TUGAS AKHIR – TJ 141502

PEMBARUAN DATA PERMAINAN BERBASIS *DOWNLOADABLE CONTENT* PADA PERMAINAN *ACTION TOWER DEFENSE : SAGA LEGACY*

Achmad Ridlo Nuur Abdillah
NRP 07211340000031

Dosen Pembimbing
Dr. Supeno Mardi Susiki Nugroho, ST., MT.
Dr. Eko Mulyanto Yuniarno, ST., MT.

Departemen Teknik Komputer
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2018

FINAL PROJECT – TJ 141502

DOWNLOADABLE CONTENT-BASED GAME DATA UPDATE IN ACTION TOWER DEFENSE GAME : SAGA LEGACY

Achmad Ridlo Nuur Abdillah
NRP 07211340000031

Supervisors
Dr. Supeno Mardi Susiki Nugroho, ST., MT.
Dr. Eko Mulyanto Yuniarno, ST., MT.

Department of Computer Engineering
Faculty of Electrical Technology
Sepuluh Nopember Institute of Technology
Surabaya 2018

PERNYATAAN KEASLIAN
TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya dengan judul "Pembaruan Data Permainan Berbasis *Downloadable Content* pada Permainan *Action Tower Defense : Saga Legacy*" adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan karya pihak lain yang saya akui sebagai karya sendiri

Semua refrensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka.

Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, Januari 2018

Achmad Ridlo Nuur Abdillah
NRP. 07211340000031

LEMBAR PENGESAHAN

Pembaruan Data Permainan Berbasis *Downloadable Content* pada Permainan *Action Tower Defense : Saga Legacy*

Tugas Akhir ini disusun untuk memenuhi salah satu syarat memperoleh gelar Sarjana Teknik di Institut Teknologi Sepuluh Nopember Surabaya

Oleh: Achmad Ridlo Nuur Abdillah (NRP: 07211340000031)

Tanggal Ujian : 4 Januari 2018 Periode Wisuda : Maret 2018

Disetujui oleh:

(Pembimbing I)

Dr. Supeno Mardi Susiki Nugroho, ST., MT.
NIP: 196906131997021003 .....

(Pembimbing II)

Dr. Eko Mulyanto Yuniarno, ST., MT.
NIP: 196806011995121009 .....

(Penguji I)

Prof. Dr. Ir. Yoyon K. Suprapto, M.Sc.
NIP: 195409251978031001 .....

(Penguji II)

Eko Pramunanto, ST., MT.
NIP: 196612031994121001 .....

(Penguji III)

Dr. Adhi Dharma Wibawa, ST., MT.
NIP: 197605052008121003 .....

Mengetahui
Kepala Departemen Teknik Komputer

Dr. I Ketut Eddy Purnama, ST., MT.
NIP: 196907301995121001

ABSTRAK

Nama Mahasiswa : Achmad Ridlo Nuur Abdillah
Judul Tugas Akhir : Pembaruan Data Permainan Berbasis *Downloadable Content* pada Permainan *Action Tower Defense : Saga Legacy*
Dosen Pembimbing : 1. Dr. Supeno Mardi Susiki Nugroho, ST., MT
2. Dr, Eko Mulyanto Yuniarno, ST., MT.

Dalam pengembangan sebuah *game*, banyak hal yang harus diperhatikan untuk membuat pengalaman bermain tetap menarik. Mulai dari jalan cerita sampai mekanisme *gameplay*. Salah satu cara paling efektif agar pemain tetap tertarik memainkan *game* yang kita buat adalah melakukan pembaruan secara berkala. Hal ini dapat memberikan pengalaman baru secara terus menerus kepada pemain. Namun, terkadang untuk merilis pembaruan, pengguna diharuskan untuk mengunduh ulang keseluaruhan aset pada *game* tersebut. Hal ini dirasa kurang efektif apabila pengembang hanya menginginkan untuk mengubah data berukuran *kilobyte* saja. Oleh karena itu, perlu adanya sistem yang dapat membantu pengembang merilis pembaruan secara cepat. Dengan sistem pengelolaan data permainan waktu nyata, pengembang dapat merilis pembaruan tanpa mengharuskan pemain untuk mengunduh ulang pada *client* pada distributor bahkan saat pemain sedang memainkan *game* tersebut. Memungkinkan pemain untuk memilih konten mana yang ingin disimpan.

Kata kunci: *Downloadable Content*, *Seamless Update*, *Real Time Database*

Halaman ini sengaja dikosongkan

ABSTRACT

Name : Achmad Ridlo Nuur Abdillah
Title : *Downloadable Content-Based Game Data Update in Action Tower Defense Game : Saga Legacy*
Advisors : 1. Dr. Supeno Mardi Susiki Nugroho, ST., MT.
2. Dr. Eko Mulyanto Yuniarno, ST., MT.

In the development of a game, many things should be noticed to make the playing experience stay interactive. From the storyline to the gameplay mechanism. One of many effective method to make players stay playing game that we developed is regular update.This could make brand new experience continuously to players. But, sometimes to release an update, user must re-download entire assets in game. This is not effective if developer only needs to change data that sized only several kilobytes. Therefore, a system needed to help developer release a quick update. With Real-time Game Data Management System, the developer could release an update without constrain players from re-download the client in distributor, even when player is playing the game. Giving player the ability to choose which content they want to keep.

Keywords: Downloadable Content, Seamless Update, Real Time Database

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Puji dan Syukur kehadirat Tuhan YME atas segala limpahan berkah, serta rahmat-Nya, penulis dapat menyelesaikan tugas akhir ini dengan judul : **Pembaruan Data Permainan Berbasis *Downloadable Content* pada Permainan *Action Tower Defense : Saga Legacy*.**

Tugas akhir ini disusun dalam rangka pemenuhan bidang riset di Jurusan Teknik Komputer ITS, Bidang Studi Game dan Perangkat Mobile, serta digunakan sebagai persyaratan menyelesaikan pendidikan S-1. Tugas akhir ini dapat terselesaikan tidak lepas dari bantuan berbagai pihak. Oleh karena itu, penulis mengucapkan terima kasih kepada:

1. Keluarga, Ibu, Bapak dan Saudara yang telah memberikan dorongan spiritual dan material serta seluruh kerabat dan kolega penulis yang banyak membantu proses dalam menyelesaikan buku penelitian ini.
2. Bapak Dr. I Ketut Eddy Purnama, S.T., M.T. selaku Kepala Departemen Teknik Komputer, Fakultas Teknologi Elektro, Institut Teknologi Sepuluh Nopember
3. Secara khusus penulis mengucapkan terima kasih yang sebesar-besarnya kepada Bapak Supeno Mardi Susiki Nugroho, ST., MT. dan Bapak Eko Mulyanto Yuniarno, ST., MT. atas bimbingan selama mengerjakan penelitian.
4. Bapak-ibu dosen pengajar Bidang Studi Game dan Perangkat mobile, atas pengajaran, bimbingan, serta perhatian yang diberikan kepada penulis selama ini.
5. Seluruh teman-teman angkatan e-53, rekan tim Agni Monkey, rekan Laboratorium Visi Komputer.

Kesempurnaan hanya milik Tuhan, untuk itu penulis memohon segenap kritik dan saran yang membangun serta menghatur maaf atas segala kekurangan yang ada dalam penulisan buku ini.

Surabaya, September 2017
Penulis

Halaman ini sengaja dikosongkan

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

Halaman sengaja dikosongkan

DAFTAR ISTILAH

DLC	Konten yang dapat diunduh
AssetBundle	Aset yang dapat dimuat saat game berjalan
JSON	Format pertukaran data yang ringan
Listener	Prosedur atau fungsi dalam program komputer yang menunggu suatu peristiwa terjadi
Enum	Tipe data terdiri dari satu set nilai bernama yang disebut elemen, anggota, enumeral, atau enumerator dari tipe
Build	versi pra-rilis dari suatu program atau berkas
Mesh	koleksi simpul, tepi dan wajah yang mendefinisikan bentuk benda polyhedral dalam grafis komputer 3D
Instance	Kejadian konkret dari objek apapun, biasanya ada saat program computer berjalan
UI	Tampilan pengguna pada permainan

Halaman sengaja dikosongkan

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Mengembangkan permainan yang terhubung internet pada zaman sekarang merupakan hal yang sudah biasa bagi para pengembang permainan. Google dan Apple sebagai perusahaan teknologi maju saat ini, telah menyediakan banyak fasilitas bagi pengembang aplikasi maupun permainan dalam mengembangkan karya mereka. Menurut data dari www.statista.com, pada tahun 2017, tercatat, sebanyak 3 juta aplikasi dan permainan yang ada pada Google Play Store[1] dan sebanyak 2.2 juta aplikasi dan permainan yang tercatat pada App Store[2]. Melalui data tersebut, dapat ditarik kesimpulan bahwa pengembang permainan dan aplikasi saat ini berlomba-lomba dalam membuat karya yang bagus sehingga diminati banyak *user*.

Sebagai pengembang permainan, untuk mempertahankan pengalaman bermain tetap menarik bagi *user* tentunya harus melakukan *update* permainan secara berkala agar permainan yang dimainkan oleh *user* tidak membosankan. Sedangkan bagi sisi *user*, dalam suatu *update* konten permainan sangat merepotkan apabila harus memasang ulang file permainan tersebut untuk mengimplementasi *update* dari pengembang. Hal tersebut kurang efektif dan efisien dalam pengembangan suatu permainan, apalagi jika update untuk *maintenance* permainan hanya hitungan *kilobyte* data saja. Selain itu, permasalahan yang lain ialah *DLC* atau *Downloadable Content*. Sistem *DLC* ini diterapkan oleh *Electronic Arts* pada permainan *The Sims Series*, dimana *user* harus memasang terlebih dahulu keseluruhan *DLC* yang ada sebelum dapat memainkannya. Hal ini akan menjadi masalah untuk permainan *mobile* dimana *DLC* menjadi satu paket dengan permainan walaupun *user* tidak membutuhkan *DLC* tersebut. Tentu para pengembang permainan membutuhkan sebuah sistem yang bisa menangani masalah manajemen data tersebut kepada *user*.

Pada pengerjaan tugas akhir ini diharapkan dapat menerapkan metode *Seamless Update* untuk memanajemen data pada permainan. Dengan sistem manajemen data yang berjalan dinamis pada permainan, maka user tidak perlu lagi repot mengunduh ulang file permainan tersebut apabila terdapat *update*. Dalam menerapkan *Seamless Update* tersebut

memerlukan *database* yang berjalan secara *real-time* dalam pengoperasiannya. Nilai data yang ada pada permainan akan disimpan pada komputasi awan pada *server*, sehingga permainan akan otomatis *update* apabila terjadi perubahan data dan data pada permainan akan dinamis terhadap *update* yang ada.

1.2 Permasalahan

Penerapan pembaruan data permainan akan merepotkan apabila mengharuskan *user* untuk merestart atau menghentikan permainan. Hal tersebut dapat merusak pengalaman bermain pengguna karena mereka harus menghentikan permainan terelebih dahulu, apalagi jika pembaruan tersebut sering terjadi dan dalam ukuran yang sangat kecil.

1.3 Tujuan

Tujuan dan manfaat dari penelitian ini adalah untuk menerapkan sistem pengelolaan data permainan secara waktu nyata berbasis DLC dalam permainan action tower defense Saga Legacy, sehingga data pada permainan menjadi dinamis dan responsif terhadap *update*. Dengan begitu akan memudahkan pengguna apabila terdapat *update*, user tidak perlu lagi untuk mengunduh ulang atau menghentikan permainan.

1.4 Batasan Masalah

Batasan masalah pada penelitian ini adalah:

1. Penelitian diimplementasikan untuk permainan yang sudah pernah dibuat sebelumnya oleh peneliti.
2. Penelitian ini diimplementasikan pada *video game* dengan *platform Android*.

1.5 Sistematika Penulisan

Laporan penelitian tugas akhir ini tersusun dalam sistematika dan terstruktur sehingga lebih mudah dipahami dan dipelajari oleh pembaca maupun seseorang yang hendak melanjutkan penelitian ini. Alur sistematika penulisan laporan penelitian ini yaitu :

1. Bab I Pendahuluan
   Bab ini berisi uraian tentang latar belakang, permasalahan, tujuan,

metodologi, dan sistematika laporan.

2. Bab II Tinjauan Pustaka
   Pada bab ini berisi tentang uraian secara sistematis teori-teori yang berhubungan dengan permasalahan yang dibahas pada tugas akhir ini. Teori-teori ini digunakan sebagai dasar dalam tugas akhir, yaitu *Remote Configuration*, *Database Management System*, *Difficulty Adjustment* dan teori-teori penunjang lainya.
3. Bab III Desain dan Implementasi Sistem
   Bab ini berisi tentang penjelasan terkait sistem yang dibuat. Guna mendukung itu digunakanlah diagram alir, *flowchart*, dan *pseudocode* agar sistem mudah dipahami dan diimplementasikan.
4. Bab IV Pengujian dan Analisa
   Bab ini menjelaskan tentang pengujian yang dilakukan terhadap sistem dalam penelitian ini dan menganalisa sistem. Spesifikasi perangkat keras dan perangkat lunak yang digunakan juga disebutkan dalam bab ini. Tujuannya adalah sebagai variabel kontrol dari pengujian yang dilakukan.
5. Bab V Penutup
   Bab ini merupakan penutup yang berisi kesimpulan yang diambil dari penelitian dan pengujian yang telah dilakukan. Saran dan kritik yang membangun untuk pengembangkan lebih lanjut juga dituliskan pada bab ini.

Halaman ini sengaja dikosongkan

BAB 2
TEORI PENUNJANG

Untuk mendukung penelitian dalam tugas akhir ini, dibutuhkan beberapa teori penunjang sebagai bahan acuan dan referensi. Dengan demikian penelitian ini lebih terarah.

2.1 DLC (*Downloadable Content*)

DLC disebut sebagai konten download yang sangat populer dan sering dipublikasikan untuk video game. Mereka dibangun sebagai perpanjangan atau *add-on* dari game, terutama terdiri dari misi, peta, senjata, karakter dan fitur baru yang digunakan untuk menemani permainan yang sudah dirilis. Mereka tidak dilepas sebagai paket ekspansi ritel namun didistribusikan melalui jaringan Internet dan konsol seperti jaringan *Xbox Live* dan *PlayStation (PSN).*

Konten yang dapat diunduh umumnya digunakan untuk mendukung setiap game individual dari genre dan dirancang khusus untuk game tersebut, sehingga cerita tambahan atau jalur misi dapat muncul dalam bentuk konten yang dapat didownload. Salah satu elemen kunci yang membuat DLC begitu populer adalah kenyataan bahwa kebanyakan dari mereka gratis dan mudah didapat atau diunduh. Kumpulan DLC ini dibuat untuk memungkinkan pemain mengakses semua konten yang dapat didownload dari permainan tertentu, namun sadar betul bahwa video tersebut tidak segera tersedia saat rilis game dan tidak semuanya tersedia untuk diunduh setelah dipromosikan dan ditawarkan untuk dijual.

Selanjutnya, untuk mendapatkan pemahaman yang lebih baik tentang paket DLC, DLC terdiri dari beberapa elemen yang dibuat untuk meningkatkan pengalaman bermain dan membuat pemain menikmati permainan lebih jauh lagi dengan memainkan konten tambahan, tidak hanya pemain memiliki elemen *multiplayer* seperti peta , karakter dan senjata pemain juga memiliki elemen pemain tunggal seperti misi, quest samping, lokasi baru dan alur cerita baru yang ada agar Anda lebih terbiasa dengan permainan, game ini pada gilirannya membuat permainan lebih awet dan menyenangkan sekaligus meningkatkan waktu hidup. nilai permainan

Kelebihan dan kekurangan yang harus diperhatikan untuk *downloadable content* adalah:

1) Keuntungan dari konten yang dapat didownload adalah:
    a. Mereka memperpanjang permainan dan umur panjang dari permainan dengan membiarkan pemain bermain dengan isi tambahan yang disediakan.
    b. Diskon yang dilampirkan pada Kumpulan DLC (pass season) adalah bonus dengan keuntungan signifikan bagi pemain atau konsumen karena mereka dapat memperoleh keseluruhan konten yang dapat diunduh dengan harga diskon.
    c. Bagi penerbit, hal itu memungkinkan mereka memperoleh pendapatan cepat dari *pre-order* tanpa segera mengirimkan konten DLC.
2) Kelemahan utama adalah:
    a. Biaya uang, seringkali banyak walaupun kontennya terkesan berharga.
    b. Beberapa DLC pada dasarnya menjual konten yang ada dalam permainan yang dapat Anda mainkan untuk dibuka, namun pada dasarnya Anda membayar untuk membukanya.
    c. Isi DLC sering dipertanyakan mengapa mereka tidak berada dalam kampanye utama sebuah game sejak awal dimana Anda harus membayar untuk memajukan kisah permainan.
    d. Setelah membeli "*Season Pass*", beberapa bagian DLC masih dibuat untuk dibeli.[3]

DLC tidak sama dengan permainan yang dapat diunduh. DLC tidak dapat diunduh karena itulah mereka sering disebut "*Add-On*". DLC tidak sama dengan User Generated Contents (UGC) karena ini mengacu pada konten dalam game yang dapat diedit atau dibuat oleh pemain sedangkan konten yang dapat didownload adalah konten tambahan yang hanya dapat ditambahkan ke permainan.

2.2 Desain Permainan *Saga Legacy*

Pada sub-bab ini, akan dijelaskan mengenai desain *game Saga Legacy* berkaitan tentang skenario dan struktur data masing-masing objek yang akan terintegrasi dengan database waktu nyata. Kisah pada *Saga Legacy* menggunakan nama dari tokoh-tokoh pada cerita rakyat, namun kisah karakter sudah dimodifikasi sehingga sesuai dengan alur cerita utama *Saga Legacy*.

2.2.1 *Storyline*

Ini adalah tahun 2016. Kehidupan berjalan seperti biasa. Namun, sinar kegelapan menyelimuti langit dan petir menyambar. Suatu sosok muncul dalam kegelapan itu membawa pasukan monster mengerikan. Dialah Ken Arok, sang penguasa tanah Jawa pendiri Kerajaan Singasari. Dengan suatu cara, Ken Arok dipanggil ke masa kini dan Ia bersumpah untuk menguasai Indonesia dengan pasukan Iblisnya. Gelombang serangan datang silih berganti di berbagai kota di Indonesia dan hampir seluruh kota sudah tumbang ke tangan Ken Arok.

Namun tiba-tiba di langit kota Surabaya, muncul cahaya yang sama ketika Ken Arok datang namun kali ini nampak berbeda. Sinar berwarna keemasan terpancar dan dari dalamnya nampak sesosok wanita cantik. Dia adalah Ken Dedes. Ia muncul di hadapan ketiga *arek-arek* Surabaya yaitu: Rhoma, Wayan, Jalil. Mereka masing-masing diberikan sebuah mustika yang dapat memanggil roh kesatria dari masa lalu untuk berperang melawan pasukan Ken Arok. Kelima muda-mudi yang berasal dari etnis yang berbeda-beda ini pun bersatu dan bertekad membebaskan tanah air dari cengkeraman Ken Arok.

Perjalanan mereka pun dimulai. Mereka mulai membebaskan kota Surabaya yang dijaga oleh Raja Buto Ijo yang ditugaskan oleh Ken Arok. Satu per satu *landmark* kota pun harus mereka kuasai untuk melemahkan kekuatan sang Raja sebelum mengalahkannya. Jalan cerita akan terus berlanjut hingga ke kota-kota besar di seluruh Indonesia. Ksatria baru akan terus diperkenalkan seiring jalannya cerita.

2.2.2 Mekanisme *Game*

Gameplay yang diusung dalam game ini adalah *Role Playing Game* (RPG), dimana pemain memegang peranan penting dalam

pengembangan karakter dan jalan cerita. Pemain berperang melawan karakter yang datang dari masa lalu.

Gameplay pada *Saga Legacy* memadukan *action*, *strategy*, dan *role playing* dengan permainan serba cepat yang mengharuskan pemain bisa bertarung sekaligus mengatur strategi untuk berperang melawan pasukan dari masa lalu. Dipadukan dengan jalan cerita yang menarik serta reward berupa *loot* maupun status memberikan kepuasan pada pemain dalam memainkan game ini. Elemen *Action* murni menjadi inti dari permainan ini. Pemain dapat menggerakkan berbagai karakter *hero* yang merupakan ksatria legendaris. Dari *hero* tipe *melee, ranged,* hingga *magic* dan dipadukan dengan kombinasi gerakan yang dinamis menjadikan permainan ini menyenangkan.

Mekanisme game adalah *tower defense* dimana stage akan selesai apabila pemain dapat menghancurkan *tower* dari kubu musuh. Pemain dapat memilih karakter ksatria yang ada di dalam game. Pemain akan menggerakan karakter ksatria untuk menghancurkan musuh dan akan dibantu oleh *troop* yang merupakan *summon* dari tower kubu pemain. *Troop* terdiri dari *Front Line, Healer, Ranged Troop*.

Pemain akan menggunakan berbagai sumber daya yang ada untuk menaklukan stage tersebut mulai dari *troop, skill,* dan *item*. Troop dibeli menggunakan uang yang berjatuhan disetiap detik pada *Stage*. Maka dibuatlah sebuah pengembangan mekanika *game* dengan mengambil tema pahlawan Indonesia.

2.2.3 Status Karakter

Setiap *hero* mempunyai status karakter yang unik. Status tersebut dapat diubah oleh pengembang menggunakan *real-time database*. Beberapa status karakter yang dapat ditingkatkan tersebut yang pertama adalah Attack Point dengan ikon yang dapat dilihat pada gambar 2.1.

Gambar 2.1 Ikon *Attack Point*

Attack point merupakan nilai serangan yang dapat dilakukan oleh karakter. Nilai ini dapat di *upgrade* dengan *gem* atau uang. Semakin besar serangan yang dipunyai oleh *hero*, maka semakin mudah pula pemain dalam membunuh musuh yang ada. Status yang lainnya adalah *Defense Poin*t dengan ikon seperti pada gambar 2.2.

Gambar 2.2 Ikon *Defense Point*

Defense point merupakan nilai pertahanan. Nilai pertahanan ini berfungsi sebagai pertahanan dari karakter. Semakin besar nilai pertahanan, maka *hero* akan semakin kuat dalam menerima serangan dari musuh. Status yang lain adalah *Health Point* dengan ikon seperti pada gambar2.3.

Gambar 2.3 Ikon *Health Point*

Health point merupakan nilai maksimum *Health Point* yang ada pada *hero*. Semakin besar nilai *Health Point*, maka *hero* akan semakin susah untuk mati. Selain status pada karakter, permainan juga memiliki mekanik disebut *Soul System* yang merupakan poin untuk melawan musuh.

Soul merupakan sebuah “mata uang” pada game ini dalam mengeluarkan pasukan. *Soul* akan terus bertambah seiring berjalannya waktu, hingga mencapai jumlah tertentu dimana pemain dapat menggunakannya untuk *summon* pasukan. Ikon *Soul System* dapat dilihat pada gambar 2.4.

Gambar 2.4 Ikon *Soul System*

Pertambahan soul seiring berjalan waktu juga dapat di *upgrade* agar *soul* akan semakin cepat bertambah. Disini pemain dituntut untuk berpikir, sebaiknya ia menggunakan *soul* untuk *summon* pasukan, atau *upgrade* kecepatan pertambahan *soul*. Sedangkan scoring pada permainan ini direpresentasikan dengan jumlah bintang yang dapat dilihat pada gambar 2.5.

Gambar 2.5 Ikon *Scoring System*

Sistem *scoring* yang diterapkan di *game* ini adalah berdasarkan satuan waktu dalam menyelesaikan *stage*. Semakin cepat selesai maka semakin besar juga skor yang didapat, begitu juga sebaliknya. Maka dari itu pemain dituntut untuk memanfaatkan *soul* dan penggunaan *skill* pada *hero* secara efektif.

2.2.4 Karakter

Dalam Saga Legacy terdapat berbagai macam karakter yang diperkenalkan dari mitos dan sejarah di Indonesia kepada pemain. Karakter-karakter tersebut digolongkan menjadi beberapa bagian sebagai berikut.

1) Karakter Kesatria (Pelindung)

Sementara ini, terdapat 3 buah karakter utama dalam game ini: Aji Saka, Sakera, Arya Kasimpar yang masing-masing melambangkan pahlawan legendaris daerah. Pahlawan ini di *summon* oleh Rhoma, Jalil, Wayan. Setiap kesatria memiliki role yang berbeda-beda dengan masing-masing 2 *skill Active* dan 2 *Skill Passive* yang dapat membantu di dalam permainan. Selain itu tiap kesatria memiliki starting status point yang berbeda-beda.

Secara garis besar beberapa role yang dimasukkan ke dalam game ini adalah:

1. *Swordsman*
2. *Rogue*
3. *Mage*
4. *Fighter*
5. *Ranger*

1. Aji Saka *(Class: Swordsman)*

Aji Saka merupakan karakter pertama yang dapat dimainkan oleh pemain. Ajisaka menggunakan pedang untuk mengalahkan musuh-musuhnya dan memiliki tingkat serangan dan pertahanan yang seimbang sehingga cocok digunakan pemula.

Ilustrasi karakter Aji Saka dapat dilihat pada gambar 2.6.

Gambar 2.6 Ilustrasi karakter Aji Saka

a. Sejarah Singkat

Disebutkan Aji Saka berasal dari Bumi Majeti. Bumi Majeti sendiri adalah negeri antah-berantah mitologis, akan tetapi ada yang menafsirkan bahwa Aji Saka berasal dari Jambudwipa (India) dari suku Shaka (Scythia), karena itulah ia bernama Aji Saka (Raja Shaka). Legenda ini melambangkan kedatangan Dharma (ajaran dan peradaban Hindu-Buddha) ke pulau Jawa. Akan tetapi penafsiran lain beranggapan bahwa kata Saka adalah berasal dari istilah dalam Bahasa Jawa *saka* atau *soko* yang berarti penting, pangkal, atau asal-mula, maka namanya bermakna "raja asal-mula" atau "raja pertama". Mitos ini mengisahkan mengenai kedatangan seorang pahlawan yang membawa peradaban, tata tertib dan keteraturan ke Jawa dengan mengalahkan raja raksasa jahat yang menguasai pulau ini. Legenda ini juga menyebutkan bahwa Aji Saka adalah pencipta tarikh Tahun Saka, atau setidak-tidaknya raja pertama yang menerapkan sistem kalender Hindu di Jawa. Kerajaan Medang Kamulan mungkin merupakan kerajaan pendahulu atau dikaitkan dengan Kerajaan Medang dalam catatan sejarah.

b. Aji Saka di Saga Legacy

Aji Saka merupakan pelindung dari karakter Rhoma, seorang arek asli Surabaya yang saat itu di serang oleh Buto ketika bermain di kawasan Tugu Pahlawan. Ken Dedes yang menyelamatkan Rhoma memberikan sebuah mustika Aji Saka kepadanya yang ketika digunakan dapat memanggil Aji Saka kapanpun ketika nyawa Rhoma terancam. Bersenjatakan pedang legendaris Medang Kamulan, dan jiwa seorang pemimpin, Aji Saka membabat habis lawan-lawannya di medan pertempuran. Bersama Rhoma, ia berkeliling kota Surabaya dan mencari arek-arek lainnya.

c. Status dan *Skill* Aji Saka

Aji Saka merupakan seorang kesatria dari *class Swordsman*, yang berarti dia mahir menggunakan pedang. Senjata andalannya adalah pedang Medang Kamulan yang dapat memberikan kekuatan bagi pemilik dan sekutu di sekitarnya. Ilustrasi Aji Saka dalam permainan dapat dilihat pada gambar 2.7.

Gambar 2.7 Karakter Aji Saka dalam permainan

Berikut adalah status awal Aji Saka ketika masih level 1

1) *Attack Point* : 50
2) *Health Point* : 250

3) *Defense Point* : 30
4) Attack Rate : 1.5 s
5) Movement Speed : 8

Aji Saka memiliki 2 skill aktif dengan ikon yang dapat dilihat pada gambar 2.8.


Gambar 2.8 Ikon skill Aji Saka

1) Morale Boost
Skill ini dapat meningkatkan damage sementara untuk Aji Saka dan *troop* di sekitarnya. Status: *Active, Buff.* Durasi*: 20s, Cooldown: 40s.*

2) HP Regen
Skill ini mengeluarkan aura yang dapat memulihkan HP Aji Saka dan *troop* di sekitarnya. Status: *Active, Buff.* Durasi*: 5s, Cooldown: 25s*.

2. Sakera *(Class: Rogue)*

a. Sejarah Singkat

Sakera adalah seorang tokoh pejuang yang lahir di kelurahan Raci Kota Bangil, Pasuruan, Jatim, Indonesia. Ia berjuang melawan penjajahan Belanda pada awal abad ke-19. Sakera adalah seorang jagoan daerah yang melawan penjajah Belanda di perkebunan tebu Kancil Mas Bangil. Legenda jagoan berdarah Bangil ini sangat populer di Jawa Timur utamanya di Pasuruan dan Madura. Sakera yang bernama asli Sadiman adalah golongan ningrat yang di sebut dengan kalas MAS, berlatar belakang Islam yang amat sholeh dan pekerja keras. Profesinya sebagai mandor di perkebunan tebu milik pabrik gula kancil Mas Bangil. Ia

dikenal sebagai seorang mandor yang baik hati dan sangat memperhatikan kesejahteraan para pekerja, sehingga dijuluki Pak Sakera. Sakera adalah pejuang yang anti penjajahan.

Suatu saat setelah musim giling selesai, pabrik gula tersebut membutuhkan banyak lahan baru untuk menanam tebu. Karena kepentingan itu orang Belanda ambisius untuk membeli lahan perkebunan yang seluas-luasnya dan dengan harga semurah-murahnya. Dengan cara yang licik, orang Belanda itu menyuruh Carik Rembang untuk bisa menyediakan lahan baru untuk Perusahaan dalam jangka waktu singkat dan murah, dengan iming-iming harta dan kekayaan. Sehingga Carik Rembang bersedia memenuhi keinginan tersebut. Carik Rembang pun menggunakan cara-cara kekerasan kepada rakyat dalam mengupayakan tanah untuk perusahaan.

Sakera pun selalu membela rakyat, karena sikap ketidakadilan yang berkali kali dilakukan oleh Carik Rembang. Sehingga Carik Rembang melaporkan hal ini kepada pemimpin perusahaan. Pemimpin perusahaan marah dan mengutus wakilnya *Markus* untuk membunuh *Sakera*. Suatu hari pekerja sedang istirahat di perkebunan, *Markus* marah-marah dan menghukum para pekerja serta menantang *Sakera*. *Sakera yang* mengetahui hal ini, marah dan membunuh *Markus* serta pengawalnya di kebon tebu. Sejak saat itu *Sakera* menjadi buronan polisi pemerintah Hindia Belanda. Suatu saat ketika *Sakera* berkunjung ke rumah ibunya, ia dikeroyok oleh Carik *Rembang* dan polisi Belanda. Karena ibu *Sakera* diancam akan dibunuh maka, *Sakera* akhirnya menyerah, dan dimasukkan ke penjara Di Bangil.

Siksaan demi siksaan dilakukan polisi Belanda kepada *Sakera* setiap hari. Selama dipenjara *Sakera* selalu kangen dengan Keluarga dirumahnya. Berbeda dengan *Sakera* yang berjiwa besar, sementara *Sakera* ada dipenjara. Perlawananpun tetap dimulai, Carik Rembang dibunuh dan dilanjutkan dengan menghabisi para petinggi perkebunan yang memeras rakyat. Bahkan kepala polisi Bangil pun ditebas tanganya dengan 'Clurit' Senjata khas yang digunakan *Sakera*, *k*etika mencoba menangkap *Sakera*.

Dengan cara yang licik pula polisi Belanda mendatangi teman seperguruan *Sakera* yang bernama *Aziz* untuk mencari kelemahan *Sakera*. Dengan iming-iming akan diberi imbalan kekayaan oleh *Government* Belanda di Bangil. Aziz menjebak Sakera dengan licik, akhirnya *Sakera* pun terjebak dan dilumpuhkan ilmunya dengan memukulkan "Janur Kuning" ke badannya. Lagi-lagi Belanda berhasil menangkap kembali

Sakera yang kemudian diadili oleh *Government* di Bangil dan diputuskan untuk dihukum gantung. *Sakera* gugur digantung di Penjara Bangil dan Ia dimakamkan di Bekacak, Kelurahan, daerah paling selatan Kota Bangil - Pasuruan - JawaTimur.

b. Sakera di *Saga Legacy*

Sakera merupakan pelindung dari Jalil, arek asli Madura. Jalil yang diberikan mustika Sakera dapat memanggilnya setiap saat ketika Jalil membutuhkannya. Sakera yang orang asli Madura menggunakan senjata khas Celurit sebagai senjata andalannya. Dengan bela diri yang mahir dan kecepatan tebasan celuritnya membuat Sakera menjadi kesatria yang gesit dan mematikan di medan pertempuran.

c. Status dan *Skill* Sakera

Sakera merupakan seorang kesatria dengan role Rogue, yang berarti ia mahir menggunakan bela diri dan Celurit serta merupakan karakter yang cepat dan lincah. Ia mampu meningkatkan kecepatan serangan dirinya dan *troop* disekitarnya. Ilustrasi Sakera pada permainan dapat dilihat pada gambar 2.9.

Gambar 2.9 Karakter Sakera dalam permainan

Berikut adalah status awal Sakera ketika masih level 1

1) *Attack Point* : 45
2) *Health Point* : 225
3) *Defense Point* : 28

4) *Attack Rate* : 1
5) *Movement Speed* : 9

Sakera memiliki 2 skill aktif dengan ikon yang dapat dilihat pada gambar 2.10.

Gambar 2.10 Ikon skill Sakera

1) *Hurricane*
Skill ini dapat meningkatkan kecepatan serangan (*Attack Speed*) Sakera dan *troop* disekitarnya. Status: *Active, Buff.* Durasi*: 10s, Cooldown: 28s.*

2) *Bleed*
Kemampuan Sakera dalam menggunakan celurit sangat tinggi. *Skill* ini dapat menambahkan status *Bleeding* pada musuhnya. Bleeding adalah status dimana musuh kehilangan darah, menerima *damage* tambahan tiap detik selama waktu durasi bleeding. Status: *Active, Offensive. Bleed Duration: 7s, Cooldown: 25s*.

3. Arya Kasimpar

a. Sejarah Singkat

Arya Kasimpar adalah seorang pangeran di Pura Kahyangan Jagat Dalem Kubontingguh di Pucangan. Sebagai pangeran di Bali yang leluhurnya berasal dari Majapahit, Arya Kasimpar menganggap pemerintahan ayahnya telah jauh menyimpang dari apa yang seharusnya menjadi tradisi Majapahit. Keluarganya, sebaliknya, menganggap asimilasi kultural sebagai hal yang positif.

Pada umur 18, Arya Kasimpar meninggalkan kerajaannya karena ketidakpuasannya dan perbedaan prinsip dengan keluarganya. Ia

dikejar obsesi untuk menjadi seorang kesatria sempurna seperti pada era Majapahit, Arya Kasimpar berkelana dari satu guru ke guru lain, melahap ilmu dan pelatihan seperti api bertemu jerami. Blambangan, Alas Purwo, hingga akhirnya menghabiskan 6 tahun terakhir di daerah Semeru.

Setelah menyelesaikan latihannya, Arya Kasimpar diberi petunjuk untuk melanjutkan perjalanan ke Kediri, sebagai akhir perjalanan sebelum kembali ke Bali. Kali ini dia akan pulang, merasa paling benar, dan akan menentang siapapun yang tidak setuju dengan cara pandangnya. Sepuluh tahun perjalanannya ternyata tidak mampu mengubah arogansinya.

Dalam perkembangannya, Arya Kasimpar bertemu dengan pendekar dari berbagai daerah di Nusantara, setiap orang dengan latar belakang, permasalahan, visi dan solusi yang berbeda-beda dan berkembang.

b. Arya Kasimpar di *Saga Legacy*

Arya adalah pelindung dari karakter Wayan, seorang bocah asal Bali yang diberikan mustika Arya Kasimpar. Arya mahir menggunakan tombak dan perisai menjadikannya kesatria tangguh dengan pertahanan yang hebat. Perisai yang digunakannya adalah perisai sakti dimana ia dapat melindungi dirinya maupun kawannya dari serangan fisik.

Adapun ilustrasi Arya Kasimpar dalam permainan dapat dilihat pada gambar 2.11.

Gambar 2.11 Arya Kasimpar dalam permainan

c. Status dan *Skill* Arya Kasimpar

Berikut adalah status awal Arya Kasimpar ketika masih level 1

1) *Attack Point* : 42
2) *Health Point* : 300
3) *Defense Point* : 45
4) *Attack Rate* : 1.8 s
5) *Movement Speed* : 6

Arya Kasimpar memiliki 2 skill aktif dengan ikon yang dapat dilihat pada gambar 2.12.

Gambar 2.12 Ikon skill Arya Kasmpar

1) *Defender*
Skill ini dapat memberikan armor tambahan kepada Arya dan *troop* disekitarnya. Status: *Active*, *Buff*. Durasi: *12s*, *Cooldown*: *30s*

2) *Power Crush*
Arya memusatkan energinya pada tombak dan melepaskannya pada musuh membuat musuh mendapatkan status *Stun*. Status: *Active*, *Offensive*. *Stun Duration: 3s. Cooldown: 18s*

2) Karakter *Troop*

1) Prajurit

Prajurit merupakan pasukan dasar yang dimiliki oleh semua hero. Pasukan ini merupakan penyerang jarak dekat, memiliki *Health Point* dan *Def* yang tinggi. Namun memiliki daya serang lumayan.

Ilustrasi karakter prajurit dalam permainan dapat dilihat pada gambar 2.13.

Gambar 2.13 Karakter Prajurit dalam permainan

1) *Attack Point* : 33
2) *Health Point* : 120
3) *Defense Point* : 20
4) *Attack Rate* : 1.5 s
5) *Attack Range* : 3
6) *Movement Speed* : 3
7) *Required Soul* : 25

2) Pemanah

Pemanah merupakan pasukan dasar yang dimiliki oleh semua hero. Pasukan ini merupakan penyerang jarak jauh, memiliki *Health Point* dan *Def* yang kecil. Namun memiliki daya serang yang tinggi. Kecepatan serang pemanah juga tidak secepat kecepatan serang dari prajurit. Ilustrasi pemanan dalam permainan dapat dilihat pada gambar 2.14.

Gambar 2.14 Karakter Prajurit Panah dalam permainan

1) *Attack Point* : 40
2) *Health Point* : 90
3) *Defense Point* : 10
4) *Attack Rate* : 3 s
5) *Attack Range* : 18
6) *Movement Speed* : 2.5
7) *Required Soul* : 35

3) Dukun

Dukun merupakan pasukan dasar yang dimiliki oleh semua hero. Pasukan ini merupakan penyerang jarak jauh, memiliki *Health Point* dan *Def* yang kecil. Namun memiliki daya serang yang tinggi. Kecepatan serang dukun juga tidak secepat kecepatan serang dari prajurit. Tetapi memiliki jarak serang yang lebih jauh daripada pemanah. Ilustrasi dukun dalam permainan dapat dilihat pada gambar 2.15.

Gambar 2.15 Karakter Dukun dalam permainan

1) *Attack Point* : 60
2) *Health Point* : 80
3) *Defense Point* : 15
4) *Attack Rate* : 3.5 s
5) *Attack Range* : 24
6) *Movement Speed* : 2.5
7) *Required Soul* : 50

3) KARAKTER MUSUH

1. Buto

Buto adalah makhluk mistis dari legenda Indonesia. Makhluk ini digambarkan dengan badan yang besar, berwarna hijau, berparas menyeramkan. Desain musuh kami desain sedemikian mungkin menyeramkan dengan manambahkan gigi tajam dan tanduk yang tajam. Buto merupakan musuh utama untuk stage pertama hingga stage kelima. Buto pada jalan cerita, ingin menghancurkan dan menguasai kota Surabaya untuk menjajah para manusia. Buto merupakan anak buah dari Ken Arok. Terdapat jenderal buto yang merupakan buto berwarna merah sebagai pemimpin pasukan buto hijau. Buto merah memiliki daya serang yang cukup kuat dan health point yang besar walaupun pertahanannya lemah. Ilustrasi buto dalam permainan dapat dilihat pada gambar 2.16.

Gambar 2.16 Karakter Buto Ijo dan Buto Merah dalam permainan

Status Buto Ijo

1) *Attack Point* : 38
2) *Health Point* : 160
3) *Defense Point* : 20
4) *Attack Rate* : 2 s
5) *Attack Range* : 3
6) *Movement Speed* : 2

Status Buto Merah

1) *Attack Point* : 40
2) *Health Point* : 500
3) *Defense Point* : 20
4) *Attack Rate* : 2 s
5) *Attack Range* : 3
6) *Movement Speed* : 2

2. Buto Pemanah

Buto pemanah merupakan *troop* enemy yang dimiliki oleh Ken Arok. Ia merupakan ahli panah bangsa buto yang tidak kalah dengan prajurit pemanah manusia. Ilustrasi buto pemanah dapat dilihat pada gambar 2.17.

Gambar 2.17 Karakter Buto Panah dalam permainan

Status Buto Pemanah

1) *Attack Point* : 40
2) *Health Point* : 90
3) *Defense Point* : 10
4) *Attack Rate* : 2 s
5) *Attack Range* : 18
6) *Movement Speed* : 2.5

3.8.5 Desain *Stage*

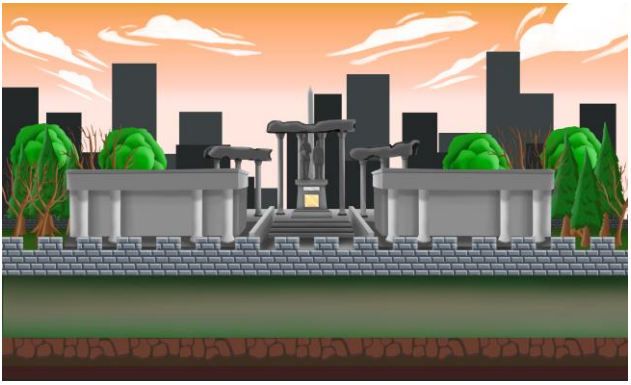
A. SURABAYA [*Stage* 1-5]

Surabaya merupakan *starting-stage* dimana invasi Ken Arok pertama-tama dilancarkan melalui kota ini. Kota yang dijuluki Kota Pahlawan ini akan memiliki beberapa stage yang mengambil setting

landmark yang terkenal di kota tersebut. *Landmark-landmark* tersebut adalah Tugu Pahlawan, Jembatan Merah dan Kampung Peneleh.

1. Tugu Pahlawan [1-3]

Tugu Pahlawan merupakan salah satu ikon Kota Surabaya, monumen ini berada di tengah - tengah kota dan di dekat kantor Gubernur Jawa Timur. Tugu Pahlawan adalah sebuah monumen yang menjadi markah tanah Kota Surabaya. Tugu Pahlawan dibangun untuk memperingati peristiwa Pertempuran 10 November 1945 di Surabaya, dimana *arek - arek Suroboyo* berjuang melawan pasukan sekutu bersama Belanda yang hendak menjajah kembali Indonesia. Ilustrasi latar belakang Tugu Pahlawan dapat dilihat pada gambar 2.18.



Gambar 2.18 Latar belakang Tugu Pahlawan

Dalam Saga Legacy, Tugu Pahlawan merupakan stage awal dimana pemain akan dikenalkan pada mekanisme dan gameplay dari game ini. Skenario stage-stage tersebut adalah sebagai berikut.

a. *Stage Prolog* [*Introduction*]

Dalam *Stage* ini, Aji Saka menghadapi Buto Ijo sendirian. Terdapat tutorial mengenai karakter kesatria utama yaitu skill dan status. Dalam stage ini, *troop* belum diperkenalkan.

[Tingkat Kesusahan : *EASY* (Menggunakan 40% kekuatan musuh), Tingkat *Spawn Rate* Musuh : 3~7 detik (*Random*)]

b. *Stage* 1

Stage selanjutnya, pasukan Buto Ijo semakin banyak. Aji Saka merasa tidak sanggup untuk melawan semuanya. Lalu Ken Dedes memanggil pasukan prajurit untuk membantu Aji Saka. Di sini mulai diperkenalkan sistem *Soul* dan *Troop*. *Troop* yang dapat di summon adalah troop Prajurit (Swordsman).

[Tingkat Kesulitan : *EASY* (Menggunakan 60% kekuatan musuh), Tingkat *Spawn Rate* Musuh : 3~7 detik (*Random*)]

c. *Stage* 2

Stage ini hanya memperkenalkan troop baru yaitu Pemanah (Archer).

[Tingkat Kesulitan: *EASY* (Menggunakan 70% kekuatan musuh), Tingkat *Spawn Rate* Musuh : 3~7 detik (*Random*)]

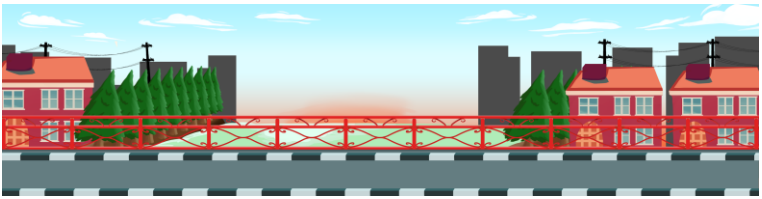
d. *Stage* 3

Stage ini akan muncul musuh baru, yaitu Buto Pemanah.

[Tingkat Kesulitan: *EASY* (Menggunakan 80% kekuatan musuh), Tingkat *Spawn Rate* Musuh : 3~7 detik (*Random*)]

2. Jembatan Merah [4-5]

Jembatan Merah merupakan salah satu monumen bersejarah di Surabaya, Jawa timur. Semasa zaman VOC dahulu Jembatan Merah menjadi sarana penghubung paling vital melewati Kalimas menuju Gedung Residesi Surabay. Ilustrasi dapat dilihat pada gambar 2.19.



Gambar 2.19 Latar belakang Jembatan Merah

Jembatan Merah juga merupakan saksi hidup dari tentara Indonesia, khususnya pahlawan - pahlawan Surabaya yang berjuang melawan kolonialisme Belanda.

a. ***Stage* 4**

Stage ini akan memperkenalkan *troop* baru yaitu Dukun (*Mage*).

[Tingkat Kesulitan: *EASY* (Menggunakan 90% kekuatan musuh), Tingkat *Spawn Rate* Musuh : 3~7 detik (*Random*)]

b. ***Stage* 5**

Pada *Stage* ini akan keluar Bos Buto Merah.

[Tingkat Kesulitan: *NORMAL* (Menggunakan 100% kekuatan musuh), Tingkat *Spawn Rate* Musuh : 3~7 detik (*Random*)]

2.3 *Database Management System*

Sebuah metode untuk mengoperasikan akses sistem database ke pluralitas database heterogen, masing-masing database tersebut memiliki struktur berdasarkan model database masing-masing dan menyimpan sejumlah entitas data yang memiliki atribut dan kejadian dalam struktur; Menyediakan *database* antarmuka, database antarmuka diisi dengan sejumlah ekspresi unik multi-karakter yang terkait dengan entitas data pluralitas *database*, dimana pluralitas ekspresi unik dan multi-karakter didefinisikan dengan menetapkan ke setiap entitas, setiap atribut dan Setiap entitas terjadi ekspresi unik, multi-karakter, ekspresi yang memiliki struktur hirarki yang telah ditentukan yang mendefinisikan hubungan antara masing-masing entitas, atribut dan kejadian entitas dengan setiap entitas, atribut dan kejadian entitas lainnya dalam database antarmuka dan menyimpan ungkapan tersebut dalam sebuah ekspresi. Atur tabel yang menghubungkan setiap elemen dari masing-masing ekspresi ke tingkat hirarkis dan posisi dalam model data[5].

2.4 *Difficulty Adjusment* Pada *Video Game*

Permainan komputer yang menyesuaikan kesulitan disesuaikan digunakan untuk terus menantang pemain sesuai kemampuan mereka. Penyesuaian kesulitan terjadi secara otomatis dalam menanggapi penilaian permainan yang sedang berlangsung terhadap kinerja pemain. Pendekatan penyesuaian kesulitan ini cenderung menjadi nilai dalam permainan komputer pendidikan sebagai alat pembelajaran perancah bagi siswa. Namun, ada penelitian terbatas yang mengevaluasi efektivitas permainan komputer pendidikan dengan penyesuaian kesulitan adaptif bila dibandingkan dengan penyesuaian kesulitan non adaptif. Hasil dari percobaan menyatakan bahwa *Difficulty Adjustment* membuat pengguna lebih termotivasi untuk menyelesaikan sebuah game dari pada yang tidak[6].

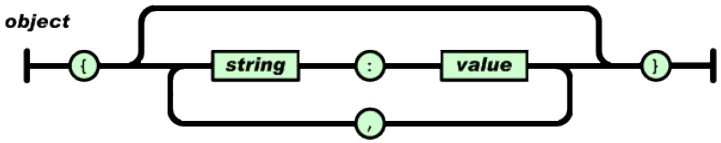
2.5 JSON (*JavaScript Object Notation*)

JSON atau *JavaScript Object Notation* merupakan *format* pertukaran data yang ringan dan lebih mudah bagi manusia untuk dibaca dan ditulis. Dan juga lebih ringan bagi mesin untuk mengurai (*parsing*) dan menghasilkan (*generate*). *Script* ini berdasarkan pada sebuah subset pada Pemrograman *Java Script*, standar ECMA-263 edisi ke-tiga – Desember 1999. JSON merupakan format teks yang bahasanya benar-benar independen namun menggunakan konvensi yang familiar bagi para *programmer* khususnya bahasa C; termasuk C, C ++, C#, *Java*, *JavaScript*, *Perl*, *Python*, dan lain-lain. Properti ini menjadikan JSON sebagai bahasa pertukaran data yang ideal.

JSON dibangun dalam dua struktur:

1. Sebuah koleksi dari sepasang nama/*value*. Dalam berbagai bahasa, ini dikenali sebagai sebuah obyek, rekaman, struktur, kamus, *hash table, keyed list* atau *associative array*.
2. Sebuah daftar *value* yang tersusun. Dalam berbagai bahasa, ini dikenali sebagai sebuah *array*, vektor, *list* atau *sequence*.

Ini merupakan struktur data yang bersifat universal. Hampir semua bahasa pemrograman modern mendukung data-data tersebut dalam satu bentuk atau bentuk yang lain. Dan masuk akal bahwa format data yang bisa ditukarkan dengan bahasa pemrograman ini juga berdasarkan pada struktur ini. Struktur array pada JSON dapat dilihat pada gambar 2.20.



Gambar 2.20 Struktur *Array* pada JSON [5]

Sebuah *array* merupakan koleksi dari *value* yang tersusun. Sebuah *array* dimulai dari ({) dan diakhiri dengan (}). Nilai ini dipisahkan dengan koma (,)[7].

2.6 *Unity AssetBundles*

AssetBundles merupakan file yang dibuat dalam *Unity editor* ketika waktu mengedit, dimana file ini dapat dipakai kembali sewaktu-waktu oleh *build* sebuah *project* ketika *run-time*. *AssetBundles* dapat berisikan aset-aset berupa model, material, tekstur dan *scene.* Namun *AssetBundles* tidak dapat diisi oleh *script*.

Secara spesifik, sebuah *AssetBundles* merupakan koleksi dari aset dan atau *scene* dari sebuah *project* yang disimpan di dalam sebuah file *compact* dengan tujuan dapat dimuat secara terpisah untuk aplikasi yang di-*build*. *AssetBundles* dapat dibuka sesuai permintaan oleh *game* atau aplikasi yang dibangun dalam *Unity*. Hal ini memungkinkan pemuatan dan pemuatan konten asinkron seperti model, tekstur, klip audio, atau bahkan keseluruhan *scene*. *AssetBundles* dapat disimpan sementara dalam *cache* dan disimpan secara lokal untuk pemuatan segera saat pertama kali menjalankan aplikasi. Tujuan utama *AssetBundles* adalah melakukan *streaming* konten berdasarkan permintaan dari lokasi yang jauh, untuk dimasukkan ke aplikasi jika diperlukan. *AssetBundles* dapat berisi jenis aset apa pun yang dikenali oleh Unity, termasuk data biner khusus. Satu-satunya pengecualian adalah bahwa *script* dari aset tidak diizinkan.

Ada banyak kasus penggunaan untuk *AssetBundles*. Konten baru dapat dimuat dan dibongkar secara dinamis dari sebuah aplikasi. DLC yang baru di-*release* dapat dengan mudah diimplementasikan. Tapak atau ukuran *disk* aplikasi dapat dikurangi saat pertama kali digunakan, dengan aset dimuat setelah pemasangan aplikasi dan hanya sesuai dengan aset yang dibutuhkan. Aset *platform* dan perangkat tertentu dapat dimuat tanpa harus mengunduh atau menyimpan aset berlebihan untuk platform atau resolusi lain. Pelokalan aplikasi menjadi mudah dengan mengunduh dan menginstal hanya aset yang dibutuhkan berdasarkan lokasi pengguna, bahasa atau preferensi. Aplikasi dapat diperbaiki, diubah atau diperbarui dengan konten baru tanpa harus mengirimkan ulang permohonan persetujuan.[8]

Gambar 2.21 menunjukkan bahwa sebuah model 3D mesh yang berbentuk kendaraan perang tank untuk permainan diaplikasikan dengan tekstur yang terdapat dalam sebuah *AssetBundles*. Maka model 3D ini bisa digunakan dan dipakai kembali sewaktu-waktu bahkan saat *run-time* sekalipun.

Gambar 2.21 Sebuah model *mesh*, yang diaplikasikan dengan tekstur. Terdapat dalam sebuah *AssetBundles*.[9]

2.7 *NoSQL Database*

NoSQL mencakup beragam teknologi *database* yang berbeda yang dikembangkan sebagai tanggapan atas tuntutan yang diajukan dalam membangun aplikasi modern:

1. Pengembang bekerja dengan aplikasi yang menghasilkan volume besar tipe data baru yang berubah dengan cepat – data

yang terstruktur, semi-terstruktur, tidak terstruktur dan polimorfik.

2. Siklus pengembangan air terjun dua belas hingga delapan belas bulan telah lama hilang. Sekarang tim kecil bekerja dalam langkah yang tangkas, pengulangan dengan cepat dan mendorong kode setiap dua minggu sekali, beberapa bahkan beberapa kali setiap hari.
3. Aplikasi yang dulunya menyajikan khalayak yang terbatas sekarang disampaikan sebagai layanan yang harus selalu ada, dapat diakses dari berbagai perangkat dan ditingkatkan secara global hingga jutaan pengguna.
4. Organisasi sekarang beralih ke arsitektur berskala besar menggunakan perangkat lunak *open-source*, server komoditas dan komputasi awan, alih-alih server monolitik dan infrastruktur penyimpanan yang besar.
5. *Database* relasional tidak dirancang untuk mengatasi tantangan skala dan kelincahan yang menghadapi aplikasi modern, dan juga tidak dibangun untuk memanfaatkan penyimpanan komoditas dan daya pemrosesan yang ada saat ini.

2.7.1 Tipe *Database NoSQL*

1. *Database* dokumen memasangkan setiap kunci dengan struktur data yang kompleks yang dikenal sebagai dokumen. Dokumen dapat berisi banyak pasangan kunci-nilai yang berbeda, atau pasangan kunci-*array*, atau bahkan dokumen bersarang.
2. *Graph Stores* digunakan untuk menyimpan informasi tentang jaringan data, seperti koneksi sosial. *Graph stores* meliputi Neo4J dan Giraph.
3. *Key-value stores* adalah *database* NoSQL yang paling sederhana. Setiap item dalam *database* disimpan sebagai nama atribut (atau 'kunci'), beserta nilainya. Contoh toko dengan nilai kunci adalah Riak dan Berkeley DB. Beberapa toko dengan nilai penting, seperti Redis, memungkinkan setiap nilai memiliki tipe, seperti 'integer', yang menambahkan fungsionalitas.
4. *Wide-column stores* seperti Cassandra dan HBase dioptimalkan untuk kueri di atas kumpulan data yang besar, dan menyimpan kolom data bersama-sama, bukan baris.

2.7.2 Keuntungan *NoSQL*

Bila dibandingkan dengan *database* relasional, *database* NoSQL lebih terukur dan memberikan kinerja superior, dan model data mereka membahas beberapa masalah yang model relasionalnya tidak dirancang untuk ditangani:

1. Volume besar dengan cepat mengubah data terstruktur, semi-terstruktur, dan tidak terstruktur
2. *Agile sprint*, iterasi skema cepat, dan kode yang sering mendorong
3. Pemrograman berorientasi obyek yang mudah digunakan dan fleksibel
4. Arsitektur berskala terdistribusi secara geografis dan bukan arsitektur monolitik yang mahal[10]

2.8 *Cloud Storage*

Cloud storage merupakan sebuah model penyimpanan berbasis internet. Data yang disimpan di *cloud*, akan disimpan di *server* penyedia layanan, dan bisa diakses dengan menggunakan koneksi internet. Pada dasarnya teknologi *Cloud Storage* merupakan pengembangan dari sistem Komputasi Awan atau yang disebut juga dengan istilah *cloud computing*. Komputasi Awan merupakan konsep dasar dari adanya layanan *Cloud Storage*. Dengan penerapan teknologi Komputasi Awan, penyedia layanan *Cloud Storage* bisa membangun media penyimpanan secara online tersebut. Mengenai komputasi awan, teknologi ini merupakan salah satu teknologi jaringan internet yang memiliki sejarah pengembangan yang cukup panjang.

Secara simpel, sistem komputasi awan menggunakan serangkaian komputer server yang telah dioptimasi dengan sistem penyimpanan yang nantinya membentuk banyak *virtual server* atau tempat penyimpanan data dalam jaringan internet. Data yang tersimpan pada *virtual server* tersebut akan tetap ada dalam *server* pusat dan jika pengguna memerlukan data tersebut, maka tinggal mengaksesnya dan akan tersimpan secara sementara pada perangkat kita.

Teknologi ini sebenarnya sudah mulai diperkenalkan sekitar tahun 1960an oleh seorang insinyur teknik komputer dari MIT bernama John McCarthy. Pada waktu itu sistem tersebut belum diterapkan pada jaringan internet namun hanya dalam sistem jaringan infrastruktur seperti

listrik dan air. Namun pada waktu itu John McCarthy sudah mulai mengung-kapkan konsep penggabungan sistem dalam media khusus yang akhirnya kini dikembangkan menjadi Komputasi Awan.

Perkembangan sistem yang mendasari *Cloud Storage* tersebut mulai diperkenalkan pada modern ini oleh perusahaan *e-Commerce Amazon* pada tahun 2000. *Amazon* menjadi salah satu pelopor penggunaan sistem tersebut sebagai penjembatan dari semua layanan e-commerce miliknya yang masuk pada layanan Amazon Web Service.

Baru beberapa waku berikutnya, perkembangan dari sistem Komputasi Awan semakin berkembang dengan pesat, seperti yang dilakukan oleh Google melalui salah satu layanannya Google Drive. Ilustrasi dari cloud storage dapat dilihat pada gambar 2.22.[11]

Gambar 2.22 Ilustrasi dari *cloud storage*[12]

2.9 *Seamless Update*

Seamless Update merupakan sebuah metode pembaruan data dimana pembaruan data tersebut terjadi pada *background*, yang artinya pengguna aplikasi dapat tetap menggunakan aplikasi meskipun proses pembaruan data terjadi. Nama metode ini terkenal seiring keluarnya *Android Nougat* dimana sudah mengimplementasikan metode *Seamless Update* dalam pembaruan *firmware* untuk *smartphone* android. Dalam skenario *seamless update* untuk *Android Nougat*, metode ini bekerja dengan menggunakan dua sistem partisi yang berbeda pada *device* android[12]. Saat pembaruan terjadi, data tersebut langsung dipindahkan ke partisi kedua. Saat pembaruan selesai, maka pengg una dapat langsung menikmati pembaruan tersebut dengan *reboot smartphone*.

Pada permainan, *seamless update* dapat digunakan sebagai metode pembaruan data permainan, sehingga pemain tidak perlu untuk keluar permainan untuk melakukan pembaruan data.

2.10 Permainan *Action-Tower Defense*

Permainan yang akan diimplementasikan dalam tugas akhir ini ialah permainan dengan jenis *Action-Tower Defense*. Jenis permainan ini hampir sama dengan *Tower Defense* biasa pada umumnya dimana pemain ditugaskan menjaga markas dari serangan musuh yang ada. Selain menjaga markas, pemain ditugaskan untuk menghancurkan markas musuh untuk meraih kemenangan. Dalam strategi menjaga markas dan menyerang markas musuh terdapat sistem poin dimana pemain dapat menggunakannya untuk mengeluarkan pasukan. Setiap pasukan memiliki keunikan kekuatan yang dapat mengantarkan pemain menuju kemenangan. Permainan *Tower Defense* yang terkenal pada saat ini ialah *Clash Royale* dan *Line Ranger*.

Tetapi, pada permainan yang akan diimplementasikan, sedikit berbeda dengan permainan *Tower Defense* yang ada. Perbedaan tersebut terletak pada *Action* yang dilakukan pemain. Selain pemain mengatur strategi dalam menahan *wave* musuh, pemain juga diharuskan untuk mengontrol *hero* atau karakter utama dalam membantu pasukan untuk memenangkan permainan. Gabungan antara *Tower Defense* dan *Action* ini diimplementasikan dalam beberapa game terkenal, contohnya *Larvae Heroes* dan *Dungeon Defender* pada gambar 2.23 dibawah ini.

(a) (b)

Gambar 2.23 Permainan (a) *Larvae Heroes* [13] dan (b) *Dungeon Defenders 2* [14] merupakan termasuk jenis game *Tower Defense*

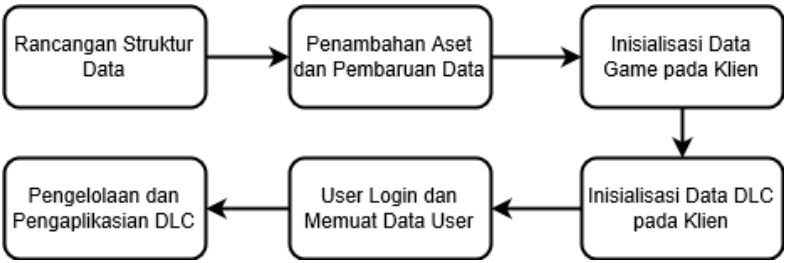
Halaman sengaja dikosongkan

BAB 3
DESAIN DAN IMPLEMENTASI SISTEM

Penelitian ini dilaksanakan sesuai dengan desain sistem berikut dengan implementasinya. Desain sistem merupakan perancangan dari sistem yang tersusun secara urut dan sistematis sehingga mempermudah dalam proses pengerjaan.

3.1 Desain Sistem

Tugas akhir ini bertujuan untuk membuat sebuah sistem *seamless update* dalam *game action tower-defense Saga Legacy*. Sistem ini terbagi menjadi 2, yaitu game data *update* yang memperbaharui data-data karakter maupun *skill*, dan *game assets update* yang meng*handle* pembaharuan assets seperti musik, sprite karakter, dan lain-lain. Semua data karakter dan *skill* dalam *game* mengacu pada *firebase real-time database* dan setiap perubahan data pada *database* akan diteruskan ke setiap *client* yang aktif. Pada *game Saga Legacy*, terdapat *listener* yang mendeteksi apabila ada perubahan data pada database, ketika terjadi perubahan data, sistem akan langsung mengambil data tersebut, lalu memperbaharui data yang ada dalam game dan mengaplikasikan nya kedalam *gameplay*. Gambaran umum dari sistem *database real-time* pada tugas akhir ini diilustrasikan dalam gambar 3.1.



Gambar 3.1 Gambaran umum desain sistem

3.2 Rancangan Struktur Data

Semua data pada sistem database waktu nyata Firebase disimpan sebagai objek *JSON*. Ketika data ditambahkan ke *JSON tree*, data tersebut menjadi sebuah node pada struktur JSON yang ada dengan kunci terkait. *Database* waktu nyata *Firebase* membolehkan *nesting* data hingga 32 tingkat kedalaman, namun ketika pengguna mengambil data pada sebuah lokasi di database, maka semua *child* pada data tersebut juga ikut terambil. Karena itu perlu, dirancang struktur data dengan jumlah nesting seminimal mungkin.

Data-data yang disimpan dalam *database* adalah sebagai berikut :

1) Data DLC (*Downloadable Content*)
2) Data *Hero*
3) Data *Troop*
4) Data *Skill*
5) Data *Stage*
6) Data Pengguna

3.2.1 Data DLC (Downloadable Content)

Tugas akhir ini mengimplementasikan sistem pengelolaan data dalam permainan *Saga Legacy* berbasis DLC. Intinya, hampir semua konten yang tersedia dalam game dapat dikelola mana yang ingin diunduh atau dihapus, menjadikan data dalam permainan dinamik dan mempermudah pengembang maupun pengguna dalam melakukan pembaruan data. Data DLC ini disimpan pada *database* dan diakses oleh klien setiap kali permainan baru dijalankan. Struktur informasi yang disimpan dalam Data DLC sesuai dengan tabel 3.1.

Tabel 3.1 Struktur **DLC**

Nama	Tipe
detail	String
icon	String
status	DLCStatus
type	DLCType
version	String

a. Nama

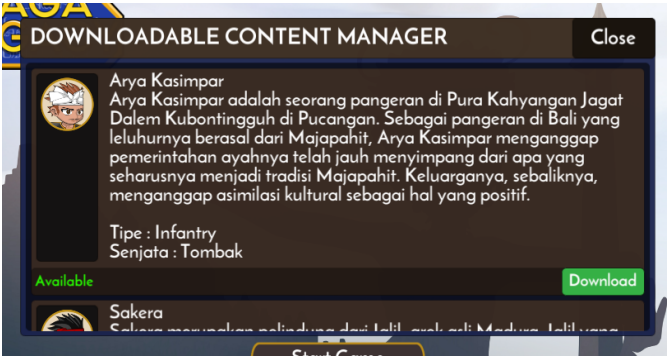
Nama dari *Downloadable Content* sekaligus nama dari *AssetBundle* dari DLC terkait dan *key* pada *NoSQL* database *Firebase*.

b. Detail

Deskripsi mengenai DLC yang menjelaskan tentang konten yang ada dalam DLC terkait.

c. Icon

Nama ikon yang akan digunakan saat pemain memuat DLC *Manager*. Ikon diunduh dari Firebase Storage menggunakan referensi nama dari nilai *Icon*. Desain DLC Manager dapat dilihat pada gambar 3.2 yang akan dijelaskan pada Sub-Bab berikutnya.



Gambar 3.2 DLC *Manager* menampilkan DLC yang dapat diunduh dan dihapus oleh pemain

d. Status

Status merupakan sebuah enum bernama **DLC*Status*** yang memiliki 3 nilai, yaitu *OnHold*, *Necessary*, dan *Downloadable*.

Yang menentukan perlakuan ketika data DLC pertama kali diinisialisasi. Parameter dalam enum **DLC*Status*** dapat dilihat pada gambar 3.3.

```
enum DLCStatus {
   OnHold, Necessary, Downloadable
}
```

Gambar 3.3 Enum **DLC*Status***

OnHold berarti DLC belum dapat diunduh dan sistem tidak akan memasukkannya kedalam daftar DLC pada klien. *Necessary* berarti DLC merupakan konten yang diperlukan agar permainan dapat berjalan dengan normal, oleh karena itu konten dengan status *Necessary* akan langsung diunduh kedalam perangkat. Sedangkan *Downloadable* merupakan DLC yang dapat diunduh sewaktu-waktu namun tidak mempengaruhi kinerja dari permainan.

e. Type

Type memiliki banyak nilai, yang menentukan bagaimana sistem bertidak setelah DLC diunduh. Nilai-nilai tersebut dapat dilihat pada gambar 3.4.

```
enum DLCType {
   Player, NPC, Troop, Stage, Language,
   BGM, SFX, Skin, Weapon, Other
}
```

Gambar 3.4 Enmu **DLC*Type***

Berdasarkan nilai dari "*Type*" tersebut, konten yang telah diunduh mendapat tindakan yang berbeda, sehingga konten dapat langsung diaplikasikan kedalam gameplay tanpa harus *restart* permainan.

f. Version

Setiap versi dari *assetBundle* ditempatkan pada folder yang berbeda dengan nama nilai versi di *Firebase Storage*, nilai *version* digunakan untuk mendapatkan referensi folder yang benar sesuai dengan kemauam pengembang permainan.

3.2.2 Data Hero

Data *Hero* menyimpan nilai status yang mempengaruhi kekuatan dari hero terkait, serta parameter-parameter yang berkaitan dengan mekanik dari permainan. Struktur ini merupakan salah satu struktur yang dapat diubah secara waktu nyata dan langsung diaplikasikan kedalam gameplay. Salah satu variabel diantaranya adalah *Attack*, yang mempengaruhi kerusakan yang terjadi ketika hero menyerang musuh. Ketika nilai *Attack* pada database waktu nyata *Firebase* dinaikkan, maka nilai tersebut akan diteruskan ke setiap perangkat klien, lalu klien akan langsung meng-aplikasikannya kedalam gameplay. Membuat tingkat kerusakan yang ditimbulkan semakin besar. Berikut adalah informasi yang disimipan pada struktur hero. Parameter-parameter yang terdapat dalam struktur ***HeroData*** dapat dilihat pada tabel 3.2.

Tabel 3.2 Struktur ***HeroData***

Nama	Tipe
heroName	String
heroStory	String
baseStat	BaseStat
skillIndex	Int32[]
prefabBundle	String
prefabName	String
iconBundle	String
iconName	String
available	Int32

Pada struktur ***HeroData,*** terdapat struktur bernama ***BaseStat*** yang menyimpan parameter-parameter dasar seperti *health*, *attack*, dan lain-lain. Parameter ini mempengaruhi tingkat kekuatan karakter dalam

permainan. Sedangkan fungsi dari masing-masing parameter dalam struktur ***HeroData*** dapat dilihat pada tabel 3.3.

Tabel 3.3 Fungsi dari masing-masing parameter pada struktur ***HeroData***

Parameter	**Fungsi**
heroName	Nama dari *hero* terkait
heroStory	Mitos atau legenda latar belakang dari *hero* terkait yang menggambarkan peran karakter dalam *game*.
baseStat	Merupakan struktur yang menyimpan data stat dasar seperti "*Attack*", "*Defense*", dll.
skillIndex	2 indeks skill yang mengacu pada database skill. Setiap hero memiliki 2 skill.
prefabBundle	Nama dari *assetbundle* dimana *prefab hero* berada.
prefabName	Nama dari *prefab* yang terdapat dalam *assetbundle*
iconBundle	Nama dari *assetbundle* dimana *icon hero* berada.
iconName	Nama dari *icon* yang terdapat dalam *assetbundle*
available	Ketersediaan dari *hero*. 0 berarti tidak tersedia, sedangkan lebih dari 0 menandakan *hero* tersedia dan dapat digunakan.

Struktur ***BaseStat*** memiliki parameter yang mempengaruhi gameplay dalam permainan seperti kecepatan karakter dan tingkat ketahanan karakter terhadap serangan. Berikut parameter-parameter yang terdapat dalam ***BaseStat*** ditampilkan pada tabel 3.4.

Tabel 3.4 Struktur ***BaseStat***

Nama	**Tipe**
health	Float
energy	Float
movementSpeed	Float
attack	Float
defense	Float
attackStamp	Float
attackSpeed	Float
speedMultiplier	Float
range	Float

Adapun fungsi dari parameter-parameter yang terdapat pada struktur ***BaseStat*** dapat dilihat pada tabel 3.5.

Tabel 3.5 Fungsi dari masing-masing parameter pada struktur ***BaseStat***

Parameter	**Fungsi**
health	Nilai maksimal *health* dari karakter, apabila nilai *health* turun hingga ke 0, maka karakter akan mati.
energy	Nilai maksimal energy yang digunakan untuk menggunakan skill
movementSpeed	Kecepatan gerak dari karakter.
attack	Nilai serangan yang dilakukan karakter. Semakin besar nilai *attack* semakin besar tingkat kerusakan yang ditimbulkan.
defense	Nilai yang menunjukkan ketahanan karakter terhadap serangan.
attackStamp	Detik dimana nilai serangan dihitung.
attackSpeed	Tingkat kecepatan serangan.
speedMultiplier	Nilai kali kecepatan saat *boon* aktif. Berpengaruh pada kecepatan serangan dan kecepatan gerak serta animasi gerakan.
range	Jarak serang dari karakter.

3.2.3 Data Troop

Data *Troop* menyimpan nilai status yang mempengaruhi kekuatan dari troop terkait, serta parameter-parameter yang berkaitan dengan mekanik dari permainan. Parameter-parameter yang terdapat dalam struktur ***TroopData*** dapat dilihat pada tabel 3.6.

Tabel 3.6 Struktur ***TroopData***

Nama	**Tipe**
troopName	String
troopStory	String
baseStat	BaseStat
prefabBundle	String
prefabName	String

Tabel 3.6 Struktur ***TroopData***

Nama	Tipe
iconBundle	String
iconName	String
isRanged	Boolean
coolDown	Float
cost	Int32
available	Int32

Tidak seperti hero, troop tidak memiliki skill dan tidak dapat dikontrol, melainkan berjalan secara otomatis, dan menyerang ketika ada musuh dalam jangkauan serang. Fungsi dari masing-masing parameter dapat dilihat pada tabel 3.7.

Tabel 3.7 Fungsi masing-masing parameter pada struktur ***TroopData***

Parameter	Fungsi
troopName	Nama dari troop terkait
troopStory	Mitos atau legenda latar belakang dari *troop* terkait yang menggambarkan peran *troop* dalam *game*.
baseStat	Merupakan struktur yang menyimpan data stat dasar seperti "*Attack*", "*Defense*", dll.
prefabBundle	Nama dari *assetbundle* dimana prefab *troop* berada.
prefabName	Nama dari *prefab* yang terdapat dalam *assetbundle*
iconBundle	Nama dari *assetbundle* dimana *icon troop* berada.
iconName	Nama dari *icon* yang terdapat dalam *assetbundle*
isRanged	Apakah *troop* bisa menyerang dari kejauhan atau tidak
coolDown	Waktu ketika *troop* dipanggil hingga *troop* bisa di panggil lagi.
cost	*"Soul"* yang dibutuhkan untuk memanggil *troop* terkait.
available	Ketersediaan dari *troop*. 0 berarti tidak tersedia, sedangkan lebih dari 0 menandakan *troop* tersedia dan dapat digunakan.

Sama seperti *hero*, *troop* memiliki baseStat yang berpengaruh pada tingkat kekuatan *troop*. *Troop* juga memiliki nilai *cooldown* yang digunakan untuk membatasi pemain agar tidak dapat memanggil troop yang sama secara beruntun, hal ini dapat berpengaruh pada strategi yang harus digunakan oleh pemain untuk memenangkan *stage*.

3.2.4 Data Skill

Data *skill* merupakan *skill* yang digunakan oleh *hero* dalam permainan. Masing-masing *hero* memiliki 2 skill yang berbeda baik dari segi efek maupun nilai dari efek yang ditimbulkan. Animasi karakter disesuaikan dengan efek dari skill yang ditimbulkan. Parameter-parameter yang terdapat dalam struktur ***DataSkill*** dapat dilihat pada tabel 3.8.

Tabel 3.8 Struktur ***SkillData***

Nama	Tipe
skillName	String
skillDescription	String
skillTrigger	SkillTrigger
skillType	SkillType
percentage	Float
duration	Int32
targetTag	String
iconBundle	String
iconName	String
coolDown	Float
castTime	Float
available	Int32

Skill memiliki struktur sendiri untuk mengurangi tingkat nesting pada database Firebase sekaligus memudahkan dalam pengolahan data.

Fungsi dari masing-masing parameter dalam struktur ***SkillData*** dapat dilihat pada tabel 3.9.

Tabel 3.9 Fungsi masing-masing parameter pada ***SkillData***

Parameter	**Fungsi**
skillName	Nama dari *skill* terkait
skillDescription	Penjelasan mengenai *skill* seperti efek, nilai serangan, dan lain-lain.
skillTrigger	Enum yang menyimpan nilai efek dari *skill* terkait. Tergantung nilainya, setiap *skill* memiliki efek yang berbeda-beda.
skillType	Enum yang menyimpan tipe area dari *skill* dimana *skill* akan efektif.
percentage	Persentase yang diambil dari *stat* karakter yang mempengaruhi tingkat kekuatan *skill*.
duration	Durasi efek dari *skill*.
targetTag	Tag target yang dituju *skill*.
iconBundle	Nama dari *assetbundle* dimana *icon skill* berada.
iconName	Nama dari *icon* yang terdapat dalam *assetbundle*
coolDown	Lama menunggu hingga *skill* dapat digunakan.
castTime	Lama karakter untuk mengaktifkan *skill*.
available	Ketersediaan dari *skill*. 0 berarti tidak tersedia, sedangkan lebih dari 0 menandakan *skill* tersedia dan dapat digunakan.

Pada ***SkillData*** terdapat 2 enum yang menentukan sifat dari skill. Enum yang pertama adalah ***SkillTrigger*** yang merupakan efek dari skill terkait seperti menyembuhkan health karakter, atau mengurangi defense musuh. Enum yang kedua adalah ***SkillType*** yang menentukan bagaimana skill terkait dapat mempengaruhi targetnya.

SkillTrigger menyimpan nilai efek yang ditimbulkan saat *skill* digunakan Hal ini sesuai dengan gambar 3.5 berikut.

```
enum SkillTrigger {
    SelfHeal, IncrementalHeal,
    AttackBuff, DefenseBuff,
    AttackSpeedBuff, Bleeding, MeleeStun
}
```

Gambar 3.5 Enum ***SkillTrigger***

SkillType memiliki 2 nilai, yaitu *WorldSkill* dan *CollisionSkill*. *WorldSkill* menandakan bahwa skill efektif tidak peduli jarak karakter terhadap sumber skill. Semua target skill akan terkena efek dari skill terkait. Sedangkan *CollisionSkill* adalah skill yang efektif ketika target menabrak skill yang digunakan oleh karakter. Enum ***SkillType*** dijelaskan pada gambar 3.6.

```
enum SkillType {
    WorldSkill, CollisionSkill
}
```

Gambar 3.6 Enum ***SkillType***

3.2.5 Data Stage

Data Stage menyimpan data troop dan tingkat kesulitan pada stage terkait. Data stage dapat diubah sewaktu-waktu pada database waktu nyata Firebase untuk menyesuaikan tingkat kesulitan dari suatu stage. Untuk mencegah suatu stage menjadi terlalu sulit atau terlalu mudah. Parameter-parameter yang terdapat dalam struktur ***StageData*** dapat dilihat pada tabel 3.10.

Tabel 3.10 Struktur ***StageData***

Nama	Tipe
stageName	String
modifier	Float

Tabel 3.10 Struktur ***StageData***

Nama	Tipe
leftTroops	Int32[]
rightTroops	Int32[]

Salah satu parameter dalam ***StageData*** adalah modifier yang digunakan untuk menambah atau mengurangi tingkat kesulitan stage dengan memodifikasi nilai stat dari troop musuh. Sehingga tergantung dari nilai modifier, pemain akan menghadapi musuh yang semakin kuat atau lemah. Sedangkan fungsi dari masing-masing parameter dalam struktur ini dapat dilihat pada tabel 3.11.

Tabel 3.11 Fungsi masing-masing parameter pada ***StageData***

Parameter	Fungsi
stageName	Nama dari stage terkait
modifier	*Modifier* merupakan pengali dari *stat troop* musuh. Semakin besar nilai *modifier*, maka musuh akan semakin kuat dan membuat stage semakin sulit.
leftTroops	*Troop* yang dipakai oleh *user*.
rightTroops	*Troop* yang digunakan musuh. Nilai modifier hanya berpengaruh pada troop musuh.

3.2.6 Data Pengguna (*User*)

Data pengguna digunakan untuk menyimpan progress pemain selama memainkan Saga Legacy. Data pengguna dikaitkan dengan akun masing-masing pemain yang telah mendaftar menggunakan surel mereka. Parameter-parameter yang terdapat pada data pengguna dapat dilihat pada tabel 3.12.

Tabel 3.12 Struktur data pengguna

Nama	Tipe
userName	String
email	String

Tabel 3.12 Struktur data pengguna

Nama	Tipe
dlc	List<String>
stage	Int32
token	Int32

Data pengguna menyimpan daftar nama dari masing-masing DLC yang telah diunduh selama memainkan Saga Legacy. Sehingga saat pemain melakukan login pada perangkat baru, DLC yang telah dimiliki akan langsung diunduh kembali. Sedangkan fungsi dari masing-masing parameter pada data pengguna dapat dilihat pada tabel 3.13.

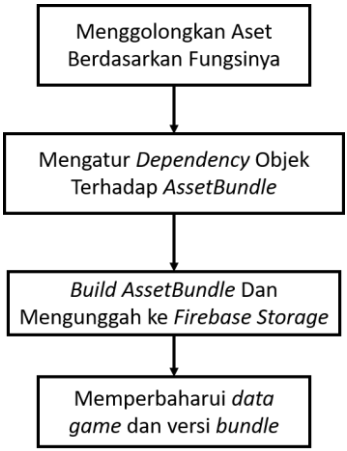
Tabel 3.13 Fungsi masing-masing parameter pada struktur data pengguna

Parameter	Fungsi
userName	Nama dari pengguna
email	Surel yang digunakan pengguna untuk mendaftar
dlc	Daftar nama DLC yang dimiliki user
stage	*Progress stage* dalam ekspedisi kisah utama.
token	Mata uang yang digunakan dalam permainan

3.3 Penambahan Aset dan Pembaruan Data

Tahap selanjutnya dalam sistem update real-time pada tugas akhir ini adalah menambahkan asset dan memperbaharui data pada cloud database. Untuk menambahkan aset, Unity3D memiliki file bernama *AssetBundle* yang menyimpan aset-aset game seperti musik, gambar, dan lain-lain. *AssetBundle* dibuild pada UnityEditor dan aset dalam *AssetBundle* dapat digunakan pada versi rilis dari game sebagai alat untuk memperbaharui aset dari game tanpa harus merilis versi baru dari game itu sendiri selama tidak diperlukan pembaharuan sistem game. Sedangkan data masing-masing karakter pada klien mengacu pada sistem *database real-time* sehingga data game tetap terupdate, selama klien memiliki koneksi internet.

Berikut adalah tahapan dalam pembuatan *AssetBundle* di ilustrasikan pada gambar 3.7.



Gambar 3.7 Alur integrasi *Unity3D* dengan *Firebase*

3.3.1 Menggolongkan Aset Berdasarkan Fungsinya

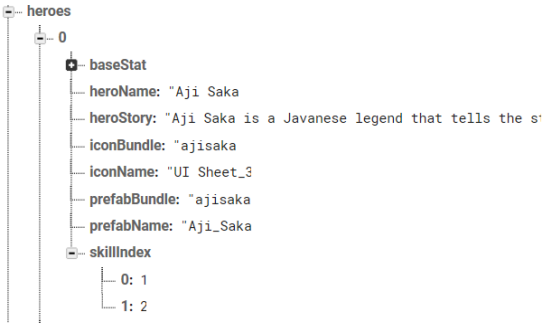
Dalam pengembangan *game*, aset-aset yang digunakan digolongkan berdasarkan fungsi dari aset tersebut didalam game. *AssetBundle* berfungsi sebagai arsip dari masing-masing kategori dari aset dalam game tersebut. Pada *game Saga Legacy*, terdapat banyak karakter mulai dari pahlawan hingga musuh, dan setiap karakter memiliki aset yang berbeda. Aset-aset tersebut nantinya digolongkan pada *AssetBundle* yang berbeda. Semua aset seperti animasi, sprite, maupun senjata digolongkan berdasarkan karakter masing-masing. Aset-aset ini nantinya akan dimuat dari *AssetBundle* saat aset tersebut dibutuhkan. Misal ikon dari salah satu karakter bernama Arya Kasimpar, akan dimasukkan kedalam *AssetBundle* aryakasimpar, sedangkan ikon karakter lain bernama sakera, akan dimasukkan kedalam *AssetBundle* sakera. Penggolongan ini nantinya akan memudahkan pengembang dalam mengatur *dependency* objek seperti antarmuka saat memilih hero yang membutuhkan ikon-ikon tersebut, sehingga dapat diakses dengan memuat *AssetBundle* sesuai dengan nama dari karakter yang dibutuhkan.

Contoh dari penggolongan ini dapat dilihat pada gambar 3.8 dibawah ini.

Gambar 3.8 Sprite pada *AssetBundle* aryakasimpar dan sakera

3.3.2 Mengatur *Dependency* Objek Terhadap *AssetBundle*

Karena aset yang digunakan dalam game merujuk pada *AssetBundle*, maka perlu adanya data yang mencatat nama *AssetBundle* dan nama file yang dibutuhkan oleh masing-masing *GameObject* agar aset pada *AssetBundle* dapat dimuat. Berdasarkan rancangan struktur data pada subbab 3.2, informasi mengenai nama *AssetBundle* dan nama aset dalam aset bundle disimpan pada 3 struktur yaitu ***HeroData, TroopData,*** dan ***SkillData***. Gambar 3.9 menunjukkan nilai data *Hero* pada *firebase database.*



Gambar 3.9 Nilai data *hero* pada *firebase database*

Sedangkan gambar 3.10 menunjukkan nilai data *skill* dan *troop* pada *firebase database*.

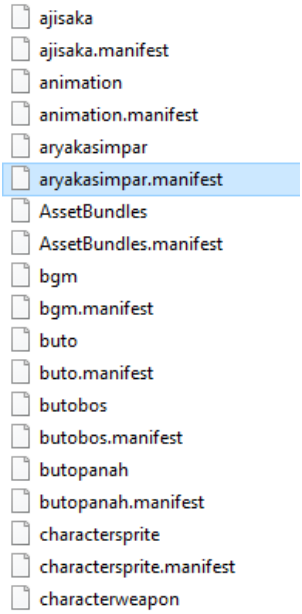
Gambar 3.10 Nilai data *skill* dan *troop* pada *firebase database*

Terdapat variabel dengan nama "*iconBundle*" dan "*iconName*", dua variabel tersebut merupakan informasi mengenai dimana ikon dari objek tersebut berada. "*iconBundle*" menyimpan nama dari *AssetBundle* sedangkan "*iconName*" menyimpan nama aset didalam *AssetBundle* dengan nama yang sudah disebutkan. Data tersebut disimpan dalam *real-time database* dengan *format json* yang biasa digunakan saat transmit data.

3.3.3 *Build AssetBundle* dan Mengunggah ke *Firebase Storage*

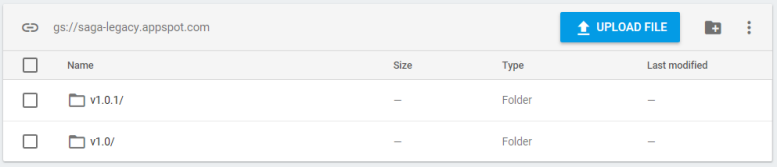
Setelah aset-aset digolongkan berdasarkan fungsinya, *AssetBundle* di-*build* menggunakan *AssetBundle BuildPipeline* yang mengarsipkan aset-aset tersebut menjadi *file* ter-serialisasi. *AssetBundle* memiliki 2 struktur sedikit berbeda tergantung apakah *AssetBundle* tersebut merupakan *normal bundle* atau *scene bundle*. *Normal bundle* merupakan *AssetBundle* yang berisi aset-aset seperti musik, gambar, dan lain-lain. Sedangkan *scene bundle* berisi *scene-scene* yang dapat dimuat

secara asinkron saat game berjalan. Menyimpan *scene* kedalam *AssetBundle* membuat aset-aset yang digunakan pada *scene* tersebut tidak ikut kedalam *versi build* sehingga ukuran *file* instalasi menjadi lebih kecil. *AssetBundle* memiliki *AssetBundleManifest* yang menyimpan data aset dan *dependency*-nya terhadap aset-aset yang lain. *AssetBundle* yang telah diunggah ke *Firebase Storage* dapat dilihat pada gambar 3.11.



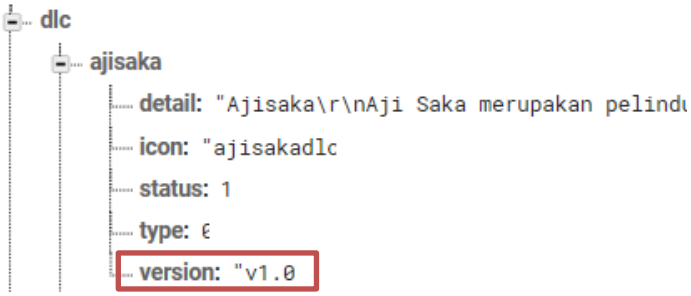
Gambar 3.11 Hasil dari *build AssetBundle* pada *Firebase Storage*

AssetBundle yang sudah di-*build* kemudian diunggah ke *Firebase Storage* berdasarkan versi yang sudah ditentukan pengembang. Hal tersebut ditunjukkan oleh gambar 3.12 berikut.



Gambar 3.12 *Folder* versi *AssetBundle* pada *Firebase Storage*

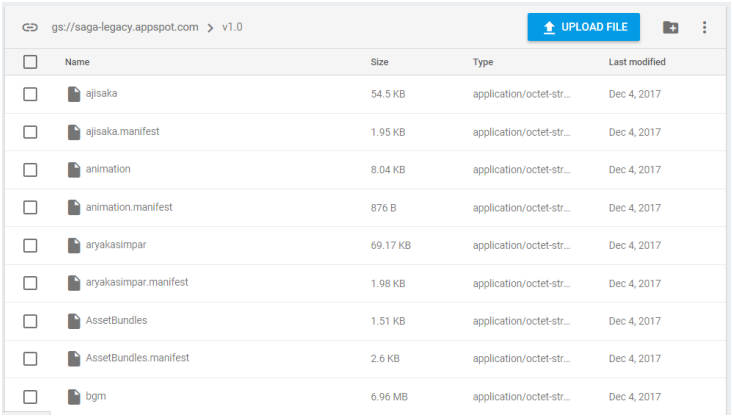
Setiap *folder* pada gambar 3.12 merepresentasikan versi dari *AssetBundle* yang dapat diakses oleh klien untuk mengunduh aset-aset yang akan digunakan didalam permainan. Versi mana yang harus diunduh oleh klien bergantung pada nilai "*version*" pada struktur DLC dalam *database* waktu nyata *Firebase*, dapat dilihat pada gambar 3.13.



Gambar 3.13 *Versi assetbundle* yang sedang aktif digunakan sebagai referensi pada *database* dan klien

Setiap file yang ada didalam *Firebase Storage* memiliki *storageReference* yang dapat digunakan untuk mengambil link dari file yang bersangkutan.

AssetBundle pada *Firebase Storage* dapat dilihat pada gambar 3.14.



Gambar 3.14 *AssetBundle* pada *Firebase Storage*

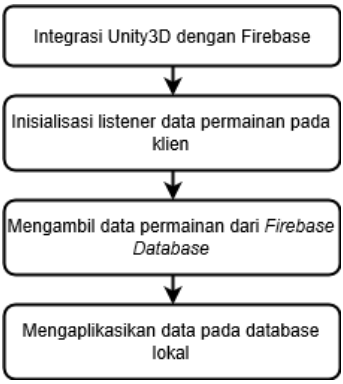
3.3.4 Memperbaharui *Data Game* dan Versi *Bundle*

Setelah AssetBundle ter-unggah kedalam *Firebase Storage*. Klien tidak akan langsung mengunduh *AssetBundle* baru tersebut. Segala perubahan terhadap dependency juga harus dirubah melalui *realtime database.* Pengembang perlu memperbaharui nilai dari "assetbundleversion" pada *Firebase Real-time Database* agar perbadaan nilai *versi* dapat ditangkap oleh *Listener* pada klien. Kemudian klien akan melakukan pengecekan ulang terhadap aset-aset yang tersimpan pada penyimpanan lokal. Verifikasi aset akan dijelaskan pada subbab berikutnya.

3.4 Inisialisasi Data Game pada Klien

Pada subbab sebelumnya, telah dijelaskan langkah-langkah dalam pengembangan sistem pengelolaan data game waktu nyata. Pada subbab ini akan dijelaskan proses integrasi klien dengan database Firebase sehingga data permainan menjadi responsive terhadap

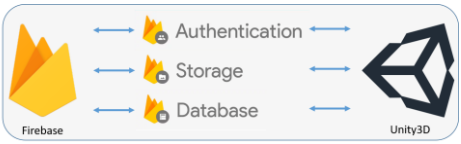
pembaruan. Berikut merupakan alur dari proses inisialisasi data permainan, diilustrasikan pada gambar 3.15.



Gambar 3.15 Alur inisialisasi data permainan pada klien

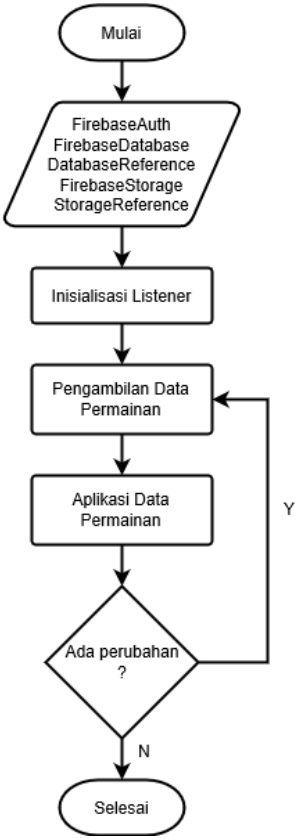
3.4.1 Integrasi *Unity3D* dengan *Firebase*

Sebelum sistem *seamless update* dapat berfungsi, pertama yang harus diatur terlebih dahulu ialah menyiapkan segala kebutuhan dari software dan plugin yang digunakan dalam pembuatan aplikasi, terutama pengintegrasian *IDE Unity3D* dengan platform pengembangan *Firebase*. *Software* dan *plugin* ini digunakan untuk menyimpan serta menghubungkan data pada game *Saga Legacy* di penyimpanan perangkat lokal dengan *cloud database* sehingga setiap perubahan data yang terjadi pada *cloud database* dapat diteruskan ke setiap perangkat dalam waktu nyata. Pada projek ini, saya memakai tiga SDK Firebase yang dapat diilustrasikan pada gambar 3.16.



Gambar 3.16 Integrasi *Unity3D* dengan *Firebase* menggunakan tiga *firebase sdk*.

Saat game pertama kali dibuka, terdapat *ConnectionHandler* yang mengatur segala koneksi *client* dengan *firebase*. Diawali dengan menginisiasi *FirebaseApp* dan mengatur *url database* untuk *real-time database*. Dilanjutkan dengan menginisiasi *Firebase Storage* dan memulai listerner yang menangani perubahan data. Data tersebut akan tersinkronisasi dengan klien sehingga apabila sewaktu-waktu terjadi perubahan, data pada klien dapat langsung diperbaharui. Proses ini dapat dilihat pada gambar 3.17.



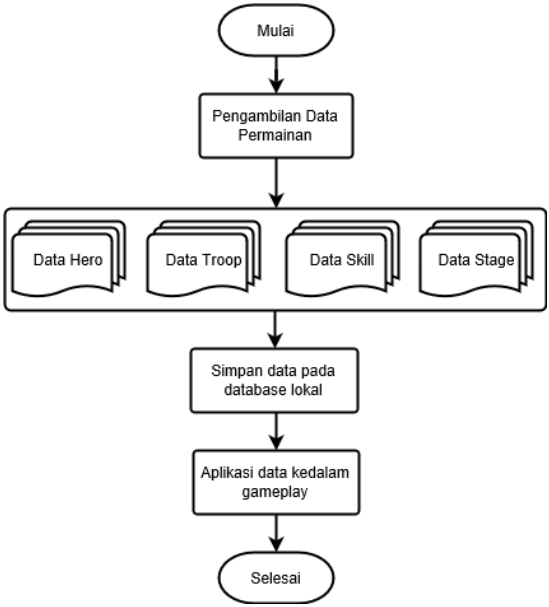
Gambar 3.17 Proses integrasi data permainan dengan *Firebase*

3.4.2 Inisialisasi *Listener* Data Permainan pada Klien

Inisiasi *Listener* dilakukan diawal ketika *game* baru saja dijalankan. Pemain akan dihadapkan pada *scene* verifikasi data, yang mencocokkan data pada penyimpanan lokal dengan database. Ada 4 listener data permainan yang diinisialisasi pada tahap ini, sesuai dengan rancangan struktur data pada subbab 3.2. Struktur data tersebut yaitu *data hero*, *data troop*, *data skill*, dan *data stage*. Nilai data dari keempat struktur ini akan senantiasa tersinkronisasi dengan *database* waktu nyata *Firebase*.

3.4.3 Mengambil Data Permainan dari *Firebase Database*

Setelah inisialisasi listener, sistem akan mengambil nilai dari keempat tersebut untuk memastikan data yang dipakai pada klien merupakan data permainan yang terbaru sebelum permainan dimulai. Proses ini dapat dilihat pada gambar 3.18.



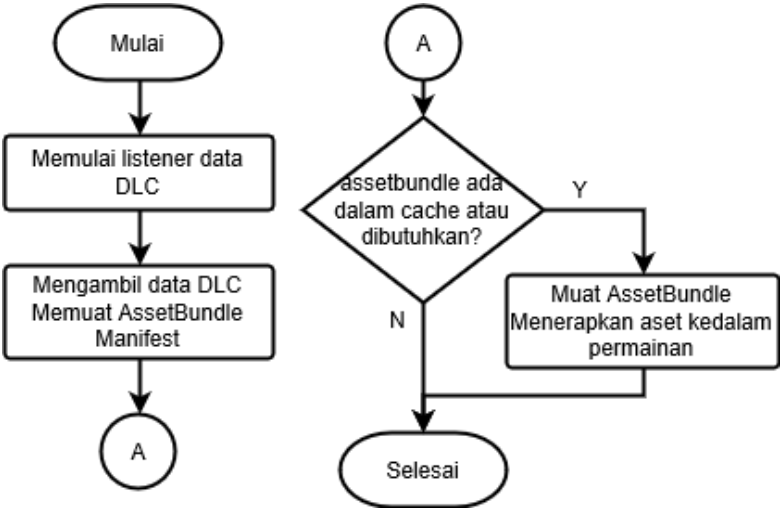
Gambar 3.18 Alur pengambilan dan pengaplikasian data permainan

3.4.4 Mengaplikasikan Data Pada *Database* Lokal

Berdasarkan struktur dari data. Data yang telah diambil dari database Firebase mendapat perlakuan yang berbeda. Untuk data hero, troop, dan skill perubahaan akan langsung diteruskan kedalam gameplay. Tidak peduli meskipun pemain sedang didalam stage, data akan langsung dirubah dan mempengaruhi gameplay. Sedangkan untuk data stage. Perubahan baru terjadi hanya pada awal stage.

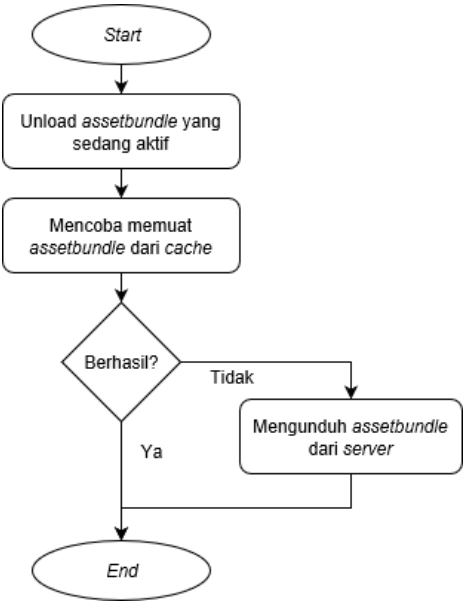
3.5 Inisialisasi Data DLC pada Klien

Setelah data permainan terinisialisasi sistem akan melakukan pengecekan aset pada penyimpanan lokal. Dimulai dengan menginisialisasi *listener* data DLC yang menyimpan informasi aset yang dibutuhkan untuk menjalankan permainan. Data ini digunakan untuk menentukan apakah suatu *assetbundle* harus diunduh atau tidak berdasarkan status dari DLC. Alur dari proses verifikasi aset ini dapat dilihat pada gambar 3.19.



Gambar 3.19 Alur inisialisasi data DLC pada klien

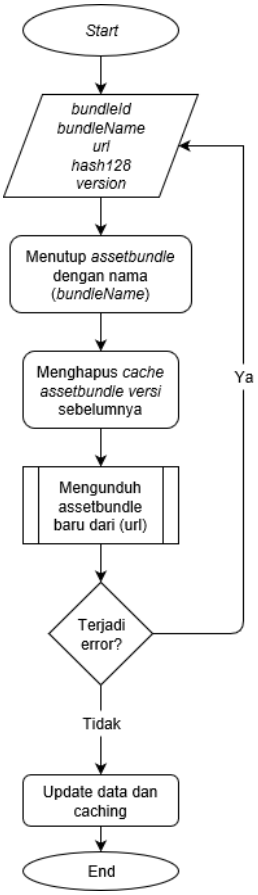
Apabila nilai *hash* dari *AssetBundle* yang tersimpan sudah sama dengan nilai pada *database*. Sistem akan mencoba untuk memuat aset dari *cache* pada penyimpanan lokal. Proses pengambilan data *AssetBundle* dari *cache* dapat dilihat pada gambar 3.20.



Gambar 3.20 Proses muat AssetBundle dari cache

Apabila sistem tidak berhasil memuat *assetbundle* dari penyimpanan lokal maka menggunakan informasi url, hash, dan versi dari bundle, sistem akan mengunduh *assetbundle* pembaharuan dari *server*, kemudian menyimpannya pada penyimpanan lokal.

Proses pengunduhan assetbundle baru dapat dilihat pada gambar 3.21.

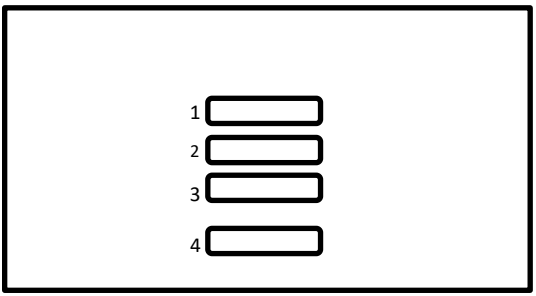


Gambar 3.21 Proses unduh assetbundle dari server

3.5 *User Login* dan Memuat Data Pengguna

Setelah semua data DLC terinisialisasi, pengguna akan dihadapkan pada panel *login*. Pada panel ini, pengguna dapat memilih

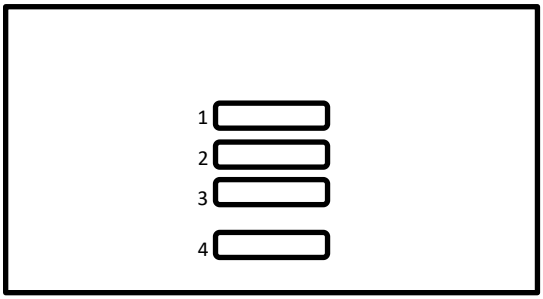
untuk melakukan *login* menggukanan surel apabila pengguna telah memiliki akun sebelumnya. Berikut adalah sketsa dari panel *login*, ditunjukkan pada gambar 3.22.



Gambar 3.22 Sketsa *panel login* menggunakan surel

1) *Input* surel pengguna
2) *Input password* akun
3) Tombol *login*
4) Tombol registrasi akun

Apabila pengguna belum memiliki akun, pengguna dapat melakukan registrasi menggunakan surel dengan menekan tombol registrasi. Berikut merupakan ilustrasi dari panel registrasi, dapat dilihat pada gambar 3.23.



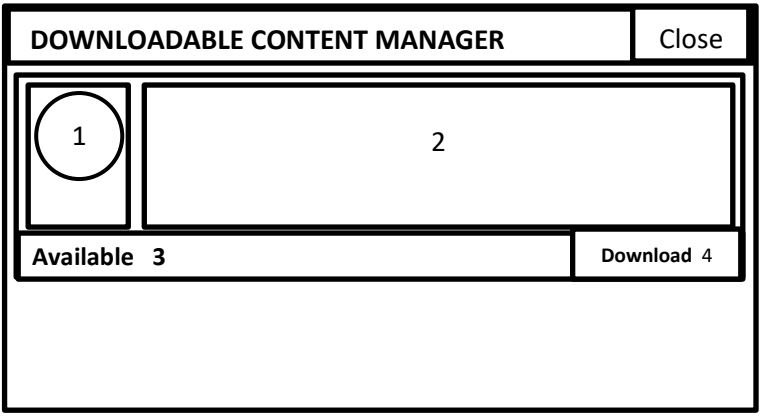
Gambar 3.23 *Panel* registrasi menggunakan surel

1) *Input* surel pengguna
2) *Input password* akun
3) Tombol registrasi menggukan surel
4) Tombol batal registrasi

Setelah pengguna berhasil masuk menggunakan surel mereka, sistem akan mengambil data pengguna dari akun terkait dan membaca daftar DLC yang pernah diunduh. Masing-masing DLC pada daftar DLC akan diunduh kedalam penyiimpanan lokal perangkan.

3.6 Pengelolaan dan Pengaplikasian DLC

Didalam permainan terdapat DLC *Manager* yang dapat digunakan oleh pemain untuk mengelola DLC yang tersedia. *Panel* DLC *Manager* ini memungkinkan pemain untuk mengunduh dan menghapus setiap DLC yang tersedia pada *Saga Legacy*. Pembaruan aset melalui DLC ini dilakukan dengan seamless update, yaitu tanpa *loading screen* ataupun mengharuskan pemain untuk membuka ulang *Saga Legacy*. Sehingga pemain dapat mengelola DLC tanpa mengganggu jalannya permainan. Desain dari DLC *Manager* dapat dilihat pada gambar 3.24 berikut.



Gambar 3.24 Sketsa DLC *Manager*

1) Ikon dari DLC
2) Deskripsi mengenai DLC
3) Status dari DLC apakah tersedia atau sudah diunduh
4) Tombol unduh atau menghapus DLC

Penunjuk status DLC juga digunakan untuk menampilkan progress unduh ketika pemain melakukan pengunduhan DLC. Setelah proses pengunduhan selesai, nama DLC akan ditambahkan pada data pengguna sebagai referensi apabila pengguna melakukan login pada perangkat lain, sehingga data tetap tersinkronisasi antar perangkat. Dengan pengelolaan DLC ini, ukuran instalasi permainan menjadi lebih kecil dan pembaruan dapat dilakukan dengan lebih cepat serta mengurangi gangguan saat proses pembaruan ketika pemain harus membuka ulang atau menunggu pada halaman memuat. Pemain dapat melanjutkan permainan sementara sistem mengunduh DLC dan mengaplikasikannya ketika proses unduh selesai.

BAB 4
PENGUJIAN DAN ANALISA

Pada bab ini dilakukan pengujian terhadap sistem pengelolaan data permainan berbasis DLC untuk mengetahui apakah fungsi dari sistem yang direncanakan telah bekerja sesuai harapan. Pengujian dilakukan dalam beberapa bagian, yaitu pengujian pembaruan data dan penambahan DLC baru pada permainan, dan pengelolaan DLC pada klien apakah masing-masing DLC yang tersedia sudah dapat diunduh dan dijalankan dengan benar dan memiliki respon yang cepat terhadap perubahan data. Hasil pengujian digunakan untuk menilai tingkat keberhasilan sistem dalam mencapai tujuan dari tugas akhir, yaitu sebagai sistem pembaruan dan penambahan data untuk game *Saga Legacy* yang responsif terhadap perubahan data.

4.1 Metode Pengujian

Pengujian pada projek tugas akhir ini dilakukan untuk mengetahui seberapa responsif klien ketika terjadi perubahan data. Ada dua macam pengujian yang dilakukan, yang pertama adalah menguji fungsionalitas sistem, apakah sistem mampu bertindak sebagai alat bagi pengembang maupun pengguna untuk mengelelola data permainan dalam waktu nyata. Ini dilakukan dengan mengetes secara langsung pengunduhan karakter baru melalui DLC dan melihat apakah karakter dimuat dengan benar dan dapat digunakan. Yang kedua akan dilakukan pengujian performa sistem berdasarkan waktu mengunduh berdasarkan kecepatan dan ukuran konten yang diunduh pada jaringan yang berbeda sebagai ukuran untuk mengetahui tingkat efisiensi sistem dalam berperan sebagai sistem pengelolaan data permainan waktu nyata. Pengujian ini dilakukan dengan cara mengunduh masing-masing DLC yang tersedia dalam permainan dari *Firebase Storage*, dan menghitung waktu serta kecepatan unduh sistem pada jaringan yang berbeda-beda untuk memastikan bahwa sistem berjalan seperti yang diharapkan dan dapat meningkatkan efisiensi waktu dalam melakukan pembaruan data permainan.

4.2 Implementasi Pembaruan Data dan Penambahan DLC

Ada beberapa langkah yang harus dilakukan untuk menambah konten baru pada permainan *Saga Legacy*. Mulai dari menggolongkan aset hingga penambahan data dlc baru pada *database* waktu nyata *Firebase*. Pada pengujian ini, akan dilakukan penambahan karakter dengan mengunduh DLC yang tersedia.

Sebelum membangun *asssetbundle*, pengembang harus mengelompokkan masing-masing aset dalam permainan berdasarkan fungsinya. Dari hasil pengelompokan tersebut, berikut adalah daftar nama dan ukuran dari masing-masing *assetbundle* yang telah dibangun ditampilkan pada tabel 4.1

Tabel 4.1 Nama dan ukuran dari hasil bangun *assetbundle*

No.	Nama AssetBundle	Ukuran	Keterangan
1	Ajisaka	106 kb	*Hero* yang dapat dikendalikan pemain
2	Animation	9 kb	Animasi tampilan pengguna dan skenerio
3	Aryakasimpar	71 kb	*Hero* yang dapat dikendalikan pemain
4	Bgm	7.125 kb	Musik latar yang digunakan didalam permainan
5	Buto	39 kb	Musuh yang dihadapi pemain dalam *stage*
6	Butobos	38 kb	Musuh yang dihadapi pemain dalam *stage*
7	Butopanah	25 kb	Musuh yang dihadapi pemain dalam *stage*
8	Charactersprite	39 kb	Aset gambar karakter
9	Characterweapon	4 kb	Aset gambar dan *prefab* senjata
10	Fonts	1.596 kb	Tipografi yang digunakan dalam game

Tabel 4.1 Nama dan ukuran dari hasil bangun *assetbundle*

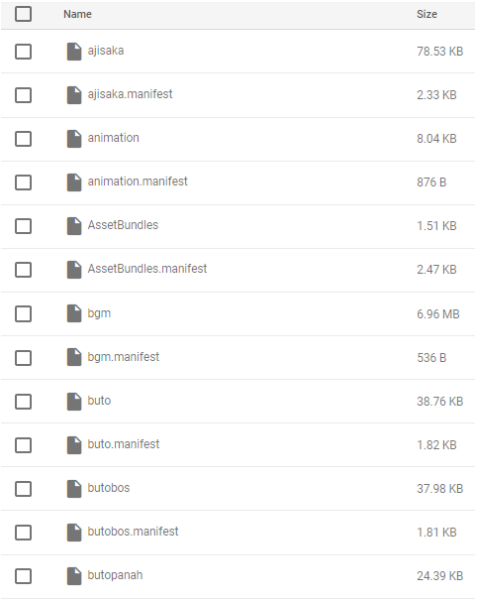
No.	Nama AssetBundle	Ukuran	Keterangan
11	Kendedes	36 kb	Karakter pembantu

			yang terlibat dalam jalan cerita.
12	Prajurit	44 kb	Pasukan yang berperang bersama *hero*.
13	Prajuritdukun	46 kb	Pasukan yang berperang bersama *hero*.
14	Prajuritpanah	50 kb	Pasukan yang berperang bersama *hero*.
15	Prefabs	3.545 kb	Template objek yang dapat dibuat *instance*-nya.
16	Rhoma	31 kb	Karakter pembantu yang terlibat dalam jalan cerita.
17	Sakera	81 kb	*Hero* yang dapat dikendalikan pemain
18	Sfx	19 kb	Efek suara yang digunakan dalam permainan.
19	Sprite	2.975 kb	Gambar latar dan *UI* dalam permainan
20	Textassets	2 kb	Aset teks yang digunakan dalam jalan cerita.
21	Weaponanimation	7 kb	Animasi senjata yang digunakan karakter.
22	Zscene	7.650 kb	Stage yang dapat dimainkan. Jalan cerita terdapat didalam stage.

Hampir semua assetbundle pada tabel diatas berstatus "*Necessary*" yang artinya sistem akan langsung mengunduh aset karena assetbundle tersebut dibutuhkan untuk menjalankan permainan. Hanya dua assetbundle yang berstatus "*Downloadable*" yaitu Sakera dan Arya Kasimpar yang dapat diunduh apabila diinginkan oleh pemain saat didalam permainan.

Selanjutnya pengembang mengunggah *assetbundle* yang sudah dibangun ke Firebase Storage dalam folder sesuai dengan versi bundle

yang dikehendaki. Nama assetbundle akan digunakan sebagai referensi dari database storage, sehingga meskinpun assetbundle pada suatu folder diperbaharui dengan link yang berbeda. Pengguna tetap dapat mengakses assetbundle terkait menggunakan referensi nama tersebut. Berikut adalah assetbundle yang telah dibangun dan diunggah ke Firebase Storage dapat dilihat pada gambar 4.1.



Gambar 4.1 *AssetBundle* pada *Firebase Storage*

Setelah aset-aset yang dibutuhkan pada permainan diunggah, pengembang lalu menambahkan data DLC pada *Firebase Database* agar *assetbundle* dapat diakses oleh klien. Data DLC memiliki key yang sama dengan nama *assetbundle* pada *Firebase Storage* sehingga dapat direferensi untuk mengambil tautan unduh dari DLC terkait. Selain itu, terdapat parameter "*version*" yang merupakan versi dari assetbundle pada DLC. *Version* merupakan referensi folder tempat *assetbundle* berada. Data DLC pada *Firebase Database* dapat dilihat pada gambar 4.2.

Gambar 4.2 Data DLC pada *Firebase Database*

4.3 Pengelolaan *DLC* (*Downloadable Content*) pada Klien

Pemain dapat mengelola *DLC* yang tersedia dalam permainan dengan membuka *DLC Manager*. *DLC Manager* memberikan kemampuan pada pemain untuk memilih mengunduh dan menghapus konten yang ada pada permainan *Saga Legacy* tanpa mengganggu jalannya permainan. Pada panel DLC *Manager* terdapat daftar DLC yang tersedia serta penjelasan mengenai DLC terkait. Panel pilih *hero* dapat dilihat pada gambar 4.3.

Gambar 4.3 *Screenshot* pilih karakter sebelum mengunduh DLC

Pada gambar 4.3, merupakan tampilan pilih hero yang menampilkan hanya satu karakter saja pada bar daftar hero, yaitu Aji Saka. Apabila pengguna ingin menambahkan karakter yang dapat dimainkan, pengguna dapat menekan tombol "*DLC Manager*" pada bagian pojok kiri bawah layar. Kemudian proses memuat "*DLC Manage*r" akan dijalankan seperti yang terlihat pada gambar 4.4.

Gambar 4.4 *Screenshot* pengguna memuat *DLC manager*

Pada panel DLC Manager terdapat daftar DLC yang tersedia dalam permainan Saga Legacy, salah satunya adalah karakter Arya Kasimpar yang merupakan salah satu hero yang dapat dimainkan. Ketika player menekan tombol download, maka proses pengununduhan assetbundle akan dilakukan tanpa mengganggu permainan. Hal tersebut ditampilkan seperti gambar 4.5.

Gambar 4.5 *Screenshot* pengguna telah mengunduh DLC Arya Kasimpar

Setelah proses unduh selesai, aset akan langsung dimuat dan ditampilkan pada tampilan pilih hero. Pengguna kemudian dapat memilih hero tersebut dan dimainkan pada stage seperti pada gambar 4.6.

Gambar 4.6 *Screenshot* Arya Kasimpar setelah pengunduhan DLC

Dengan sistem ini pengguna dapat memilih konten mana yang ingin disimpan dalam perangkat mereka namun tetap mengetahui apabila pengembang melakukan pembaruan atau penambahan konten.

4.4 Pengujian Performa Sistem

Pengujian terhadap performa sistem dibagi menjadi dua bagian. Bagian pertama ialah pengujian sistem terhadap perbedaan jaringan. Tujuan dari pengujian bagian pertama ialah untuk menguji waktu unduh berdasarkan ukuran berkas dan kecepatan jaringan. Pengujian bagian kedua ialah pengujian sistem terhadap perbandingan waktu pembaruan

menggunakan *DLC Manager* dan dengan tanpa menggunakan *DLC Manager*. Tujuan dari pengujian bagian kedua ialah untuk melihat perbedaan sistem yang menggunakan *DLC Manager* dan tidak menggunakan *DLC Manager* terhadap waktu. Pengujian ini dilakukan dengan spesifikasi perangkat sebagai berikut :

1. Operator jaringan : Telkomsel
2. Lokasi : Surabaya, Indonesia

Sedangkan untuk spesifikasi handphone yang digunakan ialah :

1. Merk Handphone : Xiao-Mi A1
2. OS Android : 8.0.0 (Oreo)
3. Chipset : Qualcomm MSM8953 Snapdragon 625
4. CPU : Octa-core 2.0 GHz Contex-A53
5. GPU : Adreno 506

4.4.1 Pengujian Sistem terhadap Perbedaan Jaringan

Pengujian ini dilakukan untuk mengetahui kecepatan respon sistem dan kecepatan unduh pada 3 jaringan yang berbeda yaitu 2G, 3G, dan 4G.

Pengujian pada jaringan 4G mendapatkan hasil yang ditunjukkan pada tabel 4.2. Dari data pada tabel dapat dilihat ada ketidakstabilan kecepatan seperti saat mengunduh *assetbundle "characterweapon"* yang memiliki ukuran hanya 4 *kilobyte*, membutuhkan waktu 1,38 detik, tidak berbeda jauh dengan waktu mengunduh prefab yaitu 1,65 detik meskipun ukuran *file*-nya yang jauh lebih besar yatiu 3.545 *kilobyte*. Selain itu waktu tercepat yang didapat adalah 0.76 detik dengan ukuran *bundle* 106 *kilobyte*, meskipun terdapat *assetbundle* dengan ukuran yang jauh lebih kecil, hal ini bisa terjadi dikarenakan terdapat *delay* saat permintaan unduhan ataupun faktor lain. Kecepatan maksimal pada jaringan 4G ialah 2873,0 *kilobyte* tiap detik saat mengunduh berkas *AssetBundle* Bgm. Sedangkan kecepatan minimum pada jaringan 4G ialah 2,0 *kilobyte* tiap detik saat mengunduh

AssetBundle Textassets. Rata-rata kecepatan pada koneksi 4G yaitu 529,6 *kilobyte* tiap detik.

Tabel 4.2 Waktu unduh konten berdasarkan kecepatan dan ukuran berkas pada jaringan 4G.

No.	Nama AssetBundle	Ukuran AssetBundle	Waktu Mengunduh	Kecepatan kb/s
1.	Ajisaka	106 kb	0.76 detik	139 kb/s
2	Animation	9 kb	1,0 detik	9,0 kb/s
3	Aryakasimpar	71 kb	1,66 detik	42,8 kb/s
4	Bgm	7.125 kb	2,48 detik	2873,0 kb/s
5	Buto	39 kb	1,45 detik	26,9 kb/s
6	Butobos	38 kb	0,82 detik	46,3 kb/s
7	Butopanah	25 kb	0,82 detik	30,5 kb/s
8	Charactersprite	39 kb	0,82 detik	47,6 kb/s
9	Characterweapon	4 kb	1,38 detik	2,9 kb/s
10	Fonts	1.596 kb	1,02 detik	1564,7 kb/s
11	Kendedes	36 kb	0,82 detik	43,9 kb/s
12	Prajurit	44 kb	0,78 detik	56,4 kb/s
13	Prajuritdukun	46 kb	0,82 detik	56,1 kb/s
14	Prajuritpanah	50 kb	1,46 detik	34,2 kb/s
15	Prefabs	3.545 kb	1,65 detik	2148,5 kb/s
16	Rhoma	31 kb	1,42 detik	21,8 kb/s
17	Sakera	81 kb	1,44 detik	56,3 kb/s
18	Sfx	19 kb	0,78 detik	24,4 kb/s
19	Sprite	2.975 kb	1,67 detik	1781,4 kb/s
20	Textassets	2 kb	1,01 detik	2,0 kb/s
21	Weaponanimation	7 kb	1,49 detik	4,7 kb/s
22	Zscene	7.650 kb	2,9 detik	2637,9 kb/s

Kecepatan unduh pada jaringan 3G ditunjukkan oleh tabel 4.3 dan mengalami penurunan daripada 4G meskipun tidak terlalu signifikan, rata-rata kecepatan turun dari 529,6 *kilobyte* tiap detik, menjadi 412,5 *kilobyte* tiap detik. Namun perbedaan cukup terlihat pada saat pengunduhan assetbundle dengan ukuran besar. Kecepatan maksimal pada jaringan 3G ialah 2233,5 *kilobyte* tiap detik saat mengunduh berkas *AssetBundle* Bgm. Sedangkan kecepatan minimum

pada jaringan 3G ialah 1,4 *kilobyte* tiap detik saat mengunduh *AssetBundle* Textassets. Pengunduhan *AssetBundle* dengan ukuran kecil masih dipengaruhi waktu jeda sebelum memulai pengunduhan. Jeda yang terjadi dikarenakan oleh faktor *request* unduh oleh perangkat ke database. Data tersebut dapat dilihat pada tabel 4.3 dibawah ini.

Tabel 4.3 Waktu unduh konten berdasarkan kecepatan dan ukuran berkas pada jaringan 3G.

No.	Nama AssetBundle	Ukuran AssetBundle	Waktu Mengunduh	Kecepatan kb/s
1.	Ajisaka	106 kb	1,41 detik	75,2 kb/s
2	Animation	9 kb	1,03 detik	8,7 kb/s
3	Aryakasimpar	71 kb	1,63 detik	43,6 kb/s
4	Bgm	7.125 kb	3,19 detik	2233,5 kb/s
5	Buto	39 kb	1,40 detik	27,9 kb/s
6	Butobos	38 kb	1,44 detik	26,4 kb/s
7	Butopanah	25 kb	0,82 detik	30,5 kb/s
8	Charactersprite	39 kb	1,01 detik	38,6 kb/s
9	Characterweapon	4 kb	0,97 detik	4,1 kb/s
10	Fonts	1.596 kb	1,40 detik	1140,0 kb/s
11	Kendedes	36 kb	1,39 detik	25,9 kb/s
12	Prajurit	44 kb	1,01 detik	43,6 kb/s
13	Prajuritdukun	46 kb	1,40 detik	32,9 kb/s
14	Prajuritpanah	50 kb	1,41 detik	35,5 kb/s
15	Prefabs	3.545 kb	2,02 detik	1755,0 kb/s
16	Rhoma	31 kb	1,38 detik	22,5 kb/s
17	Sakera	81 kb	1,44 detik	56,3 kb/s
18	Sfx	19 kb	1,05 detik	18,1 kb/s
19	Sprite	2.975 kb	2,24 detik	1328,1 kb/s
20	Textassets	2 kb	1,44 detik	1,4 kb/s
21	Weaponanimation	7 kb	0,82 detik	8,5 kb/s
22	Zscene	7.650 kb	3,61 detik	2119,1 kb/s

Saat mengunduh pada jaringan 2G yang ditunjukkan oleh tabel 4.4, kecepatan unduh turun drastis jika dibandingkan dengan pengujian sebelumnya dengan kecepatan maksimal 24,5 *kilobyte* tiap detik pada saat mengunduh *AssetBundle* Sprite dan paling lamban dengan

kecepatan 1,1 *kilobyte* tiap detik pada saat mengunduh *AssetBundle* 1,1 kb/s. Kecepatan rata-rata unduh pada jaringan 2G adalah 12,4 *kilobyte* tiap detik Meskipun waktu unduh yang lama, pembaruan berhasil diterapkan sehingga pembaruan tetap bisa dilakukan bahkan pada koneksi yang terbilang lambat.

Tabel 4.4 Waktu unduh konten berdasarkan kecepatan dan ukuran berkas pada jaringan 2G.

No.	Nama AssetBundle	Ukuran AssetBundle	Waktu Mengunduh	Kecepatan kb/s
1.	Ajisaka	106 kb	6,28 detik	16,9 kb/s
2	Animation	9 kb	2,15 detik	4,2 kb/s
3	Aryakasimpar	71 kb	6,10 detik	11,6 kb/s
4	Bgm	7.125 kb	326,52 detik	21.8 kb/s
5	Buto	39 kb	4,82 detik	8,1 kb/s
6	Butobos	38 kb	3,54 detik	10,7 kb/s
7	Butopanah	25 kb	2,47 detik	10,1 kb/s
8	Charactersprite	39 kb	2,45 detik	15,9 kb/s
9	Characterweapon	4 kb	1,60 detik	2,5 kb/s
10	Fonts	1.596 kb	75,09 detik	21,3 kb/s
11	Kendedes	36 kb	3,37 detik	10,7 kb/s
12	Prajurit	44 kb	4,26 detik	10,3 kb/s
13	Prajuritdukun	46 kb	3,28 detik	14,0 kb/s
14	Prajuritpanah	50 kb	3,00 detik	16,7 kb/s
15	Prefabs	3.545 kb	242,73 detik	14,6 kb/s
16	Rhoma	31 kb	3,54 detik	8,8 kb/s
17	Sakera	81 kb	5,43 detik	14,9 kb/s
18	Sfx	19 kb	2,14 detik	8,9 kb/s
19	Sprite	2.975 kb	121,42 detik	24,5 kb/s
20	Textassets	2 kb	1,85 detik	1,1 kb/s
21	Weaponanimation	7 kb	2,18 detik	3,2 kb/s
22	Zscene	7.650 kb	358,78 detik	21,3 kb/s

Hasil dari pengujian ini menunjukkan seberapa efektifkah kecepatan jaringan 4G, 3G dan 2G terhadap sistem pengelolaan data permainan berbasis DLC pada projek tugas akhir ini dalam membantu pengembang merilis pembaharuan data pada pengguna dengan

menerapkan *seamless update*. Hasil pengujian tersebut ialah sebagai berikut :

1. Jaringan 4G
    a. Kecepatan Maksimum : 2873,0 *kilobyte* per detik pada *AssetBundle* Bgm.
    b. Kecepatan Minimum : 2,0 *kilobyte* tiap detik pada *AssetBundle* Textassets.
    c. Kecepatan Rata-Rata : 529,6 *kilobyte* tiap detik
2. Jaringan 3G
    a. Kecepatan Maksimum : 2233,5 *kilobyte* per detik pada *AssetBundle* Bgm.
    b. Kecepatan Minimum : 1,4 *kilobyte* tiap detik pada *AssetBundle* Textassets.
    c. Kecepatan Rata-Rata : 412,5 *kilobyte* tiap detik
3. Jaringan 2G
    a. Kecepatan Maksimum : 24,5 *kilobyte* per detik pada *AssetBundle* Sprite.
    b. Kecepatan Minimum : 1,1 *kilobyte* tiap detik pada *AssetBundle* Textassets.
    c. Kecepatan Rata-Rata : 12,37 *kilobyte* tiap detik

Dari hasil tersebut dapat diambil kesimpulan bahwa kecepatan unduh paling cepat ialah pada jaringan 4G dengan kecepatan rata-rata 529,6 *kilobyte* tiap detik dan kecepatan unduh paling lambat ialah pada jaringan 2G dengan kecepatan rata-rata 12,37 *kilobyte* tiap detik. Hasil tersebut menunjukkan apabila kita mengunduh permainan tersebut dengan jaringan 4G, maka sistem akan berjalan lebih cepat daripada *user* yang menggunakan jaringan 2G. Sedangkan pada jaringan 3G, kecepatan unduh rata-rata tidak berbeda jauh dengan jaringan 4G, dengan kecepatan rata-rata 412,5 *kilobyte* tiap detik.

4.4.2 Perbandingan Waktu Pembaruan Menggunakan *DLC* Manager dengan Tidak Menggunakan *DLC Manager*

Seiring dengan penambahan konten baru, ukuran *file* instalasi akan semakin besar, membuat waktu unduh *file* instalasi semakin lama. Ukuran file instalasi awal sebelum ditambahkan DLC adalah 47,387 kb. Pengujian ini dilakukan untuk mengetahui tingkat efisiensi waktu unduh menggunakan DLC Manager dengan cara membandingkan waktu unduh

tersebut dengan mengunduh *file* instalasi ulang. Hal tersebut dijelaskan pada persamaan 4.1.

$$\square = \frac{\square\square\square\square\square\ \square\square\square\square\square\ \square\square\square\ \square\square\square\square\square\square\square}{\square\square\square\square\square\ \square\square\square\square\square\ \square\square\square\square\square\ \square\square\square\square\square\square\square\square\square\square\square\square\ \square\square\square} \times 100\% \qquad (4.1)$$

Dimana *x* merupakan presentase waktu unduh dengan menggunakan *DLC Manager* jika dibandingkan dengan waktu unduh keseluruhan *file* instalasi. Hasil dari pengujian ini akan didapatkan nilai tingkat efisiensi waktu unduh menggunakan *DLC Manager* dengan cara menghitung waktu unduh menggunakan *DLC Manager* dibandingkan dengan waktu unduh tidak menggunakan *DLC Manager*.

Pengambilan data dalam pengujian ini diambil dari waktu unduh dalam pengunduhan satu demi satu *AssetBundle* yang terdapat pada tabel 4.1. Hasil dari pengujian waktu unduh menggunakan *DLC Manager* dengan tidak menggunakan *DLC Manager* dijelaskan oleh tabel 4.5.

Tabel 4.5 Waktu unduh konten berdasarkan kecepatan dan ukuran berkas pada jaringan Wi-Fi.

No.	AssetBundle	Waktu unduh DLC	Waktu unduh tidak DLC	% waktu
1.	Ajisaka	0,76 detik	27,01 detik	2,81 %
2	Animation	1,0 detik	28,01 detik	3,55 %
3	Aryakasimpar	1,66 detik	29,67 detik	5,59 %
4	Bgm	2,48 detik	31,15 detik	7,96 %
5	Buto	1,45 detik	32,55 detik	4,45 %
6	Butobos	1,44 detik	33,99 detik	4,24 %
7	Butopanah	0,82 detik	34,81 detik	2,35 %
8	Charactersprite	0,82 detik	35,63 detik	2,30 %
9	Characterweapon	1,38 detik	37,01 detik	3,73 %
10	Fonts	1,02 detik	38,03 detik	2,68 %
11	Kendedes	0,82 detik	38,85 detik	2,11 %
12	Prajurit	0,78 detik	39,63 detik	1,97 %
13	Prajuritdukun	0,82 detik	40,45 detik	2,03 %
14	Prajuritpanah	1,46 detik	41,91 detik	3,48 %
15	Prefabs	1,65 detik	43,56 detik	3,79 %
16	Rhoma	1,42 detik	44,98 detik	3,15 %

17	Sakera	1,44 detik	46,42 detik	3,10 %
18	Sfx	0,78 detik	47,2 detik	1,65 %
19	Sprite	1,67 detik	48,87 detik	3,42 %
20	Textassets	1,01 detik	49,88 detik	1,97 %
21	Weaponanimation	1,49 detik	51,37 detik	2,90 %
22	Zscene	2,9 detik	54,27 detik	5,34 %

Pada pengujian tersebut menunjukkan bahwa rata-rata waktu unduh dengan menggunakan DLC ialah 1,32 detik. Sedangkan rata-rata waktu unduh dengan tidak menggunakan DLC ialah 39,78 detik. Sehingga didapatkan persentase waktu unduh dengan menggunakan *DLC Manager* jika dibandingkan dengan waktu unduh keseluruhan *file* instalasi menggunakan persamaan 4.1 ialah 3.31%. Dari pengujian ini dapat ditarik kesimpulan bahwa dengan menggunakan *DLC Manager*, maka hanya memerlukan waktu 3.31% dari waktu pengunduhan dan instalasi file ulang. Angka 3.31% tersebut menunjukkan bahwa dengan menggunakan *DLC Manager*, maka akan meningkatkan efisiensi waktu 30 kali lebih cepat *user* dalam menerima pembaharuan konten dari pengembang.

BAB 5
PENUTUP

5.1 Kesimpulan

Dengan adanya sistem pengelolaan data permainan waktu nyata pada Saga Legacy ini, data permainan menjadi dinamis dan responsif terhadap penambahan dan pembaruan data, sehingga memudahkan pengembang dan pemain dalam melakukan pembaruan. Pengujian performa sistem dibagi menjadi dua bagian pengujian, yaitu pengujian pengujian sistem terhadap perbedaan jaringan dan pengujian sistem terhadap perbandingan waktu pembaruan menggunakan *DLC Manager* dan dengan tanpa menggunakan *DLC Manager*. Kesimpulan pada pengujian pertama ialah sistem akan berjalan lebih cepat pada jaringan 4G dengan kecepatan rata-rata 529,6 *kilobyte* per detik, tetapi sistem akan berjalan lambat apabila *user* menggunakan jaringan 2G dengan kecepatan rata-rata 12,37 *kilobyte* per detik. Lalu pada pengujian kedua mendapatkan kesimpulan bahwa pembaruan data menggunakan DLC dengan *seamless update* hanya membutuhkan 3.31% waktu yang dibutuhkan dibandingkan dengan pembaruan yang dilakukan dengan cara mengunduh ulang keseluruhan file instalasi yang akan terus bertambah besar seiring dengan terus bertambahnya konten dalam permainan. Sistem ini memungkinkan pemain untuk memilih konten mana yang ingin disimpan pada perangkat dan konten mana yang ingin dihapus tanpa harus menghentikan permainan terlebih dahulu, sehingga meminimalisir gangguan yang terjadi saat pengembang merilis konten baru. Dari rancangan hingga terimplementasinya sistem pengolaan data waktu nyata pada permainan Saga Legacy ini dapat memudahkan pengembang menambahkan konten baru pada permainan mereka, tanpa mengharuskan pemain untuk menghentikan permainan terlebih dahulu.

5.2 Saran

Penelitian ini masih memiliki beberapa kekurangan, salah satunya adalah belum bisa menambahkan script kedalam *AssetBundle*. Sehingga ketika pengembang ingin melakukan perubahan pada sistem permainan, *user* harus mengunduh ulang paket instalasi yang baru. Permasalahan tersebut dapat dijasikan acuan untuk penelitian

selanjutnya. Sehingga keseluruhan sistem menjadi *seamless* atau mengharuskan pemain menghentikan permainan mereka terlebih dahulu.

Halaman ini sengaja dikosongkan.

DAFTAR PUSTAKA

[1] Number of available applications in the Google Play Store from December 2009 to September 2017 https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/ Diakses pada 21 Nopember 2017

[2] Number of available apps in the Apple App Store from July 2008 to January 2017 https://www.statista.com/statistics/263795/number-of-available-apps-in-the-apple-app-store/ Diakses pada 21 Nopember 2017

[3] Downloadable Content https://www.giantbomb.com/downloadable-content/3015-329/ Diakses pada 12 Desember 2017

[4] Clifford P., Rory BHANDARI, Toby Rogers, “Database Management System”, FACE Recording and Measurements, Ltd. (2015) Abstract, US, 2015.

[5] Introducing JSON https://www.json.org/ Diakses pada 8 Oktober 2017.

[6] Struktur array - Gambar https://www.json.org/ Diakses pada 8 Oktober 2017.

[7] AssetBundles and the AssetBundle Manager https://unity3d.com/learn/tutorials/topics/scripting/assetbundles-and-assetbundle-manager Diakses pada 9 Oktober 2017

[8] Mesh Model - Gambar
https://unity3d.com/learn/tutorials/topics/scripting/assetbundles-and-assetbundle-manager Diakses pada 9 Oktober 2017

[9] What is NoSQL
https://www.mongodb.com/nosql-explained Diakses pada 12 Nopember 2017

[10] Cloud Storage, Teknologi Penyimpanan Digital Masa Kini
https://www.maxmanroe.com/cloud-storage-teknologi-penyimpanan-digital-masa-kini-2.html Diakses pada 12 Nopember 2017

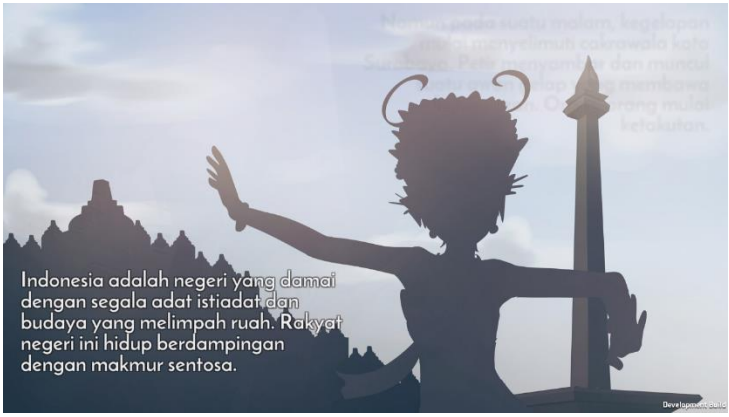
[11] Cloud Storage - Gambar
http://www.tutorialspoint.com/articles/how-cloud-storage-works Diakses pada 14 Nopember 2017

[12] Android 7.0: What are seamless updates and how do they work?
https://www.androidcentral.com/what-are-seamless-updates-and-how-will-they-work Diakses pada 2 Januari 2017

LAMPIRAN

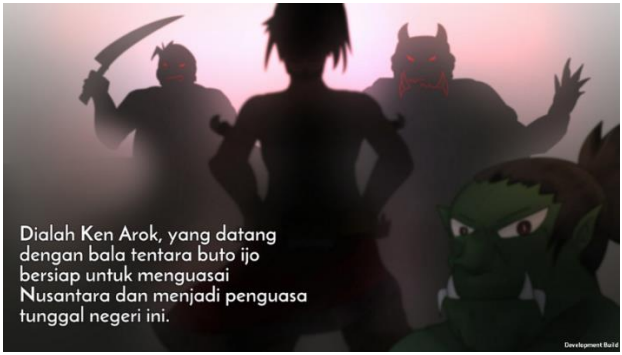
A. Petunjuk Bermain *Saga Legacy*

Pada saat *launch* game pertama kali, maka akan terdapat cerita prolog awal yang dilengkapi dengan gambar peristiwa dari cerita tersebut. Tampilan tersebut seperti gambar berikut :

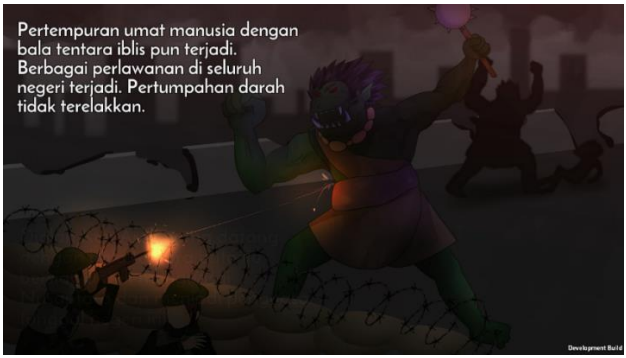


Gambar A.1 Prolog cerita 1

Gambar A.2 Prolog cerita 2

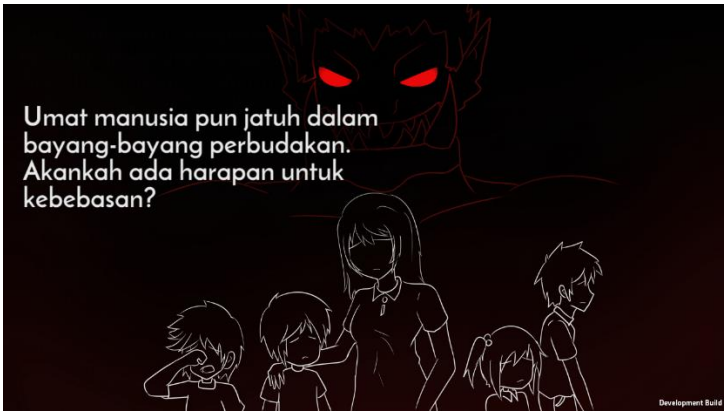


Gambar A.3 Prolog cerita 3



Gambar A.4 Prolog cerita 4

Gambar A.5 Prolog cerita 5



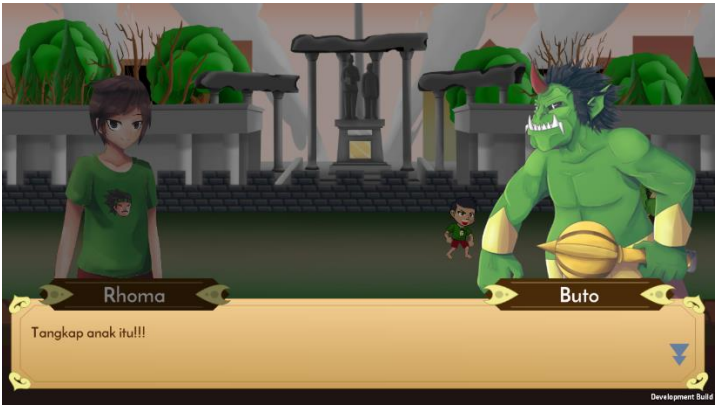
Gambar A.6 Prolog cerita 6

Setelah cerita tersebut, maka player akan memasuki *loading screen*, pada *loading screen* akan terdapat *trivia-trivia* mengenai sejarah bangsa Indonesia.

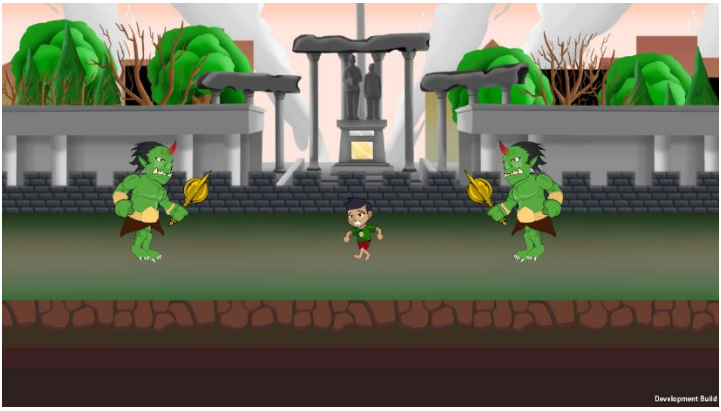
Gambar A.7 *Screenshot* tampilan loading denga trivia

Setelah *loading* selesai, maka akan keluar *button* yang bertuliskan "Done". Artinya *loading screen* telah selesai, player hanya tinggal menyentuh tombol tersebut untuk melanjutkan ke *screen* selanjutnya. Setelah itu akan muncul *cutscene* antara Rhoma dan Buto,

dimana Rhoma disini diselamatkan oleh Ken Dedes dan mendapatkan batu Mustika untuk memanggil Aji Saka.



Gambar A.8 Screenshot Rhoma dan Buto



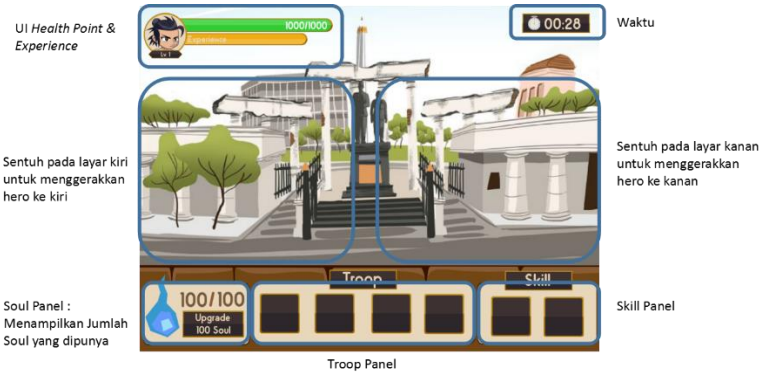
Gambar A.9 *Screenshot* Rhoma terkepung Buto

Gambar A.10 *Screenshot* munculnya Ken Dedes

Setelah *cutscene* selesai, maka *player* akan memainkan karakter Hero Aji Saka, sebagai tutorial awal dalam pertarungan, pada tahap ini, *player* akan memasuki *Stage*-Prolog. *Player* belum bisa *summon* pasukan, hanya menyerang dengan *Hero* Aji Saka saja.

Gambar A.11 *Screenshot* percakapan awal Aji Saka dan Buto

Setelah cutscene percakapan antara Aji Saka dan Buto selesai, maka pemain akan memasuki arena pertarungan dimana pemain mulai bermain untuk memainkan Hero Aji Saka. Berikut merupakan panduan *User Interface* pada saat pertarungan berlangsung.



Gambar A.12 Petunjuk mengenai elemen tampilan pengguna

Setelah bertarung dengan pasukan Buto ijo, maka akan muncul *Score Board* tanda perang sudah berakhir. Disini *player* bisa melihat waktu yang telah berhasil diselesaikan oleh *player* serta *score* yang didapatkan.

Gambar A.13 *Screenshot panel stage* selesai

Setelah itu *player* tinggal menekan tombol *back* yang ada untuk melanjutkan ke menu Home. Berikut merupakan tampilan menu Home.

Gambar A.14 *Screenshot* halaman *main menu*

Terdapat beberapa tombol pada menu Home ini, yaitu :

1. Menu Setting
   Menu setting ini ditandai dengan simbol gerigi pada pojok kanan atas bagian paling kiri. Pada menu ini, pemain bisa mematikan sound apabila tidak ingin bermain dengan menggunakan sound.
2. Help
   Menu ini untuk Help kedepannya.
3. Exit
   Menu ini untuk keluar dari permainan.
4. Adventure
   Menu ini untuk melanjutkan ke menu Stage Select. Menu Stage select tersebut tampilannya berupa pulau Indonesia, dan pada menu tersebut *player* bisa memilih *stage*. Tampilan menu memilih stage tersebut seperti tampilan berikut.

Gambar A.17 *Screenshot* halaman pemilihan daerah

Gambar A.16 *Screenshot* halaman pemilihan *stage*

Pada *Select Stage* ini, kita juga bisa memilih Hero mana yang ingin dipilih. Terdapat 3 pilihan hero dengan skill berbeda-beda, yaitu Aji Saka, Sakera dan Arya Kasimpar.

Gambar A.17 *Screenshot* halaman pemilihan *hero*

Setelah memilih hero, maka pemain dapat langsung memulai *stage* tersebut. Pemain juga bisa mengeluarkan pasukan-pasukan yang ia punya saat menyelesaikan stage yang lebih tinggi.

Gambar A.18 *Screenshot gameplay* pada *stage*

Halaman sengaja dikosongkan

BIOGRAFI PENULIS

Achmad Ridlo Nuur Abdillah, lahir pada 16 April 1995 di Lumajang. Penulis lulus dari SMP Negeri 1 Lumajang pada tahun 2010 kemudian melanjutkan pendidikan di SMA Negeri 2 Lumajang hingga lulus pada tahun 2013. Penulis kemudian melanjutkan pendidikan sarjana ke Departemen Teknik Komputer ITS Surabaya pada bidang studi Game dan Perangkat Mobile. Saat di kuliah penulis aktif dalam beberapa kegiatan organisasi termasuk Magang Asisten Lab B201 dan Commtech 2017. Selama masa kuliah penulis mengikuti beberapa perlombaan nasional seperti Gemastik, I-Fest, PKM, Comfest dan berhasil meraih beberapa penghargaan. Penulis sangat tertarik dengan pengembangan game dan segala hal yang berhubungan dengan komputer, dan berencana melanjutkan studi pada bidang yang berkaitan.