RANCANG BANGUN FIGHTING GAME WAYANG

SKRIPSI

Untuk memenuhi sebagian persyaratan untuk

Mencapai gelar Sarjana Komputer

Disusun Oleh :

Luthfi Aziz

0910680076

INFORMATIKA / ILMU KOMPUTER
PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2015

PENGESAHAN

RANCANG BANGUN FIGHTING GAME WAYANG

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
LUTHFI AZIZ
NIM: 0910680076

Skripsi ini telah diuji dan dinyatakan lulus pada
1 Oktober 2015
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Eriq M. Adams J, S.T, M.Kom	**Issa Arwani, S.Kom, M.Sc**
NIP: 19850410 201212 1 001	**NIP: 19830922 201212 1 003**

Mengetahui
Ketua Program Studi Informatika / Ilmu Komputer

Drs. Marji, M.T.
NIP: 19670801 199203 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 1 Oktober 2015

Luthfi Aziz

NIM: 0910680076

KATA PENGANTAR

Alhamdulillah,

Segala puji syukur penulis atas kehadirat Allah SWT dengan seluruh limpahan rahmat serta ridho-Nya, penulis dapat menyelesaikan skripsi ini. Dengan terselesaikannya skripsi ini tentu juga ada dukungan berupa semangat dan doa dari orang-orang yang berada di sekeliling penulis yang telah terlibat secara langsung maupun tidak langsung. Oleh karena itu penulis mengucapkan terima kasih yang tidak terhingga kepada:

1. Allah SWT.
2. Keluarga penulis, bapak Djamhur Hamid dan ibu Sri Nuring Wahyu sebagai orang tua yang telah merawat penulis dari kecil. Serta mas Affan sekeluarga dan mbak Ririn
3. Dosen Pembimbing Eriq M. Adams J., S.T., M.Kom. dan Issa Arwani, S.Kom, M.Sc. yang tanpa menyerah telah membimbing penulis selama ini. Serta tidak lupa mengucapkan terima kasih kepada seluruh dosen dan staff FILKOM dalam membantu penulis dalam menyelesaikan skripsi ini.
4. Teman teman seperjuangan angkatan 2009, geri, james, utek, ipul, gero, yuri, ayis, raka, dan teman teman lainnya yang telah berbagi keringat, rokok dan guyonan. Terimakasih atas semangat dan tawanya.
5. ATS yang selalu mengajak main Dota. Terimakasih atas *brotherhood*-nya.
6. Pevita.

Semoga Allah SWT membalas kebaikan dan ketulusan semua pihak yang telah membantu menyelesaikan skripsi ini dengan melimpahkan rahmat dan karunia-Nya dan semoga karya penelitian tugas akhir ini dapat memberikan manfaat dan kebaikan bagi banyak pihak demi kemaslahatan bersama serta bernilai ibadah di hadapan Allah SWT. Amin.

Malang, Oktober 2015

Penulis

ABSTRAKSI

Luthfi Aziz, 2015. Rancang Bangun Fighting Game Wayang. Skripsi Program Studi Teknik Informatika. Program Teknologi Informasi dan Ilmu Komputer. Universitas Brawijaya. Malang. Pembimbing: Eriq Muhammad Adams J., S.T., M.Kom. dan Issa Arwani, S.Kom, M.Sc.

Salah satu faktor penting dalam pembangunan karakter suatu bangsa adalah kuatnya budaya bangsa tersebut. Untuk itulah upaya dalam melestarikan dan memperkokoh budaya nasional menjadi suatu agenda yang sangat penting untuk dilakukan. Namun dengan pengaruh dari budaya luar yang sangat kuat, pelestarian budaya ini mengalami hambatan yang besar dalam upayanya untuk memperkenalkan kembali budaya lokal pada kaum muda. Konsep *mukokuseki* yang *berarti "nationlessness*" atau tanpa bangsa dan ras, yang dalam implementasinya berupa karakter imajinatif yang tidak berafiliasi atau berdasar pada suatu bangsa maupun ras tertentu. Konsep ini cocok untuk diterapkan pada game yang bertujuan agar karakter tidak terlalu *segmented* terhadap budaya yang bisa menghalangi game tersebut untuk dapat diterima oleh masyarakat. Game dirancang untuk mengenalkan kembali budaya asli Indonesia khususnya jawa yaitu wayang, pada masyarakat umum Indonesia yang telah bayak terpengaruh oleh budaya dari luar. Berdasarkan hasil kuisioner yang disebar pada *player* yang telah memainkan game ini, dengan *gameplay fighting* yang sudah umum *game* ini dinilai mudah dan menyenangkan untuk dimainkan. Penggunaan karakter wayang dengan jurusnya yang unik membuat game ini dapat menambah ketertarikan pengguna untuk mengenal tokoh tokoh wayang secara lebih lanjut.

Kata kunci : *mukokuseki*, *game*, wayang

ABSTRACT

Luthfi Aziz, 2015. Design and Build Game Fighting Game Wayang Undergraduate Thesis of Informatics Engineering Study Program, Information Technology and Computer Science Program. Brawijaya University. Malang. Pembimbing: Eriq Muhammad Adams J., S.T., M.Kom. dan Issa Arwani, S.Kom, M.Sc.

One important factor in national character development is the strength of the nation's culture. The effort to preserve and strengthen the national culture become a very important agenda to be done. But because the strong influence of foreign culture, the preservation of this cultural experience meets significant barriers in its efforts to reintroduce the local culture on youth. Mukokuseki means "nationlessness" or without a nation and race, which is in its implementation is the form of imaginative characters that are not affiliated with a particular nation or race. This concept aims to make a game character design that are not too segmented to a culture that could hinder the game to be accepted by society. This concept is suitable to be applied on a game that aims to reintroduce wayang, a local culture of Indonesia, to the Indonesian people that have been affected by many foreign culture. Based on the results of a questionnaire that was distributed to players who have played this game, with the familiar fighting gameplay this game is easy to learn and fun to play. The use of wayang characters and unique skill makes this game regrow the user's interest to learn more about wayang characters.

Keyword : mukokuseki, game, wayang

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR ISTILAH

Api	Elemen dalam *game* yang berfungsi sebagai target utama untuk dikumpulkan oleh pemain untuk dapat memenangkan permainan.
Area Hit	Jenis pukulan maupun jurus dari suatu karakter yang dapat memberikan efek pada suatu area dengan luas tertentu.
Mana	Jumlah energi atau kekuatan yang dimiliki oleh suatu karakter yang digunakan untuk dapat mengeluarkan jurus dari karakter dengan jumlah mana tertentu yang harus dikeluarkan.
Melee	Jenis pukulan jarak dekat yang dapat dikeluarkan oleh tiap karakter.
Power Up	Penambah kekuatan spesial yang dapat menambah jumlah mana pada suatu karakter secara instan.
Skill	Jurus atau gerakan spesial yang dimiliki oleh tiap karakter.
Summon	Tipe jurus atau gerakan yang mempunyai kekuatan dapat memanggil atau memunculkan pasukan secara instan pada stage.
Spawn	Menampilkan suatu elemen permainan dalam stage.

BAB I
PENDAHULUAN

1.1 Latar Belakang

Salah satu faktor penting dalam pembangunan karakter suatu bangsa adalah kuatnya budaya bangsa tersebut. Budaya jugalah yang merepresentasikan kepribadian suatu bangsa, entah dalam segi sosial (kehidupan bermasyarakat) maupun segi estetika (seni). Untuk itulah upaya dalam melestarikan dan memperkokoh budaya nasional menjadi suatu agenda yang sangat peting untuk dijalankan oleh Indonesia. Namun ada hal yang perlu diingat dalam upaya melestarikan budaya ini, seperti apa yang pernah Mahatma Gandhi katakan, bahwa tidak ada satupun budaya yang dapat bertahan, jika budaya tersebut berusaha menjadi suatu hal yang eksklusif [PRR-88]. Karena dunia sekarang terus tumbuh dan berkembang dengan sangat cepat, budaya pun harus menyesuaikan, salah satunya dengan akulturasi budaya. Sementara dari segi budaya, wayang merupakan seni budaya yang akan diangkat dalam game ini. Wayang adalah seni budaya asli Indonesia, seperti yang telah dicanangkan oleh UNESCO pada tahun 2003. Wayang merupakan seni pertunjukan yang menggunakan bayangan dari boneka yang terbuat dari kulit atau kayu yang dimainkan oleh seorang dalang untuk menceritakan suatu kisah. Secara umum kisah yang biasa dimainkan oleh wayang adalah kisah kisah ramayana atau mahabarata [RZM-12].

Selain itu seni wayang kulit Indonesia kini menghadapi problem yang serius. Bukan terkait jumlah dalang, tapi jumlah penonton kian lama kian menyusut. Seperti yang pernah dikatakan oleh Ketua Sekretariat Nasional Pewayangan Indonesia (Sena Wangi) Suparmin Sunjoyo, dimana jumlah dalang banyak dengan jurusan pedalangan yang tersebar di perguruan tinggi di Indonesia, tidak diimbangi dengan jumlah penonton yang sekarang semakin sedikit. Dikatakan, saat ini 80 persen penonton wayang berusia di atas 50 tahun. Untuk itu, pihak Sena Wangi telah mengusulkan untuk memasukan wayang menjadi bagian kurikulum di pelajaran sekolah. Namun sayangnya sampai sekarang belum ada respon. Menurutnya wayang perlu masuk kurikulum karena akan menjadi kewajiban dalam

pelestariannya. Sementara itu sekitar 75 jenis wayang yang menjadi kekayaan budaya Indonesia kini telah punah. Hanya sekitar 25 jenis wayang yang saat ini masih bertahan dengan jumlah komunitas dan penonton cukup banyak. Semestinya, dengan diakuinya wayang oleh Organisasi Pendidikan, Ilmu Pengetahuan, dan Kebudayaan PBB (UNESCO) sebagai mahakarya dunia yang tak ternilai dalam seni bertutur (Masterpiece of Oral and Intangible Heritage of Humanity) pada 2003, wayang bisa lebih berkembang di Tanah Air. Kenyataannya, pemerintah belum memiliki arah dan strategi yang jelas dalam pengembangan wayang. Selain kurangnya perhatian pemerintah, perkembangan zaman telah membawa perubahan kebudayaan dan peradaban sehingga wayang yang merupakan kesenian tradisional semakin ditinggalkan. Tak heran beberapa jenis wayang punah dan tak bisa lagi ditonton masyarakat, seperti wayang suket, wayang klitik, wayang krucil, wayang gedog, dan wayang beber. Adapun wayang yang masih digemari masyarakat sehingga masih cukup eksis antara lain wayang kulit purwa Jawa dengan berbagai gaya, baik Surakarta, Yogyakarta, Jawa Timuran, Banyumasan, Cirebonan, maupun Betawi. Begitu pula wayang golek Sunda, wayang Bali, dan wayang sasak Lombok masih banyak penggemarnya. Meski penggemar wayang menurun, menurut Ketua Umum Persatuan Pedalangan Indonesia (Pepadi) Ekotjipto, animo masyarakat untuk terjun ke dunia pedalangan cukup tinggi. Ini ditunjukkan dengan banyaknya peserta pada setiap lomba pencarian bibit dalang yang digelar Pepadi. Upaya yang dapat dilakukan agar wayang terhindar dari kepunahan antara lain dengan memasukkan wayang dalam pendidikan formal. Selain itu, juga memasukkan wayang dalam perangkat komunikasi modern sehingga mudah dijangkau anak-anak atau generasi muda. Saat ini terdapat 15.000 seniman pedalangan yang masih eksis. Sementara jumlah dalang di seluruh Indonesia tercatat 6.000 orang. Berbagai permasalahan itulah yang membuat wayang harus segera dilestarikan keberadaannya [HTP-14].

Dalam hal penyebaran konten, saat ini game telah menjadi kekuatan media masa baru mendampingi televisi, internet dan media cetak sebagai para pendahulunya. Sebagaimana di Amerika, Barrack Obama telah menggunakan media game sebagai sarana kampenya untuk mendulang suara pada saat pemilihan presiden [GMM-11]. Game juga mulai disadari oleh kaum akademisi sebagai media

atau sarana yang tepat untuk pembelajaran dalam kelas. Video game membuat proses belajar menjadi lebih relevan dan menarik. Mereka menempatkan gamer ke dalam suatu ruang masalah, dan gamer dituntut untuk dapat memecahkannya. Oleh karena itu game menjadi salah satu media yang dianggap pantas untuk menjalankan misi pengenalan budaya [GEB-13]. Lalu yang menjadi masalah adalah bagaimana sebuah aplikasi game dengan konten budaya, dapat diterima dengan baik oleh masyarakat umum yang telah banyak bersentuhan dengan berbagai budaya yang masuk di Indonesia

Mengenai hal ini, Jepang dalam pembuatan *karakter ataupun environment* dalam sebuah game mempunyai konsep *mukokuseki yang berarti "nationlessness*" atau tanpa bangsa dan ras, yang berarti karakter imajinatif yang tidak berafiliasi atau berdasar pada suatu bangsa maupun ras tertentu. Konsep ini bertujuan agar karakter pada game tidak terlalu *segmented* terhadap budaya yang bisa menghalangi game tersebut untuk dapat diterima oleh masyarakat [CRS-04]. Maka dari itu sisi budaya yang akan dikenalkan lebih secara implisit muncul dalam design lokasi, obstacle dan juga kostum. Gameplay juga akan menggunakan sistem atau cara bermain yang telah umum di masyarakat seperti genre *fighting.*

1.2 Rumusan Masalah

Berdasarkan uraian latar belakang diatas maka dapat diambil rumusan masalah sebagai berikut:

1. Bagaimana perancangan *gameplay* dan karakter dari *game fighting multiplayer* berbasis karakter wayang?
2. Bagaimana implementasi gameplay dari *game fighting multiplayer* berbasis karakter wayang?
3. Bagaimana pengujian *game fighting multiplayer* berbasis karakter wayang?

1.3 Batasan Masalah

Agar pembahasan tidak meluas dari pokok permasalahan, maka penelitian ini disusun dengan batasan sebagai berikut:

1. Karakter wayang yang digunakan menganut pada tokoh yang terdapat dalam kisah Mahabharata dan Ramayana versi jawa.

2. *Design* dan *gameplay* permainan dibangun pada *platform* 2D.
3. *Game* ini ditujukan untuk platform *desktop* PC dengan menggunakan *controller* dari Xbox 360.

1.4 Tujuan

Merancang dan mengimplementasi multiplayer fighting game yang dibangun dengan berdasarkan karakter wayang menggunakan game engine Unity pada platform dekstop PC.

1.5 Manfaat

1. Bagi penulis :
   a. Mengaplikasikan ilmu yang didapat selama mengikuti perkuliahan di Teknik Informatika Universitas Brawijaya khususnya dalam segi merancang sebuah aplikasi permainan atau game.
   b. Ikut melestarikan budaya Indonesia khususnya budaya Jawa.
2. Bagi pengguna :
   a. Menambah pengetahuan terhadap karakter wayang.
   b. Melestarikan budaya wayang dengan cara memperkenalkannya secara interaksi sosial memanfaatkan gameplay multiplayer

1.6 Sistematika Penulisan

Sistematika penulisan dalam skripsi ini sebagai berikut:

BAB I Pendahuluan

Memuat latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat dan sistematika penulisan.

BAB II Dasar Teori

Memaparkan teori dasar dan teori pendukung yang berhubungan dengan pembuatan *game* menggunakan *engine* Unity, dan penjelasan mengenai pengenalan budaya wayang.

BAB III Metode Penelitian dan Perancangan

Berisi tentang langkah-langkah dalam merancang pembuatan *permainan*. Serta membahas analisis kebutuhan dan perancangan yang sesuai dengan teori yang ada dalam membangun permainan

agar sesuai dengan keinginan penulis serta bermanfaat bagi pengguna.

BAB IV Implementasi

Membahas tentang implementasi dari permainan yang dibangun. Meliputi pembuatan design karakter, stage, *gameplay*, dan audio dengan menggunakan software-software yang dibutuhkan, serta pemrograman game yang menggunakan bahasa pemrograman C #.

BAB V Pengujian dan Analisis

Memuat hasil pengujian dan analisis terhadap *game* yang telah dibangun.

BAB VI Kesimpulan dan Saran

Berisi kesimpulan yang didapat dari proses pembuatan dan pengujian *game* serta saran untuk pengembangan dan penyempurnaan *game* lebih lanjut.

BAB II
DASAR TEORI

2.1 Permainan

Menurut Jesper Juul game adalah suatu sistem formal berbasis aturan dengan hasil yang kuantitatif atau terukur, di mana pada hasil yang berbeda ditetapkan nilai yang berbeda pula, pemain diberikan upaya atau kuasa untuk mempengaruhi hasil, hingga pemain merasa dekat dengan hasil atau pencapaian yang telah diraihnya [JPR-03].

Video game sendiri mulai berkembang semenjak game Pong yang dirilis oleh Atari pada tahun 1972 mengalami kesuksesan dan menyebabkan platform video game mulai populer. Perkembangan video game dimulai dengan maraknya *arcade game* yang mencapai jaman keemasannya mulai jaman akhir 1970-an hingga pertengahan jaman 1990-an yang lalu kemudian diikuti dengan merebaknya konsol *video game* seperti Nintendo, SEGA dan Playstation. Hingga kini *video game* telah berkembang cepat dengan masuknya teknologi *Kinect* yang membuat konsol dapat mendeteksi gerakan player secara *realtime*, dan ada pula Oculus Rift *virtual reality headset* yang membuat pemain seolah-olah masuk dalam dunia game tersebut [MSP-NN].

Pengklasifikasian *genre* dalam video game ada banyak macamnya. Beberapa kategorinya antara lain ditunjukkan pada tabel 2.1 berikut ini:

Basic Genres	Action, Adventure, Educational, Racing / Driving, Role-Playing (RPG), Simulation, Sports, Strategy
Perspectives and Viewpoints	1st-Person, 3rd-Person, Isometric, Platform, Side-Scrolling, Top-Down
Sports Themes	Baseball, Basketball, Bike / Bicycling, Bowling, Boxing, Cricket, etc. (29 total)
Non-Sports Themes	Adult, Anime/Manga, Arcade, BattleMech, Board / Party Game, Cards, Casino, Chess, Comics, Cyberpunk / Dark Sci-Fi, Detective / Mystery, Fighting, Flight, Game Show,

	Helicopter, Historical Battle (specific/exact), Horror, Interactive Fiction, etc. (42 total)
Educational Categories	Ecology / Nature, Foreign Language, Geography, Graphics / Art, Health / Nutrition, etc. (14 total)
Other Attributes	Add-on, Coin-Op Conversion, Compilation / Shovelware, Editor / Constructor Set, Emulator, Licensed Title

Tabel 2.1 Klasifikasi Genre Game

Basic genre adalah pengkategorian permainan berdasarkan *gameplay* yang dikembangkan dalam permainan tersebut, seperti Tekken yang bisa dikategorikan ber-*genre Fighting* ataupun Need for Speed *series* yang ber-*genre racing*. Sedangkan *Perspective* dan *Viewpoint* adalah berdasarkan pada sudut pandang pemain saat memainkan game tersebut, seperti Counter Strike *series* yang memakai sudut pandang FPS (*First Person Shooter*) ataupun GTA (*Grand Theft Auto*) series yang memakai sudut pandang *Third Person Shooter* [GGS-NN].

2.2 Fighting Games

Fighting games adalah bentuk aksi dalam game dimana dua karakter dalam layar melakukan pertarungan 1 lawan 1. *Fighting games* biasanya menampilkan perkelahian tanpa senjata seperti tinju atau seni beladiri lain, namun bisa juga menggunakan senjata seperti pedang ataupun pistol. Pemain diberikan kontrol terhadap karakter dan dapat melakukan pertarung jarak dekat dengan lawan. *Game fighting* yang pertama dirilis adalah “Heavyweight Champ” yang dibuat pada tahun 1976.

Karakter di game *fighting* seringkali memiliki gerakan beladiri super dan tenaga yang ekstrem. *Gameplay* dari *game fighting* biasanya terdiri dari beberapa ronde pada tiap pertadingan yang dimainkan. Agar efektif dalam game fighting, player seringkali harus menguasai teknik seperti kombo, tangkis, dan serangan balik.

Beberapa *game fighting* populer diantaranya adalah:

- Capcom's "Street Fighter"
- Midway Games' "Mortal Kombat"

- Namco's "Soulcalibur"

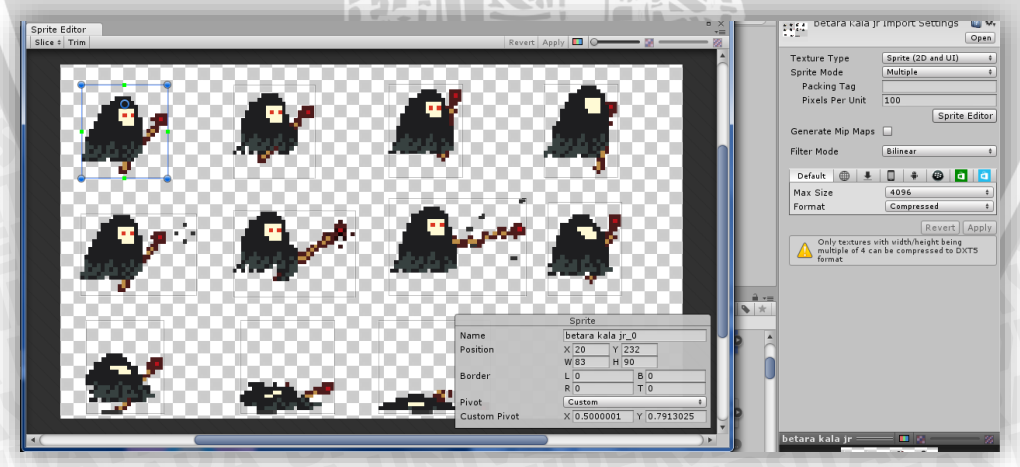
2.3 Unity

Unity merupakan salah satu *game engine* yang banyak dipakai oleh para game developer selain UDK dan CryEngine. Community support dan persyaratan lisensi yang baik, membuat Unity cocok untuk startup ataupun para game developer baru [GCM-15]. Beberapa game terkenal yang menggunakan Unity sebagai *game engine*-nya antara lain Kerbal Space Program, Besiege, dan Hearthstone: Heroes of Warcraft.

Kelebihan lain dari Unity adalah kemampuannya dalam pengembangan permainan pada banyak platform. Beberapa platform yang di-*support* oleh Unity antara lain BlackBerry 10, Windows Phone 8, Windows, OS X, Linux , Android, iOS, Unity Web Player , Adobe Flash, PlayStation 3, PlayStation 4, PlayStation Vita, Xbox 360, Xbox One, Wii U, dan juga Wii.

Mulai versi 4.3, Unity juga telah menambah fiturnya untuk pengembangan permainan dalam platform 2D, dengan mengeluarkan fitur seperti 2D *rendering* dan juga 2D *physics engine*.

Dengan menggunakan 2D *rendering*, pembuat *game* dapat menggunakan *sprite editor* untuk membuat animasi *frame to frame* sesuai dengan *spritesheet* yang ada. Tampilan *sprite editor* pada unity 4.3 dapat dilihat pada Gambar 2.1 berikut:



Gambar 2.1 Tools Sprite Editor Pada Unity 4.3

Sementara itu 2D *physics engine* menyediakan fitur-fitur yang berkaitan dengan *physics* dari sebuah *game*, fitur tersebut antara lain *box collider 2D, rigid*

body 2D, dan *physics material 2D*. Bahasa pemrograman yang dipakai dalam Unity adalah Javascript, C#, dan boo. Sisi skrip pada Unity dibangun dengan mengacu pada Mono, yaitu implementasi *open-source* dari .NET Framework [UNI-NN].

2.4 Wayang

Wayang adalah literatur seni pertunjukan asli Indonesia yang menuai perkembangan pesat di pulau Jawa dan Bali. Bahkan sebagian daerah seperti Sumatra dan Semenanjung Malaya juga memiliki budaya wayang. Semua itu, terpengaruh oleh kebudayaan Jawa dan Hindu.

Di Indonesia pada tanggal 7 November 2003, UNESCO sebagai lembaga yang membawahi kebudayaan dari PBB menetapkan wayang sebagai pertunjukan seni tersohor dari Indonesia. Seni Wayang banyak memuat ajaran moral budi pekerti serta wahana bercerita dan memberikan berbagai karakter dan tokoh dalam pementasannya. Mengisahkan nilai kehidupan dalam pewayangan penuh dengan pesan budi luhur, kesetiaan, pengabdian, kebajikan, pengorbanan, perjuangan, dan kejujuran [WYG-10].

Pertunjukan seni wayang di Jawa kebanyakan membawakan kisah tentang Mahabharata dan Ramayana yaitu karya sastra kuno terkenal yang berasal dari India. Namun di Indonesia khususnya di jawa kisah-kisah ini telah mengalami akulturasi dengan budaya jawa, sehingga terdapat beberapa kisah yang berbeda dengan naskah aslinya [JWA-NN]. Terdapat juga tokoh-tokoh wayang asli Indonesia diantranya yaitu tokoh semar, gareng, petruk, dan bagong tokoh punakawan yang dalam cerita Mahabharata versi jawa menjadi *gag character* pada tiap pementasan walau kadang juga dapat berperan sebagai penasehat para kesatria [KWG-07].

2.5 Mukokuseki

Konsep *mukokuseki* adalah salah satu konsep yang banyak digunakan oleh produk budaya populer yang berasal dari Jepang. Istilah *mukokuseki* (*無国籍*) secara harfiah bearti "*statelessness*" atau "*nationlessness*", konsep ini digunakan oleh produsen produk budaya popular Jepang untuk membuat produk-produk tersebut diterima dan dapat dipasarkan di seluruh dunia. Konsep *mukokuseki* salah satunya ini dapat dilihat dari design karakter dengan ras yang membingungkan pada

anime-anime Jepang, dimana suatu karakter mempunyai mata lebar, warna kulit terang, tubuh bagus, dan berbagai macam warna rambut.

Gambar 2.2 Konsep Mukokuseki Pada Anime Jepang

Gambar 2.2 menunjukkan contoh konsep *mukokuseki* dalam implementasinya pada karakter *anime*. *Mukokuseki* atau "*culturally odourless*" dalam konteks ini secara metafora menjelaskan tentang karakteristik yang harusnya otentik dari sebuah produk budaya Jepang justru tidak secara ekplisit ditonjolkan, melainkan tersembunyi dan diturunkan ke peran sekunder. Konsep ini berkebalikan dengan ide "*fragant*" dari kreasi budaya, yaitu dimana konten dari unsur budaya asli lebih menonjol sehinga lebih mudah diidentifikasi [MSK-NN].

2.6 Pengujian

Dalam pengembangan permainan ini proses pengujian yang akan dilakukan adalah *Black Box Testing*, *White Box Testing, dan Play Testing*.

2.6.1 Black Box Testing

Black-box testing adalah metode pengujian perangkat lunak yang menguji sisi fungsionalitas dari aplikasi. Dalam melakukan pengujian blackbox, pengetahuan khusus tentang kode aplikasi / struktur program serta pengetahuan

tentang pemrograman pada umumnya tidak diperlukan. Uji kasus dibangun dalam spesifikasi dan persyaratan tentang bagaimana aplikasi yang diuji seharusnya berjalan. Pengujian ini dapat menjadi fungsional atau non-fungsional, meskipun biasanya fungsional. Perancang uji memilih input yang valid dan tidak valid dan menentukan output yang benar.

Metode ujicoba *black box* memfokuskan pada keperluan fungsional dari software. Karena itu ujicoba *black box* memungkinkan pengembang software untuk membuat daftar kondisi input yang akan menguji seluruh syarat-syarat fungsional suatu program. Ujicoba *black box* bukan merupakan alternatif dari ujicoba *white box*, tetapi merupakan pendekatan yang melengkapi untuk menemukan kesalahan lainnya, selain menggunakan metode *white box* [BWT-13].

Ujicoba *black box* berusaha untuk menemukan kesalahan dalam beberapa kategori, diantaranya:

1. Fungsi-fungsi yang salah atau hilang
2. Kesalahan interface
3. Kesalahan dalam struktur data atau akses database eksternal
4. Kesalahan performa
5. Kesalahan inisialisasi dan terminasi

2.6.1.1 Combinatorial Testing

Tester dan *projek manager* sering dihadapkan dengan masalah apakah pengujian yang tengah dilakukan terlalu sedikit, terlalu banyak, atau sudah tepat. Kualitas dari *game* memang harus cukup baik untuk dapat dimainkan konsumen, namun pengujian tidak bisa berlangsung selamanya jika *deadline* atau tanggal rilis *game* sudah sangat dekat. Mencoba untuk menguji setiap kemungkinan kombinasi acara permainan, konfigurasi, fungsi, dan opsi *setting* lainnya akan sangat tidak praktis dan tidak ekonomis dalam situasi seperti ini. Mengambil jalan pintas atau melewatkan beberapa pengujian adalah cara yang berisiko.

Pengujian kombinasi berpasangan adalah cara untuk menemukan cacat dalam perangkat lunak permainan dengan tetap menjaga tes set relatif kecil dibandingkan dengan fungsionalitas yang dicakup tiap modul tes. Kombinasi berpasangan berarti bahwa setiap nilai yang anda gunakan untuk pengujian perlu

dikombinasikan setidaknya sekali dengan setiap nilai lain dari parameter yang tersisa.

Parameter adalah suatu elemen individu dari permainan yang akan disertakan dalam tes kombinatorial. Anda dapat menemukan parameter uji dengan melihat berbagai jenis elemen permainan, fungsi, dan pilihan seperti:

1. Game events
2. Game settings
3. Gameplay options
4. Hardware configurations
5. Character attributes
6. Customization choices

Pengujian bisa dilakukan secara homogen, yang dirancang untuk menguji kombinasi parameter dari jenis yang sama, atau heterogen, yang dirancang untuk menguji lebih dari satu jenis parameter di modul yang sama.

Misalnya, pilihan parameter pengujian dari *game options screen* untuk efeknya pada gameplay dilakukan dengan pengujian kombinatorial homogen. Sedangkan jika parameter pengujian dipilihdari berbagai menu untuk memilih karakter yang berbeda, *item*, dan pilihan untuk memainkan untuk misi tertentu, maka itu dilakukan dalam pengujian heterogen [CBT-NN].

2.6.2 White Box Testing

White Box Testing merupakan cara pengujian dengan melihat ke dalam modul untuk meneliti kode-kode program yang ada, dan menganalisis apakah ada kesalahan atau tidak. Jika ada modul yang menghasilkan output yang tidak sesuai dengan proses bisnis yang dilakukan, maka baris-baris program, variabel, dan parameter yang terlibat pada unit tersebut akan dicek satu persatu dan diperbaiki, kemudian di-*compile* ulang [WLN-13].

2.6.2.1 Basis Path Testing

Basis Path Testing memungkinkan perancang *test case* mendapatkan ukuran logis dari perancangan prosedural dan menggunakan ukuran ini sebagai petunjuk untuk mendefinisikan himpunan jalur yang akan diuji. Basis Path menggunakan notasi graf atau *flow graph* untuk menggambarkan aliran kontrolnya.

Lingkaran (node), menggambarkan satu atau lebih perintah prosedural. Urutan proses dan keputusan dapat dipetakan dalam satu node. Tanda panah (*edge*), menggambarkan aliran kontrol. Setiap *node* harus mempunyai tujuan *node*. Region adalah daerah yang dibatasi oleh *edge* dan *node*. Termasuk daerah diluar grafik alir [BPT-NN].

2.6.2.2 Cyclomatic Complexity

Cyclomatic Complexity merupakan suatu sistem pengukuran yang menyediakan ukuran kuantitatif dari kompleksitas logika suatu program. Pada Basis Path Testing, hasil dari cyclomatic complexity digunakan untuk menentukan banyaknya independent paths. Independent path adalah sebuah kondisi pada program yang menghubungkan node awal dengan node akhir [BPT-NN].

Terdapat 2 persamaan yang digunakan, yaitu:

$$V(G) = E - N + 2 \text{ atau } V(G) = P + 1$$

Keterangan:

V (G) = cyclomatic complexity untuk flow graph G

E=Jumlah Edge (panah)

N=Jumlah Node (lingkaran)

P=Jumlah Predicate Node

2.6.3 Play Testing

Play testing adalah jenis tes berbeda dari jenis pengujian game yang lain. Jenis pengujian game lain biasanya dilakukan dengan pertanyaan utama yaitu: Apakah permainan ini bekerja sebagaimana mestinya. Play testing dilakukan dengan pertanyaan yang berbeda tapi bisa dibilang lebih penting, yaitu: Apakah permainan ini bekerja dengan baik. Perbedaan antara kedua pertanyaan ini adalah jelas. Kata "baik" menyiratkan banyak sekali dalam empat huruf kecil. Karena hal ini dapat menyebabkan banyak pertanyaan lainnya: Apakah permainan terlalu mudah? Apakah permainan terlalu sulit? Apakah permainan mudah untuk dipelajari? Apakah antarmuka sudah jelas dan mudah untuk dinavigasikan? Dan pertanyaan yang paling penting dari semua: Apakah permainan ini menyenangkan? Berbeda dengan jenis lain dari pengujian game, Play Testing sendiri mempunyai hasil berupa judgement terhadap game, bukan fakta [PLT-08].

2.7 Adobe Photoshop

Adobe Photoshop adalah perangkat lunak atau software yang digunakan untuk mengedit foto atau citra yang dibuat oleh perusahaan Adobe Systems. Perangkat ini pertama kali diperkenalkan oleh Adobe pada tahun 1989 yang merupakan program grafis untuk sistem operasi Macintosh dari Apple. Aplikasi ini diciptakan oleh Thomas dan John Knoll, kakak beradik lulusan University of Michigan. Adobe membeli lisensi pada September 1988 atau awal musim gugur padahal sofware ini baru dibuat oleh Knoll bersaudara 3 bulan sebelumnya [PSP-12].

2.8 Iterative Rapid Prototyping

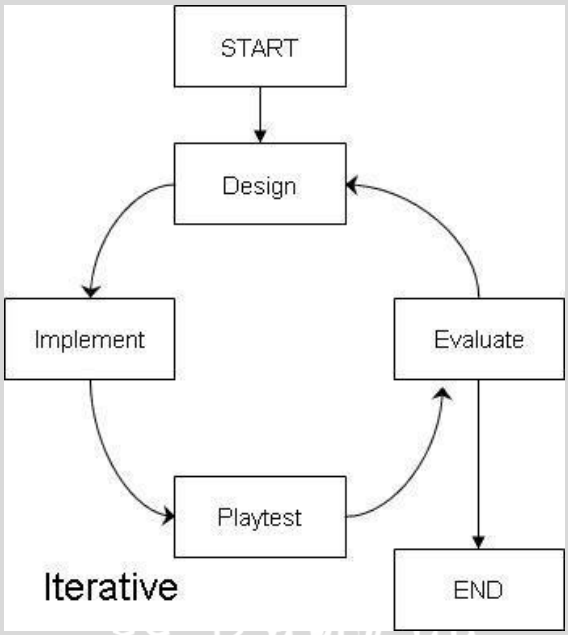
Iterative rapid prototyping merupakan proses mendesain *game* yang dimulai dari mencari ide *game*, membuat *prototype* dimana *prototype* dibagi menjadi 2 yakni *paper prototype* dan *digital prototype*, mengimplementasi *prototype* dan mengevaluasinya. Proses-proses tersebut akan dilakukan berulang-ulang hingga mendapatkan hasil *game* yang diinginkan.

Secara historis, metodologi desain terkenal pertama ialah metode waterfall. Metode ini dilakukan dengan cara mendefinisikan seluruh elemen permainan dalam bentuk rancangan, lalu mengimplementasikannya (menggunakan pemrograman dalam video game, atau menciptakan papan dan potongan untuk permainan non-digital), kemudian mengujinya untuk memastikan aturan dalam permainan bekerja dengan baik, dan menyempurnakannya dengan tampilan grafis yang bagus dan menarik, baru kemudian mem-*publish*-nya.

Metode ini dinamakan *waterfall* karena seperti layaknya aliran air pada air terjun, metode ini hanya bisa bergerak dalam satu arah. Jika *developer* sedang sibuk membuat seni grafis pada permainan namun menyadari ada aturan dalam permainan yang harus diubah, maka metedologi ini tidak memungkinkan anda untuk kembali pada fase perancangan *gameplay* permainan, sehingga sulit untuk dilakukan perubahan pada permainan saat proses *development* tengah berlangsung.

Lalu munculah ide untuk setidaknya dalam sebuah metedologi design memiliki pilihan untuk kembali dan memperbaiki beberapa hal dalam langkah-langkah sebelumnya, kemudian terciptalah apa yang dikenal sebagai pendekatan iteratif. Metode ini berjalan mirip dengan metode *waterfall*, dimana *developer*

pertama kali merancang permainan, kemudian menerapkannya, dan kemudian memastikan permainan tersebut bekerja. Namun setelah fase ini, ditambahkan langkah lain untuk mengevaluasi permainan. Dengan memainkannya, dapat diputuskan apa yang baik dan apa yang perlu diubah. Jika *developer* menilai permainan yang telah dibuat sudah memenuhi kriteria yang ingin mereka capai maka proses *development* selesai. Namun jika teridentifikasi beberapa elemen yang memerlukan perubahan, maka *developer* akan kembali ke langkah desain, lalu kemudian menemukan cara untuk mengatasi masalah-masalah yang telah diidentifikasi, meng-*implementasi* perubahan tersebut, dan kemudian mengevaluasi lagi. Proses ini dilakukan terus menerus hingga permainan yang dibuat siap untuk di-*publish*. Proses tersebut dapat dilihat pada gambar 2.3 berikut :



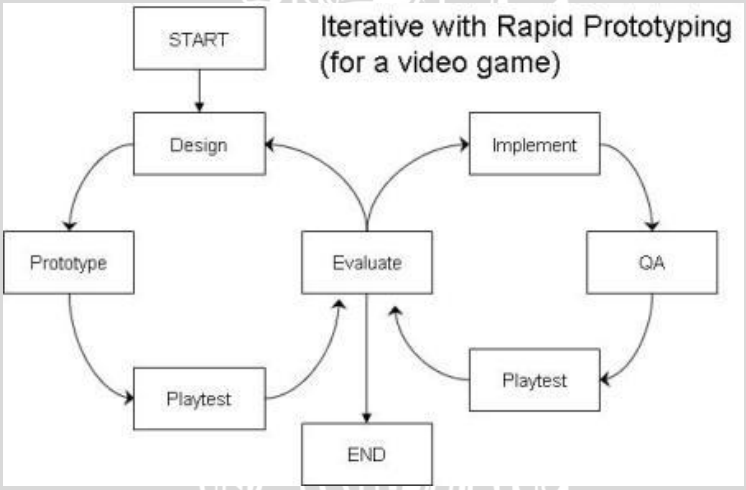
Gambar 2.3 Iterative cycle

Metode ini mirip dengan metode yang sudah ada yaitu *Scientific method*:

- Membuat observasi.
- Membuat hipotesis.
- Membuat percobaan untuk membuktikan atau menyangkal hipotesis.
- Lakukan percobaan.
- Menginterpretasikan hasil percobaan, membentuk satu set baru pengamatan. Kembali ke langkah pertama.

Dengan media non-digital (kartu dan papan), proses ini dapat bekerja dengan baik, karena pengujian bisa dilakukan dengan cepat. Pengujian dengan digital, mengalami satu masalah yaitu implementasi (pemrograman dan debugging), merupakan proses dengan biaya banyak dan memakan waktu lama.

Secara umum, semakin banyak iterasi dilakukan, semakin baik permainan yang akan dihasilkan. Oleh karena itu, setiap proses desain permainan harus melibatkan iterasi sebanyak mungkin, dan apa yang dapat *developer* lakukan yang memungkinkan proses iterasi lebih cepat, akan menghasilkan permainan yang lebih baik pada akhirnya . Karena itu, desainer video game akan lebih sering mem-*prototipe* di atas kertas pada proses pertama kali, lalu kemudian melibatkan programmer ketika mereka yakin bahwa aturan atau *gameplay* pada permainan sudah baik. Hal ini disebut sebagai *rapid prototyping* seperti terlihat pada gambar 2.4 berikut:



Gambar 2.4 Iterative with Rapid Prototyping

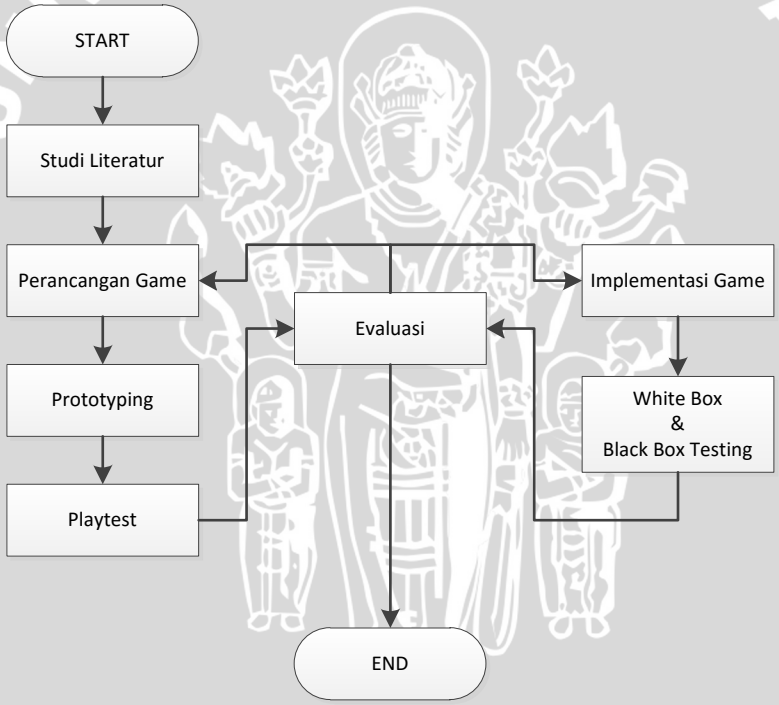
Permainan memiliki banyak jenis risiko yang terkait di dalamnya. Ada risiko desain, risiko bahwa permainan tidak akan menyenangkan dan orang-orang tidak akan menyukainya. Ada juga risiko implementasi, kemungkinan bahwa tim pengembangan tidak akan mampu meng-*implementasi* permainan sama sekali, walaupun jika rancangan peraturan permainan yang telah dibuat sangatlah *solid*. Kemudian ada risiko pasar, kemungkinan bahwa walaupun permainan yang dihasilkan sangat bagus namun tetap tidak ada yang membelinya. Tujuan dari iterasi adalah untuk menurunkan risiko desain. Semakin banyak iterasi dilakukan,

semakin *developer* dapat yakin bahwa aturan pada permainan sudah efektif. Ini semua terpusat pada satu hal penting yaitu semakin besar risiko desain permainan yang dibangun, maka semakin banyak iterasi yang dibutuhkan. Metode iterasi bukan merupakan suatu cara yang secara kritikal harus dilakukan pada permainan yang sebagian besar mekaniknya diangkat dari permainan sukses lain. Seperti sekuel ataupun ekspansi dari sebuah permainan populer adalah contoh situasi di mana pendekatan *Waterfall* dapat bekerja dengan baik [SEC-09].

BAB III
METODE PENELITIAN DAN PERANCANGAN

3.1 Metode Penelitian

Dalam penelitian ini metodologi yang digunakan adalah *iterative rapid prototyping*. *Iterative rapid prototyping* merupakan sebuah metodologi yang digunakan untuk membuat *prototyping* atau merancang konsep dari sebuah *game*. Metode ini melakukan proses iterasi atau perulangan hingga mendapatkan konsep atau rancangan *game* yang matang. Tahapan metode yang dilakukan adalah tinjauan pustaka atau studi literatur, perancangan *game*, *prototyping*, implementasi *game*, dan pengujian aplikasi berupa *playtest*, *white box* dan *black box testing*. Alur penelitian ditunjukkan dalam Gambar 3.1 berikut:



Gambar 3.1 Alur Penelitian

3.1.1 Studi Literatur

Pencarian literatur atau referensi dilakukan untuk mendapat dasar teori yang memperkuat terhadap fokus penelitian atau dalam hal ini pembuatan aplikasi *video game*. Dasar teori yang dibutuhkan dalam penelitian ini adalah :

1. *Game* (Permainan) secara umum, dan juga *fighting game* secara khusus.
2. Budaya dan wayang, sebagai dasar dari objek dari *game* yang dibangun.
3. *Mukokuseki*, sebagai bagian dari teori penyebaran atau perkenalan budaya.

4. Unity (*Game Engine*).
5. Photoshop (desain karakter dan *environment*).
6. *Game testing*.

3.1.2 Perancangan Game

Tahap perancangan dalam *game development* dilakukan guna mendapatkan gambaran umum dari permainan yang akan dibangun. Gambaran umum tersebut meliputi *target game*, *gameplay*, dan *concept art* tampilan dari permainan yang akan dibuat.

Penentuan target atau tujuan adalah berupa penentuan capaian atau keuntungan yang ingin diraih dari game yang dibuat. Capaian tersebut berupa target finansial seperti *income* dari penjualan dan *sponsorship*, dan target non-finansial seperti edukasi, penyuluhan maupun pengenalan budaya. Dari tujuan inilah lalu target pasar ditetapkan, yaitu sasaran konsumen yang mempunyai prospek dimana *game* tersebut dipasarkan sehingga tujuan atau target utama dari game tercapai.

Gameplay dibangun sebagai penjelasan bagaimana player atau pemain berinteraksi dengan game. Penjelasan rinci dalam *gameplay* meliputi peraturan atau *rule* dalam game, cerita atau plot, rintangan dan state kondisi game selesai.

Concept art adalah rancangan desain awal yang digunakan untuk mengembangkan tampilan dan menvisualisasikan *feel* dari sebuah proyek. *Concept art* terdiri dari antara lain desain karakter, *environtment*, dan HUD. Tahap ini menghasilkan sketsa mentah dari game yang kemudian dirubah menjadi representasi yang lebih akurat secara visual pada tahap implementasi.

3.1.3 Prototyping

Pada pembuatan game ini, prototipe yang dirancang terdiri dari 2 jenis prototipe. Prototipe yang pertama adalah *paper* prototipe, prototipe ini menggunakan media kertas sebagai simulator gameplay dari game yang akan dibuat. Prototipe yang kedua adalah digital prototipe dimana rancangan dibuat secara digital dengan *dummy resource*, sebatas memberikan gambaran umum terhadap gameplay yang akan dibuat.

3.1.4 Implementasi

Tahap implementasi dilakukan sesuai dengan gambaran dan konsep dasar yang telah dihasilkan pada tahap perancangan. Hal-hal tersebut meliputi :

1. Design/Art. Aplikasi : Adobe Photoshop CS6
2. Game Build. Aplikasi : Unity Game Engine 4.3
3. Scripting. Aplikasi : Mono Developer
4. Audio. Aplikasi : Audacity

3.1.5 Pengujian

Pengujian dilakukan pada game yang telah selesai dibangun untuk menemukan kesalahan-kesalahan atau bug pada game. Play testing dilakukan pada *digital prototype* untuk menguji game dari segi fungsionalitas permainan, seperti apakah game ini menyenangkan dan mencapai sasaran dari tujuan yang ingin diraih. Pengujian setelah implementasi dilakukan dengan 2 jenis pengujian, yaitu *black box* dan *white box testing.* Hasil pengujian akan menjadi bahan evaluasi untuk pengembangan game berikutnya sebelum game di-*deploy*.

3.1.6 Evaluasi

Hasil dari pengujian akan dievaluasi dan dianalisis apakah hasil yang diperoleh telah sesuai dengan target maupun sasaran yang ingin dicapai.

3.2 Rancangan Game

Rancangan game berisi rincian deskripsi elemen-elemen dalam game yang menjadi dasar dalam proses implementasi pembuatan permainan. Rincian elemen-elemen tersebut meliputi:

3.2.1 *Game Description*

Aplikasi game ini diberi nama "LAKOON" berasal dari bahasa jawa lakon yang berarti peran utama atau juga dapat berarti karangan yang berupa cerita sandiwara. Target utama dari game adalah memperkenalkan karakter-karakter wayang jawa melalui permainan yang menyenangkan, dengan tipe permainan *multiplayer* cepat dan simple.

Menggunakan *genre fighting* yang simple dan telah popular dimainkan oleh para gamer, game ini terinspirasi dari game-game sejenis seperti Samurai Gunn, Super Smash Bros, dan juga Street Fighter. Platform PC dipilih karena selain penggunanya yang lebih banyak bila dibandingkan platform lain, juga karena game ini tidak membutuhkan spesifikasi konsole yang besar untuk memainkannya, sehingga tidak hanya *gaming* PC yang bisa menjalankannya. Dengan tingkat kekerasan yang ringan dan tidak ada pemakaian kata-kata yang kotor

maupun kasar, game ini dapat dikategorikan untuk semua umur. *Minimum PC requirements* untuk dapat menjalankan *game* ini terdapat pada Tabel 3.1.

Tabel 3.1 Minimum PC Requirements

MINIMUM PC REQUIREMENTS		
CPU	:	Intel Atom dual-core processor
RAM	:	1 GB
GPU	:	Intel GMA 3600
OS	:	Windows 7

3.2.2 *Formal Elements*

Permainan dibangun dengan mengacu pada tipe permainan yang diinginkan yaitu fast multiplayer fighting game, yaitu game fighting multiplayer yang memiliki pace permainan cepat yang mengandalkan agility playernya untuk dapat memenangkan permainan. Deskripsi elemen-elemen pada gameplay adalah sebagai berikut :

3.2.2.1 *Players*

Game ini bertipe multiplayer dan membutuhkan 2 orang untuk memainkannya. Pemain memulai permainan pada waktu yang sama dan hanya bisa meninggalkan permainan ketika permainan telah selesai.

3.2.2.2 *Objectives*

Pemain saling berhadapan satu sama lain dan bertarung untuk memperebutkan poin yang berupa api. Pemain juga bisa menjatuhkan api yang telah diambil pemain lain dengan cara membunuh karakter musuh. Karakter pemain yang berhasil mengumpulkan 3 api akan memenangkan permainan.

3.2.2.3 *Rules*

Peraturan-peraturan dasar dari game ini dibagi dalam 3 macam, yaitu *rules of setup* yang menjelaskan tentang bagaimana permainan dimulai, lalu *rules for progression of play* yang menjelaskan peraturan saat permainan berlangsung, dan juga *rules for resolution* yang menjelaskan kondisi permainan berakhir. Penjelasannya adalah sebagai berikut :

1. *Rules for setup*

   Pemain akan memilih satu dari 3 karakter yang disediakan. Kedua pemain dapat memilih karakter yang sama. Pemain kemudian akan memilih arena sebagai tempat berlangsungnya permainan.

2. *Rules for progression of play*

   Peraturan-peraturan saat permainan berlangsung adalah sebagai berikut :

   a. Player dapat bertarung dengan *common attack (melee hit)*, atau pun *special move* yang berbeda pada tiap karakter.
   b. Player membutuhkan mana untuk dapat menggunakan *special move*
   c. Player dapat menggambil api yang jatuh di dalam arena yang muncul setiap 20 detik sekali.
   d. Api/skor player bertambah 1 bila dapat mengambil api.
   e. Api/skor player berkurang 1 bila mati.
   f. Bila karakter mati, maka api yang dimiliki karakter tersebut akan jatuh pada lokasi karakter tersebut terbunuh.
   g. Jika karakter pemain mati saat api/skor kosong tidak mendapat pengurangan poin.
   h. Mana poin akan beregenerasi 2 poin per detik.
   i. Api akan spawn pada lokasi randam di dalam area stage.
   j. Karakter respawn pada salah satu dari lima titik yang dipilih secara random oleh system.

3. *Rules for resolution*

   Game akan berakhir jika salah satu pemain berhasil mengumpulkan 3 api. Pemain tersebut berstatus sebagai pemenang dalam game tersebut.

3.2.2.4 *Resources and Resource Management*

Dalam game ini pemain mendapat *resource* yang dapat diatur penggunaanya oleh pemain sendiri, yaitu *mana* poin. *Mana* poin menjadi salah satu elemen dalam permainan yang krusial karena dapat mempengaruhi tipe strategi pertarungan yang dimainkan. *Special skill* yang memerlukan mana poin jauh lebih berguna dari pada *common attack (melee hit)*.

3.2.2.5 *Game State*

Game Lakoon ini mempunyai beberapa komponen atau resource di luar komponen yang dimiliki oleh karakter pemain, dan merupakan elemen peting dalam game. Komponen lain dalam game yang berpengaruh dalam permainan namun tidak dimiliki langsung oleh masing-masing pemain adalah :

1. *Mana Power-up* : *Power-up* ini dapat menambah mana sebanyak 10 poin pada karakter yang berhasil mengambilnya. *Power-up* ini muncul setiap 20 detik sekali pada lokasi acak dalam arena.
2. Dalam beberapa arena atau stage juga terdapat jebakan berupa duri-duri tajam, dan karakter pemain akan mati bila menyentuhnya.
3. Api : Objek utama yang diperebutkan dalam game. Objek ini dapat berasal dari langit arena yang muncul setiap 20 detik sekali, ataupun jatuh dari karakter pemain yang mati dan memiliki api sebelumnya.

3.2.2.6 *Information*

Informasi tentang status game maupun pemain ditampilkan seluruhnya di layar kepada semua pemain. Informasi tersebut antara lain berupa status api tiap karakter, sisa mana yang dimiliki, dan juga lokasi api ataupun *power-up mana* yang bisa dilihat tersebar dalam arena.

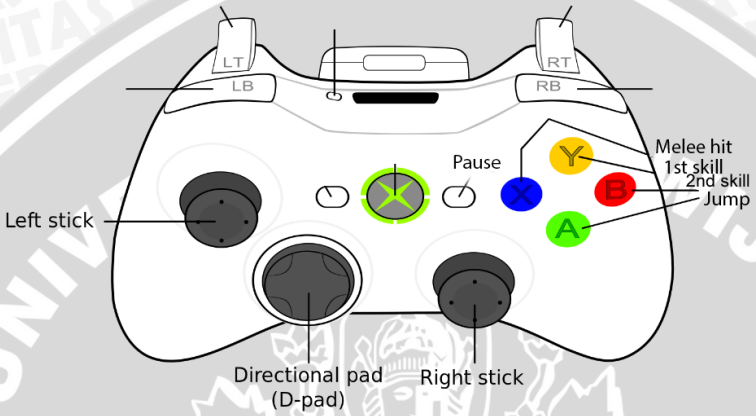
3.2.2.7 *Sequencing*

Pemain dalam game ini dapat beraksi secara *real-time* dengan bebas ketika permainan telah dimulai. Pemain dapat beraksi secara simultan bersama-sama, tanpa harus menunggu giliran.

3.2.2.8 *Player Interaction*

Player berinteraksi secara langsung dalam permainan. Interaksi tersebut berupa "*direct conflict*" antar pemain. Penjelasan *player movement* yaitu karakter dapat bergerak pada dimensi 2D. Gerakan umum yang dapat dilakukan adalah gerakan secara horizontal berupa jalan dan juga vertical berupa loncat maupun jatuh. Loncat dapat dilakukan secara *single* maupun *double jump* (loncat di udara setelah melakukan loncatan pertama pada collider). Beberapa karakter mempunyai *skill* yang memberikan *advantage* pada pemain dalam melakukan manuver pergerakan. *Skill* tersebut diantara yaitu menambah kecepatan dalam rentang waktu sementara, teleportasi, menembus *collider*, dan juga terbang.

Penjelasan sistem *fighting* pada *player* adalah, karakter dapat melakukan *melee hit* yang macamnya bervariasi pada tiap jenis karakter yang dipilih seperti penggunakan pedang, gada ataupun *unarmed melee hit*. Beberapa karakter juga memiliki *skill* yang menunjang mereka dalam *fighting*, seperti diantara yaitu panah (*range hit*), *monster summon*, ataupun *area hit*. Konfigurasi kontrol ditunjukkan pada Gambar 3.2.



Gambar 3.2 Konfigurasi Kontrol

3.2.2.9 *Theme*

Game ini dirancang dengan menampilkan suasana wayang yang dikemas secara *retro style.* Kemampuan karakter seperti srikandi yang pintar memanah juga diambil dari kisah wayang Mahabarata. Stage juga dirancang bedasar tempat yang berhubungan dengan karakter dalam game. Seperti sunyaruri yang merupakan tempat tinggal wayang, istana alengka adalah tempat dasamuka, hutan kamiyaka merupakan tempat suami srikandi yaitu arjuna di asingkan bersama 4 saudaranya yang lain.

3.2.2.10 *Game as Systems*

Game ini secara sistem mengadopsi *direct combat sistem* sebagai *gameplay* utamanya. Dimana pemain berinteraksi secara langsung dalam arena, dan bertarung untuk dapat memenangkan permainan.

3.2.3 *Game Character*

Terdapat 4 karakter yang dapat dimainkan pada game ini. Karakter tersebut adalah semar, srikandi, dasamuka, dan juga batara kala. Penjelasan terhadap karakter-karakter tersebut adalah sebagai berikut :

1. Semar

Karakter ini mempunyai keuntungan dengan mobilitasnya yang tinggi. Hal itu dikarenakan karakter ini mempunyai *skill* yang menunjang bila digunakan pada stage yang besar atau untuk menjangkau tempat-tempat yang tinggi. *Skill* tersebut adalah *skill* yang membuat semar dapat melayang di udara dengan rentang waktu bergantung pada poin mana yang tersedia. *Skill* lainnya yang menunjang adalah *special move* yang membuat semar menggebrak tanah dengan sangat cepat dan menyebabkan getaran pada area di sekitarnya.

2. Srikandi

Dengan *skill* memanahnya karakter ini mempunyai keuntungan dalam pertarungan jarak jauh. Jarak panah dapat mencapai lokasi manapun pada stage. Namun bidikan panah tersebut bergantung pula pada posisi Srikandi berada, hal itu karena *skill* tersebut hanya dapat dibidik pada sisi horizontal (sudut 0’ atau 180’) dan vertical (sudut 90’ atau 270’) dari posisi Srikandi. Karakter ini juga mempunyai *skill* lain yaitu menaikan kecepatan karakter secara instan untuk sementara waktu.

3. Dasamuka / Rahwana

Melawan karakter ini akan menjadi susah karena kemampuannya untuk memunculkan pasukan untuk melawan musuhnya. Rahwana tidak dapat mengkontrol pergerakan dari pasukan yang dia munculkan. Pasukan ini bergerak terus hingga menabrak collider yang menyebabkannya berputar dan bergerak pada arah sebaliknya. Bila musuh menabrak pasukan ini maka pasukan akan hancur namun karakter musuh juga akan terbunuh. Pasukan tidak dapat menabrak ataupun membunuh karakter yang memunculkannya. Rahwana juga mempunyai *skill* lain yaitu *teleportation*. *Skill* ini membuat karakter berpindah tempat secara instan dengan karakter musuh. *Skill* ini dapat digunakan untuk menjebak maupun mengambil keuntungan saat akan mengambil api yang dekat dengan musuh.

4. Batara Kala

Karakter ini mempunyai kemampuan menembus collider yang membuatnya sangat sulit untuk dibunuh. Kemampuan tersebut ditunjang pula dengan *skill* lain yang dia miliki, yaitu berjalan dalam tembok tanpa terpengaruh gravitasi maupun collider. Namun karakter ini mempunyai kelemahan dalam pertempuran karena hanya memiliki satu jenis serangan yaitu melee attack.

3.2.4 *Stage*

Game ini terdiri dari 10 stage yang dapat dipilih dengan bebas oleh pemain. Konsep dari 10 stage tersebut mengikuti dari tempat atau lokasi pada dunia wayang yang berhubungan dengan karakter-karakter pada game Lakoon ini.

Penjelasan stage-stage tersebut adalah sebagai berikut :

1. **Stage Sunyaruri**

Terdiri dari 2 stage yang memiliki spesifikasi kesulitan berbeda pada tiap stagenya. Salah satu stage mempunyai dunia yang luas dengan banyak path tinggi yang sulit dicapai. Hal itu membuat stage ini memiliki keuntungan bila player memainkan semar dengan tingkat mobilitasnya yang tinggi.

2. **Stage Kamiyaka**

Dengan dunia yang tidak besar dan dataran yang simple, stage ini menyajikan pertarungan-pertarungan frontal jarak dekat dengan intensitas pertarungan yang tinggi. Srikandi dan Rahwana mempunyai keuntungan pada stage ini. Dengan panah yang dapat melewati obstacle dan kecepatan tinggi yang membuat targetnya susah untuk dapat menghindar membuat srikandi dapat membunuh musuhnya dengan mudah. Begitu juga dengan Rahwana yang dapat memanggil para monsternya untuk memenuhi stage membuat musuh kesulitan mencari tempat berpijak dengan tenang.

3. **Stage Alengka**

Stage ini termasuk salah satu stage yang sulit dikarenakan datarannya yang selalu bergerak dan membuat pertempuran yang dilakukan susah untuk diprediksi. Jarak antar dataran yang sempit dan pinggiran stage yang dibatasi oleh dinding batu bata, membuat stage ini juga menguntungkan untuk rahwana yang dapat memenuhi stage dengan monster peliharaannya.

4. **Stage Setra gandamayit**

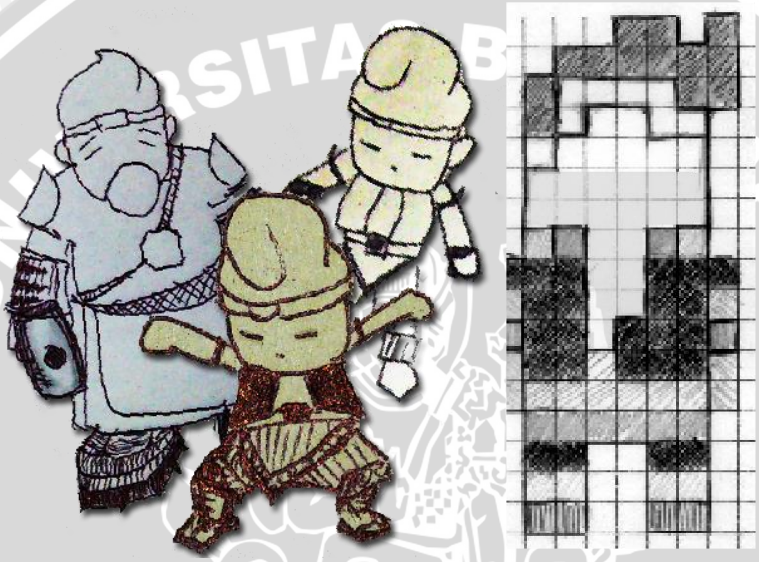
Stage ini hanya diterangi cahaya pada sekitar karakter, sehingga jalur stage tidak terlihat dan tertutup fog. Batara kala mempunyai keuntungan pada stage ini. Dengan kemampuannya menembus dinding membuatnya tidak membutuhkan cahaya untuk menemukan jalan.

3.2.4 *Concept Arts*

Game Lakoon ini dirancang dan dibangun pada dunia 2D. Design karakter dibuat dengan konsep *pixel art*. Sementara itu desain *environment*, *background*, ataupun HUD menggunakan perpaduan antara pixel art dan digital art lain. Tema dasar dari desain adalah *retro looks* dengan paduan budaya jawa dan berbaur dengan campuran budaya lain.

a. Desain Karakter

1. Semar

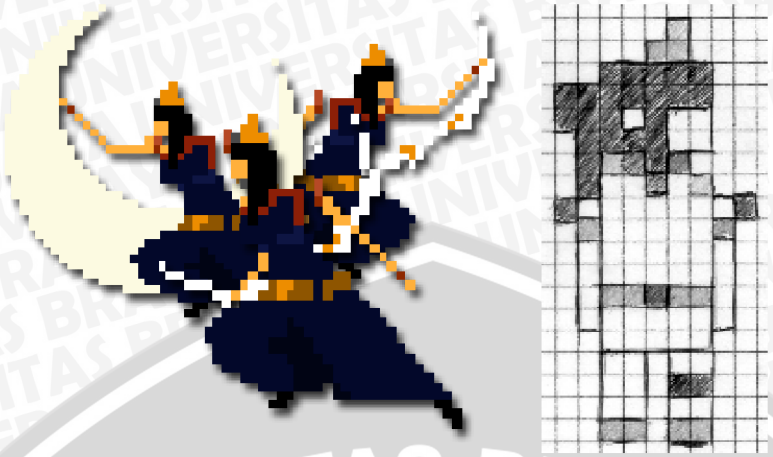


Gambar 3.4 Rancangan Semar Sebelum Revisi Dan Rancangan Akhir

Design semar yang ditunjukan pada Gambar 3.4 dirancang dengan mengikuti karakter khas dari semar yaitu jambul rambutnya yang khas. Karakter semar dibuat lebih terkesan muda dan lincah pada game ini. Hal ini sesuai dengan *skill* yang dimiliki karakter semar pada game ini, yang membuatnya menjadi karakter gesit.

2. Srikandi

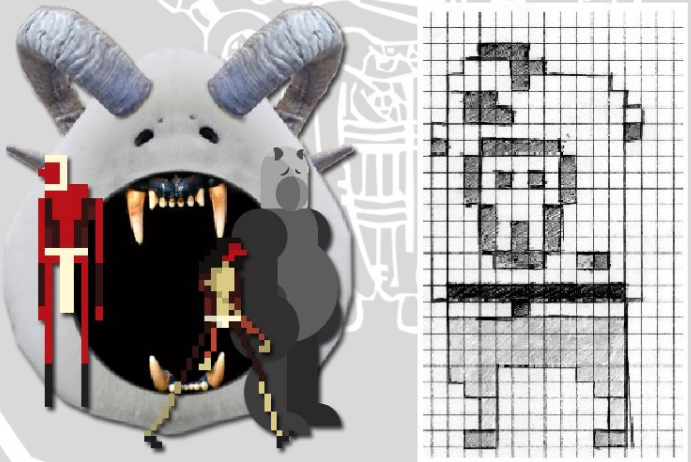
Srikandi digambarkan sebagai wanita dengan panah sebagai senjata andalannya. Rancangan desain dari srikandi ditunjukan pada Gambar 3.5.



Gambar 3.5 Rancangan Srikandi Sebelum Revisi Dan Rancangan Akhir

3. Dasamuka / Rahwana

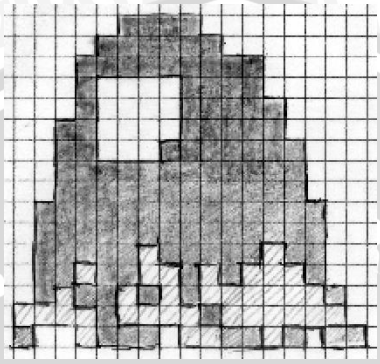
Desain rancangan rahwana yang ditunjukkan pada Gambar 3.6 diilustrasikan memiliki sosok yang tinggi besar dengan wajah beringas dan berjanggut. Namun dengan baju gambar tengkoraknya karakter ini terlihat lebih *fashionable* daripada karakter yang lain.



Gambar 3.6 Rancangan Rahwana Sebelum Revisi Dan Rancangan Akhir

4. Batara Kala

Karakter batara kala ini bertubuh hitam gelap dan berbentuk bukan manusia. Ukuran tubuhnya adalah yang paling kecil dibanding karakter lainnya. Desain rancangan dari karakter batara kala ditunjukkan pada Gambar 3.7.



Gambar 3.7 Rancangan Batara Kala

b. Stage Design

Terdapat 4 rancangan design utama dalam game ini. 4 rancangan tersebut merujuk dari 4 tipe stage yang telah disebutkan sebelumnya yaitu stage sunyaruri, stage kamiyaka, stage alengka, dan juga stage setra gandamayit. Rancangan tekstur tiap stage ditunjukkan pada gambar 3.8.

Gambar 3.8 Rancangan Tekstur Tiap Stage

1. Stage Sunyaruri

   Stage dirancang dengan background dataran tinggi dan tebing-tebing batu, yang diatasnya terdapat candi-candi.

2. Stage Kamiyaka

   Design stage kamiyaka adalah sebuah hutan lebat dengan pepohonan dan semak belukar.

3. Stage Alengka

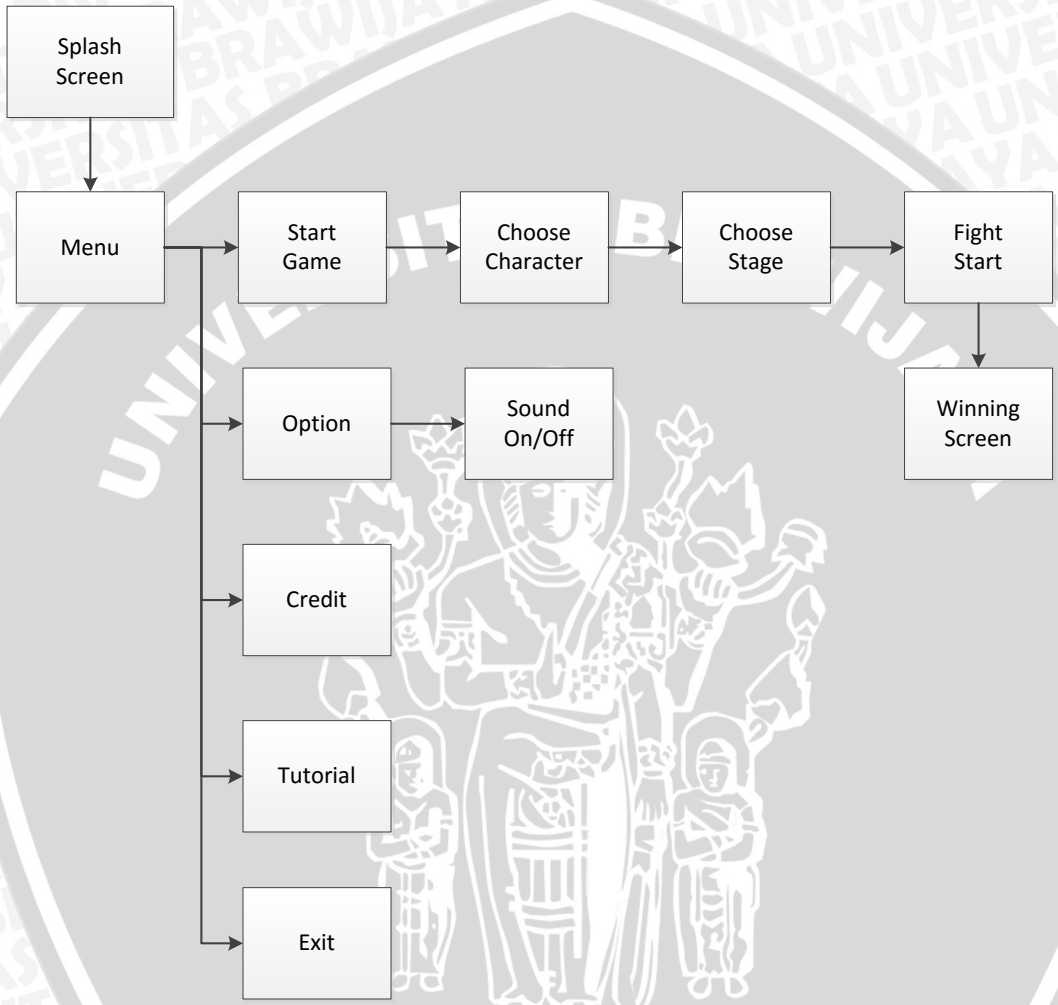
   Stage alengka bertempat di dalam gua merah yang berkesan panas dan terbakar.

4. Stage Setra Gandamayit

Stage ini berada pada hutan gelap tanpa cahaya yang di dalamnya terdapat hewan-hewan asing.

3.2.5 Game Screen Flow

Screen flow dari game ini ditunjukkan pada Gambar 3.9 berikut ini :



Gambar 3.9 Diagram Alir Permainan

Saat aplikasi dijalankan, pemain akan diperlihatkan splash screen berisi logo lab game, logo developer dan juga logo game. Setelah itu pada menu screen pemain dapat memilih untuk mengatur konfigurasi audio pada pilihan menu Option, ataupun memilih menu credit untuk melihat developer yang membuat game ini ataupun memulai permainan dengan memilih start. Pada opsi start, pemain akan dibawa pada screen untuk memilih karakter yang akan dimainkan. Setelah kedua pemain menentukan karakter yang akan dimainkan maka pemain akan menuju pada choose

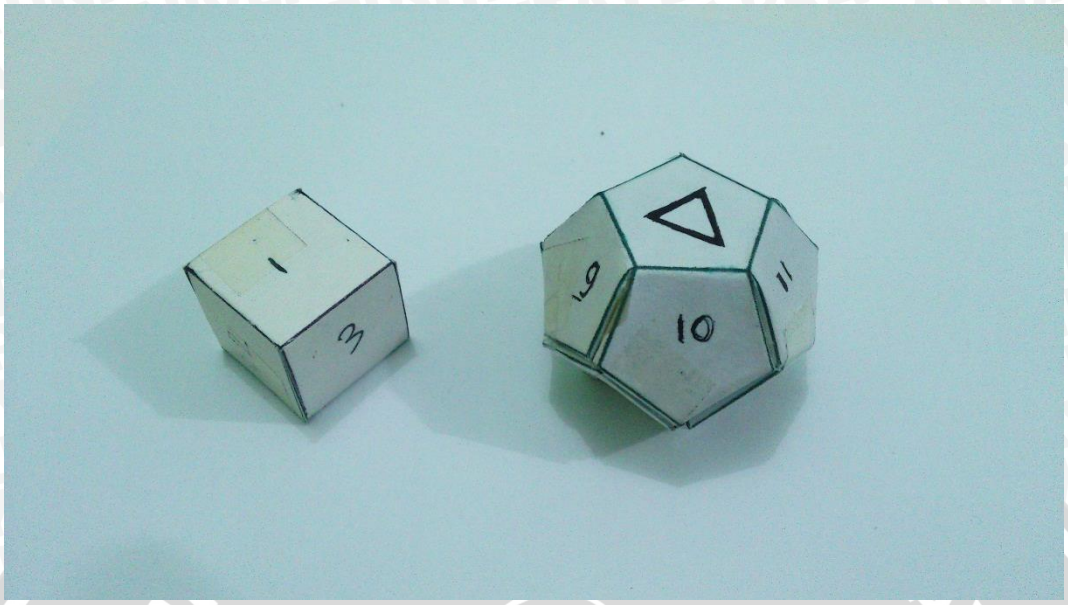
stage screen dimana pemain akan memilih area tempat pertarungan akan dilangsungkan. Pertempuran akan segera dimulai sesaat setelah pemain memilih stage atau area permainan. Winning stage akan muncul jika salah satu pemain telah berhasil ditentukan sebagai pemenangnya.

3.2.6 Paper Prototyping

Paper prototype digunakan untuk mengetahui proses interaksi langsung antara pemain dan sistem yang ditampilkan melalui visualisasi nyata. Dalam proses ini developer dapat melihat dan mengevaluasi terhadap rancangan design sistem game yang akan dibuat. *Paper prototype* pada game ini ditunjukkan pada Gambar 3.10 dan Gambar 3.11 berikut ini :

Gambar 3.10 Paper Prototyping

Gambar 3.11 Gaco Karakter dan Api

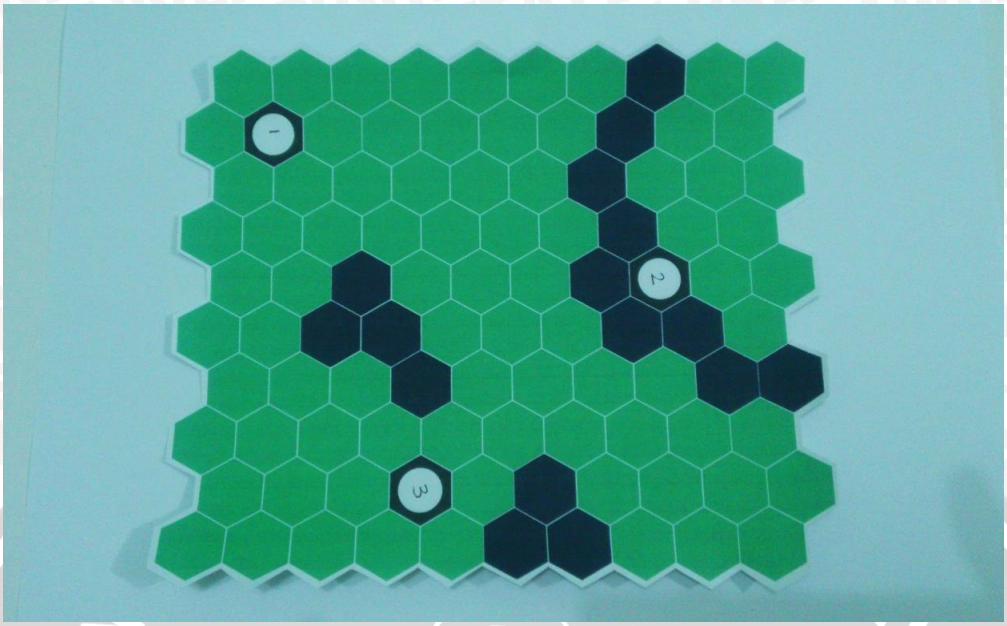


Gambar 3.12 Dadu Lokasi Spawn dan Dadu Movement

Dadu pada Gambar 3.12 digunakan untuk mensimulasikan penentuan acak lokasi spawn pemain dan juga spawn api. Dadu pertama yang berbentuk *dodecahedron* terdiri dari angka 2 hingga 11 dan 2 buah api yang disimbolkan dengan Δ. Dadu kedua yang berbentuk kubus terdiri dari angka 1 sampai 3 untuk menentukan lokasi *spawn* pemain.

Penggunaan mana pada *skill* yang dikeluarkan oleh tiap pemain disimulasikan dengan pengurangan terhadap jumlah movement yang didapat dari pelemparan dadu. Rancangan kartu dapat dilihat pada Gambar 3.13.

Gambar 3.13 Kartu Deskripsi *Skill*



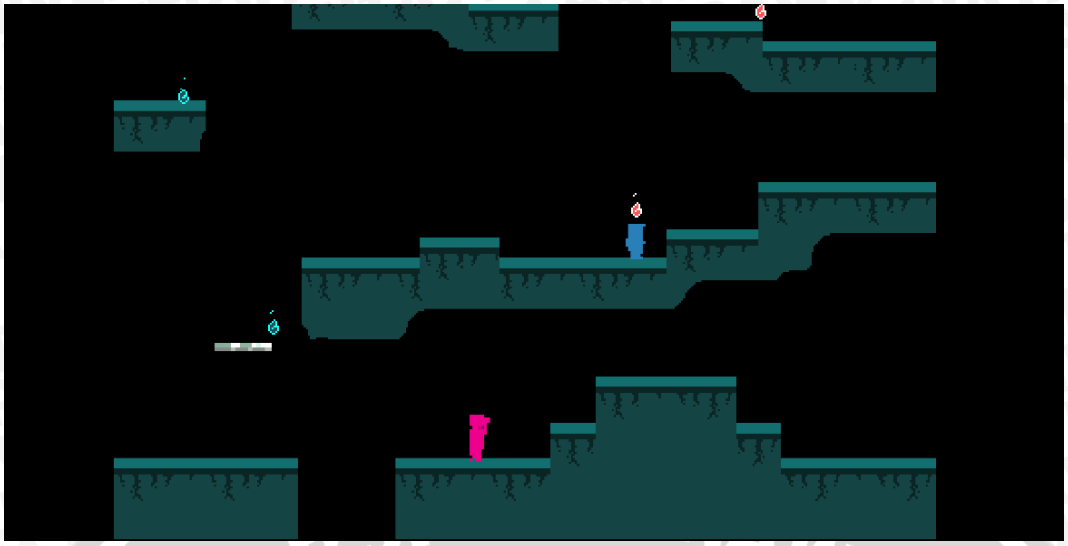
Gambar 3.14 Board Arena Permainan

Pada gambar 3.14 ditunjukkan board permainan yang terdiri dari tile 10x10 dengan tile hijau yang berarti dapat dilewati pemain dan tile hitam yang tidak dapat dilewati oleh pemain. Jika pemain bergerak melewati batas sebelah kiri maka pemain akan pindah ke sisi sebaliknya, begitu juga dengan atas dan bawah.

3.2.7 *Digital Prototyping*

Tahapan ini adalah proses implementasi secara digital dari paper prototype yang telah dibuat. Prototype ini dibangun sebatas untuk memberi gambaran umum dari gameplay permainan yang dibuat. Fokus dari digital prototype ini untuk melihat sejauh mana sistem gameplay yang dirancang dapat dan nyaman untuk dimainkan dalam implementasi digital, sehingga nantinya akan terlihat bug maupun elemen-elemen dalam game yang harus dikembangkan ataupun diubah lebih lanjut.

Pada digital prototype ini sistem hanya fokus pada gameplay saat pertarungan di arena dimulai. Sistem lain seperti character select, stage select, HUD maupun fungsi setting game belum diimplementasikan. Karakter juga memakai dummy sprite, sedangkan sistem attack, manuver, mana, dan juga poin api telah diimplementasikan. Hasil dari digital protoype ditunjukkan pada Gambar 3.15 berikut ini :



Gambar 3.15 Digital Prototype

3.2.8 *Play Testing*

Playtesting dilakukan untuk menguji game dari pandangan tester atau diluar developer. *Playtesting* terdiri dari *Bug testing, Fun testing, Balance testing,* dan *Focus testing.*

3.2.8.1 *Fun Testing*

Fun testing dilakukan untuk menguji tingkat kenyamanan dan kesenangan yang didapat pemain dari game ini. Hasil dari fun testing dijelaskan secara deskriptif atau berupa data kualitatif.

Hasil nya dengan sistem pertarungan yang simple dan mudah dipahami, juga pace permainan yang cepat membuat game menyenangkan untuk dimainkan. Tipe karakter yang beragam juga membuat game tidak bosan untuk dimainkan secara terus menerus.

3.2.8.2 *Bug testing*

Bug testing dilakukan untuk menemukan kesalahan-kesalahan pada game yang tidak berjalan seperti yang telah dirancang sebelumnya. Test dilakukan dengan melakukan playtesting sesuai dengan test case yang dibuat. Test case tersebut adalah :

1. Mulai permainan utama setelah memilih karakter dan stage/arena.
2. Apakah kedua karakter yang telah dipilih pemain muncul di stage?
3. Apakah karakter dapat digerakan?

4. Apakah jika pemain memukul karakter musuh maka karakter musuh akan mati?
5. Apakah muncul animasi mati dari karakter yang barusan dibunuh?
6. Apikah api muncul pada lokasi yang acak?
7. Apakah api mucul 10 detik sekali?
8. Apakah api dapat diambil?
9. Apakah saat mati api akan jatuh?
10. Apakah saat mana habis karakter tidak dapat mengeluarkan *skill*?
11. Apakah mana beregenerasi 1 poin setiap detik?
12. Apakah saat pemain bergerak melewati batas sebelah kiri, kanan ataupun bawah maka karakter akan muncul pada posisi batas arah sebaliknya?
13. Apakah setelah pemain mendapat 3 api permainan akan selesai dan tampilan pemenang dimunculkan?

Hasilnya tidak ditemukan bug pada game yang telah dibuat.

3.2.8.3 *Balance Testing*

Balance testing dilakukan untuk menguji elemen-elemen dalam game dan pengaruhnya terhadap permainan. Pengujian ini ditujukan untuk menjaga keseimbangan dalam permainan sehingga tidak ada *character* yang terlalu *overpowered* dibanding *character* lainnya.

Elemen utama yang berpengaruh dalam permainan pada game ini dapat dikelompokan menjadi 2 tipe yaitu elemen di luar karakter dan elemen dalam karakter. Elemen dalam karakter adalah :

a. Attacking & Manuver *skill*

   Tiap karakter mempunyai special *skill* yang berbeda-beda. Semar mempunyai *skill* terbang dan *hit area*. Sedangkan srikandi mempunyai *skill* memanah dan gerak cepat. Sementara dasamuka memiliki *skill* memanggil pasukan dan *swap position*.

b. Mana spent

   Mana dibutuhkan untuk dapat mengeluarkan special *skill*. Jumlah mana yang dibutuhkan tiap *skill*nya berbeda-beda tergantung *skill* apa yang digunakan.

c. Mana regeneration

Mana akan beregenerasi atau bertambah sejumlah poin tertentu pada tiap detiknya.

d. Movement speed

Adalah kecepatan lari pada tiap karakter

Sedangkan elemen di luar karakter adalah :

a. Api spawn time

Api akan muncul 1 kali setiap beberapa detik.

b. Arena design

Obstacle pada tiap arena berbeda-beda dengan tingkat kesulitan yang beragam.

A. Balance Testing v1.0

1. Game Parameter Setting

Maximum mana pada tiap karakter adalah 100 poin. Stage yang digunakan untuk pengujian adalah semua stage. Api manik respawn 1 kali dalam 10 detik. Nilai untuk tiap parameter yang digunakan ditunjukkan dalam Tabel 3.2.

Tabel 3.2 Parameter Setting v1.0

Character	**Special Move**	
Semar	*Skill* Terbang	Mana : 10
		Addforce : up x 600
	Skill Jatuh	Mana : 10
		Damage radius : 2.5
Srikandi	*Skill* Panah	Mana : 15
		Speed : 50
	Skill Dash	Mana : 10
		Speed :
		Duration : 2 second
Dasamuka	*Skill* Summon	Mana : 15
		Monster life time : 20 second
	Skill Teleport	Mana : 30

2. Result

Dasamuka mendapatkan presentase win rate 46.67%, dan merupakan presentase win rate paling besar dibanding karakter yang lain. Srikandi di urutan kedua dengan win rate 30%, dan yang terakhir adalah Semar dengan 23.33%. Dasamuka tercatat dapat mendominasi permainan pada stage yang mempunyai arena sempit seperti stage Alengka 2 dan juga Sunyaruri 2. Di beberapa arena yang besar dan terbuka pun Dasamuka dapat memenangkan permainan walau tidak mendominasi. Semar dan srikandi mendapat status yang cukup berimbang dengan saling mengalahkan di hampir semua stage. Rata-rata durasi permainan adalah 1.05 menit, dan membuat permainan terasa terlalu cepat selesainya.

3. Analisa

Balance testing pada game Lakoon v1.0 ini menunjukkan bahwa masih terdapat ketidak seimbangan kekuatan pada setiap karakter di semua stage. Spam pasukan oleh dasamuka di arena sempit terasa begitu *overpower* karena tidak adanya celah untuk menghindar. Sedangkan pada stage yang besar pun Dasamuka menggunakan *skill*nya yang lain, yaitu dapat menukar posisi antar player pada permainan. Hal ini dapat dimanfaatkan untuk menjebak karakter musuh pada posisi yang berbahaya, ataupun menggunakannya untuk merebut api yang didekati oleh musuh.

B. Balance Testing v1.1

1. Game Parameter Setting

Banyak mana yang dibutuh oleh rahwana untuk mengeluarkan *skill* teleport dinaikan sebanyak 10 poin. Mana untuk *skill* terbang semar dikurangi sebanyak 5 poin. Durasi spawn api diperlambat menjadi 1 per 20 detik. *Skill* terbang semar sekarang juga dapat menembus collider. Nilai untuk tiap parameter yang digunakan ditunjukkan dalam Tabel 3.3.

Tabel 3.3 Parameter Setting v1.1

Character	**Special Move**	
Semar	*Skill* Terbang	Mana : 5
		Addforce : up x 600
	Skill Jatuh	Mana : 10

		Damage radius : 2.5
Srikandi	*Skill* Panah	Mana : 15
		Speed : 50
	Skill Dash	Mana : 10
		Speed :
		Duration : 2 second
Dasamuka	*Skill* Summon	Mana : 15
		Monster life time : 20 second
	Skill Teleport	Mana : 40

2. Result

Win rate antar karakter sudah berimbang dengan persentase 33.3% untuk semar, 30% untuk srikandi dan 36.7% untuk dasamuka. Permainan juga berimbang disetiap stage dengan rata-rat durasi permainan yaitu 2 menit.

3. Analisa

Penambahan mana yang dibutuhkan untuk *skill* teleport dasamuka membuatnya tidak dapat terlalu boros dalam mengeluarkan mana, sehingga *skill* teleport yang banyak digunakan untuk menjebak musuh pada v1.0 tidak lagi terlalu sering namun cukup untuk dapat melakukan pertarungan. Semar dengan pengurangan mana *skill* terbangnya juga dapat melakukan stategi sebagai aggro semar, yang secara agresif mengejar musuhnya hingga kena. Walaupun begitu karena durasi permainan yang menjadi lebih lama dan regenerasi mana yang sedikit membuat pace pertarungan terasa menjadi lebih lambat.

C. Balance Testing v1.2

1. Game Elemen

Penambahan elemen dilakukan dengan implementasi powerup penambah mana. Powerup mana tersebut dapat menambah mana karakter yang mengambilnya sebanyak 10 poin. Powerup ini muncul 1 kali setiap detik pada lokasi acak dalam stage.

2. Result

Pace permainan lebih terasa cepat dengan pemakaian *skill* yang dapat lebih sering digunakan, karena adanya *power up* mana.

3. Analisa

Implementasi *power up* mana menambah sisi tantangan dalam game yaitu memperebutkan *power up*, sehingga *filler action* antar jatuhnya api dapat lebih terisi. Pemain juga dapat lebih bebas menggunakan *skill*nya karena tidak takut untuk kehabisan mana.

3.2.8.4 *Focus testing*

Focus testing adalah pengujian yang dilakukan pada target *audience* atau sasaran pengguna dari game yang dibangun. Daftar pertanyaan yang diberikan adalah sebagai berikut :

1. Apakah anda mengetahui karakter wayang Ramayana atau Mahabarata?
2. Jika tahu (nomor 2) apakah anda mengetahui kekuatan karakter tersebut?
3. Tingkat kesenangan saat memainkan game
4. Ketertarikan untuk mengetahui karakter wayang secara lebih lanjut

Hasil nya dari 5 orang yang telah dilakukan wawancara, 3 orang menyatakan kurang mengetahui karakter wayang dan 2 lainnya cukup mengetahui. Namun mengenai kekuatan karakter tersebut, para koresponden menjawab tidak tahu. Tingkat kesenangan dinilai baik dengan 100% dari koresponden menyatakan game mudah untuk dimainkan dan menyenangkan. Beberapa *skill* unik dari karakter membuat 100% dari mereka menyatakan ketertarikannya untuk mengenal tokoh tokoh wayang secara lebih lanjut.

BAB IV
IMPLEMENTASI

4.1 Pemilihan Teknologi dan Platform

Dalam pembuatan game ini dibutuhkan spefikasi perangkat keras dan perangkat lunak yang dapat menunjang dalam penyelesaian implementasi aplikasi. *Game* ini dibangun dan dikembangkan dengan lingkup spesifikasi hardware yang ditunjukkan pada Tabel 4.1 berikut ini :

Tabel 4. 1 Tabel Spesifikasi Hardware

Spesifikasi *Hardware*	
Processor	Intel Core i3-2330M CPU @2.20GHz (4CPU)
Memory (RAM)	4096MB
Hard disk	580GB HDD
Graphic Card	NVIDIA GeForce GT 525M

Spesifikasi *software* yang digunakan dalam pengembangan *game* ini meliputi *game engine, graphic editor* maupun *audio editor* dan yang lain ditunjukkan pada Tabel 4.1 berikut ini :

Tabel 4.2 Tabel Spesifikasi Software

Spesifikasi *Software*	
Operating System	*Windows 7 Home Premium 64bit*
DirectX Version	*DirextX 11*
Programming Language	*C#*
Graphics Editor	*Adobe Photoshop CS6*
Game Engine	*Unity 4.6*
Integrated Development Environment	*Mono Developer*
Audio Editor	*Audacity*

Beberapa software pendukung dibutuhkan untuk membangun game Lakoon ini. Pemilihan Unity 4.6 sebagai game engine dalam pembuatan game Lakoon karena sejak versi 4.3 keatas unity telah menyediakan tools-tools pendukung untuk

pembuatan game 2d, sehingga memudahkan developer untuk mengimplementasi game yang akan dibuat. Forum dan komunitas yang aktif membuat unity menjadi lebih mudah untuk dipelajari.

4.2 Implementasi Prosedur Program

Terdapat beberapa faktor utama pada proses implementasi prosedur program pada game ini, antara lain adalah implementasi gameplay, implementasi *character behavior*, implementasi design environment dan karakter, lalu implementasi sound effect dan musik.

4.2.1 Implementasi Karakter

Tabel 4. 2 Pseudo code Gerak Karakter

Pseudo Code Gerak	
DECLARATION count is INT jumpButton is STRING kecepatan is FLOAT power is FLOAT	
	DESCRIPTION
1.	START
2.	IF (Input axis < 0) THEN
3.	transform.scale.x = -1.0f;
4.	ELSE IF (Input axis > 0) THEN
5.	transform.scale.x = 1.0f;
6.	END IF;
7.	transform.position += transform.right * kecepatan *
8.	nilai input axis * Time.deltaTime;
9.	IF(Input.GetButtonDown(jumpButton)) THEN
10.	IF(tanah) THEN
11.	rigidbody2D.AddForce(transform.up*power);
12.	ELSE IF(!tanah && count == 1) THEN
13.	rigidbody2D.AddForce(transform.up*power);
14.	END IF;
15.	END
16.	
17.	

Pseudo code yang ditunjukkan pada tabel 4.3 merupakan implementasi dari gerak dasar pada setiap karakter. Karakter dapat bergerak secara horizontal pada axis x. Karakter dapat melakukan lompatan bila gameobject berada atau menyentuh platform game, karakter dapat juga melakukan lompatan di udara hanya satu kali setelah karakter tersebut meninggalkan platform.

Tabel 4. 3 Pseudo code Prosedur Hit

Pseudo Code Hit	
DECLARATION	
	nextFire is FLOAT
	fireRate is FLOAT
1.	**DESCRIPTION**
2.	START
3.	IF(Input.GetButtonDown(pukul) && Time.time > nextFire) THEN
4.	nextFire = Time.time + fireRate;
5.	collider.enabled <- true;
6.	PelanSebentar();
7.	END IF;
8.	END

Pada tabel 4.4, player dapat melakukan pukulan jarak dekat (*melee hit*) pada rentang waktu setiap 0.5 detik. Implementasi hit action pada engine yaitu dengan cara meng-*enable* collider2d di depan karakter dengan jarak tertentu dan men-*disable*-nya kemudian. Kecepatan player juga dikurangi saat hit action berjalan untuk dapat menciptakan feel aksi yang lebih riil.

Tabel 4. 5 Pseudo code Prosedur Cek hit

Pseudo Code Cek Hit
DECLARATION
apiClone is GAMEOBJECT
SkorApi is INT

```
1.  DESCRIPTION
2.  START
3.        IF (collider hitbox bertabrakan dengan collider
          dengan tag pukul) THEN
4.              apiClone <- Find ("fire") on Parent;
5.              transform.parent.position =
6.  OtherLocation[Rand()];
7.              IF (apiClone) THEN
8.                  SkorApi--;
9.                  apiClone.parent <- null;
10.             END IF;
11.     END IF;
12. END
```

Pada tabel 4.5, saat player terpukul oleh collider hit player lain, jika player tersebut memiliki api maka api tersebut akan terjatuh atau lepas dari parent atau player. Player juga akan respawn pada titik yang secara acak dipilih dari 5 titik koordinat pada array *OtherLocation[]* yang telah di tentukan pada tiap stage.

Tabel 4. 6 Pseudo code Prosedur Ambil Api

```
Pseudo Code Ambil Api
DECLARATION
     Api is GAMEOBJECT
1.  DESCRIPTION
2.  START
3.        IF(collider player berinteraksi dengan collider
          dengan tag api) THEN
4.              api.parent <- transform;
5.              api.transform.position <- transform.position;
6.        END IF;
7.  END
```

Pada tabel 4.6, Jika player collider berinteraksi dengan collider api maka gameobject api tersebut akan menjadi child dari karakter, dan merupah *position* api sama dengan *position* dari *parent.*

Implementasi gerakan khusus pada tiap karakter semar, srikandi dan juga dasamuka yang memiliki special move / skill yang memberikan advantage tersendiri terhadap karakter lain adalah sebagai berikut :

Tabel 4. 7 Pseudo code Prosedur Skill Semar

Pseudo Code Skill Semar	
DECLARATION col is COLLIDER2D	
1.	**DESCRIPTION**
2.	START
3.	IF(Input.GetButtonDown(ulticek) && manaChar.mana > 5) THEN
4.	rigidbody2D.AddForce(transform.up*power);
5.	manaChar.SubMana(5);
6.	END IF;
7.	IF(Input.GetButtonDown(skill)) THEN
8.	anim.SetTrigger("Ulti");
9.	col.enabled = true;
10.	rigidbody2D.AddForce(transform.up*power*-30);
11.	StartCoroutine("hitwait");
12.	END IF;
13.	END

Pada tabel 4.7, semar memiliki 2 special move yang keduanya memanfaatkan fungsi *addforce*. Skill yang pertama, karakter dapat bergerak bebas walau tidak menyentuh platform dengan kebutuhan mana sebanyak 5 setiap kali skill dikeluarkan. Skill kedua adalah karakter dapat jatuh dan menyentuh *platform* atau tanah dengan cepat, saat ia melakukannya *collider* pukul akan muncul dalam sementara waktu. Gambaran implementasinya pada scene ditunjukkan pada Gambar 4.1.

Gambar 4.1 Implementasi Skill Terbang Semar

Tabel 4. 8 Pseudo code Prosedur Skill Srikandi

Pseudo Code Skill Srikandi	
DECLARATION Kecepatan is FLOAT Bow is GAMEOBJECT PanahPressed is BOOL UpPressed is BOOL DownPressed is BOOL	
1.	**DESCRIPTION**
2.	START
3.	IF(INPUT KEY skill 1) THEN kecepatan = 15;
4.	CepatLama();
5.	END IF
6.	IF (INPUT KEY skill 2) THEN
7.	bow.SetActive (true);
8.	PanahPressed = true;
9.	END IF
10.	IF(PanahPressed && Input ButtonDown(atas)) THEN
11.	bow.rotation = Quaternion.Euler(0,0,90);
12.	bow.position = transform.position;
13.	UpPressed = true;

```
14.       END IF
15.       IF (Input ButtonUp(atas)) THEN
16.             bow.rotation =  Quaternion.Euler(0, 0, 0);
17.             bow.localPosition =  AnakPos;
18.             UpPressed = false;
19.       END IF
20.       IF (PanahPressed && Input.GetButtonDown(bawah)) THEN
21.             bow.rotation =  Quaternion.Euler(0, 0, 270);
22.             bow.position.y = transform.position.y-1.5f;
23.       END IF
24.       IF (Input.GetButtonUp(bawah)) THEN
25.             bow.rotation =  Quaternion.Euler(0, 0, 0);
26.             anak.transform.localPosition =  AnakPos;
27.             DownPressed = false;
28.       END IF
29.       IF (Input.GetButtonUp(arc)) THEN
30.             IF(UpPressed)THEN
31.                 Instantiate(spawnedObject,
32.                 transform.position,  Quaternion.Euler(0,
33.                 0, 90));
34.             END IF
35.             ELSE IF(DownPressed) THEN
36.                 Instantiate(spawnedObject,
37.                 transform.position,  Quaternion.Euler(0,
38.                 0, 270));
39.             END IF
40.             ELSE IF(transform.localScale.x == -1) THEN
41.                 Instantiate(spawnedObject,
42.                 transform.position,  Quaternion.Euler(0,
43.                 0, 180));
44.            END IF
45.             ELSE{
46.                  Instantiate(spawnedObject,
47.                 transform.position,  Quaternion.Euler(0,
48.                 0, 0));
49.           END IF
50.       END IF
51.       bow.SetActive(false);
52. END
```

Tabel 4.8 menunjukkan pseudo code untuk implementasi skill karakter srikandi. Srikandi memiliki 2 special move yang membantunya dalam pertarungan. Skill pertama adalah penambahan kecepatan secara instan pada aselang waktu sementara. Skill kedua yaitu, srikandi dapat menembakan panah dengan arah horizontal dan vertical yaitu derajat 0, 90, 180, dan juga 270 dihitung dari sumbu x positif, dengan pivot sumbu z. Gambar implementasi secara visual ditunjukkan pada Gambar 4.2.

Gambar 4.2 Implementasi Skill Panah Srikandi

Tabel 4. 9 Pseudo Code Prosedur Skill Rahwana

Pseudo Code Skill Rahwana	
DECLARATION Ulticek is STRING Skill is STRING	
1.	**DESCRIPTION**
2.	START
3.	IF(Input.GetButtonDown(ulticek) && manaChar.mana >
4.	5) THEN
5.	var tempPosition = transform.position;
6.	transform.position = musuh.transform.position;
7.	musuh.transform.position = tempPosition;
8.	END IF;
9.	IF(Input.GetButtonDown(skill)) THEN
10.	IF(transform.localScale.x == -1) THEN
11.	Vector3 temp = transform.position;
12.	temp.x -= 0.5f;
13.	var res = Instantiate (spawnedObject,
14.	temp, transform.rotation) as GameObject;
15.	var tempscale =
16.	res.transform.localScale;
17.	tempscale.x = -1;
18.	res.transform.localScale = tempscale;

```
19.             END IF;
20.             ELSE
21.                     Vector3 temp = transform.position;
22.                     temp.x += 0.5f;
23.                     temp.y -= 0.5f;
24.                     var res = Instantiate (spawnedObject,
25.         temp, transform.rotation) as GameObject;
26.                     var tempscale =
27.         res.transform.localScale;
28.                     tempscale.x = 1;
29.                     res.transform.localScale = tempscale;
30.              END IF;
31.  END
```

Pseudo code yang ditunjukkan pada Tabel 4.9 merupakan implementasi dari skill dasamuka. Dasamuka atau rahwana juga mempunyai 2 special move seperti karakter yang lain, skill yang pertama diimplementasikan dengan menukar *transform.position* masing-masing karakter. Skill ke 2 rahwana meng-*instantiate* monsternya yang dideskripsikan *behavior*-nya pada kelas *DasamukaMonster.cs* yaitu terus berjalan *transform.right* dan bila monster collide dengan yang lain localscale.x menjadi negative sehingga monster akan berjalan kearah sebaliknya. Monster ini juga mempunyai collider2d pukul. Implementasinya secara visual ditunjukkan pada Gambar 4.3.

Gambar 4.3 Implementasi Skill Rahwana

4.2.2 Implementasi Gameplay

Ada beberapa script yang menunjang proses gameplay pada game ini, script penunjang itu termasuk penyimpanan data antar scene, pendefinisian atribut stage, rules, dan juga status karakter atau player.

Tabel 4. 10 Pseudo code Prosedur Game Master

Pseudo Code GameMaster.cs	
1.	public class GameMaster : MonoBehaviour {
2.	static int a = 0;
3.	public GameObject Player1;
4.	public GameObject Player2;
5.	public string Player1Name;
6.	public string Player2Name;
7.	Animator[] AnimApi = new Animator[2];
8.	public int[,] charStats = new int[4,3];
9.	bar[] manaScript = new bar[2];
10.	public float xKiri;
11.	public float xKanan;
12.	public float[] yStage;
13.	public GameObject apiManik;

```
14.     public bool apiHidup = false;
15.     Master MasterScript;
16.     public Vector3[] spawn;
17.     public GameObject WinStage;
18.     public float batas;
19.     public float batasKiri;
20.     public float batasKanan;
21.
22.     void Start () {
23.
24.         GameObject Master = GameObject.Find("Master");
25.         MasterScript = Master.GetComponent<Master>();
26.         Player1Name = MasterScript.Player1Name;
27.         Player2Name = MasterScript.Player2Name;
28.
29.         Player1 =
30.         Instantiate(Resources.Load(Player1Name),
31.         spawn[0], transform.rotation) as GameObject;
32.         gerak grkscript =
33.         Player1.GetComponent<gerak>();
34.         grkscript.SetControl ("P1", batas, batasKiri,
35.         batasKanan);
36.
37.         Player2 =
38.         Instantiate(Resources.Load(Player2Name),
39.         spawn[2], transform.rotation) as GameObject;
40.         gerak grkscript2 =
41.         Player2.GetComponent<gerak>();
42.         grkscript2.SetControl ("P2", batas, batasKiri,
43.         batasKanan);
44.
45.         GameObject Cam = GameObject.Find("Single
46.         Camera");
47.         zoom camSc = Cam.GetComponent<zoom>();
48.         camSc.SetCam (Player1, Player2);
49.
50.         GameObject manaParent =  GameObject.Find
51.         ("/CameraHUD/P1/Image");
52.
```

```
53.             manaScript[0] =
54.             manaParent.GetComponent<bar>();
55.             GameObject manaParent2 =  GameObject.Find
56.             ("/CameraHUD/P2/Image");
57.             manaScript[1] =
58.             manaParent2.GetComponent<bar>();
59.
60.             charStats [0, 1] = 100;
61.             charStats [1, 1] = 100;
62.
63.             GameObject apiParent =
64.             Instantiate(Resources.Load("life"+Player1Name)
65.             , transform.position,  Quaternion.identity) as
66.             GameObject;
67.             apiParent.transform.parent = GameObject.Find
68.             ("/CameraHUD").transform;
69.             Vector3 tempPos = new Vector3 (-0.6f, -4.2f,
70.             204);
71.
72.             apiParent.transform.localPosition = tempPos;
73.             AnimApi[0] =
74.             apiParent.GetComponent<Animator>();
75.
76.             GameObject apiParent2 =
77.             Instantiate(Resources.Load("life"+Player2Name)
78.             , transform.position,  Quaternion.identity) as
79.             GameObject;
80.             apiParent2.transform.parent = GameObject.Find
81.             ("/CameraHUD").transform;
82.             Vector3 tempPos2 = new Vector3 (0.6f, -4.2f,
83.             204);
84.             apiParent2.transform.localPosition = tempPos2;
85.             AnimApi[1] =
86.             apiParent2.GetComponent<Animator>();
87.
88.             }
89.
90.         void Update () {
91.             if(!apiHidup) StartCoroutine("spawnManik");
```

```
92.     }
93.
94.     IEnumerator spawnManik(){
95.             apiHidup = true;
96.             Vector3 tempKoor = new
97.             Vector3(Random.Range(xKiri,xKanan),
98.             yStage[Random.Range(0, yStage.Length-1)], 0);
99.             var tempManik = Instantiate(apiManik,
100.            tempKoor, Quaternion.identity) as GameObject;
101.            yield return new WaitForSeconds(15);
102.            apiHidup = false;
103.    }
104.
105.    public void charName (string p1, string p2){
106.            Player1Name = p1;
107.            Player2Name = p2;
108.    }
109.
110.    public void api(int charApi, int berapa){
111.            charStats [charApi, 0] += berapa;
112.            if (charStats [charApi, 0] == 3) {
113.                 if(charApi == 0){
114.
115.            MasterScript.winning(Player1Name);}
116.                 else{
117.                      MasterScript.winning(Player2Name);
118.                 }
119.                 WinStage.SetActive(true);
120.            }
121.            string triggerApi = charStats [charApi,
122.            0].ToString();
123.            AnimApi[charApi].SetTrigger (triggerApi);
124.
125.    }
126.
127.    public void mana(int charMana, int berapa){
128.            charStats [charMana, 1] += berapa;
129.            manaScript[charMana].manaBar(-berapa);    }
130.
```

```
131.            public Vector3 spawnChar (){
132.            return spawn[Random.Range(0,4)];
        }
```

Pada tabel 4.10 atribut-atribut penunjang game didefinisikan dan di-*instantiate* beberapa prefab yang dibutuhkan. Definisi atribut player1 dan player2 diambil dari kelas master yang menyimpan datanya sejak pemain melakukan pemilihan karakter di scene *character select.* Kelas master melakukan penyimpan data dengan fungsi DontDestroyOnLoad() membuat data yang telah didefinisikan sebelumnya tidak terhapus walaupun telah berganti scene. Api dimunculkan dengan rentang waktu setiap 15 detik, posisi *instantiate* api ditentukan secara random dengan range yang telah ditentukan pada variable xKiri, xKanan dan juga yStage. Proses penambahan dan pengurangan mana di-*trigger* oleh kelas scorePlayer yang dipasang pada tiap karakter, proses tersebut kemudian akan diteruskan pada kelas bar yang dipasang pada objek HUD untuk dapat menampilkan progres pengurangan maupun penambahan mana. Titik koordinat spawn karakter juga didefinisikan pada kelas ini yang nantinya oleh kelas cekHit dipanggil untuk mendapatkan titik spawn karakter yang baru.

4.2.3 Implementasi Camera Works

Kamera pada game ini menggunakan *orthographic projection*. Kamera dibagi menjadi 3 state utama yang dibedakan dari ukuran *orthographic* kamera. State tersebut berubah sesuai dengan jarak dari kedua pemain secara dinamis. Selain itu kamera juga mengikuti titik tengah koordinat antar pemain yang diambil dari titik x player sebelah kiri + (*distance* antar player / 2). Kode lengkap dari cara kerja camera ditunjukkan pada Tabel 4.11 berikut ini :

Tabel 4. 11 Pseudo Code Prosedur Camera Works

```
Pseudo Code Camera
DECLARATION
    player1 is TRANSFORM
    player2 is TRANSFORM
    centerKotak is FLOAT
    zoomArea is INT
    normal is INT
```

```
    aboveNormalVal is INT
    isZoomed is BOOL
    cekZoom is BOOL
    aboveNormal is BOOL
1.  DESCRIPTION
2.  START
3.      var distKotak =
4.      Distance(player1.position,player2.position);
5.      var distVertical = Mathf.Abs(player1.position.y -
6.      player2.position.y);
7.      IF(player1.position.x> player2.position.x) THEN
8.            centerKotak = player2.position.x +
9.      (distKotak/2);
10.     ELSE
11.           centerKotak = player1.position.x +
12.     (distKotak/2);
13.     END IF;
14.     IF(player1.position.y>player2.position.y) THEN
15.           centerKotak1 = player2.position.y +
16.     (distVertical/2);
17.     ELSE
18.           centerKotak1 = player1.position.y +
19.     (distVertical/2);
20.     END IF;
21.     IF(camera.orthographicSize > 11) THEN
22.           camera.position.y =
23.           Mathf.Lerp(camera.position.y, centerKotak1,
24.           Time.deltaTime*5);
25.     END IF;
26.     IF(distKotak >= 20 && distKotak <= 40) THEN
27.           IF(isZoomed == true || aboveNormal == true)
28.     THEN
29.                 isZoomed = !isZoomed;
30.                 aboveNormal = false;
31.           END IF;
32.     END IF;
33.     IF(distKotak <= 20 && isZoomed == false) THEN
34.           isZoomed = !isZoomed;
35.           aboveNormal = false;
```

```
36.         END IF;
37.         IF(distKotak >= 40 && aboveNormal == false) THEN
38.               isZoomed = false;
39.               aboveNormal = !aboveNormal;
40.         END IF;
41.         IF(isZoomed == true) THEN
42.               camera.orthographicSize =
43.               Mathf.Lerp(camera.orthographicSize,zoomArea,Tim
44.               e.deltaTime*smooth);
45.               camera.position.x =
46.               Mathf.Lerp(camera.position.x, centerKotak,
47.               Time.deltaTime*smooth)
48.         ELSE IF(aboveNormal == true) THEN
49.               camera.orthographicSize =
50.               Mathf.Lerp(camera.orthographicSize,aboveNormalV
51.               al,Time.deltaTime*smooth);
52.         ELSE
53.               camera.orthographicSize =
54.               Mathf.Lerp(camera.orthographicSize,normal,Time.
55.               deltaTime*smooth);
56.               camera.position.x =
57.               Mathf.Lerp(camera.transform.position.x,
58.               centerKotak, Time.deltaTime*smooth);
59.         END IF;
60.
61.         public void SetCam(GameObject p1a, GameObject p2a){
62.               other = p1a.transform;
63.               other2 = p2a.transform;}
64.   END
```

4.2.4 Implementasi Tutorial

Stage tutorial dikontrol dengan beberapa kondisi yang men-*trigger step* selanjutnya yang digunakan untuk membantu player dalam memahami cara bermain dalam game. Perintah dan object tutorial disimpan dalam bentuk gameobject yang nantinya di-*enable* oleh *script. Pseudo code* dari tutorial ditunjukkan pada Tabel 4.12 berikut ini :

Tabel 4. 12 Pseudo code Prosedur Tutorial

Pseudo Code tutorial.cs

```
DECLARATION
    AksiJalan is BOOL
    AksiPukul is BOOL
    AksiLoncat is BOOL
    AksiSkill is BOOL
    pickImage is GAMEOBJECT[]
```

```
1.  DESCRIPTION
2.  START
3.      IF (Input.GetButtonDown("JalanP1") && AksiJalan) THEN
4.          pickImage[0].SetActive(true);
5.          AksiJalan = false;
6.          finish++;
7.      END IF;
8.
9.      IF (Input.GetButtonDown("HitP1") && AksiPukul) THEN
10.         pickImage[2].SetActive(true);
11.         AksiPukul = false;
12.         finish++;
13.     END IF;
14.     IF (Input.GetButtonDown("JumpP1") && AksiLoncat) THEN
15.         pickImage[1].SetActive(true);
16.         AksiLoncat = false;
17.         finish++;
18.     END IF;
19.
20.     IF (Input.GetButtonDown("SkillP1") && AksiSkill) THEN
21.         pickImage[3].SetActive(true);
22.         AksiSkill = false;
23.         finish++;
24.     END IF;
25.     IF (Input.GetButtonDown("UltiP1") && AksiUlti) THEN
26.         pickImage[4].SetActive(true);
27.         AksiUlti = false;
28.         finish++;
29.     END IF;
30.
31. END
32.
```

4.3 Implementasi Art

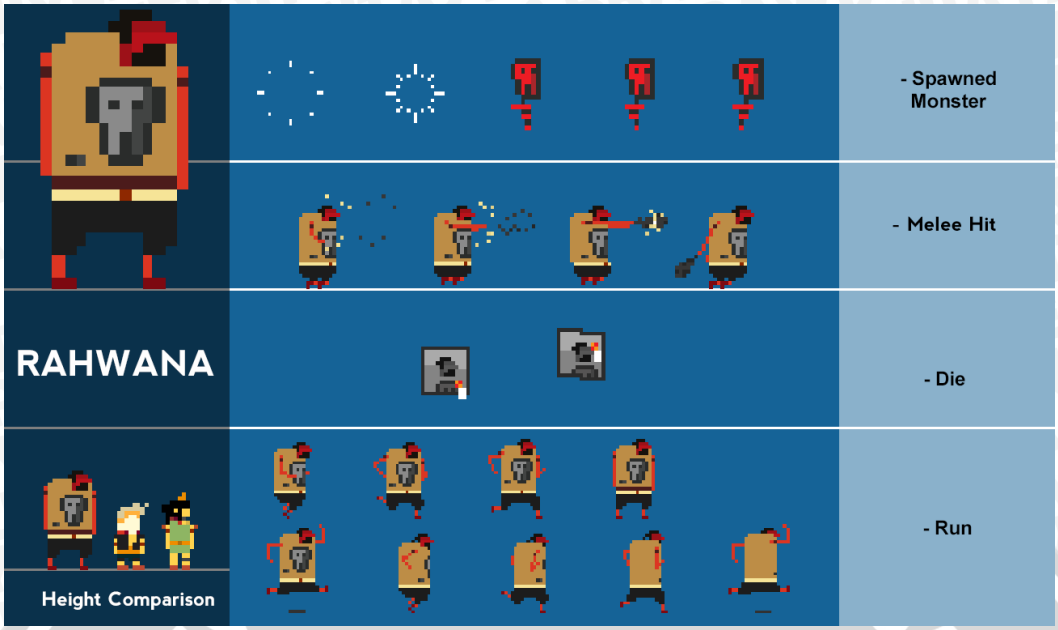
Dalam implementasi sisi art dari game lakoon, terdapat beberapa fokus utama dalam pembuatannya yaitu desain karakter, animasi karakter dalam hal ini menggunakan sprite, desain stage, desain menu, dan juga HUD.

4.3.1 Implementasi Desain Karakter

Desain karakter pada game Lakoon ini terdiri dari 4 karakter utama yaitu semar, srikandi, rahwana dan betara kala. Implementasi desain dari karakter semar dapat dilihat pada Gambar 4.4 berikut ini :

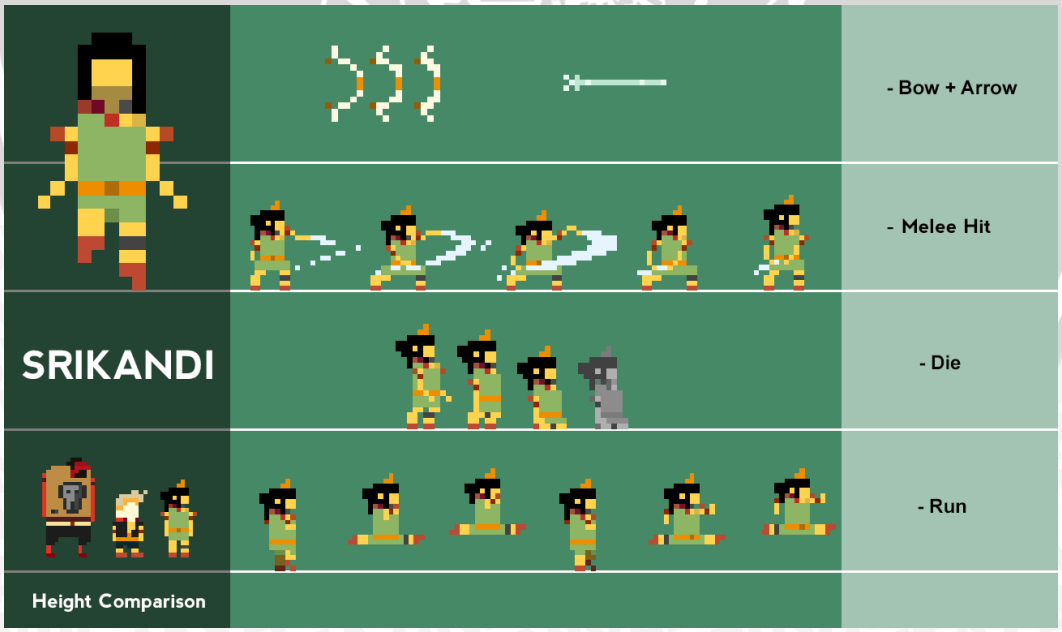
Gambar 4.4 Implementasi Sprite Semar

Desain asli dari semar dengan rambut putih dan jambul sebagai ciri khas karakter tersebut tetap dipertahankan pada desain karakter game lakoon ini. Namun ciri khas tersebut dimodifikasi dengan membuat *looks*-nya lebih muda dan lincah.



Gambar 4.5 Implementasi Sprite Dasamuka

Gambar 4.5 menunjukkan implementasi dari desain rahwana. Rahwana didesain dengan sosoknya yang besar dengan *looks* layaknya *gangster*. Sebagai sosok buto yang jahat, rahwana didesain dengan warna kulit merah yang membuatnya memiliki tampilan yang garang.



Gambar 4.6 Implementasi Sprite Srikandi

Gambar 4.6 menunjukkan implementasi desain dari karakter srikandi. Sebagai seorang pemanah, srikandi didesain sebagai sosok wanita lincah dengan panah dan busurnya. Sesuai dengan cerita dalam pewayangan yang mengkisahkan

tentang hidupnya di hutan kamikaya, srikandi didesain dengan warna hijau yang dapat menyamarkan dirinya dengan lingkungan sekitar.

Gambar 4.7 Implementasi Sprite Betara Kala

Gambar 4.7 menunjukkan implementasi desain dari karakter betara kala .Betara kala yang dalam pewayangan dikisahkan sebagai penguasa alam gaib didesain dengan tampilan yang seram dengan jubah hitamnya.

4.3.2 Implementasi Desain HUD

HUD (*Head-up Display*) merupakan komponen dari game berupa user interface yang berfungsi untuk menampilkan informasi dari game yang digunakan untuk membantu pemain untuk memenangkan permainan. Dalam game ini HUD yang akan ditampilkan adalah mana bar dan fire counter. Hasil implementasinya secara visual ditunjukkan pada Gambar 4.8.

Gambar 4.8 Implementasi HUD

4.3.3 Implementasi Desain Main Menu

Main menu adalah *starting point* pada sebuah game yang berisi opsi atau pilihan komponen dalam game. Dalam game Lakoon komponen-komponen yang ada pada *main menu* adalah :

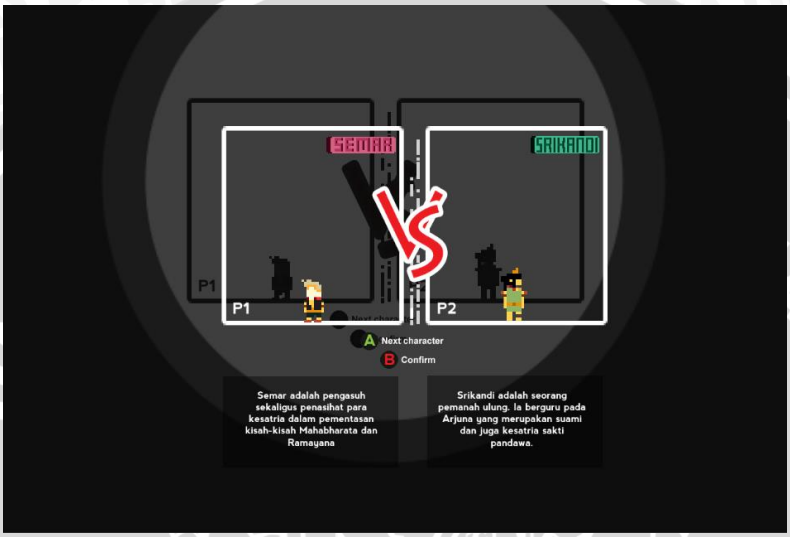
1. *Play*, pilihan ini digunakan untuk memulai permainan utama dalam game, dan membawa pemain pada tampilan *select character* untuk dapat memilih karakter yang akan dimainkan.
2. *Option*, digunakan untuk mengaktifkan atau menonaktifkan fungsi audio pada game.
3. *Tutorial*, menu ini akan aktif saat pemain berjalan melewati batas sebelah kanan pada menu stage. Pilihan ini akan membawa pemain pada stage tutorial yang berisi cara untuk memainkan game Lakoon ini, termasuk didalamnya informasi kontrol dan aturan dasar permainan.
4. *Credit*, menu ini akan aktif saat pemain berjalan melewati batas sebelah kiri pada menu stage. Pilihan ini akan membawa pemain pada stage credit yang berisi informasi tentang developer dari permainan ini.
5. *Exit*, menu ini akan aktif saat pemain masuk pada pintu bertuliskan exit. Pilihan ini digunakan untuk keluar dari permainan dan menutup aplikasi

Implementasi komponen desain scene menu awal dapat dilihat pada Gambar 4.9 berikut ini :

Gambar 4. 9 Implementasi Desain Menu Utama

4.3.4 Implementasi Desain Select Character

Pada scene ini pemain dapat memilih karakter yang akan dimainkan saat permainan dimulai. Kedua pemain dapat memilih character secara bersamaan pada scene yang sama. Ditampilkan pula deskripsi atau profil dari tiap karakter dalam game. Pemain juga dapat mencoba kontrol dasar gerak pada tiap karakter yang akan dipilih. Hasil implementasinya secara visual ditunjukkan pada Gambar 4.10.



Gambar 4. 10 Implementasi Desain Select Character

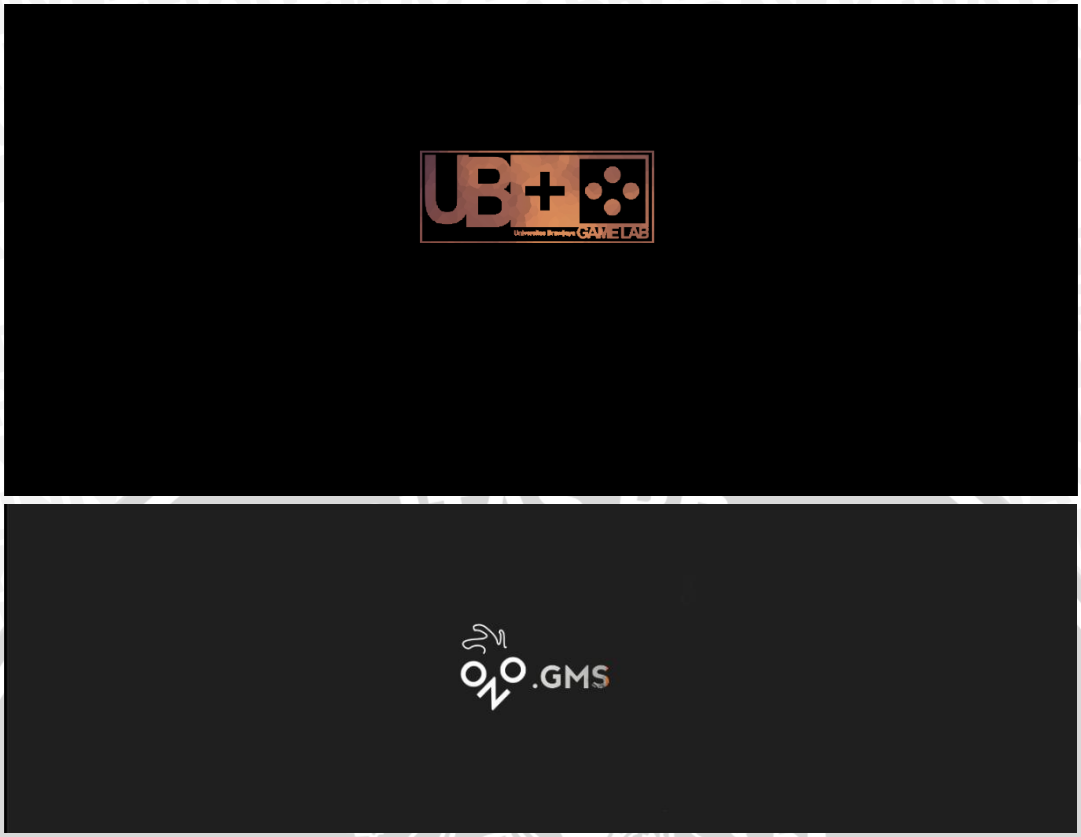
4.3.5 Implementasi Desain Select Stage

Pemain dapat memilih stage atau arena bertanding pada scene ini. Scene ini juga berisi deskripsi dari tiap stage yang ada. Pemain memilih satu dari sepuluh stage yang ada. Terdapat 5 jenis stage dalam game ini yang pada tiap jenis terdapat 2 versi arena. 5 stage itu adalah stage sunyaruri, stage kamikaya, stage alengka, stage menu, dan juga stage setra gandamayit. Implementasi dari desain dapat dilihat pada Gambar 4.11.

Gambar 4. 11 Implementasi Desain Select Stage

4.3.6 Scene Splash Screen

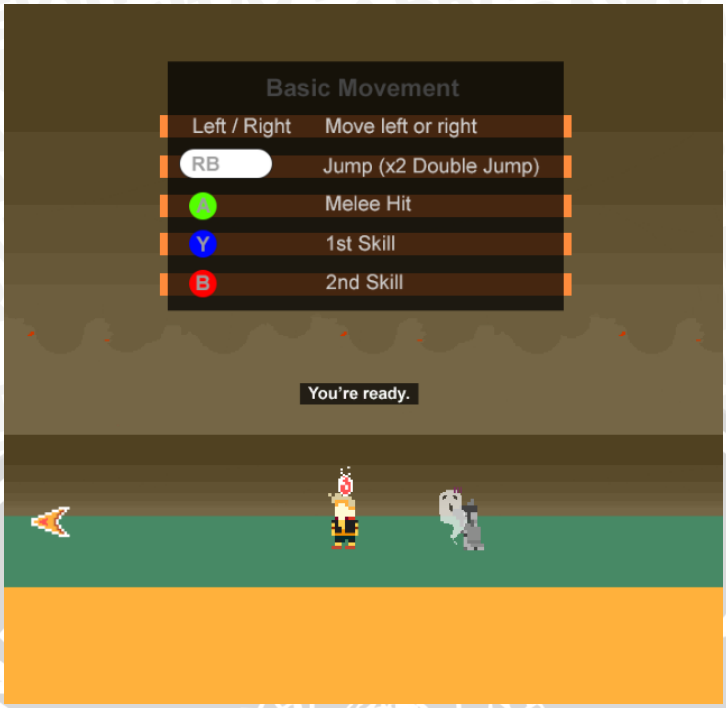
Splash screen merupakan scene yang terdapat pada awal game dijalankan dan biasanya menampilkan studio developer, sponsor, ataupun teknologi yang digunakan. Unity free license dalam hal ini akan selalu mencantumkan logonya di splash screen untuk game-game yang menggunakan unity sebagai game enginenya. Hasil implementasinya secara visual ditunjukkan pada Gambar 4.12.



Gambar 4. 12 Implementasi Scene Splash Screen

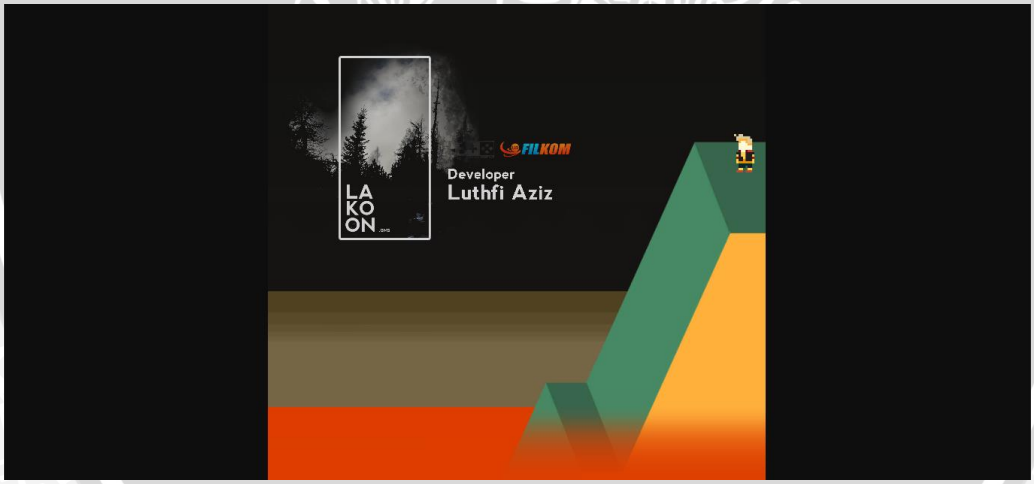
4.3.7 Tutorial

Saat player memasuki scene tutorial maka yang pertama diinformasikan adalah tentang kontrol key yang digunakan, dengan demo langsung terhadap gerakan-gerakan dasar pemain yaitu lari, pukul, loncat, dan juga mengaktifkan special skill 1 dan 2 karakter. Pada scene ini juga diterangkan tentang gameplay permainan yaitu bagaimana cara mengambil api, dan bagaimana cara membunuh pemain lain. Tutorial *guide* dimunculkan secara berurutan mulai dari dasar konfigurasi kontrol hingga sistem pertarungan. Player dapat meninggalkan scene ini setelah semua objective telah terpenuhi. Gambar 4.13 menunjukkan implementasi desain dari scene tutorial.



Gambar 4. 13 Implementasi Scene Tutorial

4.3.8 Credit



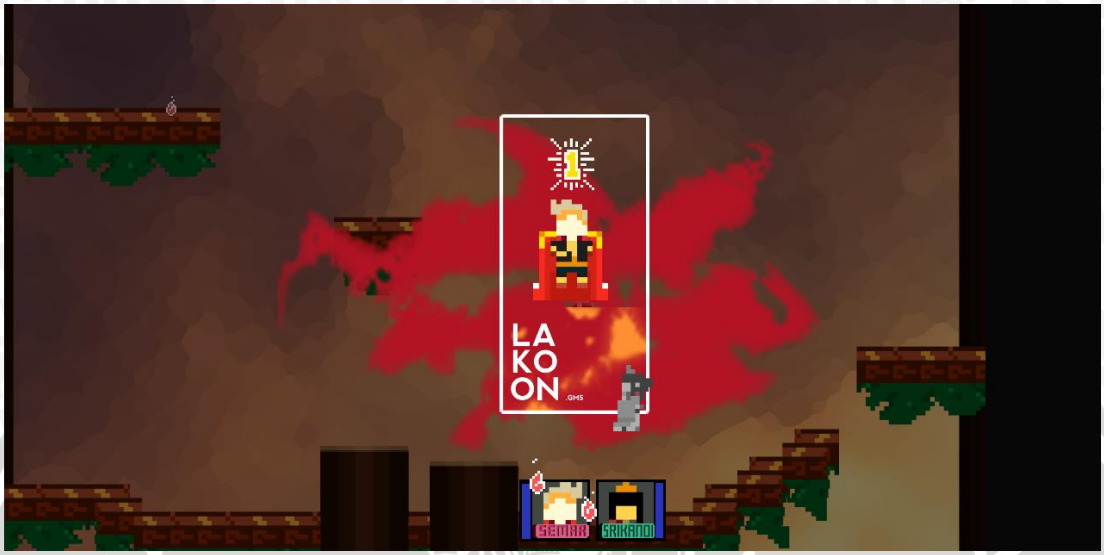
Gambar 4. 14 Implementasi Scene Credit

Gambar 4.14 menunjukkan implementasi dari scene credit yang berisi tentang informasi developer, studio game, dan juga sumber-sumber asset yang digunakan pada game ini.

4.3.9 Winning Screen

Ini adalah tampilan saat salah satu pemain berhasil mengumpulkan 3 api dan memenangkan permainan. Karakter pemenang dari permainan akan diberi jubah

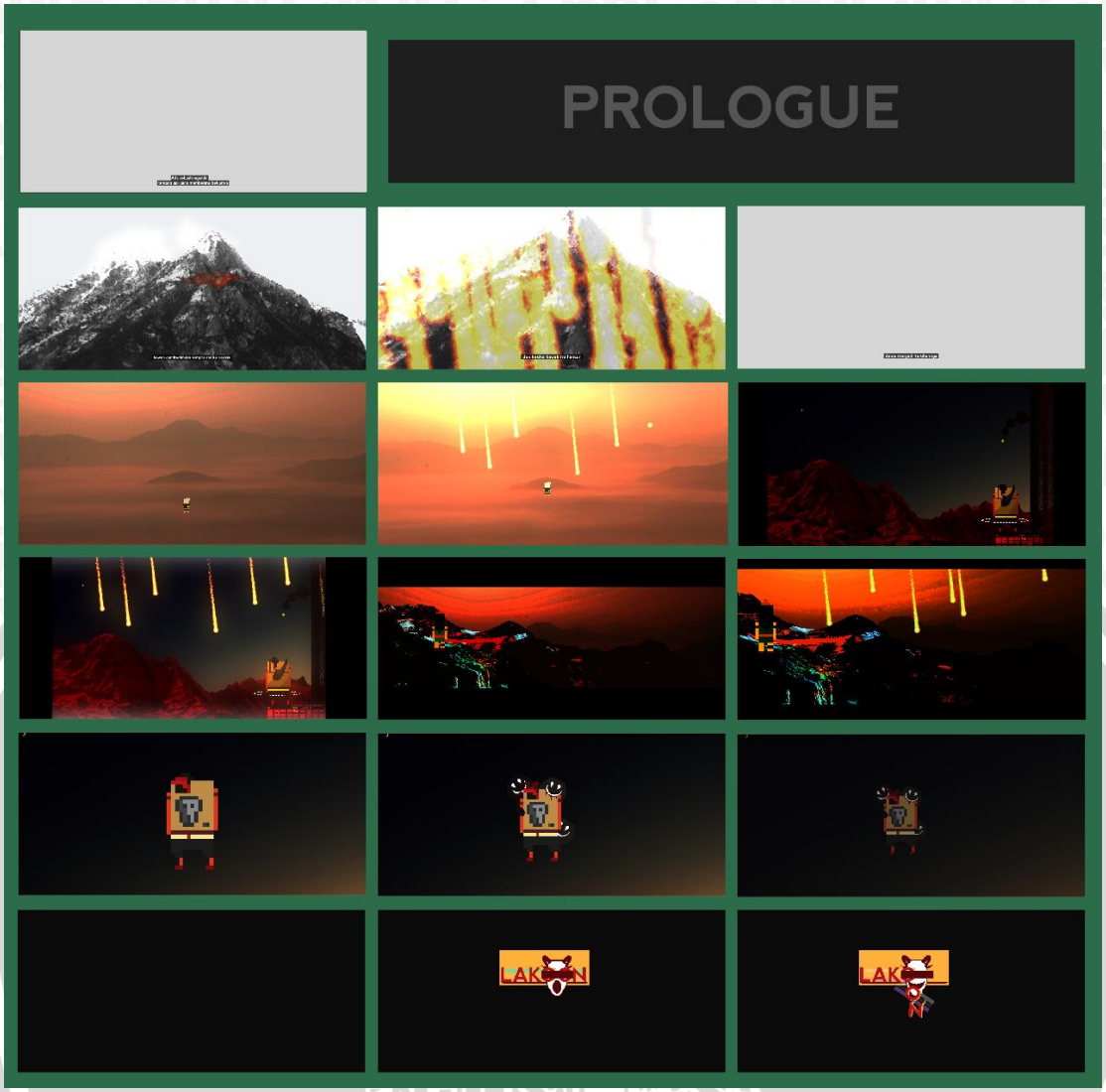
dan karakter kalah akan berada dibawahnya dalam state dead. Tampilan winning muncul di scene yang sama dengan stage berlangsungnya permainan. Tampilan dari winning screen dapat dilihat pada Gambar 4.15.



Gambar 4. 15 Implementasi Tampilan Menang

4.3.10 Implementasi Prolog

Prolog ditampilkan pada awal game dijalankan. Prolog berisi cerita tentang bagaimana latar belakang dari game ini berasal. Desain prolog berupa animasi 2d dan juga narasi singkat tentang cerita pada game. Narasi singkat tersebut adalah : "Ada sebuah legenda tentang api yang memberimu kekuatan. Kawah candradimuka tempat api itu berasal. Dan ketika kawah itu hancur, dunia menjadi taruhannya. Akankah menjadi terang, atau dalam kegelapan". Sedangkan desain prolog tersebut dapat dilihat pada Gambar 4.16 berikut ini :



Gambar 4. 16 Implementasi Scene Prolog

Scene ini mengisahkan tentang api yang konon katanya siapapun yang memilikinya dapat mempunyai kekuatan yang sangat hebat. Api ini terdapat di kawah candradimuka, dan saat gunung tempat kawah candradimuka berada itu hancur, maka api itupun ikut tersebar kemanana. Hingga setiap kesatria di dunia berlomba-lomba untuk mendapatkannya, entah itu dengan niat untuk kebaikan ataupun untuk kejahatan.

4.4 Implementasi Game Stage

Terdapat 10 stage dalam game Lakoon ini. Stage tersebut terdiri dari 5 tipe stage yang alam setiaptipenya mempunyai 2 versi arena. Kelima tipe itu adalah Sunyaruri, Kamikaya, Alengka, Menu stage, Setra Gandamayit.

4.4.1 Stage Sunyaruri

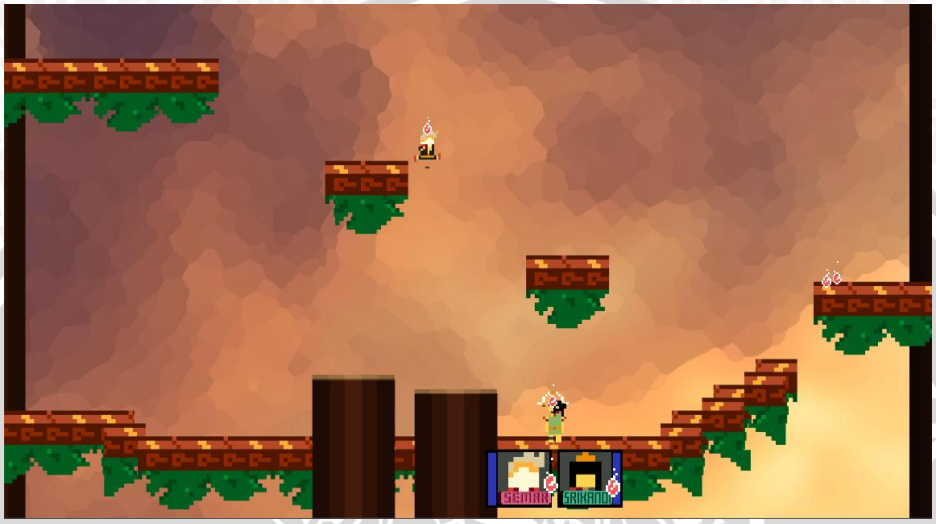
Gambar 4. 17 Implementasi Stage Sunyaruri

Gambar 4.17 menunjukkan inplementasi dari stage sunyaruri. Stage ini mempunyai 2 versi dengan kontur arena yang berbeda. Stage versi 1 mempunyai dunia yang luas dengan kontur berupa pijakan-pijakan batu tinggi yang dapat menampilkan tipe permainan dengan mobilitas tinggi. Sedangkan versi 2 lebih fokus pada kecekatan dan pertarungan jarak dekat. Sedangkan implementasi stage pada versi kedua ditunjukkan pada Gambar 4.18.

Gambar 4. 18 Implementasi Stage Sunyaruri II

4.4.2 Stage Kamikaya

Stage ini didesain dengan tampilan sesuai dengan nama dari stage ini yaitu Kamikaya. Kamikaya adalah sebuah hutan tempat Srikandi dan Arjuna pernah melarikan diri di dalamnya. Implementasinya ditunjukkan pada Gambar 4.19 berikut :



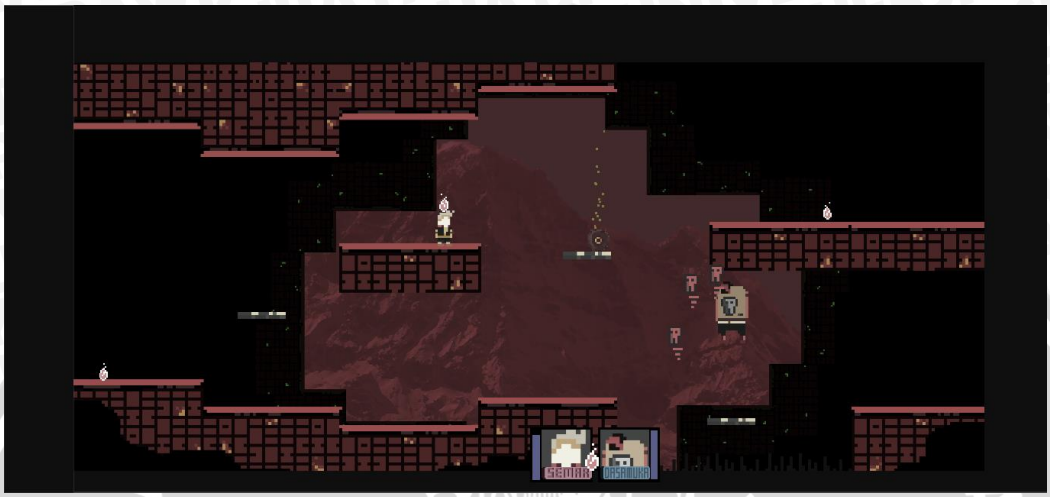
Gambar 4.19 Implementasi Stage Kamikaya

Kedua versi dari stage ini memiliki tipe arena yang relatif seragam, yaitu arena terbuka dengan obstacle yang minimal. Sedangkan implementasi stage pada versi kedua ditunjukkan pada Gambar 4.20.

Gambar 4. 20 Implementasi Stage Kamikaya II

4.4.3 Stage Alengka

Stage ini memiliki area yang sempit dan membuat rahwana yang dapat memanggil bala tentaranya mempunyai keuntungan bila bermain di stage ini. Imlementasi dari stage alengka ditunjukkan pada Gambar 4.21.



Gambar 4. 21 Implementasi Stage Alengka

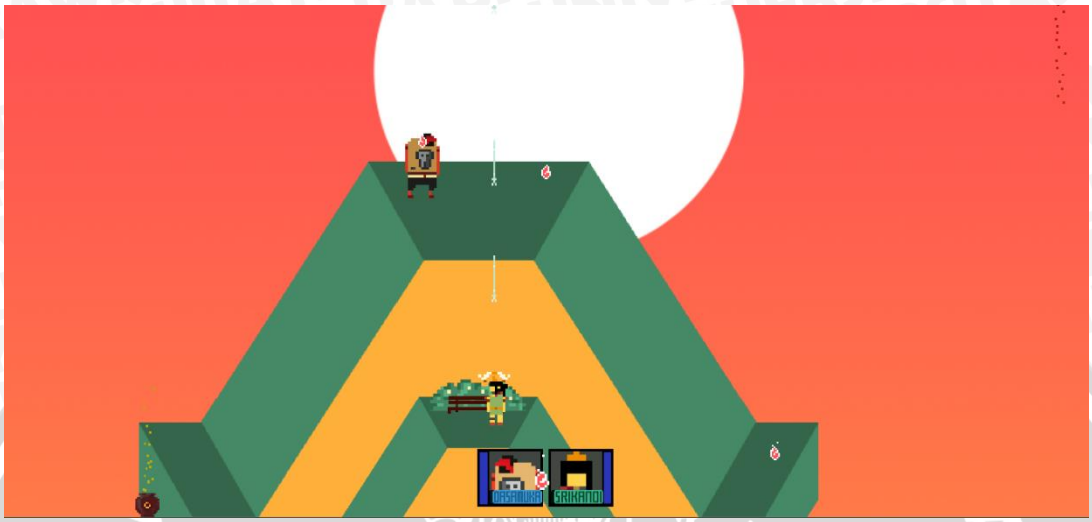
4.4.4 Stage Setra Gandamayit

Stage ini mempunyai tingkat kesulitan tersendiri yang unik, karena sumber cahaya hanya bersifat spotlight dengan intensitas dan radius yang kecil dan bergerak mengikuti masing-masing player. Hal ini membuat path atau kontur dari arena tidak terlihat secara utuh. Implementasinya dapat dilihat pada Gambar 4.22.

Gambar 4. 22 Implementasi Stage Setra Gandamayit

4.4.5 Stage Menu

Stage ini dibuat dengan dasar desain yang sama dengan desain menu awal. Stage ini memiliki luas yang medium dan obstacle yang tidak terlalu rumit. Implementasi pada stage menu ditunjukkan pada Gambar 4.23.



Gambar 4. 23 Implementasi Stage Menu

4.5 Implementasi Sound Effect, Suara dan Music

Audio yang digunakan pada game ini merupakan free license audio dengan license minimal yaitu *Licensed under Creative Commons Attribution 4.0 International*. Audio license ini berisi tentang kebebasan pengguna untuk memakai audio source baik untuk pribadi maupun komersial dengan pencantuman credit terhadap pencipta.

Daftar *audio* yang digunakan pada *game* Lakoon adalah sebagai berikut:

1. **Prolog :**

a. http://www.freesound.org/people/BrainClaim/sounds/273804/

b. http://www.freesound.org/people/Mortifreshman/sounds/237210/

c. Music used: Battle of Kings by Per Kiilstofte : https://machinimasound.com/music/battle-of-kings

2. **Main Game :**

a. iMakeSoundFX: http://www.reddit.com/r/gamedev/comments/2l5irc/76_sfx_to_download_constantly_growing_also_taking/

BAB V
PENGUJIAN DAN ANALISIS

Pada bab ini akan dibahas proses pengujian terhadap game yang telah dibuat. Pengujian tersebut terdiri dari 2 jenis pengujian, yaitu pengujian *white box* dan pengujian *black box*. Pada pengujian white box, game akan diuji menggunakan unit testing. Sedangkan pada pengujian black box akan dilakukan combinatorial testing dan juga test flow diagrams.

5.1 White Box Testing

Pengujian white box dengan unit testing ini menggunakan teknik pengujian basis path testing. Teknik ini dilakukan untuk dapat mendefinisikan basis path dari unit yang diuji.

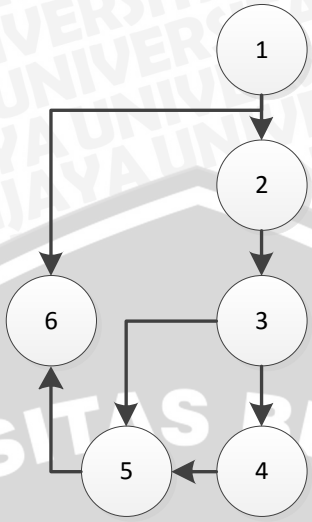
5.1.1 Check Hit Character Unit Testing

Check hit berfungsi untuk mendeteksi tumbukan yang terjadi pada karakter merupakan bentuk tumbukan hit atau hanya tumbukan biasa dengan collider lain. Script ini juga berfungsi untuk menjatuhkan api bila karakter yang terkena hit memilikinya. *Pseudo code* dari check hit ditunjukkan pada Tabel 5.1.

Tabel 5.1 Pseudocode CekHit

Pseudo Code Cek Hit	
DECLARATION apiClone is GAMEOBJECT SkorApi is INT	
	DESCRIPTION
1.	START
	IF (collider hitbox bertabrakan dengan collider
2.	dengan tag pukul) THEN
	apiClone <- Find ("fire") on Parent;
3.	transform.parent.position =
4.	OtherLocation[Rand()];
	IF (apiClone) THEN
5.	SkorApi--;
6.	apiClone.parent <- null;
	END IF;
	END IF;

	END



Gambar 5.1 *Flow Graph* Prosedur Cek hit

Dari flow graph yang ditunjukkan pada Gambar 5.1 akan dihitung *cyclomatic complexity-nya* dengan menggunakan persamaan V(G) = E – N + 2, dimana V(G) merupakan nilai *cyclomatic complexity*, sedangkan E adalah sisi atau garis penghubung antar *node* dan N merupakan jumlah *node.*

V(G) = E – N + 2

= 7 – 6 + 2

= 3

Berdasarkan dari nilai *cyclomatic complexity* maka 3 basis path yang didapat adalah :

Jalur 1: 1-6

Jalur 2: 1-2-3-5-6

Jalur 3: 1-2-3-4-5-6

Didapat tes case yang dijelaskan pada Tabel 5.2 berikut ini :

Tabel 5.2 Tabel Kasus Uji Untuk Pengujian *Check Hit*

Jalur	Kasus uji	Hasil yang diharapkan	Hasil yang didapatkan	Status
1	Karakter bertumbukan dengan collider	Tidak terjadi apa-apa	Tidak terjadi apa-apa terhadap karakter	Valid

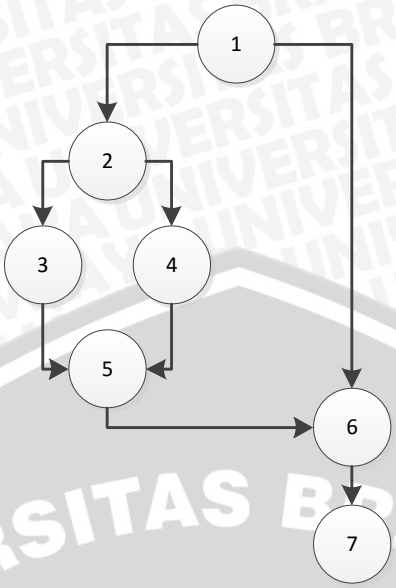
	dengan tag bukan pukul	terhadap karakter		
2	Karakter bertumbukan dengan collider dengan tag pukul dan tidak memiliki api	Karakter pindah tempat dan tidak ada api yang jatuh dari karakter	Karakter pindah tempat dan tidak ada api yang jatuh dari karakter	Valid
3	Karakter bertumbukan dengan collider dengan tag pukul dan memiliki api	Karakter pindah tempat dan api yang dimilikinya jatuh	Karakter pindah tempat dan api yang dimilikinya jatuh	Valid

5.1.2 Pengujian Unit Kondisi Menang

Salah satu pemain dinyatakan sebagai pemenangnya ketika api yang didapatnya sudah mencapai 3 buah. *Pseudo code* dari pengecekan kondisi menang ditunjukkan pada Tabel 5.3.

Tabel 5.3 Pseudocode Kondisi Menang

```
Pseudo Code Kondisi Menang
DECLARATION
    MasterScript is MASTERSCRIPT
    charStats is INT
    DESCRIPTION
    START
1.      IF (charStats [charApi, 0] == 3) THEN
2.          IF(charApi == 0) THEN
3.              MasterScript.winning(Player1Name);
4.          ELSE
                MasterScript.winning(Player2Name);
5.          END IF;
6.          WinStage.SetActive(true);
7.      END IF;
    END
```



Gambar 5.2 *Flow Graph* Kondisi *Menang*

Dari flow graph yang yang ditunjukkan pada Gambar 5.2 akan dihitung *cyclomatic complexity-nya* dengan menggunakan persamaan V(G) = E – N + 2, dimana V(G) merupakan nilai *cyclomatic complexity*, sedangkan E adalah sisi atau garis penghubung antar *node* dan N merupakan jumlah *node.*

V(G) = E – N + 2

= 8 – 7 + 2

= 3

Berdasarkan dari nilai *cyclomatic complexity* maka 3 basis path yang didapat adalah :

Jalur 1: 1-6-7

Jalur 2: 1-2-3-5-6-7

Jalur 3: 1-2-4-5-6-7

Didapat tes case yang dijelaskan pada Tabel 5.4 berikut ini :

Tabel 5.4 Tabel Kasus Uji Untuk Pengujian *Menang*

Jalur	Kasus uji	Hasil yang diharapkan	Hasil yang didapatkan	Status
1	Karakter belum mempunyai api 3	Tidak akan terjadi apa-apa	Tidak terjadi apa-apa	Valid

2	Karakter mempunyai 3 api dan merupakan player 1	Muncul tampilan pemenang permainan yaitu player 1	Muncul tampilan pemenang permainan yaitu player 1	Valid
3	Karakter mempunyai 3 api dan merupakan player 2	Muncul tampilan pemenang permainan yaitu player 2	Muncul tampilan pemenang permainan yaitu player 2	Valid

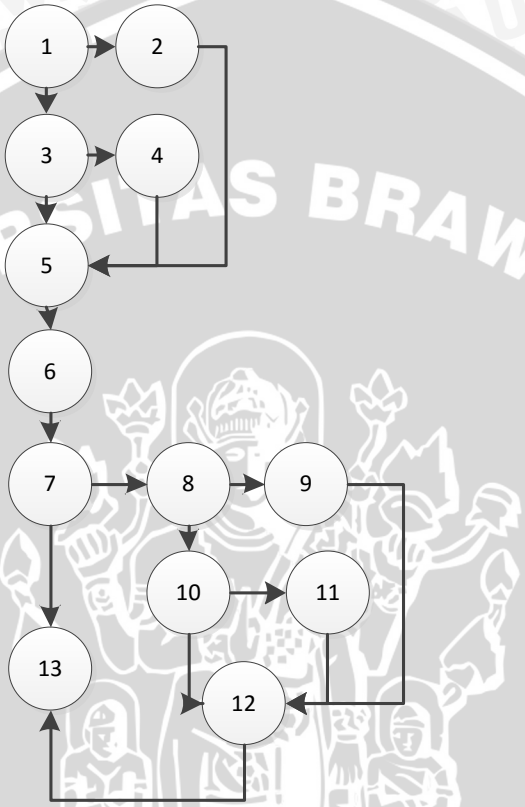
5.1.2 Pengujian Unit Gerak

Script ini berfungsi untuk mengkontrol dasar gerakan yang dapat dilakukan oleh tiap karakter. *Pseudo code* dari proses gerak ditunjukkan pada Tabel 5.5.

Tabel 5.5 Pseudocode Gerak

```
Pseudo Code Gerak
DECLARATION
    count is INT
    jumpButton is STRING
    kecepatan is FLOAT
    power is FLOAT
    DESCRIPTION
    START
1.      IF (Input axis < 0) THEN
2.          transform.scale.x = -1.0f;
3.      ELSE IF (Input axis > 0) THEN
4.          transform.scale.x = 1.0f;
5.      END IF;
6.      transform.position += transform.right * kecepatan *
        nilai input axis * Time.deltaTime;
7.      IF(Input.GetButtonDown(jumpButton)) THEN
8.          IF(tanah) THEN
9.              rigidbody2D.AddForce(transform.up*power);
                count++;
10.         ELSE IF(!tanah && count == 1) THEN
```

```
11.                 rigidbody2D.AddForce(transform.up*power);
                    count++;
12.             END IF;
13.         END IF;
    END
```



Gambar 5.3 *Flow Graph* Gerak

Dari flow graph yang yang ditunjukkan pada Gambar 5.3 akan dihitung *cyclomatic complexity-nya* dengan menggunakan persamaan V(G) = E – N + 2, dimana V(G) merupakan nilai *cyclomatic complexity*, sedangkan E adalah sisi atau garis penghubung antar *node* dan N merupakan jumlah *node.*

V(G) = E – N + 2

= 17 – 13 + 2

= 6

Berdasarkan dari nilai *cyclomatic complexity* maka 6 basis path yang didapat adalah :

Jalur 1: 1-2-5-6-7-13

Jalur 2: 1-3-4-5-6-7-13

Jalur 3: 1-3-5-6-7-13

Jalur 4: 1-3-5-6-7-8-9-12-13

Jalur 5: 1-3-5-6-7-8-10-11-12-13

Jalur 6: 1-3-5-6-7-8-10-12-13

Didapat tes case yang dijelaskan pada Tabel 5.6 berikut ini :

Tabel 5.6 Tabel Kasus Uji Untuk Pengujian *Trigger System*

Jalur	Kasus uji	Hasil yang diharapkan	Hasil yang didapatkan	Status
1	Input button kiri, tidak loncat	Karakter bergerak ke kiri	Karakter bergerak ke kiri	Valid
2	Input button kanan, tidak loncat	Karakter bergerak ke kanan	Karakter bergerak ke kanan	Valid
3	Tidak ada inputan button gerak	Tidak terjadi apa-apa	Tidak terjadi apa-apa	Valid
4	Karakter di atas tanah, input loncat	Karakter loncat ke atas	Karakter loncat ke atas	Valid
5	Karakter di udara, input loncat, count = 1	Karakter loncat di udara	Karakter loncat di udara	Valid
6	Karakter di udara, input loncat, count != 1	Karakter tidak loncat	Karakter tidak loncat	Valid

5.2 Pengujian *Black Box*

Pada game ini pengujian black box dilakukan dengan menggunakan metode combinatorial testing dan juga test flow diagram.

5.2.1 Combinatorial Testing

Parameter yang akan diuji adalah jenis karater, mana status, skill, hit target dan juga api status. Nilai parameter secara keseluruhan ditunjukkan pada tabel 5.7.

Tabel 5.7 Tabel Gameplay Combinatorial

No	Character	Skill	Hit Target	Mana Stats
1	Semar	Melee Hit	Semar	Empty
2	Srikandi	Skill 1	Srikandi	Full
3	Dasamuka	Skill 2	Dasamuka	Full
4	Semar	Skill 1	Dasamuka	Empty
5	Srikandi	Skill2	Semar	Empty
6	Dasamuka	Melee Hit	Srikandi	Empty
7	Semar	Skill 2	Srikandi	Full
8	Srikandi	Melee hit	Dasamuka	Full
9	Dasamuka	Skill 1	Semar	Full

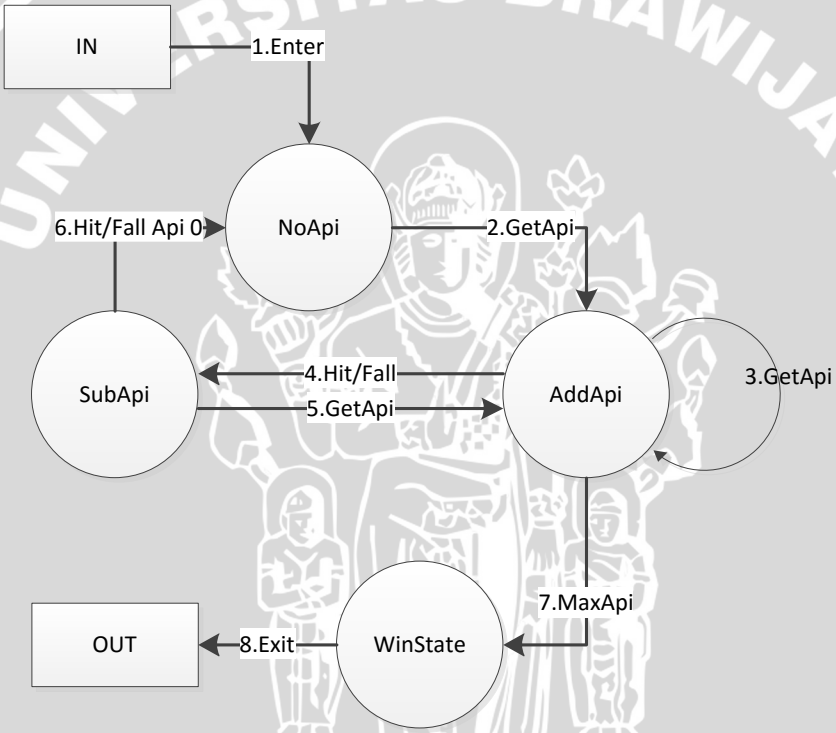
Semua case pada Tabel 5.7 akan diuji untuk menemukan elemen yang tidak bekerja sebagaimana mestinya. Hasil dari pengujian tersebut adalah sebagai berikut :

1. Pada kombinasi ini semar dapat melakukan melee hit pada hit target. Hit target menampilkan state mati setelah collider hit mengenainya. Mana tidak mempengaruhi skill, karena melee hit tidak menggunakan mana untuk mengeluarkannya.
2. Srikandi dengan mana full dapat mengeluarkan skill 1 nya. Hit target tidak mengalami tumbukan, karena skill 1 srikandi merupakan non-attacking skill.
3. Pada kombinasi ketiga, dasamuka berpindah tempat dengan hit target sesuai dengan skill 2 dari semar yaitu swap location.
4. Dengan mana empty semar tidak dapat mengeluarkan skill 1 nya.
5. Srikandi juga tidak dapat mengeluarkan skill 2 nya saat mana status menunjukan empty.
6. Melee hit tidak memerlukan mana poin untuk dapat mengeluarkannya, sehingga dasamuka dapat menggunakanya dan hit target srikandi berada pada state mati ketika terkena collider hit.

7. Skill 2 semar merupakan non-attacking skill, sehingga hit target tidak berganti state. Skill dapat dikeluarkan karena mana status menunjukkan full.
8. Srikandi berhasil mengeluarkan melee hitnya pada hit target dasamuka.
9. Dasamuka dengan mana full dapat mengeluarkan skill 1 nya yaitu summon pasukan. Hit target memasuki state mati ketika pasukan dasamuka bertumbukan dengannya.

Berdasarkan dari hasil pengujian dari beberapa kemungkinan pada tabel 5.7, game bekerja sebagaimana mestinya dan tidak ditemukan kesalahan.

5.2.2 Api TFD



Gambar 5.4 Test Flow Diagram Api

TFD api pada Gambar 5.4 menunjukan state dimana pemain memulai permainan dengan tidak mempunyai api, hingga state menang. State menang diperoleh ketika player mendapat *MaxApi* atau dalam hal ini 3 api. Data dictionary dari TFD tersebut ditunjukkan pada Tabel 5.8 sebagai berikut :

Tabel 5.8 Tabel *Dictionary TFD* Api

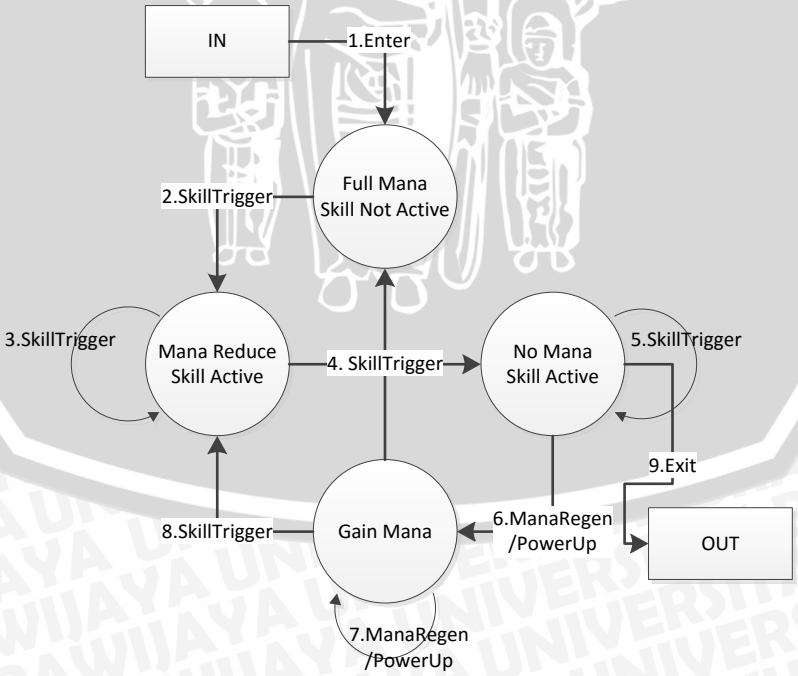
Events	**Deskripsi**

Enter	Game dimulai dengan inisiasi karakter dan set location
GetApi	Pemain bertumbukan dengan collider api dan status api +1
Hit/fall	Pemain mati dan api jatuh
MaxApi	Pemain mengambil api sebanyak 3 api
Exit	Permainan selesai

Dari TFD tersebut didapatkan path sebagai berikut :

1. Flow : 1-2-3-3-7-8 (Minimun Path)
   Pemain mendapatkan maximal api dengan jalur paling cepat tanpa mengalami reduksi poin akibat terkena hit.
2. Flow : 1-2-3-4-5-3-7-8
   Pemain mengalami reduksi poin akibat terkena, setelah mendapatkan api. Api yang akan didapatkan akan terjatuh dan dapat diambil pemain lain.
3. Flow : 1-2-4-6-2-3-7-8
   Pemain tereduksi hingga tidak mempunya poin atau api, sehingga harus memulai state dari path 2 kembali.

5.2.2 Skill TFD



Gambar 5.5 Test Flow Diagram Skill

TFD skill pada Gambar 5.5 menunjukan state pemain dimulai dengan status mana penuh hingga mana habis. Data dictionary dari TFD tersebut ditunjukkan pada tabel 5.9 sebagai berikut :

Tabel 5.9 Tabel *Dictionary TFD* Skill

Events	**Deskripsi**
Enter	Game dimulai dengan inisiasi karakter dan set location
SkillTrigger	Input skill
ManaRegen	Karakter mendapat mana regen sebanyak 1 poin/detik
PowerUp	PowerUp keluar setiap 20 detik sekali dan menambah 10 poin

1. Flow : 1-2-3-4-5-9 (Minimun Path)

   Pemain dapat kehabisan mana saat pemakaian skill terlalu cepat untuk dikeluarkan
2. Flow : 1-2-3-4-5-6-7-8-4-9

   Penambahan mana bisa melalui 2 cara yaitu mana regeneration ataupun power up mana.

BAB VI
KESIMPULAN DAN SARAN

6.1 Kesimpulan

Berdasarkan pengamatan yang dilakukan selama proses perancangan, implementasi, dan pengujian, maka didapat kesimpulan sebagai berikut:

1. Pada proses perancangan, paper prototyping dan digital prototyping sangat dibutuhkan untuk dapat melihat bug, kenyamanan permainan dan *balance* dari game yang dibuat. Proses ini ditunjang pula oleh beberapa tahapan lainnya yaitu pendefinisian game pada *formal elements* dan juga *concept art* dari karakter, *environment,* maupun HUD dalam game.
2. Game Lakoon ini adalah permainan bergenre fighting yang dimainkan secara multiplayer pada platform dekstop PC. Game ini berlatar belakang pada tokoh-tokoh yang dikisahkan pada cerita wayang Ramayana dan juga Mahabarata.
3. Hasil pengujian pada game Lakoon ini menunjukan bahwa aplikasi dapat berjalan dengan baik sesuai dengan rancangan dan kebutuhan yang diharapkan. Secara fungsionalitas, game Lakoon juga dapat menumbuhkan rasa keingintahuan pemain terhadap budaya Jawa khususnya budaya wayang.

6.2 Saran

Saran untuk pengembangan game Lakoon lebih lanjut adalah:

1. Permainan multiplayer dapat dilakukan secara online dan tidak hanya terbatas pada 1 platform saja.
2. Penambahan karakter wayang yang dapat dipilih, khususnya tokoh-tokoh karakter asli indonesia seperti petruk, gareng, bagong, antareja atau antasena.
3. Penambahan stage yang dapat dipilih, dengan rintangan yang unik daripada stage lain.
4. Penambahan pilihan gameplay lain yang dapat dipilih oleh penguna. Seperti pilihan single player yang membawa pemain pada petualangan mencari api dengan melewati rintangan-rintangan dalam stage.

5. Bila telah dapat dilakukan multiplayer secara online, maka dapat diimplementasikan pembuatan game pada platform lain seperti platform android, ios, windows phone, nintendo, ps4, xbox, dan lain sebagainya.

DAFTAR PUSTAKA

[CRS-04] Kohler, Chris. (2004). Power Up: How Japanese Video Games Gave the World an Extra Life. London: BradyGames.

[RZM-12] Aizid, Rizem. (2012). Atlas Tokoh Tokoh Wayang. Yogyakarta: Diva Press

[PRR-88] Prabhu, K., Rao, R. (1988) The Mind of Mahatma Gandhi. Austin: Greenleaf Books.

[GCM-15] Vaccari, Giacomo. What Game Engine To Use? A Beginner Game Developer Primer. 27 Februari 2014; *http://giacomovaccari.tumblr.com/post/1838 0022743/what-game-engine-to-use-a-beginner-game-developer* [20 April 2015].

[WLN-13] Agissa, Wildan. White Box Testing & Black Box Testing. 01 Mei 2013; http://bangwildan.web.id/berita-176-white-box-testing--black-box-testing.html [4 Oktober 2014].

[KNR-11] Patel, Kaner. Gaming Is The New Mass Media. 06 April 2011; http://adage.com/article/special-report-digital-conference/ea-ceo-riccitiello-gaming-mass-media/226832/ [23 Februari 2014].

[SHZ-13] Schwartz, Dan. Cook Triblet Inlet Tribal Council To Share Culture, Educate Youth Through Video Games. 2 September 2013; http://peninsulaclarion.com/news/2013-09-02/cook-inlet-tribal-council-to-share-culture-edducate-youth-through-video-games [15 Maret 2015].

[JPR-03] Juul, Jesper. The Game, The Player, The World: Looking For A Heart of Gameness. 6 November 2003. Keynote At Level Up Conference. Utrecht.

[MSP-NN] Game History; http://www.museumofplay.org/icheg-game-history/timeline/ [4 Oktober 2014].

[GGS-NN] Game Genres; http://www.mobygames.com/glossary/genres [4 Oktober 2014].

[UNI-NN] Learn About Unity; http://unity3d.com/learn [4 Oktober 2014].

[MSK-NN] Mukokuseki; http://tvtropes.org/pmwiki/pmwiki.php/Main/Mukokuseki [4 Oktober 2014]

[GMD-09] Second Lesson, Game Design: Iterative Rapid Prototyping. 7 Februari 2009; http://gamedesignconcept.wordpress.com/2009/07/02/level-2-game-design-iteration-and-rapid-prototyping/ [15 Mei 2015].

[PSP-12] Mengenal Software Editing Foto Adobe Photoshop. 23 September 2012; http://belajar-komputer-mu.com/mengenal-software-editing-foto-adobe-photoshop/ [4 Oktober 2014].

[BPT-NN] Basis Path Testing; http://users.csc.calpoly.edu/~jdallbey/206/Lectures/BasisPathTutorial/ [4 Oktober 2014].

[CBT-NN] Combinatorial Testing; http://web.niaccist.niacc.edu/~milleste/classroom/testingconcepts/section7/ [22 Oktober 2015]

[HTP-14] Trieddiantoro, Hendro. Wayang Kulit Terancam Punah. Januari 2014; http://www.researchgate.net/publication/272505006_Wayang_Kulit_Terancam_Punah [22 Oktober 2015]