Dan Dada

Pilih vertex di bawah kepala, lalu lakukan extrude dengan Alt + Mouse kanan. Bentuk leher dan lakukan Loop Cut (CTRL+R). Lakukan sampai sesuai dengan bentuk leher.



Gambar 4.46 Bentuk leher

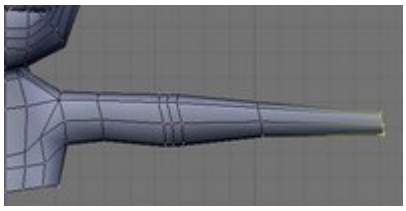
Lakukan extrude bagian bawah leher untuk membentuk dada dan bahu Untuk membentuk badan dilakukan banyak extrude/pelebaran dari vertex-vertex sehingga membentuk bagian yang kita inginkan. Disini semua tergantung keinginan sang animator untuk membuat modelnya seperti apa. Cara untuk melakukan extrude sudah dijelaskan sebelumnya. Dan hasilnya akan seperti ini:



Gambar 4.47 Bentuk dada

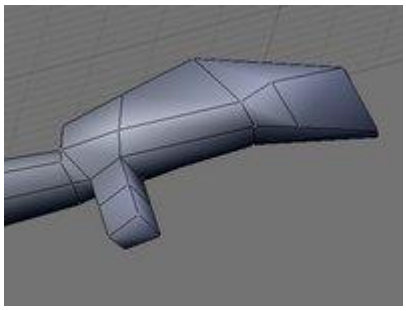
7. Membuat Tangan

Pilih vertex lingkaran pangkal lengan, lalu lakukan extrude ke kanan lakukan penskalaan pada beberapa bagian sehingga menyerupai lengan.



Gambar 4.48 Bentuk lengan

Untuk membuat tangan, juga lakukan extrude hingga mencapai hasil berikut

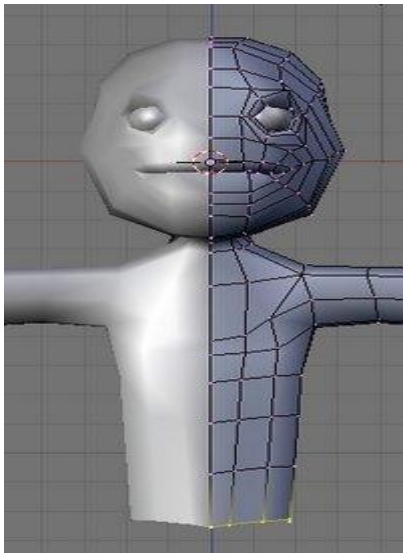


Gambar 4.49 Bentuk tangan

tutup lubang di ujung kanan dengan menekan tombol F agar lebih terlihat sebagai tangan, kita lakukan pembentukan dan lakukan dalam mode Subsurf menyala.

8. Membuat Perut

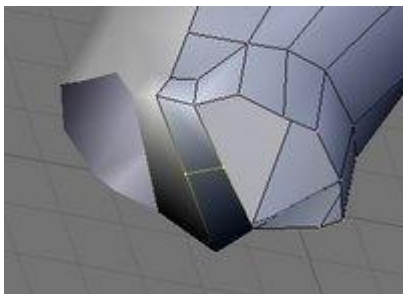
Untuk membuat perut cukup mudah, caranya hanya dengan melakukan extrude/pelebaran vertex-vertex di bawah dada sehingga membentuk perut. Selain itu juga lakukan penskalaan pada beberapa titik.



Gambar 4.50 Bentuk Perut

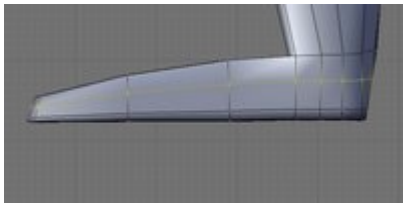
9. Membuat Kaki

Pertama tutup terlebih dahulu sedikit bagian tengah dari pinggang dengan tombol CTRL+R.



Gambar 4.51 Proses bentuk Kaki

Untuk membuat kaki, extrude lingkaran (lubang kaki) ke bawah hingga membentuk kaki bentuklah kaki hingga menyerupai kaki dengan beberapa titik membesar (lakukan skala).



Gambar 4.52 Bentuk kaki

Untuk membuat kakinya lebih berbentuk, kita bisa lakukan Face Loop dengan menekan tombol Ctrl+Alt+mouse kanan. Lakukan juga penskalaan agar bentuk kakinya semakin bagus.