TUGAS AKHIR – K141502

PERMAINAN REALITAS VIRTUAL MAGUS MENGGUNAKAN TEKNOLOGI OCULUS RIFT DAN LEAP MOTION DENGAN KENDALI HAND GESTURE

SULTAN BONAR MARTINUS
NRP 5114100006

Dosen Pembimbing
Dr. Eng. Darlis Herumurti, S.Kom., M.Kom.
Imam Kuswardayan, S.Kom., M.T.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

TUGAS AKHIR – K141502

PERMAINAN REALITAS VIRTUAL MAGUS MENGGUNAKAN TEKNOLOGI OCULUS RIFT DAN LEAP MOTION DENGAN KENDALI HAND GESTURE

SULTAN BONAR MARTINUS
NRP 5114100006

Dosen Pembimbing
Dr. Eng. Darlis Herumurti, S.Kom., M.Kom.
Imam Kuswardayan, S.Kom., M.T.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

FINAL PROJECT – K141502

VIRTUAL REALITY GAME MAGUS USING OCULUS RIFT AND LEAP MOTION WITH HAND GESTURE CONTROL

SULTAN BONAR MARTINUS
NRP 5114100006

Advisor
Dr. Eng. Darlis Herumurti, S.Kom., M.Kom.
Imam Kuswardayan, S.Kom., M.T.

INFORMATICS DEPARTMENT
Faculty of Information and Communication Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

PERMAINAN REALITAS VIRTUAL MAGUS MENGGUNAKAN TEKNOLOGI OCULUS RIFT DAN LEAP MOTION DENGAN KENDALI HAND GESTURE

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Interaksi, Grafika dan Seni
Program Studi S-1 Departemen Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh:
SULTAN BONAR MARTINUS
NRP: 5114100006

Disetujui oleh Dosen Pembimbing Tugas Akhir:

Dr. Eng. Darlis Herumurti, S.Kom., M.Kom. NIP: 19771217 200312 1 001	..... (pembimbing 1)
Imam Kuswardayan, S.Kom., M.T. NIP: 19761215 200312 1 001	..... (pembimbing 2)

SURABAYA
MEI 2018

[Halaman ini sengaja dikosongkan]

PERMAINAN REALITAS VIRTUAL MAGUS MENGGUNAKAN TEKNOLOGI OCULUS RIFT DAN LEAP MOTION DENGAN KENDALI HAND GESTURE

Nama Mahasiswa	: Sultan Bonar Martinus
NRP	: 5114100006
Departemen	: Informatika FTIK-ITS
Dosen Pembimbing 1	: Dr. Eng. Darlis Herumurti, S.Kom., M.Kom.
Dosen Pembimbing 2	: Imam Kuswardayan, S.Kom., M.T.

ABSTRAK

Perkembangan teknologi dengan menggunakan realitas virtual sudah semakin maju. Tidak hanya untuk membantu manusia dalam mensimulasikan aktifitas di dunia nyata, juga menjadi sarana untuk bermain permainan yang terasa nyata. Untuk merasakan teknologi realitas virtual ada banyak macam alat yang dapat digunakan, salah satunya adalah Oculus Rift. Alat ini berfungsi untuk menampilkan dunia maya kepada pengguna.

Ide yang digunakan dalam tugas akhir ini adalah membangun sebuah permainan First Person Shooter (FPS) yang memiliki control Leap Motion dan Voice Recognition. Permainan ini menjadikan tangan dan suara sebagai sumber masukan dalam melakukan interaksi. Permainan ini dibangun menggunakan Unity dan Leap Motion SDK. Tujuan dari permainan ini adalah untuk meningkatkan adrenalin pemain untuk mengalahkan musuh *yang ada sebelum* musuh *mengalahkannya lebih dulu.*

Hasil dari pengujian ini akan berupa sebuah permainan yang dapat berjalan di perangkat personal computer high-end. Permainan ini dibangun dengan Unity Versi 2017.3.1.1f1 dengan

bahasa pemrograman C#, Library Windows.Speech, dan Leap Motion SDK. Proses pembuatan asset permainan sebagian besar mengambil dari asset store dan sebagian dibuat menggunakan tools Blender, Corel Draw X7, dan Adobe After Effects 2017cc. Dengan pengujian beta dan juga kuisioner dapat disimpulkan permainan telah mengimplementasikan perancangan dengan baik.

Kata kunci: Oculus Rift, Realitas Virtual, Voice Recognition,Leap Motion, Hand Gesture

VIRTUAL REALITY GAME MAGUS USING OCULUS RIFT AND LEAP MOTION TECHNOLOGY WITH HAND GESTURE CONTROL

Name : Sultan Bonar Martinus
NRP : 5114100006
Department : Informatics FTIK-ITS
Supervisor I : Dr. Eng. Darlis Herumurti, S.Kom., M.Kom.
Supervisor II : Imam Kuswardayan, S.Kom., M.T.

ABSTRACT

The development of technology by using virtual reality is increasingly advanced. Not only to help humans in simulating activities in the real world, it also becomes a means to play a Permainan that feels real. To experience virtual reality technology there are many different tools that can be used, one of them is Oculus Rift. This tool serves to show virtual world to users.

The idea used in this final project is to build a First Person Shooter (FPS) Permainan that has Leap Motion and Voice Recognition control. This Permainan makes hands and sounds as a source of input in the interaction. The Permainan is built using Unity and Leap Motion SDK. The purpose of the Permainan is to increase the adrenaline of players to defeat the enemies before the enemy defeats them first.

The result of this project test will be a Permainan that can run on high-end personal computer devices. The Permainan is built with Unity Version 2017.3.1.1f1 with c# programming language, Windows.Speech Library, and Leap Motion SDK. The process of making asset Permainans mostly take from the asset store and partly made using tools Blender, Corel Draw X7, and Adobe After Effects 2017cc. With beta testing and also the questionnaire can be concluded the Permainan has implemented the design properly.

Keywords: Oculus Rift, Realitas Virtual, Voice Recognition,Leap Motion, Hand Gesture

KATA PENGANTAR

Puji syukur penulis panjatkan ke hadirat Tuhan Yang Maha Esa karena atas karunia dan rahmat-Nya penulis dapat menyelesaikan tugas akhir yang berjudul:

PERMAINAN REALITAS VIRTUAL MAGUS MENGGUNAKAN TEKNOLOGI OCULUS RIFT DAN LEAP MOTION DENGAN KENDALI HAND GESTURE

Melalui lembar ini, penulis ingin menyampaikan ucapan terima kasih dan penghormatan yang sebesar-besarnya kepada:

1. Tuhan Yang Maha Esa, karena limpahan rahmat dan karunia-Nya penulis dapat menyelesaikan Tugas Akhir dan juga perkuliahan di Departemen Informatika ITS.
2. Ayah dan Ibu penulis, Sahatan Tamba dan Hotmian Nainggolan yang tiada hentinya memberikan dukungan doa, moral, dan material kepada penulis sehingga penulis dapat menyelesaikan Tugas Akhir ini.
3. Bapak Dr. Eng. Darlis Herumurti S.Kom., M.Kom. dan Bapak Imam Kuswardayan S.Kom., M.T. selaku pembimbing I dan II yang telah membimbing dan memberikan motivasi, nasihat dan bimbingan dalam menyelesaikan Tugas Akhir ini.
4. Teman teman penulis yang selalu mendorong dan membantu penulis untuk menyelesaikan Tugas Akhir ini.
5. Kakak kakak kelas penulis yang telah memberikan ilmunya untuk penulis.
6. Adik adik kelas penulis yang telah menghibur penulis.
7. Serta semua pihak yang telah turut membantu penulis dalam menyelesaikan Tugas Akhir ini.

Bagaimanapun juga penulis telah berusaha sebaik-baiknya dalam menyelesaikan tugas akhir ini. Namun, penulis mohon maaf apabila terdapat kekurangan ataupun kesalahkan yang penulis lakukan. Kritik dan saran yang membangun dapat disampaikan sebagai bahan perbaikan untuk ke depannya.

Surabaya, Mei 2018

Sultan Bonar Martinus

DAFTAR ISI

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

[Halaman ini sengaja dikosongkan]

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

DAFTAR KODE SUMBER

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

1.1. Latar Belakang

Permainan digital merupakan salah satu media hiburan yang sedang berkembang pesat, dan bermain permainan digital merupakan salah satu cara untuk mengisi waktu luang. Hingga saat ini ada banyak jenis permainan digital yang sudah ada, dan juga teknologi yang digunakan dalam pembuatan permainan telah berkembang.

Perkembangan teknologi dari waktu ke waktu mengalami kemajuan yang sangat pesat. Seiring dengan perkembangan itu pula, teknologi yang dipakai untuk membuat permainan terus berkembang dan semakin banyak. Teknologi permainan yang sedang berkembang dan menarik perhatian dari para *gamers* adalah permainan berbasis realitas virtual.

Realitas virtual merupakan teknologi yang dapat membuat pengguna berinteraksi dengan suatu lingkungan yang disimulasikan oleh komputer, suatu lingkungan sebenarnya yang ditiru atau benar-benar suatu lingkungan yang hanya ada dalam imajinasi.

Dalam hal ini penulis ingin membuat sebuah permainan yang berjudul Magus. Magus merupakan sebuah permainan realitas virtual dengan genre *survival* [1], dengan kendali *voice recognition* dan deteksi *hand gesture* untuk menggunakan kemampuan yang dimiliki pemain di dalam permainan.

1.2. Rumusan Permasalahan

Rumusan masalah yang diangkat dalam tugas akhir ini adalah sebagai berikut:

1. Bagaimana merancang *Gameplay* Magus dengan fitur 3D *First Person Shooter* (FPS) menggunakan konsep Realitas Virtual?
2. Bagaimana merancang mekanik dari pemain dengan menggunakan Leap Motion Controller untuk mendapatkan *hand gesture* pemain ?
3. Bagaimana implementasi dari rancangan di atas diselesaikan dengan *Game Engine Unity* ?

1.3. Batasan Permasalahan

Batasan masalah pada tugas akhir ini antara lain:

1. Permainan yang dibuat merupakan permainan yang bekerja di *Personal Computer* (PC) yang mumpuni.
2. Permainan yang dibuat merupakan aplikasi realitas virtual yang membutuhkan Oculus Rift untuk realitas virtual dan Leap Motion sebagai kendali *hand gesture*.
3. Masukan *hand gesture* berupa gerakan kedua tangan pemain, untuk mendeteksi arah telapak tangan pemain sebagai kendali dalam Permainan.
4. Lingkungan pengembangan yang digunakan menggunakan aplikasi Unity 3D lisensi gratis dan bahasa pemrograman C#.
5. Kondisi musuh terbatas hanya diam dan menyerang pemain.
6. Obyek musuh dibagi menjadi dua jenis, yaitu musuh yang kuat dan musuh yang lemah.
7. Terdapat 3 musuh kuat yang dianggap *boss* dan beberapa musuh lemah tersebar di area permainan yang dianggap *creep*.
8. Pemain memiliki *health bar* dan *mana bar*, yang apabila pemain terkena serangan maka *health bar* akan berkurang dan apabila mengaktifkan sihir maka *mana bar* akan berkurang.
9. Pemain memiliki berbagai macam sihir untuk digunakan.

10. Permainan akan berakhir apabila *health bar* pemain habis atau pemain berhasil mendapatkan 3 *item* yang didapat setelah mengalahkan seluruh *boss* yang ada.
11. Sepanjang permainan, pemain memiliki kesempatan untuk mendapatkan barang/*item* pembantu untuk menambahkan *health bar* dan/atau *mana bar.*

1.4. Tujuan

Tujuan dari pembuatan Tugas Akhir ini adalah untuk membuat permainan realitas virtual dengan tampilan *First Person Shooter* yang mengimplementasikan deteksi *hand gesture* sebagai pengendali gerakan pemain dan pengaktifan sihir atau *item* dalam *gameplay* permainan Magus menggunakan Unity.

1.5. Manfaat

Manfaat dari tugas akhir ini yaitu sebagai berikut:

1. Memberikan *gameplay* baru dengan menambahkan fitur deteksi *hand gesture* sebagai kendali pada permainan realitas virtual.
2. Memberikan pengalaman baru pada pemain permainan realitas virtual.
3. Mengasah kemampuan pengambilan keputusan dan strategi pemain untuk menyelesaikan/memenangkan permainan.
4. Sebagai sarana hiburan untuk para pemain.

1.6. Metodologi

Langkah-langkah yang ditempuh dalam pengerjaan tugas akhir ini adalah sebagai berikut:

1. **Studi literatur**

   Pada studi literatur ini, akan dipelajari sejumlah referensi yang diperlukan dalam pembuatan aplikasi yaitu mengenai *Game Engine* Unity3D, *Library* Windows.Speech, Bahasa Pemrograman C#, *Survival*

Game, *First Person Shooter* (FPS), Realitas Virtual, dan *Leap Motion Controller*.

2. Analisis dan desain sistem

Pada tahap ini dilakukan analisis dan pendefinisian kebutuhan sistem untuk masalah yang dihadapi, terutama analisis terkait bagaimana skenario latihan berbicara di depan umum yang akan diterapkan pada sistem. Selanjutnya, dilakukan perancangan sistem dengan beberapa tahap sebagai berikut:

a. Mempelajari konsep dan cara kerja serta batasan penggunaan Leap Motion Controller.
b. Mempelajari dokumentasi dan tutorial Oculus Rift.
c. Mempelajari dokumentasi dan tutorial Unity3D.
d. Perancangan *gameplay* realitas virtual Magus dengan menambahkan fitur deteksi *hand gesture* menggunakan Leap Motion Controller.

3. Implementasi sistem

Pada tahap ini akan dilakukan pembangunan sistem. Sistem yang dimaksud disini, yaitu permainan realitas virtual yang dibangun dengan menggunakan *tools* Unity 3D dan perangkat Oculus Rift dengan OS Windows 10.

4. Pengujian dan evaluasi

Tahap Pengujian dan evaluasi berisi pengujian aplikasi dan evaluasi berdasarkan hasil pengujian. Pada tahap ini akan dilakukan pengujian dari fungsionalitas perangkat lunak, apakah sesuai dengan yang diharapkan serta tidak diharapkan terdapat *bug*. Pengujian akan dilakukan kepada 5 mahasiswa Departemen Informatika dan 5 mahasiswa non Departemen Informatika, mereka akan menjadi penguji dan memainkan permainan Magus. Pengujian dilakukan untuk mengukur tingkat ketepatan pengenalan *gesture* dari posisi tangan pemain yang diterapkan.

5. **Penyusunan buku Tugas Akhir**

   Pada tahap ini dilakukan proses dokumentasi dan pembuatan laporan dari seluruh konsep, tinjauan pustaka, metode, implementasi, proses yang telah dilakukan, pengujian, evaluasi dan hasil-hasil yang telah didapatkan selama pengerjaan tugas akhir.

1.7. Sistematika Penulisan

Buku tugas akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan tugas akhir. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku tugas akhir terdiri atas beberapa bagian seperti berikut ini:

Bab I Pendahuluan

Bab ini berisi latar belakang masalah, rumusan masalah, tujuan dan manfaat pembuatan tugas akhir, batasan masalah, metodologi yang digunakan, dan sistematika penyusunan tugas akhir.

Bab II Tinjauan Pustaka

Bab ini menjelaskan beberapa pustaka-pustaka yang dijadikan penunjang dan berhubungan dengan pokok pembahasan yang mendasari pembuatan tugas akhir.

Bab III Desain dan Perancangan Sistem

Bab ini membahas mengenai desain dan perancangan sistem yang akan dibangun.

Bab IV Implementasi Sistem

Bab ini membahas mengenai bagaimana implementasi sistem dari desain yang sudah dirancang.

Bab V Pengujian dan Evaluasi

Bab ini membahas pengujian dari metode yang ditawarkan dalam tugas akhir untuk mengetahui kesesuaian metode dengan data yang ada.

Bab VI Kesimpulan dan Saran

Bab ini berisi kesimpulan dari hasil pengujian yang telah dilakukan. Bab ini juga membahas saran-saran untuk pengembangan sistem lebih lanjut.

Daftar Pustaka

Merupakan daftar referensi yang digunakan untuk mengembangkan tugas akhir.

Lampiran

Merupakan bab tambahan yang berisi data atau daftar istilah yang penting pada tugas akhir ini.

BAB II
TINJAUAN PUSTAKA

Bab ini membahas pustaka/teori-teori yang menjadi dasar dalam pembuatan tugas akhir.

2.1. Realitas Virtual

Realitas virtual adalah sebuah teknologi yang membuat pengguna dapat berinteraksi dengan lingkungan yang ada dalam dunia maya yang disimulasikan oleh komputer, sehingga pengguna merasa berada di dalam lingkungan tersebut. Teknologi realitas virtual sejatinya telah banyak diterapkan di beberapa sektor industri seperti kedokteran, penerbangan, pendidikan, arsitek, militer, hiburan, dan lain sebagainya. Realitas virtual sangat membantu dalam menyimulasikan sesuatu yang sulit untuk dihadirkan secara langsung dalam dunia nyata. Tentunya ini dapat membuat lebih praktis dan lebih ekonomis [2].

Kelebihan utama dari realitas virtual adalah pengalaman yang membuat pengguna merasakan sensasi dunia nyata dalam dunia virtual/maya. Bahkan perkembangan teknologi realitas virtual saat ini memungkinkan tidak hanya indra penglihatan dan pendengaran saja yang bisa merasakan sensasi nyata dari dunia maya dari realitas virtual, namun juga indra yang lainnya.

Untuk memunculkan sensasi nyata dari realitas virtual diperlukan perangkat pendukung seperti Oculus Rift, HTC Vive, Google Cardboard dan lainya. Terdapat 4 elemens penting dalam realitas virtual. Adapun 4 elemen itu adalah sebagai berikut:

a. *Virtual world*, sebuah konten yang menciptakan dunia virtual dalam bentuk *screenplay* maupun *script.*
b. *Immersion*, sebuah sensasi yang membawa pengguna teknologi realitas virtual merasakan ada di sebuah lingkungan nyata yang padahal fiktif. *Immersion* dibagi dalam 3 jenis, yakni:

- *Mental immersion*, membuat mental penggunanya merasa seperti berada di dalam lingkungan nyata.
- *Physical immersion*, membuat fisik penggunanya merasakan suasana di sekitar lingkungan yang diciptakan oleh realitas virtual tersebut.
- *Mentally immersed*, memberikan sensasi kepada penggunanya untuk larut dalam lingkungan yang dihasilkan realitas virtual.

c. *Sensory feedback*, berfungsi untuk menyampaikan informasi dari *virtual world* ke indera penggunanya. Elemen ini mencakup visual (penglihatan), audio (pendengaran) dan sentuhan.
d. *Interactivity*, bertugas untuk merespons aksi dari pengguna, sehingga pengguna dapat berinteraksi langsung dalam medan fiktif atau *virtual world*.

Sebuah teknologi dapat dikatakan sebagai realitas virtual jika sudah memenuhi beberapa persyaratan, seperti tampilan gambar/grafis/visualisasi 3D tampak nyata dan sesuai dengan perspektif dari penggunanya, dan mampu mendeteksi semua gerakan dan respons dari pengguna baik itu gerakan kepala atau bola mata pengguna.

2.2. Unity 3D

Unity 3D adalah sebuah aplikasi yang digunakan untuk mengembangkan *Permainan* berbasis *multi-platform*. Unity dapat digunakan untuk membuat sebuah *Permainan* yang bisa digunakan pada perangkat komputer, ponsel pintar android, iPhone, PS3, dan bahkan X-BOX. [3]

Unity merupakan sebuah *tool* yang terintegrasi, untuk membuat *Permainan*, arsitektur bangunan dan simulasi. Unity juga bisa digunakan untuk permainan PC dan permainan *online*. Untuk permainan *online* diperlukan sebuah *plugin*, yaitu Unity Web Player, sama halnya dengan Flash Player pada browser.

Unity tidak dirancang untuk proses desain atau *modelling*, dikarenakan Unity bukanlah sebuah *tools* untuk mendesain, Unity

hanyalah sebuah *Permainan engine* 2D atau 3D. Banyak hal yang bisa dilakukan dengan Unity dengan berbagai fitur yang dimilikinya, seperti adanya fitur *audio reverb zone, particle effect*, dan *sky box* untuk menambahkan langit. Fitur *scripting* yang disediakan mendukung 3 bahasa pemrograman, JavaScript, C#, dan Boo.

2.3. Bahasa Pemrograman C#

C# (dibaca: c sharp) merupakan sebuah bahasa pemrograman berorientasi objek yang dikembangkan oleh Microsoft sebagai bagian dari insiatif kerangkan .NET *framework*. Bahasa pemrograman ini dibuat berbasiskan bahasa C++ yang telah dipengaruhi oleh aspek-aspek ataupun fitur yang terdapat pada bahasa-bahasa pemrograman lainnya seperti Java, Delphi, Visual Basic dan lain-lain dengan beberapa penyederhanaan. [4]

2.4. First Person Shooter

Pengembangan *First Person Shooter* (FPS) dimulai pada tahun 1973, dan 1974, Spasim. Setelah itu, lebih banyak judul bermunculan seperti MIDI Maze tahun 1987, kemudian bersatu dengan genre kekerasan membentuk Wolfenstein 3D tahun 1992, akan tetapi lebih kalah populer dengan permainan Doom. Half Life (1998) dan Half Life 2 (2004) meningkatkan narasi dan elemen teka-teki. [5]

2.5. Hand Gesture Recognition

Hand Gesture Recognition (Pengenalan pose tangan) adalah sebuah topik dalam ilmu computer dan teknologi bahasa dengan tujuan menginterpretasikan pose manusia dengan algoritma matematika. Hand Gesture Recognition memberikan sebuah cara yang natural, inovatif dan modern dari komunikasi non verbal. Pengaturanya terdiri dari satu kamera untuk menangkap

gerakan yang dibentuk pengguna dan mengambil gambar tangan ini sebagai masukan pada algoritma yang digunakan. [6]

2.6. Leap Motion Controller

Leap Motion Controller Merupakan suatu perangkat yang di kembangkan oleh Leap motion, Inc pada tahun 2008 oleh David Holz. Leap Motion Controller sendiri digunakan sebagai *Input* dari komputer tanpa sentuh. Bisa dikatakan bahwa Leap Motion ini merupakan pengganti *Mouse* pada komputer karena memiliki fungsi yang sama. [7]

Leap Motion Merupakan Sebuah Perangkat Sensor hardware yang mendukung gerakan tangan dan jari sebagai *Input*, tetapi tidak memerlukan kontak tangan atau sentuhan langsung terhadap komputer.

Leap Motion dari sisi *hardware* sebenarnya cukup sederhana. Inti dari Leap Motion ini terletak pada pemanfaatan dua kamera *stereo* dan terdapat tiga lampu pemancar inframerah yang menyebar secara konvergen sehingga mampu untuk menjangkau area yang lebih luas. Jadi pada tahap ini inframerah akan menyebar untuk membentuk sebuah area seperti setengah lingkaran dengan jarak jangkauan maksimal 50 cm.

Dalam tugas akhir ini, Leap Motion dimanfaatkan untuk mendapatkan gerakan kedua tangan sebagai acuan kendali pergerakan dan mengeluarkan serangan sihir saat bermain. Gerakan tangan ini diamati oleh sensor pada Leap Motion Controller sebagai input yang dapat memicu kondisi tertentu di dalam Permainan ini.

2.7. Library Leap Motion

SDK Leap Motion berisi dua *library* dasar yang mendefinisikan API ke data pelacakan Leap Motion. Satu *library* ditulis dengan bahasa C++, yang kedua ditulis dalam bahasa. *Class* pembungkus untuk *library* ini mendefinisikan bahasa pengikat untuk C# dan

Objective-C. Dalam Unity 5, baik versi Pro atau Personal mendukung *plugins*. *Plugin* Unity menggunakan definisi kelas C# dalam folder LeapC dari *asset* inti. Library ini memuat library leapC.dll (Windows) [8].

2.8. Oculus Rift

Oculus Rift adalah sistem realitas virtual yang benar-benar menenggelamkan anda ke dalam dunia maya. Rift adalah peranti layar ikat kepala untuk menampilkan realitas virtual yang saat ini dikembangkan oleh Oculus VR [9].

[Halaman ini sengaja dikosongkan]

BAB III
DESAIN DAN PERANCANGAN SISTEM

Bab ini membahas tentang desain dan perancangan Sistem Realitas Virtual untuk Magus. Pembahasan yang dilakukan meliputi analisis sistem, perancangan sistem, skenario simulasi, dan perancangan antar muka sistem.

3.1. Analisis Sistem

Sub bab ini akan membahas tentang analisis kebutuhan sistem, meliputi spesifikasi kebutuhan sistem, baik itu kebutuhan fungsional sistem maupun kebutuhan non-fungsional sistem, dan identifikasi pengguna sistem.

3.1.1. Spesifikasi Kebutuhan Sistem

Pada sistem ini terdapat beberapa kebutuhan fungsional dan kebutuhan non-fungsional yang mendukung berjalannya sistem. Kebutuhan fungsional sistem dapat dilihat pada Tabel 3.1, sedangkan kebutuhan non-fungsional sistem dapat dilihat pada Tabel 3.2.

Tabel 3.1 Kebutuhan Fungsional Sistem

Kode	Deskripsi
F1	Pemain dapat melihat menu utama
F2	Pemain dapat memilih pilihan pada menu utama
F3	Pemain dapat melihat *gameplay*
F4	Pemain dapat melihat daftar sihir
F5	Pemain dapat keluar dari permainan
F6	Pemain dapat memulai permainan
F7	Pemain dapat memulai kembali permainan yang telah selesai
F8	Pemain dapat bergerak maju dan mundur

Kode	Deskripsi
F9	Pemain dapat memberikan perintah berupa sihir dalam Bahasa inggris
F10	Pemain memiliki *health bar* dan mana bar yang dapat berkurang
F11	Karakter pemain dapat mati dalam permainan.
F12	Pemain dapat memenangkan permainan dengan mengumpulkan tiga buah *gem* dalam Permainan.
F13	Musuh dapat menyerang pemain
F14	Musuh dapat berjalan mendekati pemain apabila pemain berada pada radius tertentu.
F15	Musuh kuat atau *Boss* dapat menyerang pemain menggunakan sihir
F16	Tiap musuh yang ada akan menjatuhkan item secara random

Tabel 3.2 Kebutuhan Non-Fungsional Sistem

Kode	Deskripsi
NF1	Sistem dapat dijalankan di Windows 10
NF2	Sistem memiliki antar muka yang mudah dipahami

3.1.2. Identifikasi Pengguna

Pengguna yang dapat memainkan permainan Magus ini adalah siapa saja (umum). Sehingga, pengguna berhak menggunakan seluruh fungsionalitas yang terdapat pada sistem.

3.2. Perancangan Sistem

Sub bab ini membahas tentang bagaimana sistem ini dirancang, meliputi deskripsi umum sistem dan arsitektur sistem.

3.2.1. Deskripsi Umum Sistem

Magus merupakan permainan realitas virtual berbasis *personal computer* ber-*genre* FPS yang memanfaatkan Leap Motion dan *Voice Recognition* sebagai kendalnya. Sistem ini merupakan permainan realitas virtual yang menghadirkan atmosfer dan suasana ketika berada di dunia fantasi.

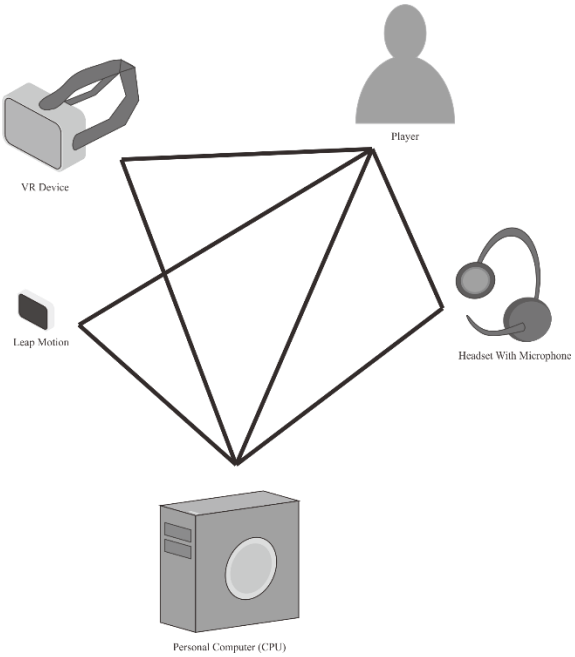
Pembangunan sistem ini dimulai dari membuat terrain yang bertema fantasi yang akan dijadikan sebagai lingkungan nyata di dalam permainan VR.

Proses pembangunan sistem selanjutnya yaitu dengan merancang *gameplay* yang sesuai dengan tema dari permainan ini. Mencari *asset-asset* yang sesuai, dan melengkapi *asset* dengan membuat manual.

3.2.2. Arsitektur Sistem

Magus mengintegrasikan antara *Personal Computer*, Oculus Rift, Leap Motion, dan *Microphone external*. Diproses di PC dan ditampilkan di Oculus Rift, dengan menggunakan Leap Motion, permainan Magus dapat mendeteksi gerakan tangan pemain yang berada depan sensor Leap. Lalu dengan menggunakan *microphone*, permainan Magus dapat mendengarkan apa yang dikatakan pemain. Dengan begini, permainan Magus lebih terasa real, karena menggunakan tangan dan suara sendiri untuk berinteraksi dalam permainan.

Seperti yang dijelaskan diatas, Gambar 3.1 berikut merupakan tampilan diagram dari alur sistem permainan Magus.



Gambar 3.1Arsitektur Sistem

Dari pemain, sistem menerima input berupa perintah suara melalui *microphone* dan isyarat gerakan tangan melalui leap motion, lalu diproses di CPU untuk dijadikan data, hasil dari pemrosesan data tersebut di keluarkan berupa tampilan 3D realitas virtual melalui Oculus Rift dan efek suara melalui *headphone*.

3.3. Perancangan Perilaku *Environment*

Pada sub bab ini menjelaskan tentang skenario komponen-komponen yang ada di dalam *game*. Komponen yang dibahas merupakan komponen pelengkap yang dikerjakan penulis untuk menyempurnakan *game* Magus ini, adapun komponen tersebut meliputi mekanik pemain, mekanik musuh, *health*, *mana* dan *item*.

3.3.1. Perilaku Tangan Pemain

Dalam permainan ini, pemain menggunakan tanganya untuk mengendalikan pergerakanya dan mengeluarkan atau menggunakan barang yang ada di dalam game. Kedua tangan dapat digunakan untuk berinteraksi dengan *UI Menu* dalam game dengan menyentuhnya, adapun saat dalam bermain, kedua tangan pemain memiliki fungsi yang berbeda yang dibagi menjadi,

1. Tangan Kiri

Tangan kiri digunakan pemain untuk menggerakan posisinya maju dan mundur, untuk bergerak maju pemain menghadapkan telapak tangan kiri kedepan, dan untuk mundur pemain menghadapkan telapak tangan kiri kebelakang.

2. Tangan Kanan

Tangan kanan digunakan pemain untuk menggunakan sihir dan *item* yang ada atau didapatkanya di dalam game, cara menggunakannya pemain hanya perlu mengarahkan telapak tangan kanan kedepan dan menyebut sihir untuk menggunakan *item* pembantu ataupun menyerang pemain dengan sihir yang tersedia. Jika pemain menggunakan sihir, *mana bar* pemain akan berkurang, jika *mana* pemain habis, pemain tidak akan bisa mengeluarkan sihir. Pemain dapat menghadapkan tangan kanan kedepan dan menyebutkan *item* pembantu mana yang akan digunakan untuk meningkatkan *mana bar* atau *health bar* nya.

3.3.2. Perilaku Musuh

Dalam permainan Magus, ada dua tipe musuh yang akan menghalangi pemain memenangkan permainan ini, yaitu musuh tipe lemah /*creep* dan musuh tipe kuat / *boss*.

1. Musuh Lemah

Musuh lemah tersebar di tempat permainan ini, musuh lemah bertipe serangan dekat, musuh lemah akan mendatangi dan menyerang pemain jika pemain berada di arah penglihatan dan jarak serang si musuh. Ada tiga jenis musuh lemah dalam permainan Magus ini, tiap tingkat musuh dari *level* satu sampai tiga dibedakan dari jumlah *health* nya dan besar *damage* yang ditimbulkan apabila musuh menyerang pemain. Jika pemain berhasil mengalahkan musuh lemah ini, barang pembantu dalam permainan secara acak akan muncul ditempat musuh mati, pemain dapat mengambilnya dengan melintasi tempat barang itu berada dan menggunakanya jika dibutuhkan.

2. Musuh Kuat

Terdapat tiga musuh kuat yang tersebar di titik tertentu di dalam permainan. Tiga musuh kuat ini yang harus pemain kalahkan untuk memenangkan permainan dengan mengambil *item* permata yang jatuh setelah musuh kuat ini dikalahkan. Musuh kuat memiliki tipe serangan jarak jauh dengan mengeluarkan sihir untuk menyerang. Musuh kuat memiliki tubuh yang besar dan *health bar* yang besar. Musuh kuat akan langsung menyerang pemain jika pemain berada di jarak pandangan dan jarak serangan si musuh kuat. Musuh kuat tidak akan mengejar pemain. Bila pemain mengalahkan musuh kuat, sebuah *item* kristal akan muncul dan pemain harus mendapatkan ketiga *item* tersebut dari ketiga musuh kuat untuk memenangkan permainan.

3.3.3. Perilaku *Health*, *Mana Bar* dan *Item*

Dalam permainan ini pemain dan musuh memiliki *health bar*, hanya pemain yang memiliki tambahan *mana bar* yang digunakan untuk mengetahui sebarapa banyak *health* dan *mana bar* yang dimilikinya saat itu. *Item* atau barang akan muncul ketika musuh dikalahkan oleh pemain. Adapun mekaniknya adalah sebagai berikut,

1. *Health bar* dan *Mana bar*

Health bar dan *Mana bar* ada sebagai indikator untuk pemain mengetahui kondisi dirinya dan kondisi musuh. *Health bar* akan berkurang jika pemain atau musuh menerima serangan, jumlah *bar* yang berkurang sesuai dengan nilai serangan yang diberikan. *Mana bar* akan berkurang jika pemain mengeluarkan sihirnya untuk menyerang musuh.

2. *Item* atau barang

Item didapat setelah pemain mengalahkan musuh dalam permainan, ada tiga *item* yang mungkin pemain peroleh yaitu ada, *heal item, mana item,* dan kristal. Pemain dapat menggunakan *heal item* dan *mana item* untuk meningkatkan *health* atau *mana bar* miliknya demi bertahan hidup di dalam permainan. *Item* Kristal didapat dengan mengalahkan musuh kuat dalam permainan Magus ini, kumpulkan tiga kristal dari tiap musuh kuat untuk memenangkan permainan ini.

3.4. Perancangan Skenario Simulasi

Pada sub bab ini menjelaskan tentang skenario permainan untuk menentukan kondisi menang atau kalah. Selain itu akan dibahas pula aturan permainan dari permainan.

3.4.1. Alur Simulasi

Alur permainan dari permainan Magus antara lain:

1. Saat permainan dijalankan maka pemain akan melihat Menu Utama yang memiliki 4 tombol interaksi yaitu, tombol *Play*, *How To Play, Skill List,* dan *Exit*.
2. Untuk mengetahui *gameplay* pemain menekan tombol *How To Play*. Pemain dapat membaca beberapa petunjuk mengenai cara cara menyelesaikan permainan.

3. Untuk mengetahui daftar sihir yang dapat digunakan, pemain dapat menekan tombol *Skill List*.
4. Untuk bermain pemain menakan tombol *Play*.
5. Setelah memilih tombol main, maka pemain akan menuju halaman utama permainan.
6. Di dalam halaman utama terdapat objek objek yang dapat berinteraksi dengan pemain diantaranya adalah musuh, item, serta beberapa tombol.
7. Pada awalnya musuh akan berada pada posisi *idle*, menunggu interaksi dari pemain.
8. Musuh akan melakukan sebuah aksi apabila pemain berada didekat musuh.
9. Ketika musuh mendeteksi pemain, musuh akan berjalan menuju pemain lalu menyerang pemain.
10. Pemain diharuskan untuk mengalahkan semua musuh yang ada dengan menggunakan sihir.
11. Di dalam permainan pemain diharapkan untuk mengumpulkan item berupa *gem* yang akan jatuh setelah mengalahkan *Boss*.
12. Untuk melepaskan sihir, pemain dapat menyebutkan sihir yang ada dengan mengisyaratkan tangan kanan menghadap kedepan.
13. Untuk menggunakan *item heal* dan *mana*, pemain dapat menyebutkan heal dan regen mana.
14. Pemain dikatakan kalah apabila *health bar* dari pemain telah habis.
15. Pemain dikatakan menang apabila telah berhasil mengumpulkan 3 buah *gem* di dalam permainan.

3.4.2. Aturan Permainan

Dalam memainkan permainan ini, terdapat aturan sebagai berikut:

1. Pada awal permainan, *health* dan *mana* pemain akan berada pada kondisi 100%.

2. Pemain dapat melakukan apa saja, terbatas dengan fitur yang ada.
3. Pemain harus dapat menjaga *health bar* agar tidak habis.
4. Pemain harus mengalahkan semua Bos yang ada dan mendapatkan *item* dari bos tersebut.
5. Jumlah musuh yang ada tidak diketahui.
6. Musuh tersebar diseluruh tempat.
7. Besar *health poin* musuh biasa 100 health poin musuh bos 200.
8. Besar *health poin* musuh tidak diketahui.
9. Untuk melakukan interaksi pemain harus mengisyaratkan tangan di depan Leap Motion dan/atau menyebutkan sihir yang ada.
10. Setiap membunuh Musuh, beberapa barang/*item* penambah *health poin* dan *mana poin* yang kemungkinan muncul ditempat musuh terbunuh dan pemain dapat mengambil *item* tersebut dengan melewati *item* tersebut.
11. Pemain dapat menggunakan barang penambah *health poin* dan *mana* poin dengan cara sama seperti mengeluarkan sihir.

3.5. Perancangan Tampilan Antarmuka

Sub bab ini membahas bagaimana rancangan antarmuka pengguna yang akan digunakan untuk tugas akhir. Rancangan antarmuka yang dibahas meliputi ketentuan masukan dan rancangan halaman tampilan. Di dalam aplikasi ini terdapat beberapa tampilan, yaitu tampilan menu permainan, tampilan *gameplay*, tampilan daftar sihir, tampilan permainan (inti permainan), dan tampilan akhir permainan.

3.5.1. Tampilan Menu Permainan

Tampilan menu permainan merupakan tampilan yang pertama kali muncul ketika aplikasi dijalankan. Pada tampilan awal terdapat empat tombol, yaitu tombol *Play, How To Play, Skill List*,

dan *Exit*. Tampilan rancangan antarmuka dapat dilihat pada Gambar 3.2 berikut:

Gambar 3.2 Rancangan Tampilan Menu Permainan

Berikut penjelasan dari Gambar 3.2 :

1. Tombol *Play*, berfungsi untuk memulai permainan.
2. Tombol *How To Play*, berfungsi untuk menampilkan/menyembunyikan tampilan *gameplay* di bagian kiri player.
3. Tombol *Skill List*, berfungsi untuk menampilkan/menyembunyikan tampilan daftar sihir di bagian kanan *player*.
4. Tombol *Exit*, berfungsi untuk keluar dari permainan.

3.5.2. Tampilan Cara Bermain

Tampilan cara bermain merupakan halaman yang akan muncul setelah pemain menekan tombol *How To Play*. Halaman ini berisikan informasi *gameplay* yang perlu diketahui untuk dapat memainkan permainan. Tampilan rancangan antarmuka dapat dilihat pada Gambar 3.3 berikut:

Gambar 3.3 Rancangan Tampilan How To Play

3.5.3. Tampilan Daftar Sihir

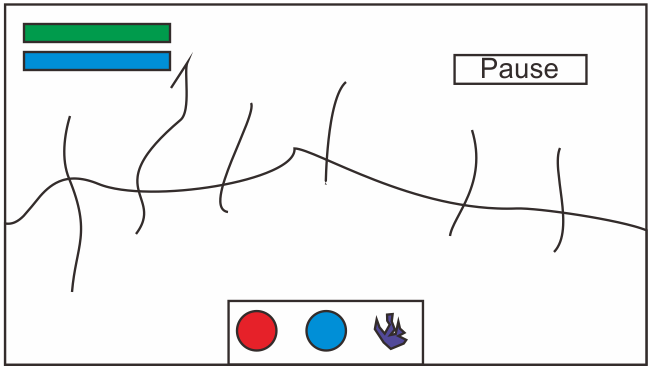
Tampilan daftar sihir merupakan halaman yang akan muncul setelah pemain menekan tombol *Skill List*. Halaman ini berisikan informasi sihir yang perlu diketahui untuk dapat memainkan permainan. Tampilan rancangan antarmuka dapat dilihat pada Gambar 3.4 berikut:

Gambar 3.4 Rancangan Tampilan Daftar Sihir

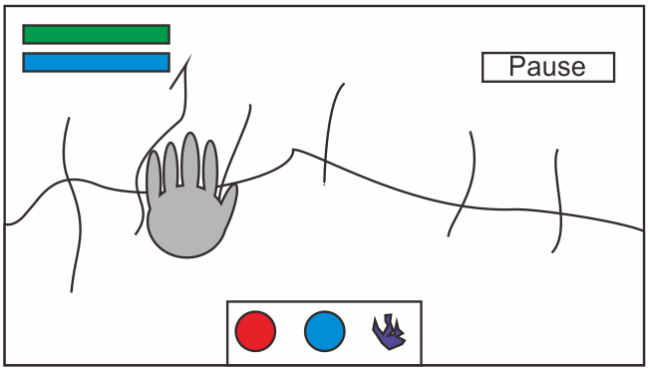
3.5.4. Tampilan Permainan

Tampilan permainan merupakan halaman yang muncul setelah pemain menekan tombol *Play* pada menu permainan. Halaman ini merupakan halaman utama dari permainan, dimana pemain mulai melakukan interaksi terhadap dunia virtual.

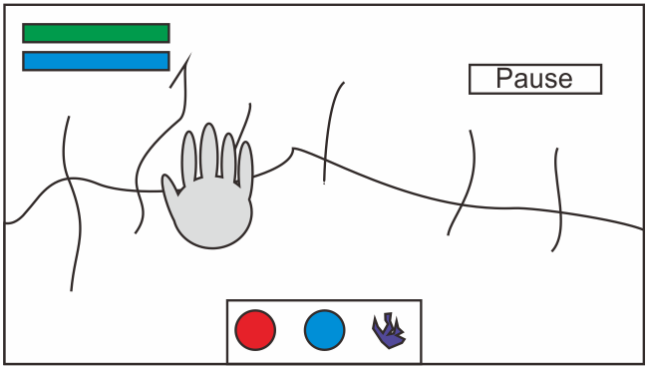
Pada tampilan ini terdapat enam tampilan antarmuka, tampilan rancangan antarmuka dapat dilihat pada Gambar 3.5 – 3.10 berikut:



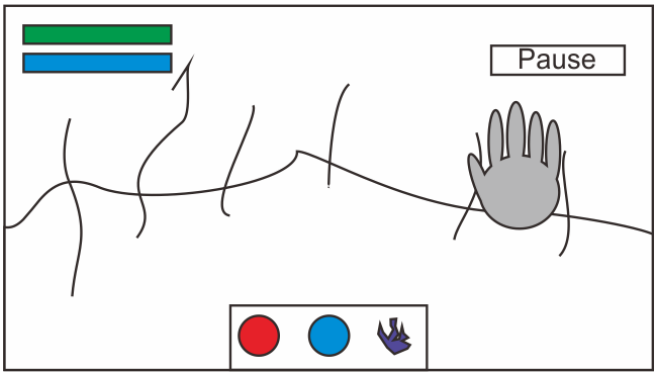
Gambar 3.5 Rancangan Tampilan Dalam Permainan



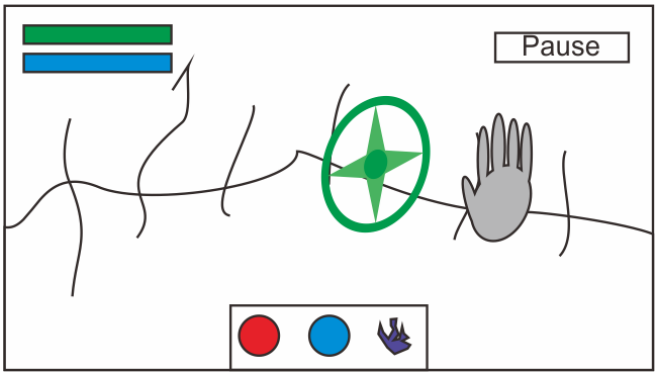
Gambar 3.6 Rancangan Tampilan Dalam Permainan, Tangan Kiri Menghadap Kedepan



Gambar 3.7 Rancangan Tampilan Dalam Permainan, Tangan Kiri Menghadap Kebelakang



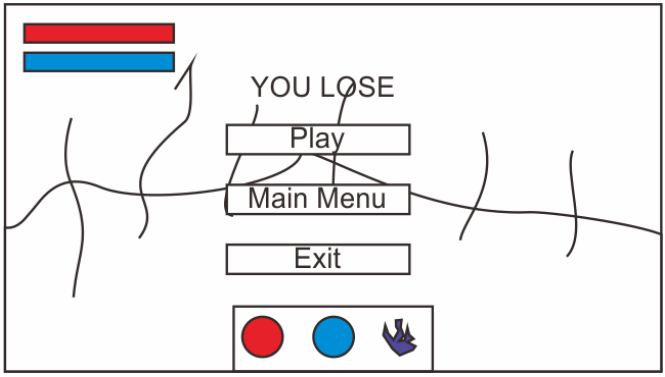
Gambar 3.8 Rancangan Tampilan Dalam Permainan, Tangan Kanan Tidak Aktif



Gambar 3.9 Rancangan Tampilan Dalam Permainan, Tangan Kanan Aktif

3.5.5. Tampilan Akhir Permainan

Tampilan akhir permainan merupakan halam ketika pemain mati karena *health bar* habis. Tampilan rancangan antarmuka dapat dilihat pada Gambar 3.10 berikut:



Gambar 3.10 Rancangan Tampilan Dalam Permainan, Permainan Selesai

Berikut merupakat penjelasan tombol tombol yang ada pada Gambar 3.10:

1. Tombol *Play*, berfungsi untuk mengulangi permainan.
2. Tombol *Main Menu*, berfungsi untuk kembali ke tampilan menu permainan.
3. Tombol *Exit*, berfungsi untuk keluar dari permainan.

[Halaman ini sengaja dikosongkan]

BAB IV
IMPLEMENTASI SISTEM

Bab ini membahas implementasi dari perancangan sistem sesuai dengan perancangan yang telah dibuat. Bahasa pemrograman yang digunakan untuk implementasi sistem adalah Bahasa pemrograman C#.

4.1. Lingkungan Implementasi

Lingkungan implementasi sistem yang digunakan untuk mengembangkan tugas akhir ini memiliki spesifikasi perangkat keras dan perangkat lunak yang ditunjukkan oleh Tabel 4.1.

Tabel 4. 1 Spesifikasi Perangkat

Perangkat	Spesifikasi
Perangkat Keras	• Prosesor: Intel® Core™ i7-7700U CPU @ 3.60GHz (4 CPUs), ~3.6GHz • Memori: 8192MB • VGA : NVIDIA GeForce GTX 1060 3Gb • Oculus Rift DK2 • Leap Motion • Headphone with Build in Microphone
Perangkat Lunak	• Sistem Operasi Microsoft Windows 10 64-bit • Perangkat Pengembang Unity3D 2017.3.1 • Perangkat Pembantu Visual Studio Community 2017, Microsoft Word 2017, Corel Draw X7, Adobe After Effects 2017cc, dan Blender

4.2. Implementasi Permainan

Implementasi dari masing-masing fungsi utama dituliskan menggunakan kode berbahasa C#. implementasi fungsi diurut berdasarkan antarmuka-antarmuka yang ada pada permainan.

4.2.1. Implementasi Menu Permainan

Gambar 4.1 merupakan tampilan implementasi dari menu permainan yang akan dijelaskan, sebagai berikut:

Gambar 4.1 Tampilan Menu Permainan

Pada Gambar 4.1 terdapat 4 tombol yaitu, *Play*, How To *Play*, Skill List, dan *Exit*. Berikut penjelasan dari tiap tombol:

1. *Play* untuk memulai permainan.
2. *How To Play* untuk memunculkan/menyembunyikan panel *gameplay*.
3. *Skill List* untuk memunculkan/menyembunyikan panel daftar sihir.
4. *Exit* untuk keluar dari permainan.

Pada Kode Sumber 4.1, terdapat beberapa fungsi untuk menjalankan halaman awal permainan ini. Diantaranya yaitu untuk pertama kali memulai permaian, fungsi untuk menampilkan panel

cara bermain, fungsi untuk menampilkan panel daftar sihir, dan untuk keluar dari permainan.

```
1.      public void GoPlay()
2.      {
3.          SceneManager.LoadScene("Main");
4.          Time.timeScale = 1;
5.      }
6.      public void GoMain()
7.      {
8.          SceneManager.LoadScene("Main Menu");
9.          Time.timeScale = 1;
10.     }
11.     public void GoHow2Play()
12.     {
13.         if (flaghow == false)
14.         {
15.             how2play.SetActive(true);
16.             flaghow = true;
17.         }
18.         else
19.         {
20.             how2play.SetActive(false);
21.             flaghow = false;
22.         }
23.     }
24.     public void SkillList()
25.     {
26.         if (flagskill == false)
27.         {
28.             skill.SetActive(true);
29.             flagskill = true;
30.         }
31.         else
32.         {
33.             skill.SetActive(false);
34.             flagskill = false;
35.         }
36.     }
37.     public void Quit()
38.     {
39.         Debug.Log("Has quit Permainan");
```

```
40.         Application.Quit();
41.     }
```

Kode Sumber 4.1 Fungsi Halaman Awal

4.2.2. Implementasi Cara Bermain

Tampilan cara bermain yang diimplementasikan dalam permainan dapat dilihat pada Gambar 4.2, berikut:

Gambar 4.2 Tampilan Gameplay

Seperti yang ditampilkan pada Gambar 4.2 disampaikan beberapa informasi cara bermain. Untuk kembali ke menu utama pemain cukup melihat kedepan.

4.2.3. Implementasi Daftar Sihir

Tampilan daftar sihir yang diimplementasikan dalam permainan dapat dilihat pada Gambar 4.3, berikut:

Gambar 4.3 Tampilan Daftar Sihir

Seperti yang ditampilkan pada Gambar 4.3 disampaikan beberapa informasi sihir sihir yang dapat digunakan dalam permainan. Untuk kembali ke menu utama pemain cuku melihat kedepan.

4.2.4. Implementasi Permainan

Tampilan permainan merupakan halaman tempat pemain melakukan interaksi. Pada halaman permainan, pemain dapat melakukan berbagai hal seperti jalan dan melepaskan sihir. Tampilan permainan yang diimplementasikan dapat dilihat pada Gambar 4.4 – 4.9.

Gambar 4.4 Tampilan Dalam Permainan,dengan satu tombol pause.

Pada Gambar 4.4 terdapat 1 tombol yaitu, *Pause*.Berikut penjelasan dari tombol:

1. *Pause* untuk menghentikan permainan sementara.

Gambar 4.5 Tampilan setelah pemain menyentuh tombol Pause

Pada Gambar 4.5 terdapat 3 tombol yaitu, *Resume*, *Main Menu* dan *Exit*. Berikut penjelasan dari tombol-tombol tersebut:

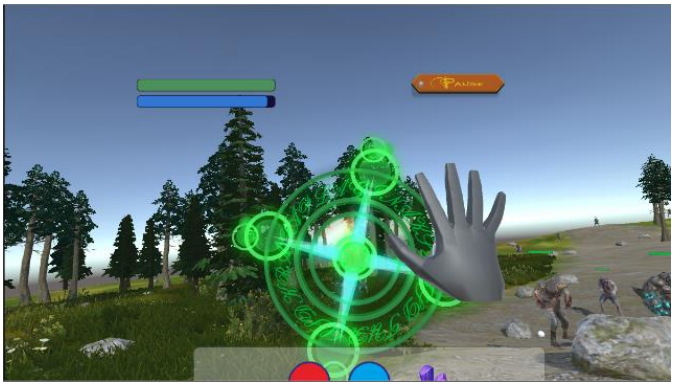
1. *Resume* untuk kembali melanjutkan permainan.
2. *Main menu* untuk pemain kembali ke *scene Main menu*.

3. *Exit* untuk keluar dari permainan.

Gambar 4.6 Tampilan Dalam Permainan, Interaksi Tangan Kiri Menghadap Kedepan

Gambar 4.7 Tampilan Dalam Permainan, Interaksi Tangan kiri, Menghadap Kebelakan

Gambar 4.8 Tampilan Dalam Permainan, Interaksi Tangan kanan, Sebelum Mengucapkan Sihir



Gambar 4.9 Tampilan Dalam Permainan, Interaksi Tangan Kanan, Setelah Mengucapkan Sihir

Terdapat beberapa obyek yang ada di dalam permainan, yaitu:

2. Musuh
3. *Boss* Musuh
4. *Health bar* pemain
5. *Health bar* Musuh

6. *Mana bar*
7. Informasi *item*
8. Tangan
9. Sihir

Implementasi fungsi dari tiap obyek pada Gambar 4.4 – Gambar 4.8 dapat dilihat di Kode Sumber 4.2 – Kode Sumber 4.10 . Diantaranya yaitu,

1. Tangan pemain

Tangan pemain dalam Permainan ini berfungsi sebagai control dalam permainan Magus ini, kedua tangan memiliki fungsi tersendiri,

1.1. Tangan Kiri

Tangan kiri digunakan sebagai kendali pergerakan pemain, jika telapak tangan pemain menghadap ke depan pemain akan bergerak maju dan jika telapak pemain menghadap ke belakang pemain akan bergerak mundur. Implementasi fungsi deteksi tangan kiri apakah dia maju atau mundur dapat dilihat di Kode Sumber 4.2 – Kode Sumber 4.3.

```
1.      public void PrintActiveMessage() {
2.          movement = true;
3.          Debug.Log("Maju :" + movement);
4.      }
5.
6.      public void PrintActiveMessageDua()
7.      {
8.          backmove = true;
9.          Debug.Log("Mundur : " + backmove);
10.     }
11.
12.     public void PrintDeactivateMessage() {
13.         movement = false;
14.         Debug.Log("Maju : " +movement);
15.     }
16.
```

```
17.     public void PrintDeactivateMessageDua()
        {
18.         backmove = false;
19.         Debug.Log("Mundur : " +backmove);
20.     }
21. }
```

Kode Sumber 4.2 Script atur boolean pergerakan moving

```
1.  public virtual void UpdateMovement()
2.  {
3.          boolmoving = scriptPrintJalan.movemen
    t;
4.          boolbackmove = scriptPrintJalan.backm
    ove;
5.
6.          Debug.Log("boolmoving : " + boolmovin
    g);
7.          if (HaltUpdateMovement)
8.           return;
9.
10.     if (EnableLinearMovement)
11.     {
12.          bool moveForward = Input.GetKey(KeyC
    ode.W) || Input.GetKey(KeyCode.UpArrow);
13.          bool moveLeft = Input.GetKey(KeyCode
    .A) || Input.GetKey(KeyCode.LeftArrow);
14.          bool moveRight = Input.GetKey(KeyCod
    e.D) || Input.GetKey(KeyCode.RightArrow);
15.          bool moveBack = Input.GetKey(KeyCode
    .S) || Input.GetKey(KeyCode.DownArrow);
16.
17.          bool dpad_move = false;
18.
19.          if (OVRInput.Get(OVRInput.Button.Dpa
    dUp))
20.          {
21.              moveForward = true;
22.              dpad_move = true;
23.
```

```
24.         }
25.
26.         if (OVRInput.Get(OVRInput.Button.Dpa
    dDown))
27.         {
28.             moveBack = true;
29.             dpad_move = true;
30.         }
31.           MoveScale = 1.0f;
32.
33.         if ((moveForward && moveLeft) || (mo
    veForward && moveRight) ||
34.             (moveBack && moveLeft) || (moveB
    ack && moveRight))
35.             MoveScale = 0.70710678f;
36.
37.         if (!Controller.isGrounded)
38.             MoveScale = 0.0f;
39.
40.         MoveScale *= SimulationRate * Time.d
    eltaTime;
41.           float moveInfluence = Acceleration
     * 0.1f * MoveScale * MoveScaleMultiplier;
42.
43.         if (dpad_move || Input.GetKey(KeyCod
    e.LeftShift) || Input.GetKey(KeyCode.RightSh
    ift))
44.             moveInfluence *= 2.0f;
45.
46.         Quaternion ort = transform.rotation;

47.         Vector3 ortEuler = ort.eulerAngles;

48.         ortEuler.z = ortEuler.x = 0f;
49.         ort = Quaternion.Euler(ortEuler);
50.
51.         if (boolmoving)
52.             MoveThrottle += ort * (transform
    .lossyScale.z * (moveInfluence * 2) * Vector
    3.forward);
53.         if (boolbackmove)
```

```
54.             MoveThrottle += ort * (transform
    .lossyScale.z * (moveInfluence * BackAndSide
    Dampen * 2) * Vector3.back);
55.
56.         if (moveLeft)
57.             MoveThrottle += ort * (transform
    .lossyScale.x * moveInfluence * BackAndSideD
    ampen * Vector3.left);
58.         if (moveRight)
59.             MoveThrottle += ort * (transform
    .lossyScale.x * moveInfluence * BackAndSideD
    ampen * Vector3.right);
```

Kode Sumber 4.3 Update movement player dari script Oculus Player Controller

1.2. Tangan Kanan

Tangan kanan digunakan pemain sebagai kendali terhadap sihir yang akan dikeluarkan pemain untuk menyerang atau lainya. Dalam hal ini jika telapak tangan kanan pemain menghadap ke depan, pemain masuk ke kondisi dapat mengeluarkan sihir atau item, hanya perlu diikuti kondisi ketersediaan mana pemain ataupun item pada pemain. Implementasi fungsi untuk deteksi apakah pemain dalam kondisi bias mengeluarkan sihir dari tangan kanan dapat dilihat dari Kode Sumber 4.4 – Kode Sumber 4.6.

```
1.      public void PrintActiveMessage()
2.      {
3.          castskill = true;
4.          Debug.Log("Castskill : " +castskill);
5.      }
6.      public void PrintDeactivateMessage()
7.      {
8.          castskill = false;
9.          Debug.Log("Castskill : " +castskill);
10.     }
```

Kode Sumber 4.4 Script atur boolean skill dapat dikeluarkan atau tidak

```
1.  private void Update()
2.  {
3.      mananow = pickUp.tmana;
4.      hpnow = pickUp.thp;
5.      boolCast = scriptCastSkill.castskill;
6.
7.      if (boolCast && _health.manaOn == true)
8.      {
9.          switch (word)
10.         {
11.             case "fire ball":
12.                 printCommand(word);
13.                 magicCircle.SetActive(true);
14.                 _health.consume(5);
15.                 cast(0);
16.                 StartCoroutine(Mati());
17.                 break;
18.             case "Lighting ball":
19.                 printCommand(word);
20.                 magicCircle.SetActive(true);
21.                 _health.consume(20);
22.                 cast(1);
23.                 StartCoroutine(Mati());
24.                 break;
25.             case "meteor":
26.                 printCommand(word);
27.                 magicCircle.SetActive(true);
28.                 _health.consume(30);
29.                 updateMeteorSource();
30.                 cast(2);
31.                 StartCoroutine(Mati());
32.                 break;
33.             case "green core":
34.                 printCommand(word);
35.                 magicCircle.SetActive(true);
36.                 selfCast(3);
37.                 _health.heal(50);
38.                 _health.consume(25);
39.                 StartCoroutine(Mati());
40.                 break;
41.             case "sunstrike":
42.                 printCommand(word);
```

```
                magicCircle.SetActive(true);
                _health.consume(20);
                cast(4);
                StartCoroutine(Mati());
                break;
            case "heal":
                if (hpnow > 0)
                {
                    printCommand(word);
                    _health.heal(1);
                    pickUp.useItem(1);
                    break;
                }
                else return;
            case "regen mana":
                if (mananow > 0)
                {
                    printCommand(word);
                    _health.heal(2);
                    pickUp.useItem(2);
                    break;
                }
                else return;
        }
    }
    else return;
}
```

Kode Sumber 4.5 Script cast sihir setelah boolean terpenuhi

```
private void cast(int magicIndex) {
    Vector3 pos;
    float yRot = transform.rotation.eulerAngles.y
;
    Vector3 forwardY = Quaternion.Euler(0.0f, yRo
t, 0.0f) * Vector3.forward;
    Vector3 forward = magicSpawner.transform.forw
ard;
    Vector3 right = magicSpawner.transform.right;
```

```
7.      Vector3 up = magicSpawner.transform.up;
8.      Quaternion rotation = Quaternion.identity;
9.      magicPrefab = PermainanObject.Instantiate(mag
    icPrefabs[magicIndex]);
10.     magicPrefabScript = magicPrefab.GetComponent<
    FireConstantBaseScript>();
11.
12.     if (magicPrefabScript == null)
13.     {
14.         magicPrefabScript = magicPrefab.GetCompon
    ent<FireBaseScript>();
15.         if (magicPrefabScript.IsProjectile)
16.         {
17.             rotation = magicSpawner.transform.rot
    ation;
18.             pos = magicSpawner.transform.position
    ;
19.         }
20.         else
21.         {
22.             pos = transform.position + (forwardY
    * 25.0f);
23.         }
24.     }
25.     else
26.     {
27.         pos = transform.position + (forwardY * 10
    .0f);
28.         rotation = transform.rotation;
29.         pos.y = 0.0f;
30.     }
31.
32.     FireProjectileScript projectileScript = magic
    Prefab.GetComponentInChildren<FireProjectileScrip
    t>();
33.     if (projectileScript != null)
34.     {
35.         projectileScript.ProjectileCollisionLayer
    s &= (~UnityEngine.LayerMask.NameToLayer("Friendl
    yLayer"));
36.     }
37.
```

```
38.     magicPrefab.transform.position = pos;
39.     magicPrefab.transform.rotation = rotation;
40. }
41. void selfCast(int magicIndex) {
42.     magicPrefab = PermainanObject.Instantiate(mag
    icPrefabs[magicIndex]);
43.     magicPrefab.transform.position = transform.po
    sition + transform.forward;
44.     magicPrefab.transform.rotation = transform.ro
    tation;
45.     source.Play();
46.     Destroy(magicPrefab, 5.0f);
47. }
```

Kode Sumber 4.6 Script atur posisi muncul sihir dan arah sihir

2. Musuh

Musuh dalam permainan ini terdapat dua tipe, 1 musuh tipe kuat/bos dan musuh tipe biasa/lemah, kedua tipe Musuh ini dapat mengetahui keberadaan *player* dari jarak pandang mereka, mereka pun akan bereaksi jika *player* menyerang mereka lebih dulu. Implementasi fungsi yang digunakan untuk mengendalikan musuh-musuh dapat dilihat pada Kode Sumber 4.6 - Kode Sumber 4.8.

```
1.      void Start () {
2.          attack = false;
3.          pursuing = false;
4.          if (player == null)
5.          {
6.              player = PermainanObject.FindPermaina
    nObjectWithTag("Player").GetComponent<Transform>(
    ) as Transform;
7.          }
8.          rb = GetComponent<Rigidbody>();
9.          rb.freezeRotation = true;
10.         animator = GetComponent<Animator>();
11.         rb.isKinematic = true;
12.     }
```

```
13.     void Update () {
14.         Vector3 direction = player.position - thi
    s.transform.position;
15.         direction.y = 0;
16.         float angle = Vector3.Angle(direction, he
    ad.transform.forward);
17.         float temps = Vector3.Distance(player.pos
    ition, this.transform.position);
18.
19.         if (Vector3.Distance(player.position, thi
    s.transform.position) < 10 && (angle < 30 || purs
    uing == true ))
20.         {
21.             pursuing = true;
22.             animator.SetBool("isIdle", false);
23.             this.transform.rotation = Quaternion.
    Slerp(this.transform.rotation, Quaternion.LookRot
    ation(direction), 0.1f);
24.              if(direction.magnitude > 2)
25.             {
26.                 rb.isKinematic = false;
27.                 this.transform.position += transf
    orm.forward * 1 * Time.deltaTime;
28.                 animator.SetBool("isWalking", tru
    e);
29.                 animator.SetBool("isAttacking", f
    alse);
30.               }
31.             else
32.             {
33.                 rb.isKinematic = true;
34.                 animator.SetBool("isAttacking", t
    rue);
35.                 animator.SetBool("isWalking", fal
    se);
36.             }
37.         }
38.         else if(attack == true)
39.         {
40.             pursuing = true;
41.             animator.SetBool("isIdle", false);
```

```
42.             this.transform.rotation = Quaternion.
    Slerp(this.transform.rotation, Quaternion.LookRot
    ation(direction), 0.1f);
43.
44.             if (direction.magnitude > 2)
45.             {
46.                 rb.isKinematic = true;
47.                 this.transform.position += transf
    orm.forward * 1 * Time.deltaTime;
48.                 animator.SetBool("isWalking", tru
    e);
49.                 animator.SetBool("isAttacking", f
    alse);
50.             }
51.             else
52.             {
53.                 rb.isKinematic = true;
54.                 this.transform.position = this.tr
    ansform.position;
55.                 animator.SetBool("isAttacking", t
    rue);
56.                 animator.SetBool("isWalking", fal
    se);
57.             }
58.         }
59.         else
60.         {
61.             animator.SetBool("isIdle", true);
62.             animator.SetBool("isWalking", false);

63.             animator.SetBool("isAttacking", false
    );
64.             pursuing = false;
65.         }
66.     }
67. }
```

Kode Sumber 4.7 Skript kontrol pergerakan dan animasi dari Musuh *biasa*

```
1.      void Start()
2.      {
3.          rb = GetComponent<Rigidbody>();
4.          rb.freezeRotation = true;
5.          animator = GetComponent<Animator>();
6.          if (player == null)
7.          {
8.              player = PermainanObject.FindPermaina
    nObjectWithTag("Player").GetComponent<Transform>(
    ) as Transform;
9.          }
10.     }
11.       void Update()
12.     {
13.         Vector3 direction = player.position - thi
    s.transform.position;
14.         direction.y = 0;
15.         float temps = Vector3.Distance(player.pos
    ition, this.transform.position);
16.
17.         if (Vector3.Distance(player.position, thi
    s.transform.position) < 20 || pursuing == true)
18.         {
19.             pursuing = true;
20.             animator.SetBool("isIdle", false);
21.             this.transform.rotation = Quaternion.
    Slerp(this.transform.rotation, Quaternion.LookRot
    ation(direction), 0.1f);
22.             if (direction.magnitude > 2 && direct
    ion.magnitude < 20)
23.             {
24.                 if (!magicCircle.activeSelf) {
25.                     StartCoroutine(processTask())
    ;
26.                 }
27.             }
28.             else
29.             {
30.                 Debug.Log("Masuk ELSE 2");
31.                 animator.SetBool("isIdle", true);
```

```
32.                 animator.SetBool("isAttacking2",
    false);
33.             }
34.         }
35.         else
36.         {
37.             animator.SetBool("isIdle", true);
38.             animator.SetBool("isAttacking", false
    );
39.             pursuing = false;
40.         }
41.     }
42.
43.     void flipBool() {
44.         if (isCasting)
45.         {
46.             isCasting = false;
47.         }
48.         else
49.         {
50.             isCasting = true;
51.         }
52.     }
53.
54.     IEnumerator processTask()
55.     {
56.         int i = 3;
57.
58.         if(y == true)
59.         {
60.             animator.SetBool("isIdle", false);
61.             animator.SetBool("isAttacking2", true
    );
62.             magicCircle.transform.LookAt(player.p
    osition);
63.             magicCircle.SetActive(true);
64.
65.             StartCoroutine(turnOffMagicCircle());

66.             y = false;
67.         }
68.         else if(y == false)
```

```
69.         {
70.             flipBool();
71.             animator.SetBool("isIdle", true);
72.             animator.SetBool("isAttacking2", fals
    e);
73.             yield return new WaitForSeconds(i);

74.             y = true;
75.         }
76.     }
77.       void cast(int magicIndex) {
78.         Vector3 pos;
79.         float yRot = magicCircle.transform.rotati
    on.eulerAngles.y;
80.         Vector3 forwardY = Quaternion.Euler(0.0f,
     yRot, 0.0f) * Vector3.forward;
81.         Vector3 forward = magicCircle.transform.f
    orward;
82.         Vector3 right = magicCircle.transform.rig
    ht;
83.         Vector3 up = magicCircle.transform.up;
84.         Quaternion rotation = Quaternion.identity
    ;
85.         magicObject = PermainanObject.Instantiate
    (magic[magicIndex]);
86.         magicPrefabScript = magicObject.GetCompon
    ent<FireConstantBaseScript>();
87.
88.         if (magicPrefabScript == null)
89.         {
90.             magicPrefabScript = magicObject.GetCo
    mponent<FireBaseScript>();
91.             if (magicPrefabScript.IsProjectile)
92.             {
93.                 rotation = magicCircle.transform.
    rotation;
94.                 pos = magicCircle.transform.posit
    ion;
95.             }
96.             else
97.             {
```

```
98.                  pos = transform.position + (forwa
    rdY * 25.0f);
99.              }
100.             }
101.             else
102.             {
103.                 pos = transform.position + (fo
    rwardY * 10.0f);
104.                 rotation = transform.rotation;

105.                 pos.y = 0.0f;
106.             }
107.
108.             FireProjectileScript projectileScr
    ipt = magicObject.GetComponentInChildren<FirePro
    jectileScript>();
109.             if (projectileScript != null)
110.             {
111.                 projectileScript.ProjectileCol
    lisionLayers &= (~UnityEngine.LayerMask.NameToLay
    er("FriendlyLayer"));
112.             }
113.
114.             magicObject.transform.position = p
    os;
115.             magicObject.transform.rotation = r
    otation;
116.         }
117.            IEnumerator turnOffMagicCircle()
118.         {
119.             yield return new WaitForSeconds(2)
    ;
120.             cast(0);
121.             magicCircle.SetActive(false);
122.         }
123.     }
```

Kode Sumber 4.8 Skript kontrol pergerakan dan animasi dari Musuh *bos*

3. *Health bar* dan *Mana bar*

Health dan *mana bar* yang ada di permainan baik pada pemain dan musuh diatur agar setiap *health bar* mencapai nol, musuh akan mati, atau pemain akan mengalami *gameover*, jika *mana bar* pemain kosong, pemain tidak mungkin dapat mengeluarkan sihir, dan jika pemain menggunakan *item* penambah *health* dan *mana*, *mana* dan *health player* akan bertambah mengikuti jumlah *item* yang pemain gunakan. Implementasi fungsi dari *health* dan *mana bar* yang digunakan pemain dan musuh dapat dilihat dari Kode Sumber 4.9.

```
1.      private void Start()
2.      {
3.          if(audiopleyer == null)
4.          {
5.              return;
6.          }
7.          enemyHealthBar.value = maxHealth;
8.          anim = GetComponent<Animator>();
9.          ai = GetComponent<AIFix>();
10.         ap = audiopleyer.GetComponent<AudioPlayer
    >();
11.     }
12.     public void takeDamage(int amount)
13.     {
14.         maxHealth -= amount;
15.         ai.attack = true;
16.         if (maxHealth < minHealth)
17.         {
18.             anim.SetBool("isDead", true);
19.             Destroy(this.PermainanObject, 3.0f);

20.             Debug.Log(item.Length);
21.             if (item.Length > 0)
22.             {
23.                 spawnRandom();
24.             }
25.         }
26.         enemyHealthBar.value = maxHealth;
```

```
27.     }
28.
29.     public void takeDamageBos(int amount)
30.     {
31.
32.         maxHealth -= amount;
33.         enemyHealthBar.value = maxHealth;
34.         Debug.Log("HP " + enemyHealthBar.value);

35.         if (enemyHealthBar.value <= 0)
36.         {
37.             anim.SetBool("isDead", true);
38.             Destroy(this.PermainanObject, 3.0f);

39.             spawnItem();
40.         }
41.     }
42.       public void heal(int a)
43.     {
44.         if (a == 1)
45.         {
46.             maxHealth += 25;
47.             enemyHealthBar.value = maxHealth;
48.         }
49.         if( a == 2)
50.         {
51.             maxMana += 25;
52.             enemyManaBar.value = maxMana;
53.         }
54.         if( a == 50)
55.         {
56.             maxHealth += 50;
57.             enemyHealthBar.value = maxHealth;
58.         }
59.     }
60.         public void consume(int cons)
61.     {
62.         if (maxMana - cons >= 0 && maxMana >= con
    s)
63.         {
64.             maxMana -= cons;
65.             enemyManaBar.value = maxMana;
```

```
66.             if (maxMana == 0)
67.             {
68.                 manaOn = false;
69.             }
70.             else manaOn = true;
71.
72.             Debug.Log(enemyManaBar.value + "Caste
    d");
73.         }
74.         else
75.         {
76.             manaOn = false;
77.             Debug.Log(enemyManaBar.value);
78.         }
79.     }
80.
81.     public void takeDamagePlayer(int amount)
82.     {
83.         maxHealth -= amount;
84.         enemyHealthBar.value = maxHealth;
85.         if (maxHealth < minHealth)
86.         {
87.             Debug.Log("DEAD");
88.             ap.playLose();
89.             Time.timeScale = 0;
90.         }
91.     }
92.
93.     void spawnRandom()
94.     {
95.         int totalspawn = Random.Range(0, 5);
96.         Debug.Log(totalspawn);
97.         for (int i = 0; i < totalspawn; i++)
98.         {
99.             int randomInt = Random.Range(0, item.
    Length);
100.                 Vector3 itemPos = new Vector3(
    this.transform.position.x + Random.Range(-
    1.0f, 1.0f),
101.
      this.transform.position.y + 0.3f, this.transfor
    m.position.z + Random.Range(-1.0f, 1.0f));
```

```
102.                    Instantiate(item[randomInt], i
    temPos, Quaternion.identity);
103.                }
104.            }
105.            void spawnItem()
106.            {
107.                    Vector3 itemPos = new Vector3(
    this.transform.position.x + Random.Range(-
    1.0f, 1.0f),
108.
      this.transform.position.y + 0.3f, this.transfor
    m.position.z + Random.Range(-1.0f, 1.0f));
109.                    Instantiate(item[0], itemPos,
    Quaternion.identity);
110.            }
111.            public void healingItems(int amount)
112.            {
113.                maxHealth += amount;
114.                enemyHealthBar.value = maxHealth;
115.            }
116.            public void healingmanaItems(int amoun
    t)
117.            {
118.                maxMana += amount;
119.                enemyManaBar.value = maxMana;
120.            }
121.        }
```

Kode Sumber 4.9 Skript kontrol health bar & mana bar Musuh *dan player*

3. *Pick Item*

Setelah pemain membunuh musuh, secara random musuh akan menjatuhkan *item*, berupa *item* penambah darah atau *mana*. Untuk mengambil *item*, pemain cukup melewati *item* tersebut dan penanda jumlah *item* di *interface layer* pemain akan bertambah sejumlah *item* yang didapat. Untuk menggunakan *item* pemain hanya perlu mengarahkan tangan seperti ingin mengeluarkan sihir dan mengatakan "*heal*"

untuk menambah *health* atau "*regen mana*" untuk menambah *mana*.
Implementasi fungsi efek mengambil *item* dan penggunaan *item* untuk panel *interface* pemain dapat dilihat di Kode Sumber 4.10.

```
1.      private void Start()
2.      {
3.          apx = ap.GetComponent<AudioPlayer>();
          }
4.
5.      private void OnTriggerEnter(Collider othe
    r)
6.      {
7.          foreach (Transform child in inventory
    Panel.transform)
8.          {
9.              if (child.PermainanObject.tag ==
    other.PermainanObject.tag)
10.             {
11.                 if (child.PermainanObject.tag
     == "hp")
12.                 {
13.                     string c = child.Find("Te
    xt").GetComponent<Text>().text;
14.                     int tcount = System.Int32
    .Parse(c) + 1;
15.                     thp = tcount;
16.                     child.Find("Text").GetCom
    ponent<Text>().text = "" + tcount;
17.                     Destroy(other.PermainanOb
    ject);
18.                     return;
19.                 }
20.                 else if(child.PermainanObject
    .tag == "mana")
21.                 {
22.                     string c = child.Find("Te
    xt").GetComponent<Text>().text;
23.                     int tcount = System.Int32
    .Parse(c) + 1;
```

```
24.                         tmana = tcount;
25.                         child.Find("Text").GetCom
    ponent<Text>().text = "" + tcount;
26.                         Destroy(other.PermainanOb
    ject);
27.                         return;
28.                     }
29.                     else if (child.PermainanObjec
    t.tag == "bositem")
30.                     {
31.                         string c = child.Find("Te
    xt").GetComponent<Text>().text;
32.                         int tcount = System.Int32
    .Parse(c) + 1;
33.                         tboss = tcount;
34.                         child.Find("Text").GetCom
    ponent<Text>().text = "" + tcount;
35.                         Destroy(other.PermainanOb
    ject);
36.                         if(tcount >= 3)
37.                         {
38.                             apx.playWin();
39.                             Time.timeScale = 0;

40.                         }
41.                         return;
42.                     }
43.                 }
44.             }
45.         }
46.           public void useItem(int a)
47.         {
48.             if(a == 1)
49.             {
50.
51.                 foreach (Transform child in inven
    toryPanel.transform)
52.                 {
53.                     if (child.PermainanObject.tag
     == "hp")
54.                     {
```

```
55.                         Debug.Log("HEALTH INCREAS
    E");
56.                         string c = child.Find("Te
    xt").GetComponent<Text>().text;
57.                         int tcount = System.Int32
    .Parse(c) - 1;
58.                         thp = tcount;
59.                         child.Find("Text").GetCom
    ponent<Text>().text = "" + thp;
60.                         return;
61.                     }
62.                 }
63.             }
64.             if (a == 2)
65.             {
66.
67.                 foreach (Transform child in inven
    toryPanel.transform)
68.                 {
69.                     if (child.PermainanObject.tag
     == "mana")
70.                     {
71.                         Debug.Log("MANA INCREASE"
    );
72.                         string c = child.Find("Te
    xt").GetComponent<Text>().text;
73.                         int tcount = System.Int32
    .Parse(c) - 1;
74.                         tmana = tcount;
75.                         child.Find("Text").GetCom
    ponent<Text>().text = "" + tmana;
76.                         return;
77.                     }
78.                 }
79.             }
80.         }
81. }
```

Kode Sumber 4.10 Script kontrol ambil dan pakai item dari Musuh

4.2.5. Implementasi Akhir Permainan

Tampilan akhir permainan memiliki dua opsi, yaitu kalah dan menang, jika pemain berhasil mengalahkan musuh dan mendapat 3 *item* bos, maka panel *win* akan muncul dan permainan berakhir jika pemain kalah atau mencapai *health bar* nol, maka panel *lose* akan muncul dan permainan berakhir. Saat panel muncul pemain dapat memilih ingin bermain lagi atau kembali ke *Main menu* awal. Implementasi fungsi kendali tampilan panel *win* dan *lose* dapat dilihat di Kode Sumber 4.11.

```
1.      void Start () {
2.          MusicSource.clip = MusicClip[0];
3.          MusicSource.loop = true;
4.          MusicSource.volume = 1;
5.          MusicSource.playOnAwake = true;
6.          MusicSource.Play();
7.      }
8.      public void playLose()
9.      {
10.
11.         MusicSource.clip = MusicClip[2];
12.         MusicSource.volume = 1;
13.         MusicSource.loop = false;
14.         MusicSource.Play();
15.         PanelLose.SetActive(true);
16.     }
17.
18.     public void playWin()
19.     {
20.         MusicSource.Stop();
21.         MusicSource.clip = MusicClip[1];
22.         MusicSource.volume = 1;
23.         MusicSource.loop = false;
24.         MusicSource.Play();
25.         PanelWin.SetActive(true);
26.     }
27. }
```

Kode Sumber 4.11 Script kendali pemunculan panel win dan lose

```
1.  public void takeDamagePlayer(int amount)
2.  {
3.      maxHealth -= amount;
4.      enemyHealthBar.value = maxHealth;
5.      if (maxHealth < minHealth)
6.      {
7.          Debug.Log("DEAD");
8.          ap.playLose();
9.          Time.timeScale = 0;
10.     }
11. }
```

Kode Sumber 4.12 tampilkan panel lose ketika darah player habis

```
1.  else if (child.PermainanObject.tag == "bositem")

2.  {
3.      string c = child.Find("Text").GetComponent<Te
    xt>().text;
4.      int tcount = System.Int32.Parse(c) + 1;
5.      tboss = tcount;
6.      child.Find("Text").GetComponent<Text>().text
    = "" + tcount;
7.      Destroy(other.PermainanObject);
8.      if(tcount >= 3)
9.      {
10.         apx.playWin();
11.         Time.timeScale = 0;

12.     }
13.     return;
14. }
```

Kode Sumber 4.13 Fungsi Win terpanggil ketika pemain berhasil mendapatkan 3 bos item

Tampilan akhir permainan baik menang ataupun kalah yang diimplementasikan dapat dilihat pada Gambar 4.9 – 4.10.

Gambar 4.10 Panel Win muncul ketika pemain berhasil menyelesaikan game

Gambar 4.11 Panel lose muncul ketika health bar pemain mencapai nol

Pada Gambar 4.9 dan Gambar 4.10 terdapat 3 tombol yaitu, *Play*, *Main Menu*, *Exit*. Berikut penjelasan dari tiap tombol:

1. *Play* untuk memulai permainan baru.
2. *Main Menu* untuk kembali kehalaman awal menu permainan Permainan.
3. *Exit* untuk keluar dari permainan.

[Halaman ini sengaja dikosongkan]

BAB V
PENGUJIAN DAN EVALUASI

Pada bab ini akan dijelaskan mengenai rangkaian uji coba dan evaluasi yang dilakukan. Proses pengujian dilakukan menggunakan metode *blackbox* berdasarkan skenario yang telah ditentukan.

5.1. Lingkungan Pengujian

Lingkungan penguijan sistem pada pengerjaan tugas ini dilakukan pada lingkungan dan alat kakas pada Table 5.1 berikut:

Table 5.1 Tabel Lingkungan Pengujian Sistem

Perangkat	Spesifikasi
Perangkat Keras	• Prosesor: Intel® Core™ i7-7700U CPU @ 3.60GHz (4 CPUs), ~3.6GHz • Memori: 8192MB • VGA : NVIDIA GeForce GTX 1060 3Gb • Oculus Rift DK2 • Leap Motion • Headphone with Build in Microphone
Perangkat Lunak	• Sistem Operasi Microsoft Windows 10 64-bit • Perangkat Pengembang Unity3D 2017.3.1 • Perangkat Pembantu Visual Studio Community 2017, Microsoft Word 2017, Corel Draw X7, Adobe After Effects 2017cc, dan Blender

5.2. Pengujian Fungsionalitas

Pengujian fungsionalitas sistem dilakukan dengan menyiapkan sejumlah skenario sebagai tolok ukur keberhasilan

pengujian. Pengujian fungsionalitas dilakukan dengan mengacu pada sub bab 3.3. Pengujian fungsionalitas yang terdapat pada permainan dijabarkan sebagai berikut.

5.2.1. Uji Coba Menu Permainan

Pada sub bab ini dijelaskan secara detil mengenai skenario yang dilakukan dan hasil yang didapatkan dari pengujian fungsionalitas perangkat lunak yang dibangun pada halaman awal. Penjelasan disajikan dengan menampilkan kondisi awal, masukan, keluaran, hasil yang dicapai, dan kondisi akhir.

Pada menu permainan yang akan diuji adalah fungsionalitas tombol yang terdapat di menu utama, yaitu tombol *Play*, *How To Play*, *Skill List*, dan *Exit*. Tampilan menu permainan dapat dilihat pada Gambar 4.1. Skenario yang telah diuji terdapat pada Table 5.2.

Table 5.2 Hasil Uji Coba Menu Permainan

ID	**UF-001**
Nama	Uji Coba Pada Menu Permainan
Tujuan uji coba	Pengguna mengetahui fungsionalitas tombol yang ada pada menu permainan
Kondisi awal	Pemain berada pada menu permainan
Skenario 1	***Pemain memilih tombol Play***
Masukan	Menekan tombol *Play* pada dunia virtual
Keluaran yang diharapkan	Pemain berpindah ke halaman permainan
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain berada pada halaman permainan
Skenario 2	***Pemain memilih tombol How To Play***
Masukan	Menekan tombol *How To Play* pada dunia virtual

Keluaran yang diharapkan	Muncul panel berupa informasi *gameplay* di sebelah kiri pemain
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain dapat membaca panel informasi
Skenario 3	***Pemain memilih tombol Skill List***
Masukan	Menekan tombol Skill List pada dunia virtual
Keluaran yang diharapkan	Muncul panel berupa informasi daftar sihir di sebelah kanan pemain
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain dapat membaca panel informasi
Skenario 4	***Pemain memilih tombol Exit***
Masukan	Menekan tombol *Exit* pada dunia virtual
Keluaran yang diharapkan	Pemain berpindah keluar dari permainan
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain berpindah keluar permainan

Hasil uji dari skenario 1 berpindah ke halaman permainan dapat dilihat pada Gambar 4.4, skenario 2 muncul panel informasi *gameplay* dapat dilihat pada Gambar 4.2, skenario 3 muncul panel informasi daftar sihir dapat dilihat pada Gambar 4.3, dan skenario 4 keluar dari permainan, pemain akan keluar dari permainan.

5.2.2. Uji Coba Permainan

Pada sub bab ini dijelaskan mengenai skenario yang dilakukan dan hasil yang didapatkan dari pengujian fungsionalitas pada permainan. Penjelasan disajikan dengan menampilkan kondisi

awal, masukan, keluaran, hasil yang dicapai, dan kondisi akhir. Skenario yang telah diuji terdapat pada Table 5.3.

Table 5.3 Hasil Uji Coba Permainan Fitur Utama

ID	**UF-002**
Nama	Uji Coba Pada Permainan
Tujuan uji coba	Pengguna mengetahui fungsionalitas interaksi yang ada pada permainan
Kondisi awal	Pemain berada pada halaman permainan
Skenario 1	***Pemain bergerak maju***
Masukan	Memajukan tangan kiri menghadap kedepan
Keluaran yang diharapkan	Pemain bergerak maju sesuai arah pandang
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain bergerak maju sesuai arah pandang
Skenario 2	***Pemain bergerak mundur***
Masukan	Memajukan tangan kiri menghadap kebelakang
Keluaran yang diharapkan	Pemain bergerak mundur sesuai arah pandang
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain bergerak mundur sesuai arah pandang
Skenario 3	***Pemain melepaskan sihir***
Masukan	Memajukan tangan kanan menghadap kedepan lalu mengucapkan kata sihir
Keluaran yang diharapkan	Muncul sihir disekitar pemain
Hasil uji coba	Berhasil
Kondisi Akhir	Muncul sihir disekitar pemain

Skenario 4	***Pemain mengalahkan* musuh**
Masukan	Memajukan tangan kanan menghadap ke musuh lalu mengucapkan kata sihir.
Keluaran yang diharapkan	*Health bar* musuh berkurang sesuai dengan nilai kerusakan sihir
Hasil uji coba	Berhasil
Kondisi Akhir	*Health bar* musuh berkurang sesuai dengan nilai kerusakan sihir
Skenario 5	***Pemain dikalahkan oleh* musuh**
Masukan	Pemain bergerak menuju musuh
Keluaran yang diharapkan	Musuh melakukan serangan kepada pemain
Hasil uji coba	Berhasil
Kondisi Akhir	Musuh melakukan serangan kepada pemain
Skenario 6	***Musuh menjatuhkan item***
Masukan	Pemain mengalahkan musuh
Keluaran yang diharapkan	Musuh menjatuhkan *item* secara *random*
Hasil uji coba	Berhasil
Kondisi Akhir	Musuh menjatuhkan *item* secara *random*
Skenario 7	***Pemain menggunakan item penambah darah***
Masukan	Memajukan tangan kanan menghadap kedepan lalu mengucapkan "*Heal*"
Keluaran yang diharapkan	*Health bar* pemain bertambah
Hasil uji coba	Berhasil
Kondisi Akhir	*Health bar* pemain bertambah

Skenario 8	***Pemain menggunakan item penambah mana***
Masukan	Memajukan tangan kanan menghadap kedepan lalu mengucapkan "*Regen Mana*"
Keluaran yang diharapkan	*Mana bar* pemain bertambah
Hasil uji coba	Berhasil
Kondisi Akhir	*Mana bar* pemain bertambah
Skenario 9	***Menampilkan kondisi kalah***
Masukan	*Health bar* pemain menjadi merah seutuhnya
Keluaran yang diharapkan	Muncul panel informasi kekalahan yang berisi tombol *Play*, *Main Menu*, dan *Exit*
Hasil uji coba	Berhasil
Kondisi Akhir	Muncul panel informasi kekalahan yang berisi tombol *Play*, *Main Menu*, dan *Exit*
Skenario 10	***Pemain kondisi menang***
Masukan	Pemain berhasil mengumpulkan tiga buah *gem*
Keluaran yang diharapkan	Muncul panel informasi kemenangan yang berisi tombol *Play*, *Main Menu*, dan *Exit*
Hasil uji coba	Berhasil
Kondisi Akhir	Muncul panel informasi kekalahan yang berisi tombol *Play*, *Main Menu*, dan *Exit*
Skenario 11	***Pemain memilih tombol pause***
Masukan	Menekan tombol *Pause* pada dunia virtual
Keluaran yang diharapkan	Permainan berhenti sementara. UI *Pause* Menu muncul yang berisi tombol *Resume*, *Main menu*, *Exit*.
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain dapat menghentikan permainan sementara.

Skenario 12	***Pemain menekan tombol Resume pada UI Pause Menu***
Masukan	Menekan tombol *Resume* pada dunia virtual
Keluaran yang diharapkan	Pemain dapat melanjutkan kembali permainan yang dihentikan. UI *Pause Menu* hilang.
Hasil uji coba	Berhasil.
Kondisi akhir	Pemain dapat melanjutkan permainan kembali.
Skenario 13	***Pemain menekan tombol Main Menu pada UI Pause Menu***
Masukan	Menekan tombol *Main menu* pada dunia virtual
Keluaran yang diharapkan	Pemain berpindah ke tampilan *main menu* di awal permainan.
Hasil uji coba	Berhasil.
Kondisi akhir	Pemain dapat berpindah ke *main menu* awal.
Skenario 14	***Pemain menekan tombol Exit pada UI Pause Menu***
Masukan	Menekan tombol *Exit* pada dunia virtual
Keluaran yang diharapkan	Pemain dapat keluar dari permainan
Hasil uji coba	Berhasil.
Kondisi akhir	Pemain keluar dari permainan.

Hasil dari skenario 1 yaitu pemain bergerak maju. Selanjutnya skenario 2 yaitu pemain bergerak mundur. Selanjutnya skenario 3 yaitu pemain memunculkan sihir berupa Fire Ball, Lighting Ball, Meteor, Green Core, atau Sunstrike. Selanjutnya skenario 4 yaitu *health bar* musuh akan berkurang, jika *health bar* musuh menjadi merah semua maka musuh akan mati. Selanjutnya skenario 5 yaitu *health bar* pemain akan berkurang apabila pemain terkena serangan dari musuh. Selanjutnya skenario 6 yaitu musuh

menjatuhkan *item* secara *random*, item yang terjatuh berupa *item mana* dan darah yang akan digunakan pada skenario 7 dan skenario 8. Selanjutnya skenario 7 yaitu pemain menggunakan item penambah darah untuk menambah *health bar*. Selanjutnya skenario 8 yaitu pemain menggunakan item penambah mana untuk menambah mana bar. Selanjutnya skenario 9 yaitu *health bar* pemain menjadi merah seutuhnya, pada kondisi ini pemain telah kalah dari permainan. Selanjutnya skenario 10 yaitu pemain berhasil mengumpulkan tiga buah *gem* yang didapatkan dari tiap *Boss*, dalam kondisi ini pemain memenangkan permainan.

5.2.3. Uji Coba Akhir Permainan

Pada sub bab ini dijelaskan secara detil mengenai skenario yang dilakukan dan hasil yang didapatkan dari pengujian fungsionalitas perangkat lunak yang dibangun pada halaman awal. Penjelasan disajikan dengan menampilkan kondisi awal, masukan, keluaran, hasil yang dicapai, dan kondisi akhir.

Pada menu permainan yang akan diuji adalah fungsionalitas tombol yang terdapat di akhir permainan, yaitu tombol *Play*, *Main Menu*, dan *Exit*. Tampilan menu permainan dapat dilihat pada Gambar 4.9 dan Gambar 4.10. Skenario yang telah diuji terdapat pada Table 5.4

Table 5.4 Hasil Uji Coba Akhir Permainan

ID	**UF-001**
Nama	Uji Coba Pada Akhir Permainan
Tujuan uji coba	Pengguna mengetahui fungsionalitas tombol yang ada pada akhir permainan
Kondisi awal	Pemain berada pada halaman permainan dalam kondisi menang/kalah
Skenario 1	***Pemain memilih tombol Play***
Masukan	Menekan tombol *Play* pada dunia virtual

Keluaran yang diharapkan	Pemain berpindah ke halaman permainan
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain berada pada halaman permainan
Skenario 2	***Pemain memilih tombol Main Menu***
Masukan	Menekan tombol *Main Menu* pada dunia virtual
Keluaran yang diharapkan	Pemain berpindah ke halaman menu permainan
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain berpindah ke halaman menu permainan
Skenario 3	***Pemain memilih tombol Exit***
Masukan	Menekan tombol *Exit* pada dunia virtual
Keluaran yang diharapkan	Pemain berpindah keluar dari permainan
Hasil uji coba	Berhasil
Kondisi Akhir	Pemain berpindah keluar permainan

Hasil uji dari skenario 1 berpindah ke halaman permainan, setelah memenangkan ataupun kalah dari permainan, pemain dapat mengulangi permainan. Selanjutnya skenario 2 yaitu pemain berpindah ke halaman menu utama. Selanjutnya skenario 4 keluar dari permainan, pemain akan keluar dari permainan.

5.2.4. Hasil Uji Coba

Pada sub bab ini diberikan hasil evaluasi dari pengujian yang dilakukan pada permainan. Hasil evaluasi dapat dilihat pada Table 5.5

Table 5.5 Hasil Evaluasi

ID	Deskripsi	Kemungkinan / Skenario	Perilaku Terlaksana
UF-001	Uji Coba Menu Permainan	Skenario 1	Ya
		Skenario 2	Ya
		Skenario 3	Ya
		Skenario 4	Ya
UF-002	Uji Coba Permainan	Skenario 1	Ya
		Skenario 2	Ya
		Skenario 3	Ya
		Skenario 4	Ya
		Skenario 5	Ya
		Skenario 6	Ya
		Skenario 7	Ya
		Skenario 8	Ya
		Skenario 9	Ya
		Skenario 10	Ya
UF-003	Uji Coba Akhir Permainan	Skenario 1	Ya
		Skenario 2	Ya
		Skenario 3	Ya

5.3. Skenario Uji Coba

Pengujian pada permainan yang dibangun tidak hanya dilakukan pada fungsionalitas yang dimiliki, tetapi juga ditujukan kepada pengguna untuk mencoba secara langsung. Pengujian ini berfungsi sebagai pengujian subjektif yang bertujuan untuk mengetahui tingkat keberhasilan permainan yang dibangun dari sisi pengguna. Hal ini dapat dicapai dengan meminta penilaian dan tanggapan dari pengguna terhadap sejumlah aspek permainan yang ada.

5.3.1. Skenario Pengujian Pengguna

Dalam melakukan pengujian permainan, pengguna diminta mencoba memainkan permainan untuk mencoba semua fungsionalitas dan fitu yang ada. Pengujian permainan oleh

pengguna dilakukan dengan sebelumnya memberikan informasi seputar permainan, kegunaan, dan fitur-fitur yang dimiliki. Setelah informasi tersampaikan, pengguna kemudian diarahkan untuk langsung mencoba permainan dengan spesifikasi lingkungan yang sama dengan yang telah diuraikan pada Tabel 5.1 Lingkungan Uji Coba.

Jumlah pengguna yang terlibat dalam pengujian perangkat sebanyak 10 orang. Dalam melakukan pengujian pengguna melakukan percobaan lebih dari satu kali penggunaan untuk masing-masing pengguna.

Dalam memberikan penilaian dan tanggapan, pengguna diberikan kuesioner pengujian permainan. Kuesioner pengujian ini dilakukan secara online melalui kuisioner dengan tautan https://intip.in/KuisionerMagus dan hasilnya akan ditampilkan pada tautan https://intip.in/TanggapanKuisionerMagus . Kuesioner pengujian ini memiliki beberapa aspek penilaian seputar desain antarmuka, *immersivity*, dan tingkat kenyamanan permainan. Nilai yang diberikan rentang nilai 1 hingga 6 dengan rincian pada Table 5.6. pada bagian akhir terdapat saran untuk perbaikan fitur. Detil kuesioner pengguna dapat dilihat pada Tabel 5.7.

Table 5.6 Rentang Nilai

No	Keterangan	Nilai
1	Sangat Tidak Setuju (STS)	1
2	Tidak Setuju (TS)	2
3	Kurang Setuju (KS)	3
4	Cukup Setuju (CS)	4
5	Setuju (S)	5
6	Sangat Setuju (SS)	6

Table 5.7 Format Kuesioner

No	Parameter	STS	TS	KS	CS	S	SS
1	Permainan memiliki objek dan latar belakang yang sesuai	0	0	2	1	5	2
2	Permainan memiliki tampilan, warna, dan desain yang menarik	0	0	2	3	4	1
3	Permainan memiliki antarmuka yang mudah dilihat/dikenali	0	0	3	2	4	1
4	Saya merasa seperti penyihir	0	0	0	1	5	4
5	Saya merasakan sensasi menjadi penyihir	0	0	0	1	7	2
6	Permainan dapat berjalan lancar tanpa adanya lag dan atau crash	0	0	4	3	1	2
7	Saya merasa terbantu dengan adanya petunjuk yang ada	0	1	1	3	4	1
8	Saya merasa nyaman selama memainkan permainan ini	0	0	2	3	5	0
9	Saya merasa tertarik untuk memainkan permainan ini untuk selanjutnya	0	0	1	1	5	3

5.3.2. Daftar Penguji Permainan

Pada sub bab ini ditunjukkan daftar pengguna yang bertindak sebagai penguji coba permainan yang dibangun. Dalam pengujian ini tidak terdapat kriteria atau keahlian khusus yang harus dimiliki pengguna karena permainan ini ditunjukkan kepada

berbagai kalangan pengguna baik yang suka bermain permainan ataupun tidak. Daftar nama penguji permainan ini dapat dilihat pada Tabel 5.8.

Table 5.8 Daftar Penguji Permainan

No	Nama	Pekerjaan
1	Muhammad Alfi Maulana Fikri	Mahasiswa
2	Muhammad Roychan Meliaz	Mahasiswa
3	Fintanto Cendekia	Mahasiswa
4	Panji Rimawan	Mahasiswa
5	Aditya Gunawan	Mahasiswa
6	Romi Yehezkiel Purba	Mahasiswa
7	Rahmat Rijal	Mahasiswa
8	Aufar	Mahasiswa
9	Deni Ismail	Mahasiswa
10	Wira Mahardika	Mahasiswa

5.3.3. Hasil Pengujian Pengguna

Sistem penilaian didasarkan pada skala perhitungan satu sampai enam dimana skala satu menunjukkan nilai terendah dan skala enam menunjukkan skala tertinggi. Penilaian akhir kemudian dilakukan dengan menghitung berapa banyak penguji yang memilih suatu skala tertentu dan kemudian dicari nilai rata-ratanya. Hasil uji coba dipaparkan secara lengkap dengan disertai tabel yang dapat dilihat pada Tabel 5.9 dan Tabel 5.10.

Table 5.9 Hasil Pengujian Pengguna

No	Pernyataan	Penilaian						Rata-Rata
		1	2	3	4	5	6	
1	Permainan memiliki objek dan latar belakang yang sesuai	0	0	2	1	5	2	4.7
2	Permainan memiliki tampilan, warna, dan desain yang menarik	0	0	2	3	4	1	4.4
3	Permainan memiliki antarmuka yang mudah dilihat/dikenali	0	0	3	2	4	1	4.3
4	Saya merasa seperti penyihir	0	0	0	1	5	4	5.3
5	Saya merasakan sensasi menjadi penyihir	0	0	0	1	7	2	5.1
6	Permainan dapat berjalan lancar tanpa adanya lag dan atau crash	0	0	4	3	1	2	4.1
7	Saya merasa terbantu dengan adanya petunjuk yang ada	0	1	1	3	4	1	4.3
8	Saya merasa nyaman selama memainkan permainan ini	0	0	2	3	5	0	4.3
9	Saya merasa tertarik untuk memainkan permainan ini untuk selanjutnya	0	0	1	1	5	3	5

Table 5.10 Hasil Akhir Pengujian Pengguna

No	Pernyataan	Rata-Rata	Total	Total (%)
Parameter Antarmuka				
1	Permainan memiliki objek dan latar belakang yang sesuai	4.7	4.47	74.5%
2	Permainan memiliki tampilan, warna, dan desain yang menarik	4.4		
3	Permainan memiliki antarmuka yang mudah dilihat/dikenali	4.3		
Parameter Immersity				
4	Saya merasa seperti penyihir	5.3	5.2	86.6%
5	Saya merasakan sensasi menjadi penyihir	5.1		
Parameter Kenyamanan				
6	Permainan dapat berjalan lancar tanpa adanya lag dan atau crash	4.1	4.42	73.7%
7	Saya merasa terbantu dengan adanya petunjuk yang ada	4.3		
8	Saya merasa nyaman selama memainkan permainan ini	4.3		
9	Saya merasa tertarik untuk memainkan permainan ini untuk selanjutnya	5		

5.3.4. Kritik dan Saran Pengguna

Dalam memberikan penilain dan tanggapan, pengguna diberikan kuesioner pengujian permainan. Kuesioner pengujian permainan ini terdapat bagian kritik dan saran untuk perbaikan fitur

kedepannya. Kritik dan sara penggunan dapat dilihat pada Tabel 5.11.

Table 5.11 Kritik dan Saran Pengguna

No	Nama	Kritik dan Saran
1	Muhammad Alfi Maulana Fikri	Posisi tangan kurang pas dengan penglihatan sisanya udah bagus, tinggal poles aja
2	Muhammad Roychan Meliaz	Kalo bisa si perkenal fitur satu satu ga langsung bisa dipake semua. Tinggal polish
3	Fintanto Cendekia	Memberikan indikator pada saat how to play dan skill list diklik. Beberapa pohon bisa ditembus.
4	Panji Rimawan	Overall sudah bagus dan keren, meskipun masih ada bug-bug kecil di beberapa kondisi. Audio saat bermain kurang memberikan kesan seperti yang ada di permainan. Seharusnya diberikan audio / backsound yang menegangkan seperti yang ada di permainan. UI status item kurang terlihat, lebih baik ditempatkan agak ke atas agar terlihat. Begitu juga dengan loading icon ketika berganti dari permainan ke *main menu*.
5	Aditya gunawan	Metode maju atau mundur melelahkan pemain

		seharusnya ditambahkan maju atau mundur otomatis setelah maju beberapa detik.
6	Romi Yehezkiel Purba	Cara menangkap spelling lebih luas agar casting lebih mudah
7	Rahmat Rijal	aplikasi memiliki design enviroment yang menarik, dan cara menggunakan sihir juga mudah. tetapi kalau bisa untuk berhenti berjalan, sebaiknya menggunakan isyarat menggemgang tangan saja
8	Aufar	sudah mantab kok, keren cuy
9	Deni Ismail	Sensor Leap terkadang kurang presisi
10	Wira Mahardika	Pergerakan pemainya agak sulit

5.4. Evaluasi Pengujian

Sub bab ini membahas mengenai evaluasi terhadap pengujian-pengujian yang telah dilakukan. Dalam hal ini, evaluasi menunjukkan data rekapitulasi dari hasil pengujian fungsionalitas. Rekapitulasi disusun dalam bentuk tabel yang dapat dilihat pada Tabel 5.5. Dari data yang terdapat pada tabel tersebut, diketahui bahwa aplikasi yang dibuat telah berjalan sesuai dengan skenario yang diharapkan.

[Halaman ini sengaja dikosongkan]

BAB VI
KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diperoleh selama pengerjaan tugas akhir dan saran mengenai pengembangan yang dapat dilakukan terhadap tugas akhir ini di masa yang akan datang

6.1. Kesimpulan

Dari hasil pengamatan selama proses perancangan, implementasi, dan pengujian yang dilakukan, dapat diambil kesimpulan sebagai berikut.

1. Permainan yang dimainkan mempunyai tampilan *First Person Shooter* dalam lingkungan realitas virtual.
2. Permainan dapat mendeteksi gerakan tangan yang didapat dari Leap Motion untuk berinteraksi dengan objek dalam permainan.
3. Permainan berhasil dibuat dengan *Game Engine Unity.*

6.2. Saran

Berikut merupakan beberapa saran untuk pengembangan sistem di masa yang akan dating. Saran-saran ini didasarkan pada hasil perancangan, implementasi, dan pengujian yang telah dilakukan.

1. Menambahkan lebih banyak lagi sihir sihir agar menjadi lebih seru.
2. Menstabilkkan gerakan tangan serta obyek pada permainan agar terlihat lebih nyata.
3. Mengoptimalkan *Frame Per Second* dari permainan Magus ini, dikarenakan penyusunan *source* atau *code* yang digunakan dalam pengembangan permainan ini kurang bagus,

terdapat dampak yang signifikan yaitu turunnya *frame per second* dari permainan.

4. Menambahkan kontrol pergerakan menggunakan tangan lebih dari sekedar maju mundur.

DAFTAR PUSTAKA

[1] Unity, "Unity Survival Game," Unity, [Online]. Available: https://en.wikipedia.org/wiki/Survival_game. [Diakses 9 December 2017].

[2] K. G. D. Herlangga, "Virtual Reality dan Perkembangannya," CodePolitan, 7 Maret 2016. [Online]. Available: https://www.codepolitan.com/virtual-reality-dan-perkembangannya. [Diakses Desember 2017].

[3] Unity, "Unity," Unity, [Online]. Available: https://unity3d.com/unity. [Diakses Desember 2017].

[4] Wikipedia, "Wikipedia C Sharp," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/C_Sharp_(programming_language). [Diakses 9 December 2017].

[5] Wikipedia, "Wikipedia First Person Shooter," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/First-person_shooter. [Diakses 9 Desember 2017].

[6] P. S. M. Meenakshi Panwar, "IEEE Xplore Hand Gesture Recognition," 22 December 2011. [Online]. Available: https://ieeexplore.ieee.org/document/6108940/. [Diakses 9 December 2017].

[7] Leap Motion, "Leap Motion," Motion Control, [Online]. Available: https://www.leapmotion.com/. [Diakses 9 December 2017].

[8] Leap Motion, "SDK Leap Motion," Motion Company, [Online]. Available: https://developer-archive.leapmotion.com/documentation/csharp/devguide/Leap_SDK_Overview.html. [Diakses 1 January 2018].

[9] Oculus Rift, "Oculus Rift," Oculus, [Online]. Available: https://www.oculus.com/rift/. [Diakses 9 December 2017].

[Halaman ini sengaja dikosongkan]

LAMPIRAN

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS

Sultan Bonar Martinus, lahir di Bogor pada tanggal 27 Maret1996. Lulus dari SMAN 1 Jakarta pada tahun 2014 dan melanjutkan studinya di Departemen Informatika Institut Teknologi Sepuluh Nopember Surabaya.
Aktif mengikuti organisasi antara lain Staf Departemen Media Informasi Himpunan Mahasiswa Teknik Computer-Informatik (HMTC) 2015/2016, Staf ahli Departemen Media Informasi Himpunan Mahasiswa Teknik Computer-Informatik (HMTC) 2016/2017, Anggota Unit Kegiatan Mahasiswa (UKM) Paduan Suara mahasiswa (PSM) ITS 2014/2015.

Dalam menyelesaikan Pendidikan sarjana, penulis mengambil bidang minat Interaksi, Grafika dan Seni (IGS). Penulis dapat dihubungi melalui alamat *e-mail*: sultanbonar@gmail.com.