

**RANCANG BANGUN GAME ADVENTURE *GUNSLINGER*
DENGAN METODE *FINITE STATE MACHINE***

SKRIPSI



**Disusun Oleh:
Zulio Raynaldi Pakar Herlambang
11.18.001**

**PROGRAM STUDI TEKNIK INFORMATIKA S-1
FAKULTAS TEKNIK INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2015**

LEMBAR PERSETUJUAN DAN PENGESAHAN

**RANCANG BANGUN GAME ADVENTURE GUNSLINGER
DENGAN METODE FINITE STATE MACHINE**

SKRIPSI

*Disusun dan Diajukan untuk melengkapi dan memenuhi persyaratan guna
mencapai Gelar Sarjana Komputer Strata Satu (S-1)*

Disusun Oleh :

Zulio Raynaldi Pakar Herlambang

11.18.001

Diperiksa dan Disetujui,

Ketua Prodi Teknik Informatika S-1

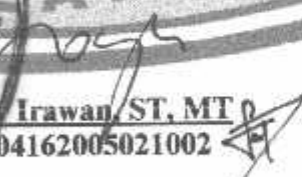
Dosen Pembimbing I

Dosen Pembimbing II


Ir. Yusuf Ismail Nakhoda, MT
NIP. Y. 1018800189


Sonny Prasetyo, ST.MT.
NIP. P. 1031000429

Ketua Prodi Teknik Informatika S-1


Joseph Dedy Irawan, ST, MT
NIP. 197404162005021002

**PROGRAM STUDI TEKNIK INFORMATIKA S-1
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2015**

LEMBAR KEASLIAN
PERNYATAAN KEASLIAN SKRIPSI

Saya yang bertanda tangan di bawah ini:

Nama : Zulio Raynaldi Pakar Herlambang
Nim : 11.18.001
Program Studi : Teknik Informatika S-1
Fakultas : Teknologi Industri

Menyatakan dengan sesungguhnya bahwa Skripsi saya yang berjudul:

“RANCANG BANGUN GAME ADVENTURE *GUNSLINGER* DENGAN METODE *FINITE STATE MACHINE*”

Adalah skripsi saya sendiri bukan duplikat serta mengutip atau menyadur seluruhnya karya orang lain kecuali dari sumber aslinya.

Malang, 17 Maret 2015

Yang membuat pernyataan



Zulio Raynaldi Pakar Herlambang

RANCANG BANGUN GAME ADVENTURE GUNSLINGER DENGAN METODE FINITE STATE MACHINE

Zulio Raynaldi Pakar Herlambang (1118001)

Jurusan Teknik Informatika S-1

Fakultas Teknologi Industri

Institut Teknologi Nasional Malang

Jl. Raya Karanglo Km. 2 Malang

Email: zulioraynaldi@gmail.com

Abstrak

Seiring perkembangan jaman, musuh-musuh dalam game sudah semakin cerdas. Hal ini dimaksudkan dengan tujuan membuat game menjadi lebih menarik dan memberikan tantangan kepada pemain. Saat game pertama kali dibuat, musuh dalam game hanya diam atau bergerak secara monoton tanpa bisa mengejar pemain. Untuk membuat sebuah musuh yang dapat merespon keberadaan pemain, maka teknologi Artificial Intelligence (AI) diterapkan kedalam sebuah game dan salah satu metode yang sring digunakan adalah Finite State Machine. Game Adventure dengan grafis 2D merupakan salah satu genre game yang menggunakan metode Finite State Machine sebagai musuh pemain.

Metode Finite State Machine (FSM) diterapkan kepada musuh dengan memberikan sensor sebagai indikator respon terhadap gerakan pemain. Game adventure ini dibangun menggunakan aplikasi Adobe flash CS6 dan bahasa pemrograman Actionscript 2.0 sehingga proses perancangan karakter, stage, dan metode FSM dapat dicapai. Dalam game ini terdapat beberapa jenis musuh yaitu minion dan elite minion, serta disetiap stage memiliki boss yang memiliki keunikan. Perilaku yang dilakukan oleh musuh meliputi melihat pemain, mengejar pemain, dan menyerang pemain.

Untuk pengujian sistem diambil 10 orang responden untuk menguji program ini. Dari hasil pengujian pengguna dapat disimpulkan untuk pergerakan karakter 40% responden mengatakan sangat baik, 50% mengatakan cukup baik, dan 10% mengatakan kurang baik. Untuk respon musuh atau AI 40% mengatakan sangat baik, 40% mengatakn cukup baik, dan 20% mengatakan kurang baik.

Kata Kunci: *Adventure Game, Finite State Machine, Actionscript 2.0, Adobe Flash CS6.*

KATA PENGANTAR

Puji syukur penulis panjatkan kehadirat Tuhan Yang Maha Esa atas berkat dan rahmat-Nya, penyusunan skripsi yang berjudul “RANCANG BANGUN GAME ADVENTURE *GUNSLINGER* DENGAN METODE *FINITE STATE MACHINE*” dapat diselesaikan dengan baik.

Penulis menyadari bahwa dalam proses penulisan skripsi ini banyak mengalami kendala, namun berkat bantuan, bimbingan dan kerjasama dari berbagai pihak sehingga kendala – kendala yang dihadapi tersebut dapat diatasi. Untuk itu penulis menyampaikan ucapan terima kasih kepada orangtua penulis yang senantiasa mendoakan, memberikan bantuan moril, materi, dan nasehat selama penulis menjalani pendidikan.

Selanjutnya ucapan terima kasih penulis sampaikan pula kepada:

1. Ir. Soeparno Djiwo, MT, selaku Rektor Institut Teknologi Nasional Malang.
2. Ir. Anang Subardi, MT, selaku Dekan Fakultas Teknologi Industri, Institut Teknologi Nasional Malang.
3. Joseph Dedy Irawan, ST, MT, selaku Ketua Program Studi Teknik Informatika, Institut Teknologi Nasional Malang.
4. Sonny Prasetio, ST, MT, selaku Sekertaris Program Studi Teknik Informatika, Institut Teknologi Nasional Malang dan Dosen Pembimbing II yang selalu memberikan bimbingan dan masukan.
5. Ir. Yusuf Nakhoda, MT, selaku Dosen Pembimbing I, yang selalu memberikan bimbingan dan masukan.
6. Semua dosen Program Studi Teknik Informatika yang telah membantu dalam penulisan dan masukan.
7. Semua teman-teman berbagai angkatan yang telah memberikan doa dan dukungannya dalam menyelesaikan skripsi ini.

Dengan segala kerendahan hati, penulis menyadari masih banyak terdapat kekurangan-kekurangan, sehingga penulis mengharapkan adanya saran dan kritik yang bersifat membangun demi kesempurnaan skripsi ini.

Malang 17 Maret 2015

Penulis

DAFTAR ISI

ABSTRAKSI.....	iv
KATA PENGANTAR.....	v
DAFTAR ISI.....	vii
DAFTAR GAMBAR	ix
DAFTAR TABEL	x
DAFTAR LAMPIRAN	xi
BAB I PENDAHULUAN	1
1.1 Latar Belakang Masalah	1
1.2 Rumusan Masalah.....	2
1.3 Tujuan	2
1.4 Batasan Masalah	2
1.5 Metodologi Penelitian.....	2
1 Studi Literatur	2
2 Analisa Kebutuhan.....	3
3 Perancangan Aplikasi	3
4 Pengujian Program.....	3
1.6 Sistematika Penulisan	3
BAB II LANDASAN TEORI	5
2.1 ActionScript.....	5
2.2 Flash.....	5
2.3 CorelDRAW	6
2.4 <i>Game</i>	7
2.5 <i>Adventure Game</i>	8
2.6 <i>Artificial Intelligence</i>	8
2.7 <i>Finite State Machine</i>	9
2.8 Animasi	10

BAB III ANALISA PERANCANGAN	12
3.1 Analisis Sistem	12
3.1.1 Analisis Masalah	12
3.1.2 Pengenalan <i>Game Gunslinger</i>	13
3.1.3 <i>Storyline</i>	13
3.2 Perancangan Sistem	13
3.2.1 Perancangan Karakter	14
3.2.2 Perancangan <i>Score</i>	15
3.2.3 Perancangan Struktur Menu	15
3.2.4 Perancangan Alur <i>Game</i>	16
3.2.5 Alur <i>Finite State Machine</i>	17
3.2.6 Perancangan Antarmuka	19
BAB IV IMPLEMENTASI DAN PENGUJIAN	23
4.1 Implementasi Hasil	23
4.1.1 Tampilan Menu Utama	23
4.1.2 Tampilan Menu Pengaturan	24
4.1.3 Tampilan Mulai Petunjuk	24
4.1.4 Tampilan Menu Tentang	25
4.1.5 Tampilan Menu Keluar	25
4.1.6 Tampilan <i>Stage</i>	26
4.1.7 Tampilan Halaman Menang	29
4.1.8 Tampilan Halaman Kalah	30
4.2 Pengujian Sistem	30
4.2.1 Pengujian Metode	30
4.2.2 Pengujian Fungsional	31
4.2.3 Pengujian <i>Performance</i>	32
4.2.4 Pengujian <i>Control Player</i>	33
4.2.5 Pengujian Pada Pengguna	34
BAB V PENUTUP	36
5.1 Kesimpulan	36
5.2 Saran	37

DAFTAR PUSTAKA	38
LAMPIRAN	39

DAFTAR GAMBAR

Gambar 3.1	Perancangan Struktur Menu	15
Gambar 3.2	Alur <i>Game Gunslinger</i>	16
Gambar 3.3	Alur Aksi Musuh	17
Gambar 3.4	Alur aksi <i>boss</i>	18
Gambar 3.5	Tampilan menu utama	19
Gambar 3.6	Tampilan <i>stage 1</i>	20
Gambar 3.7	Tampilan <i>stage 2</i>	20
Gambar 3.8	Tampilan <i>stage 3</i>	21
Gambar 3.9	Tampilan <i>stage 4</i>	21
Gambar 3.10	Tampilan <i>stage 5</i>	22
Gambar 3.11	Tampilan <i>Game Over</i>	22
Gambar 4.1	Tampilan menu utama <i>game Gunslinger</i>	23
Gambar 4.2	Tampilan menu pengaturan	24
Gambar 4.3	Tampilan Menu Petunjuk	24
Gambar 4.4	Tampilan Menu Tentang	25
Gambar 4.5	Tampilan Menu Keluar	25
Gambar 4.6	Tampilan <i>Stage 1</i>	26
Gambar 4.7	Tampilan <i>Stage 2</i>	27
Gambar 4.8	Tampilan <i>Stage 3</i>	27
Gambar 4.9	Tampilan <i>Stage 4</i>	28
Gambar 4.10	Tampilan <i>Stage 5</i>	29
Gambar 4.11	Tampilan Halaman Menang	29
Gambar 4.12	Tampilan Halaman Kalah	30

DAFTAR TABEL

Tabel 3.1	Pengenalan Karakter.....	14
Tabel 4.1	Pengujian Metode.....	31
Tabel 4.2	Pengujian Fungsional	32
Tabel 4.3	Pengujian <i>Performance</i>	33
Tabel 4.4	Pengujian <i>Control Player</i>	33
Tabel 4.5	Pengujian pengguna	34

DAFTAR LAMPIRAN

Lampiran 1. <i>Script Pemain</i>	40
Lampiran 2. <i>Script Minion</i>	42
Lampiran 3. <i>Script Elite Minion</i>	43
Lampiran 4. <i>Script Boss</i>	44
Lampiran 5. <i>Script V-Cam</i>	45

BAB I

PENDAHULUAN

1.1. Latar Belakang Masalah

Game merupakan sebuah media hiburan untuk berbagai kalangan. Seiring perkembangan jaman, musuh-musuh dalam *game* sudah semakin cerdas. Hal ini dimaksudkan dengan tujuan membuat *game* menjadi lebih menarik dan memberikan tantangan kepada pemain. *Game Adventure* dengan grafis 2D merupakan salah satu genre *game* yang menggunakan metode *Finite State Machine* sebagai musuh pemain.

Game Adventure adalah genre *game* yang lebih menonjolkan jalan cerita dibandingkan aksi dalam *game*. *Adventure Game* adalah salah satu genre *game* yang bisa dibilang paling tua. Beberapa genre *game* seperti *First Person Shooter (FPS)*, *Role-Playing Game (RPG)* dan *Real Time Strategy (RTS)* merupakan hasil pengembangan dari *Game Adventure* ini.

Dalam setiap *game* tentunya tidak akan terasa menarik jika tidak ada lawan, baik pemain lain ataupun musuh-musuh yang sudah diprogram untuk melawan pemain yang biasanya disebut *Artificial Intelligence (AI)*. Dalam *Game Adventure*, *AI* biasanya digunakan sebagai musuh pemain dan teman yang mengarahkan pemain untuk menyelesaikan alur cerita dalam *game*. Metode *AI* yang biasanya digunakan dalam sebuah *game adventure* adalah metode *Finite State Machine*.

Finite State Machine merupakan sebuah metodologi untuk memberikan perilaku buatan kepada *AI* agar dapat merespon kehadiran pemain dengan syarat tertentu. Konsep dari *Finite State Machine* adalah musuh atau *AI* bisa berubah transisi mulai dari diam, mengejar pemain, hingga menyerang pemain jika pemain melalui kondisi tertentu seperti mendekati musuh.

Dari uraian diatas membuat penulis tertarik untuk membuat sebuah *game* dengan *AI* berjenis *Finite State Machine* untuk mengetahui bagaimana *AI* ini bekerja pada sistem *game*.

1.2. Rumusan Masalah

Berdasar latar belakang yang telah dikemukakan sebelumnya, maka penulis akan merumuskan masalah yang akan dibahas sebagai berikut :

- 1) Bagaimana merancang *Game Adventure* dengan metode *Finite State Machine*.
- 2) Bagaimana membangun *game* dengan menggunakan aplikasi *Adobe Flash CS6*.

1.3. Tujuan

Tujuan dari pembuatan *game* yaitu untuk mengimplimentasikan metode *Finite State Machine* pada *game Adventure* agar membuat *game* menjadi lebih menantang dan lebih menarik bagi para pemain.

1.4. Batasan Masalah

Dalam penyusunan skripsi agar menjadi sistematis dan mudah dimengerti, maka akan diterapkan beberapa batasan masalah. Batasan-batasan masalah itu antara lain :

1. *Game* ini dibangun menggunakan *Adobe Flash CS6*.
2. *Game* ini dibuat untuk berjalan pada sistem operasi Windows.
3. Grafik dalam *game* ini adalah 2D.
4. Tampilan dalam *game* ini adalah *sidescroll* atau *sideview*.

1.5 Metode Penelitian

Metode penelitian yang digunakan dalam menyusun penelitian skripsi ini adalah sebagai berikut:

1. Studi literatur

Pengumpulan dilakukan dengan mencari bahan-bahan referensi dari berbagai sumber sebagai landasan teori yang berhubungan dengan permasalahan yang dijadikan objek penelitian.

2. Analisa kebutuhan

Data dan informasi yang telah diperoleh akan dianalisa agar didapatkan suatu kerangka yang digunakan untuk acuan perancangan perangkat lunak

3. Perancangan aplikasi

Data-data yang telah terkumpul diimplementasikan kedalam program bersama dengan pembuatan algoritma FSM.

4. Pengujian Program

Jika aplikasi selesai, selanjutnya adalah tahap pengujian dengan tujuan memastikan *Game* sudah berjalan dengan benar.

1.6 Sistematika Penulisan

Sistematika penyusunan proposal ditujukan untuk memberikan gambaran dan uraian dari proposal skripsi secara garis besar yang meliputi bab-bab sebagai berikut:

BAB I : PENDAHULUAN

Menguraikan Latar belakang, Rumusan Masalah, Batasan Masalah, Maksud dan Tujuan Penelitian, Metodologi Penelitian, Sistematika Penyusunan Laporan Penelitian.

BAB II : LANDASAN TEORI

Menguraikan tentang teori-teori yang menunjang judul, dan pembahasan secara detail. Landasan teori dapat berupa definisi-definisi atau model yang langsung berkaitan dengan ilmu atau masalah yang diteliti. Pada bab ini juga dituliskan tentang *software* yang digunakan dalam pembuatan program atau keperluan saat penelitian.

BAB III : ANALISIS DAN PERANCANGAN PROGRAM

Bab ini berisi uraian mengenai rancangan aplikasi yang akan dibuat relevansi dari permasalahan yang dikaji. Selain itu pada bab ini juga membahas analisis masalah yang akan menguraikan tentang analisis terhadap permasalahan pada kasus yang sedang diteliti.

BAB IV : IMPLEMENTASI DAN PENGUJIAN

Berisi pembahasan mengenai pembuatan aplikasi penyewaan kamar menggunakan Microsoft Visual Studio 2008 dan penyimpanan data yang menggunakan Mysql, serta me-maparkan hasil-hasil dari tahapan pembuatan aplikasi, dari tahap analisis, desain, implementasi desain, hasil testing dan implementasinya, berupa penjelasanteoritik, baik secara kualitatif, kuantitatif, atau secara statistik. Dan Selain membandingkan dengan hasil penelitian yang masih manual.

BAB V : PENUTUP

Menguraikan kesimpulan dan saran-saran yang diperoleh dari hasil analisa, agar nantinya dapat digunakan sebagai bahan penelitian berikutnya.

BAB II

LANDASAN TEORI

2.1 ActionScript

ActionScript 3.0 diperkenalkan pada tahun 2006 dan telah menjadi bahasa pemrograman utama untuk Flash sejak saat itu. Versi asli dari ActionScript diperkenalkan pada tahun 1996 dengan rilis dari Flash 4. Sebelumnya tidak disebut dengan ActionScript, dan kita bahkan tidak bisa mengetikkan jenis kode kita. Sebaliknya, kita memilih pernyataan dari serangkaian menu *drop-down*. [7]

Flash 5 pada tahun 2000 meningkat secara baik pada pengenalan formal dengan ActionScript1,0. Ini bahasa *scripting* berisi semua fitur-fitur yang berlebihan yang merupakan pengembangan berbasis bahasa web, seperti Lingo Macromedia Director dan Sun Java. Tapi, hal tersebut hadir secara mendadak dengan kecepatan dan kekuatan. [7]

Flash *MX 2004*, juga dikenal sebagai Flash 7, membawa kita menuju ActionScript 2.0, yang jauh lebih kuat versi bahasa pemrogramannya yang membuat lebih mudah untuk membuat program berorientasi objek. Ini jauh lebih dekat dengan ECMA Script, sebuah standar untuk bahasa pemrograman yang dikembangkan oleh Asosiasi Produsen Komputer Eropa (*Europe Computer Programming Association*). JavaScript, bahasa pemrograman yang digunakan di browser, juga didasarkan pada ECMA Script. [7]

ActionScript 3.0 adalah puncak dari tahun pembangunan. Versi berikutnya memperhitungkan bagaimana pengembang yang menggunakan Flash dan apa kelemahan dari versi ActionScript saat ini.

2.2 Flash

Flash merupakan *software* yang memiliki kemampuan menggambar sekaligus menganimasikannya, serta mudah. Flash tidak hanya digunakan dalam pembuatan animasi, tetapi pada zaman sekarang ini Flash juga banyak digunakan untuk keperluan lainnya seperti dalam pembuatan *game*, presentasi, membangun web, animasi pembelajaran, bahkan juga dalam pembuatan film. Animasi yang dihasilkan flash adalah animasi berupa file *movie*. *Movie* yang dihasilkan dapat

berupa grafik atau teks. Grafik yang dimaksud disini adalah grafik yang berbasis vektor, sehingga saat diakses melalui internet, animasi akan ditampilkan lebih cepat dan terlihat halus. Selain itu Flash juga memiliki kemampuan untuk mengimpor file suara, video maupun file gambar dari aplikasi lain. Flash adalah program grafis yang diproduksi oleh Macromedia Corp., yaitu sebuah *vendor software* yang bergerak dibidang animasi web. Macromedia Flash pertama kali diproduksi pada tahun 1996. Macromedia Flash telah diproduksi dalam beberapa versi. Versi terakhir dari Macromedia Flash adalah Macromedia Flash 8. Sekarang Flash telah berpindah *vendor* menjadi Adobe. [2]

2.3 CorelDRAW

Coreldraw adalah *software editor* grafis berbasis vektor, dikembangkan dan dipasarkan oleh Corel Corp. yang berbasis di Ottawa, Kanada. Yang kemudian menjadi nama paket editor grafis Corel. Versi terakhir, X4, diluncurkan pada pertengahan Februari 2008 di Indonesia. [3]

CorelDraw sejak awal dikembangkan untuk Windows dan saat ini dapat berjalan pada Windows 2000 dan versi selanjutnya. Versi untuk Mac OS dan Mac OS X pada awalnya juga tersedia, namun dihentikan karena minimnya penjualan. Versi Mac OS hanya berlanjut sampai versi 5.0. Versi terakhir untuk Linux terakhir dibuat tahun 2000. Corel pada Linux tidak berjalan langsung di atas platform, namun harus menggunakan Wine, semacam *crossover* seperti yang digunakan untuk meng-*install* Photoshop pada Linux. Pada 1985, Dr. Michael Cowpland mendirikan Corel untuk menjual sistem desktop-publishing berbasis Intel. [3]

Pada 1987, Corel merekrut beberapa pengembang *software* (programmer) untuk membangun sebuah *software* grafis berbasis vektor untuk dijadikan satu dengan paket *desktop-publishing* Corel. Program itu, yang akhirnya diberi nama CorelDraw, pertama kali diluncurkan ada 1989. Program itu diterima luas oleh masyarakat dan pada akhirnya Corel hanya fokus pada pengembangan *software*. [3]

CorelDraw dibuat untuk Windows bersamaan dengan diluncurkannya Windows 3.1. dengan dimasukkannya *TrueType* ke dalam Windows 3.1

menjadikan Corel sebagai program ilustrasi yang mampu menggunakan *fonts* yang ada tanpa membutuhkan software tambahan seperti Adobe TypeWriter. [3]

Beberapa inovasi untuk ilustrasi berbasis vektor pada CorelDraw : *Note-edit tool, stroke before fill, mesh fill* dan sebagainya. [3]

CorelDraw memiliki perbedaan mencolok dibandingkan kompetitornya. Yang pertama bahwa CorelDraw adalah suatu paket *software* grafis, bukan hanya sebuah editor gambar berbasis vektor. Peralatan – peralatan yang ada memungkinkan penggunaanya untuk mengatur kontras, keseimbangan warna bahkan mengubah dari mode *RGB (Red Green Blue)* menjadi *CMYK (Cyan Magenta Yellow)*. Khusus untuk gambar *bitmap* dapat diubah dengan Corel PhotoPaint. [3]

2.4 Game

Permainan *video (video game)* adalah permainan yang menggunakan interaksi dengan antarmuka pengguna melalui gambar yang dihasilkan oleh piranti *video*. Permainan *video* umumnya menyediakan sistem penghargaan – misalnya skor – yang dihitung berdasarkan tingkat keberhasilan yang dicapai dalam menyelesaikan tugas-tugas yang ada di dalam permainan.

Kata “*video*” pada “permainan *video*” pada awalnya merujuk pada piranti tampilan raster. Namun dengan semakin dipakainya istilah “*video game*”, kini kata permainan *video* dapat digunakan untuk menyebut permainan pada piranti tampilan apapun. Sistem elektronik yang digunakan untuk menjalankan permainan *video* disebut *platform*, contohnya adalah komputer pribadi dan konsol permainan.

Game bertujuan untuk menghibur, biasanya *game* banyak disukai oleh anak – anak hingga orang dewasa. *Games* sebenarnya penting dalam perkembangan otak, untuk meningkatkan konsentrasi dan melatih untuk memecahkan masalah dengan tepat dan cepat karena dalam *game* terdapat berbagai konflik atau masalah yang menuntut kita untuk menyelesaikannya dengan cepat dan tepat. Tetapi *game* juga bisa merugikan karena apabila kita sudah kecanduan *game* kita akan lupa waktu dan akan mengganggu kegiatan atau aktifitas yang sedang kita lakukan.[6]

2.5 *Adventure Game*

Adventure Game adalah sebuah genre permainan dalam *video game* dimana pemain berperan sebagai protagonis dalam sebuah cerita interaktif diiringi berbagai teka-teki untuk dipecahkan.

Di dunia barat, popularitas genre ini berada dipuncak sekitar tahun 1980 hingga pertengahan tahun 1990-an meski kini petualangan dianggap memiliki pasar yang kecil seiring dengan berkembangnya genre lain yang lebih diminati seperti *Role Playing Games*, *First Person Shooter* atau Simulasi. Akan tetapi di Jepang, genre ini tetap populer dalam bentuk novel visual dimana 70% dari keseluruhan judul untuk jenis ini diluncurkan di negeri sakura tersebut. Masalahnya adalah, kebanyakan *Game* buatan Jepang menggunakan bahasa Jepang yang menuntut pemahaman huruf kanji pemain membuat kebanyakan orang tidak mampu menikmatinya.[11]

2.6 *Artificial Intelligence*

Kecerdasan buatan (*Artificial Intelligence* atau *AI*) didefinisikan sebagai kecerdasan yang ditunjukkan oleh suatu *entitas* buatan. Sistem seperti ini umumnya dianggap komputer. Kecerdasan diciptakan dan dimasukkan ke dalam suatu mesin (komputer) agar dapat melakukan pekerjaan seperti yang dapat dilakukan manusia. Beberapa macam bidang yang menggunakan kecerdasan buatan antara lain sistem pakar, permainan komputer (*games*), *logika fuzzy*, jaringan syaraf tiruan dan robotika.

Pada tahun 1769, dataran Eropa dikejutkan dengan suatu permainan catur yang dapat menjawab langkah-langkah permainan catur yang belum ditentukan terlebih dahulu. Mesin ini disebut dengan Maelzel Chess Automation dan dibuat oleh Wolfgang Von Kempelen (1734-1804) dari Hungaria. Akan tetapi mesin ini akhirnya terbakar pada tahun 1854 di Philadelphia Amerika Serikat. Banyak orang tidak percaya akan kemampuan mesin tersebut. Dan seorang penulis dari Amerika Serikat, Edgar Allan Poe (1809-1849) menulis sanggahan terhadap mesin tersebut, dia dan kawan-kawannya ternyata benar, bahwa mesin tersebut adalah tipuan, dan kenyataannya bukanlah *automation*, tetapi merupakan konstruksi

yang sangat baik yang dikontrol oleh seorang pemain catur handal yang bersembunyi di dalamnya.

Usaha untuk membuat konstruksi mesin permainan terus dilanjutkan pada tahun 1914, dan mesin yang pertama kali didemonstrasikan adalah mesin permainan catur. Penemu mesin ini adalah Leonardo Torres Y Quevedo, direktur dari Laboratorio de Automatica di Madrid, Spanyol. Beberapa tahun kemudian, ide permainan catur dikembangkan dan diterapkan di komputer oleh Arthur L. Samuel dari IBM dan dikembangkan lebih lanjut oleh Claude Shannon.

Perkembangan *game* saat ini tidak lepas dari kecerdasan buatan (*artificial intelligence*). Kecerdasan buatan merupakan salah satu bagian dari ilmu komputer yang membuat mesin (komputer) dapat melakukan pekerjaan seperti manusia dan komputer dimungkinkan untuk dapat berfikir.

Berdasarkan perkembangan *game* yang pesat pada masa ini, maka tidak dipungkiri bahwa dibutuhkan sesuatu yang berbeda pada rule permainannya. Hal ini sangat berkaitan dengan kecerdasan buatan (*artificial intelligence*) yang diterapkan pada *game*. Sebelumnya, sebuah sistem *game*, jika sudah dimainkan sampai tuntas oleh seorang, maka ketika *player* yang sama memulai lagi permainan dari awal, maka *rule* permainannya akan sama. Namun, untuk saat ini sesuai dengan perkembangan *game* dan kecerdasan buatan yang diterapkan, sistem dalam *game* sudah dapat belajar mengenali pola permainan dari *player* dan ketika *player* tersebut memulai permainan kembali, maka sistem ini akan menggunakan *rule* yang berbeda untuk pemain yang sama ini, sehingga *game* menjadi lebih menarik dan menantang untuk dimainkan.[2]

2.7 Finite State Machine

FSM adalah sebuah metodologi perancangan sistem kontrol yang menggambarkan tingkah laku atau prinsip kerja sistem dengan menggunakan tiga hal berikut: *State* (Keadaan), *Event* (kejadian) dan *Action* (aksi). Pada satu saat dalam periode waktu yang cukup signifikan, sistem akan berada pada salah satu *state* yang aktif. Sistem dapat beralih atau bertransisi menuju *state* lain jika mendapatkan masukan atau *event* tertentu, baik yang berasal dari perangkat luar atau komponen dalam sistemnya itu sendiri. Transisi keadaan ini umumnya juga

disertai oleh aksi yang dilakukan oleh sistem ketika menanggapi masukan yang terjadi. Aksi yang dilakukan tersebut dapat berupa aksi yang sederhana atau melibatkan rangkaian proses yang relatif kompleks. [4]

Finite State Machine bukanlah metode yang baru. *FSM* sudah lama ada dan konsep dekomposisi biasanya sudah dipahami dan sering digunakan oleh orang-orang yang memiliki pengalaman dalam membuat program komputer atau desain program komputer. Ada beberapa teknik pemodelan abstrak yang bisa digunakan untuk membantu definisi atau pemahaman dan desain dari *FSM*, mayoritas teknik ini berasal dari disiplin ilmu desain atau matematika. [4]

Salah satu macam elemen yang disertakan pada pembangunan *game* adalah *movieclip*. *Movieclip* merupakan simbol sebagai obyek utama animasi wajah. Dalam bahasa *ActionScript* pada dasarnya setiap *movie clip* yang disertakan dari library atau melakukan duplikasi *movieclip* atau membuat *movieclip* baru melalui fungsi *createEmptyMovieClip()*, diharuskan mempunyai *depth* yang unik. Hal ini dikarenakan jika ada *movieclip* dengan *depth* yang sama, maka salah satu *movieclip* tersebut otomatis akan di-*remove* dan digantikan dengan *movieclip* yang baru (diasumsikan kedua *movieclip* tersebut berada dalam *timeline* utama). Dalam pembuatan *game* ini, diperlukan banyak *movieclip* dengan *depth* yang unik. Untuk mendapatkan *depth* unik tersebut, maka digunakan fungsi *getNextHighestDepth()*. Fungsi ini dimaksudkan agar *movieclip* menempati posisi layer masing-masing sesuai dengan nilai *depth* yang diberikan sehingga akan tampil ketika *movieclip* tersebut dibutuhkan. Sebelum *ActionScript* aplikasi dibuat, maka terlebih dahulu diperlukan deklarasi fungsi yang digunakan. [4]

2.8 Animasi

Animasi adalah gambar bergerak berbentuk dari sekumpulan objek (gambar) yang disusun secara beraturan mengikuti alur pergerakan yang telah ditentukan pada setiap pertambahan hitungan waktu yang terjadi. Gambar atau objek yang dimaksud dalam definisi di atas bisa berupa gambar manusia, hewan, maupun tulisan. Pada proses pembuatannya sang pembuat animasi atau yang lebih dikenal dengan animator harus menggunakan logika berfikir untuk menentukan alur gerak suatu objek dari keadaan awal hingga keadaan akhir objek tersebut.

Perencanaan yang matang dalam perumusan alur gerak berdasarkan logika yang tepat akan menghasilkan animasi yang menarik untuk disaksikan.

Dalam bidang grafika pemodelan visual dapat dikategorikan sebagai dua kelompok yaitu pemodelan geometrik dan pemodelan penampilan (appearance). Pemodelan geometrik merupakan representasi dari bentuk objek yang ingin ditampilkan sedangkan pemodelan penampilan membuat representasi sifat visual atau penampilan objek tersebut. Contoh sifat visual diantaranya warna dan tekstur. Berdasarkan definisi animasi di atas bahwa sebuah animasi disusun oleh himpunan gambar yang ditampilkan secara berurut maka animasi dapat dikatakan sebuah fungsi terhadap waktu. Gambar dapat didefinisikan sebagai koleksi deskripsi geometris dan visual ataupun dapat berupa citra. Pada gambar yang merupakan koleksi deskripsi, maka animasi didefinisikan sebagai fungsi yang memetakan waktu kepada perubahan parameter-parameter dari deskripsi. Pada gambar yang merupakan citra, animasi didefinisikan sebagai fungsi yang memetakan waktu kepada tiap elemen citra.[5]

BAB III

ANALISA DAN PERANCANGAN

3.1 Analisis Sistem

Analisis sistem merupakan penguraian dari suatu informasi yang utuh ke dalam bagian komponen-komponennya dengan maksud untuk mengidentifikasi permasalahan, baik analisis kebutuhan non-fungsional maupun fungsional.

3.1.1 Analisis Masalah

Analisis masalah merupakan proses idenifikasi serta evaluasi terhadap *game* sejenis dan *Game* yang akan dibangun penulis.

Dalam *Game Adventure*, pemain diharuskan untuk menyelesaikan semua level yang ada dalam *Game*. Setiap level memiliki tingkat kesulitan tertentu dan memiliki syarat tertentu seperti menyelesaikan misi di setiap level untuk dapat menuju ke level selanjutnya. Di setiap *Game Adventure*, akan terasa kurang menarik jika kita hanya melawan musuh-musuh kecil yang mudah mati atau biasanya disebut *minion*. Oleh karena itu dibuatlah sebuah *AI* yang tidak mudah untuk dikalahkan player dan memiliki status diatas rata-rata bahkan melebihi player yang biasanya disebut *boss*. *Boss* berada di setiap akhir *level* dan untuk mengalahkannya pemain tidak bisa ceroboh dalam menyerang. Untuk mengalahkan semua *boss* di setiap *level*, pemain secara tidak langsung dituntut untuk mengasah kemampuan analisa dan strateginya.

Game dengan genre *Adventure* dengan *sideview* atau *sidecroller* yaitu *Game* yang terlihat dari samping sudah tidak asing lagi dalam dunia *Game*, khususnya Indonesia. Dalam perkembangannya genre *Game* seperti ini telah berkembang pesat karena jenis *Game* seperti ini terbilang mudah dan menyenangkan untuk dimainkan oleh masyarakat terutama anak-anak. Beberapa *Game* yang memakai genre ini antara lain *Super Mario*, *Captain Commando*, *Super Contra*, dan *Metal Slug*.

3.1.2 Pengenalan *Game Gunslinger*

Game yang akan dibangun adalah *Game Gunslinger* dengan genre *Adventure Sidescroller Game*. *Game* ini dibangun dengan mengaplikasikan teknologi sebagai sarana untuk meningkatkan kemampuan koordinasi antara mata dan tangan serta analisa pemain. Fitur-fitur yang digunakan pada *Game* ini adalah:

1. Grafik yang digunakan adalah 2D.
2. *Game* ini bergenre *Adventure*.
3. Metode yang digunakan adalah *Finite State Machine*.
4. *FSM* terletak pada perilaku musuh ketika sensor mengetahui keberadaan pemain, mendekati pemain, dan menyerang pemain.

3.1.3 *Storyline*

Dalam *Game Gunslinger* ini terdapat 5 *stage* yang harus dihadapi oleh pemain. Pemain diharuskan untuk membunuh *boss* yang ada di setiap *stage* untuk melanjutkan ke *stage* berikutnya. Pada *stage* 1, pemain diharuskan melawan sebuah *boss* dan beberapa anak buahnya. Setelah mengalahkan *boss* pada *stage* 1 maka pemain akan dialihkan menuju *stage* 2 dimana *monster* semakin banyak dan *boss* yang semakin kuat. Pada *stage* 3, pemain akan menghadapi *boss* yang memiliki serangan yang besar dan memiliki anak buah yang lebih banyak dari sebelumnya. Setelah menyelesaikan *stage* 3, maka pemain akan menghadapi *boss* dan anak buah yang lebih banyak. Di *stage* terakhir, pemain akan melawan *boss* terkuat dari semua *stage*. Perbedaan dari setiap *stage* adalah background, jumlah musuh, dan *boss* yang memiliki keunikan tersendiri.





3.2. Perancangan Sistem

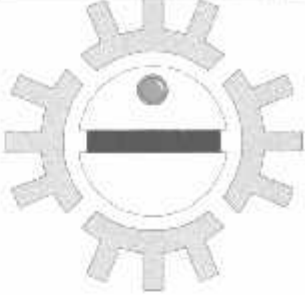

Perancangan sistem adalah suatu bagian dari metodologi pengembangan suatu perangkat lunak yang dilakukan untuk memberikan gambaran secara terperinci tentang *Game Gunslinger*.

3.2.1 Perancangan karakter

Perancangan karakter merupakan pembahasan mengenai karakter yang terlibat dalam *Game Gunslinger*. Karakter pada *Game Gunslinger* dapat dilihat pada Tabel 3.1.

Tabel 3.1. Perancangan Karakter

No	Gambar	Keterangan
1		Pemain
2		<i>Boss stage 1</i>
3		<i>Boss stage 2</i>
4		<i>Boss stage 3</i>

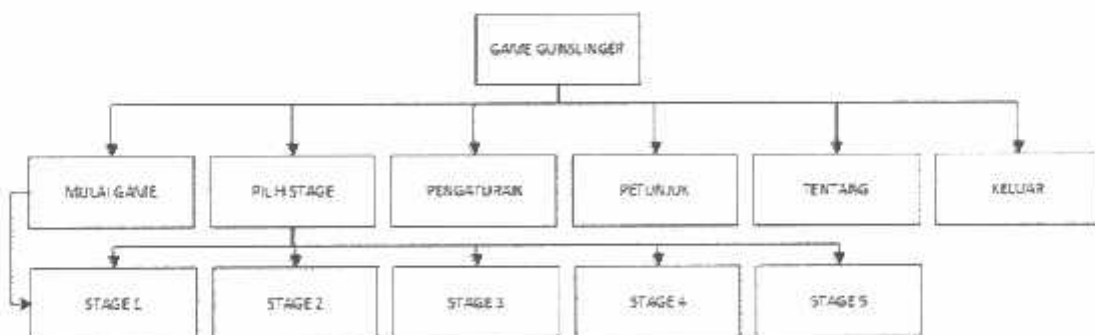
5		<i>Boss stage 4</i>
6		<i>Boss stage 5</i>

3.2.2 Perancangan Score

Pada *Game Gunslinger*, pemain juga dituntut untuk meraih *score* setinggi-tingginya dalam *Game*. *Score* bisa didapatkan dengan mengumpulkan koin yang terdapat pada setiap *stage* dan membunuh musuh. Untuk musuh memiliki nilai poin yang berbeda tergantung tingkat kesulitannya dan *boss* memiliki nilai *score* tertinggi. Pemain bisa melihat *score* yang sudah didapat pada kolom *score*.

3.2.3 Perancangan Struktur Menu

Perancangan Struktur menu adalah perancangan tata urutan menu dari *Game Gunslinger* seperti yang terlihat pada gambar 3.1.

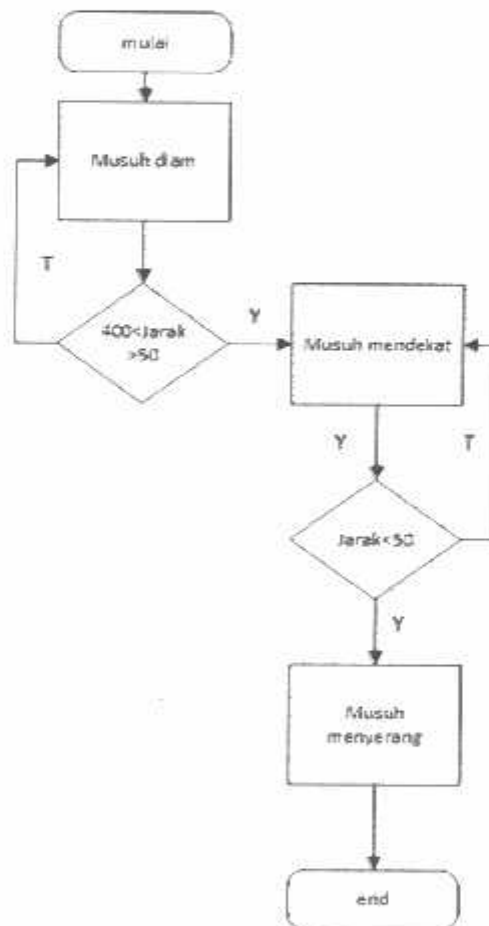


Gambar 3.1. Perancangan Struktur Menu

Pada gambar 3.3, program dimulai dari *start* kemudian dilanjutkan pada menu utama dimana terdapat beberapa tampilan menu yaitu *start Game*, pengaturan, dan keluar. Jika memilih pilihan menu Mulai maka pemain akan langsung diarahkan menuju *stage 1*, setelah berhasil menyelesaikan *stage 1* maka akan dilanjutkan ke *stage* berikutnya hingga pemain bisa menyelesaikan *stage 5*. Setelah pemain bisa menyelesaikan *stage 5* maka pemain akan diarahkan menuju tampilan *Game Over*. Jika pemain kalah, maka akan diarahkan menuju tampilan *Game Over* dan tampilan untuk mengulang *game*.

3.2.5 Alur *Finite State Machine*

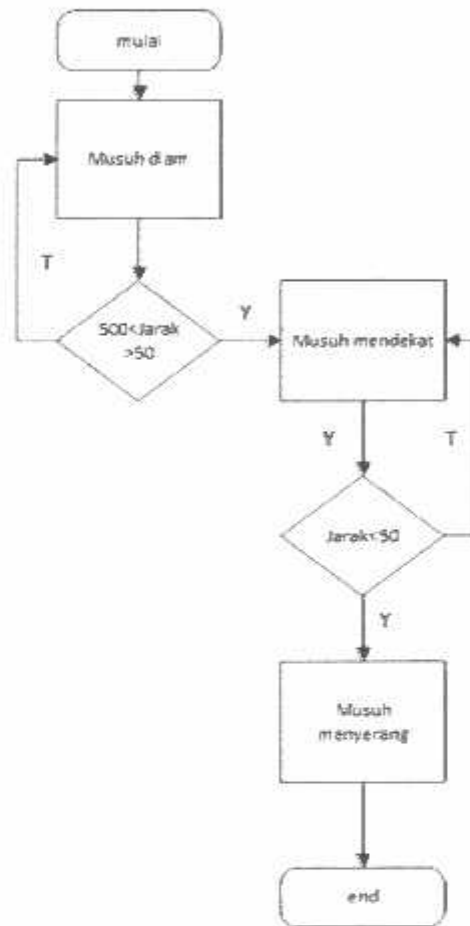
Alur metode *Finite State Machine* secara umum yang terdapat pada setiap musuh yang berada di setiap *stage Game Gunslinger* seperti yang ditunjukkan pada Gambar 3.3.



Gambar 3.3. Alur Aksi Musuh

Aksi awal musuh adalah diam, jika pemain berada pada jarak kurang dari 400 *pixel* dari musuh maka musuh akan mulai mendekati pemain, jika musuh sudah semakin mendekat hingga jaraknya kurang dari 50 *pixel* maka musuh akan mulai menyerang pemain.

Alur *Finite State Machine* pada *boss* setiap *stage* akan dijelaskan melalui *flowchart* dalam Gambar 3.4.



Gambar 3.4. Alur aksi *boss*

Pada Gambar 3.5, aksi *boss* dimulai dari *start* kemudian dilanjutkan dengan aksi awal yaitu diam. Jika pemain berada di jarak lebih dari 500 *pixel* dari *boss*, maka *boss* akan melakukan aksi diam. Jika pemain berada di jarak kurang dari 500 *pixel* dan lebih dari 50 *pixel*, maka *boss* akan melakukan aksi mendekati atau mengejar pemain hingga jarak *boss* dan pemain kurang dari 50 *pixel* maka *boss* akan mulai menyerang pemain.

3.2.6. Perancangan Antarmuka

Perancangan antarmuka bertujuan untuk memberikan gambaran tentang *Game Gunslinger*.

1. Perancangan Menu Utama

Perancangan tampilan menu utama dari *Game Gunslinger* dimana terdapat 3 tombol antara lain Mulai, Pengaturan, Petunjuk, Tentang, dan Keluar. Jika pemain memilih tombol Mulai maka pemain akan diarahkan menuju permainan *stage 1*. Jika pemain memilih tombol pengaturan, maka pemain akan diarahkan menuju tampilan menu pengaturan. Jika Pemain memilih tombol Tentang maka pemain akan diarahkan menuju tampilan Profil. Jika pemain memilih keluar maka pemain akan keluar dari *game*. Rancangan menu utama dari *Game Gunslinger* ditunjukkan pada Gambar 3.5.



Gambar 3.5. Tampilan menu utama

2. Perancangan antarmuka *stage 1*

Perancangan antarmuka *stage 1* merupakan perancangan tampilan *stage 1* dari *Game Gunslinger*. Pada *stage 1* terdapat *health bar*, *score bar*, *boss health bar* dan tombol *pause*. *Health bar* dan *boss health bar* bertujuan mengetahui darah dari pemain dan boss, jika darah pemain mencapai 0 maka pemain akan diarahkan menuju tampilan *Game over* dan jika darah *boss* mencapai 0 maka pemain akan diarahkan menuju *stage 2*. *Score bar* berfungsi untuk menampilkan skor yang sudah didapat pemain. Tombol *pause* berfungsi untuk menghentikan

permainan sementara. Pada setiap *stage*, pemain bisa mengendalikan karakter dengan menggunakan tombol tertentu seperti perintah berjalan, menyerang, dan melompat. Pada *stage 1* terdapat 1 *minion* dan *boss*. Tampilan rancangan *stage 1* terdapat pada Gambar 3.6.



Gambar 3.6. Tampilan *stage 1*

3. Perancangan antarmuka *stage 2*

Perancangan antarmuka *stage 2* merupakan perancangan tampilan *stage 2* dari *Game Gunslinger*. Pada *stage 2* terdapat *health bar*, *score bar*, *boss health bar* dan tombol *pause*. Pada *stage 2* terdapat 2 *minion* dan *boss*. Tampilan rancangan *stage 2* terdapat pada Gambar 3.7.



Gambar 3.7. Tampilan *stage 2*

4. Perancangan antarmuka *stage 3*

Perancangan antarmuka *stage 3* merupakan perancangan tampilan *stage 3* dari *Game Gunslinger*. Pada *stage 3* terdapat *health bar*, *score bar*, *boss health*

bar dan tombol *pause*. Pada *stage 3* terdapat 3 *minion* dan *boss*. Tampilan rancangan *stage 3* terdapat pada Gambar 3.8.



Gambar 3.8. Tampilan *stage 3*

5. Perancangan antarmuka *stage 4*

Perancangan antarmuka *stage 4* merupakan perancangan tampilan *stage 4* dari *Game Gunslinger*. Pada *stage 4* terdapat *health bar*, *score bar*, *boss health bar* dan tombol *pause*. Pada *stage 4* terdapat 3 *minion*, *elite minion* dan *boss*. Tampilan rancangan *stage 4* terdapat pada Gambar 3.9.



Gambar 3.9. Tampilan *stage 4*

6. Perancangan antarmuka *stage 5*

Perancangan antarmuka *stage 5* merupakan perancangan tampilan *stage 5* dari *Game Gunslinger*. Pada *stage 5* terdapat *health bar*, *score bar*, *boss health bar* dan tombol *pause*. Pada *stage 5* terdapat 3 *minion*, 3 *elite minion* dan *boss*. Tampilan rancangan *stage 5* terdapat pada Gambar 3.10.



Gambar 3.10. Tampilan *stage 5*

7. Perancangan antarmuka *Game Over*

Perancangan antarmuka *Game Over* adalah perancangan tampilan ketika pemain telah menyelesaikan semua *stage* pada *Game Gunslinger* atau pemain kalah dalam *Game*. Dalam tampilan rancangan antarmuka *Game Over*, terdapat tampilan skor, tombol ulangi permainan dan tombol kembali ke menu utama. Tampilan skor berfungsi untuk menampilkan total skor yang sudah didapatkan oleh pemain. Tombol ulangi permainan berfungsi untuk mengarahkan pemain kembali ke dalam permainan. Tombol kembali ke menu utama berfungsi untuk mengarahkan pemain kembali ke tampilan menu utama. Tampilan rancangan *Game Over* terdapat pada Gambar 3.11.



Gambar 3.11. Tampilan *Game Over*

BAB IV

IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi Hasil

Implementasi merupakan sebuah tahapan akhir dimana tahapan tersebut akan diperlihatkan berupa tampilan yang telah dibangun berdasarkan *software* Adobe Flash dan CorelDRAW sebagai *software* untuk desain dari isi *game* ini. Adapun tahap implementasi tersebut memuat tampilan-tampilan sebagai berikut :

4.1.1 Tampilan Menu Utama

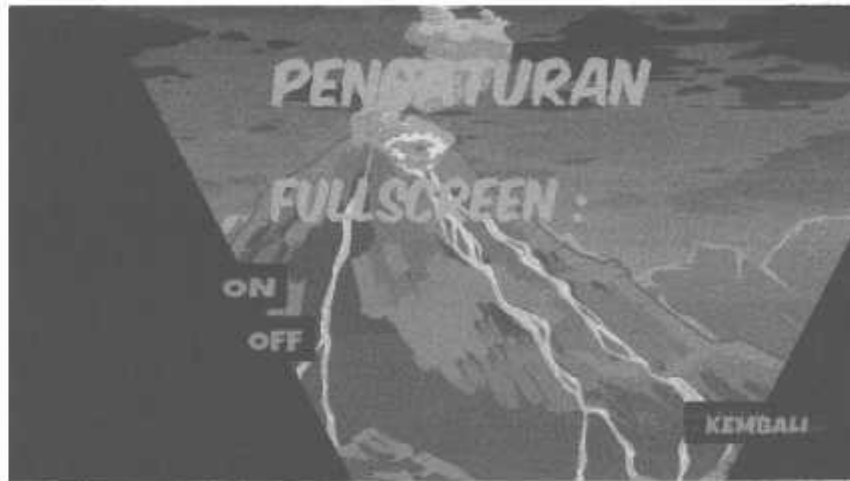
Tampilan menu utama adalah tampilan awal yang muncul pada saat membuka *Game Gunslinger*. Pada tampilan awal ini berisikan tombol menu utama untuk langsung bermain, tombol pilih *stage* untuk memilih *stage*, tombol pengaturan untuk mengatur pengaturan dalam *Game Gunslinger*, tombol petunjuk untuk melihat petunjuk bermain *Game Gunslinger*, tombol tentang untuk melihat profil penulis, dan tombol keluar untuk keluar dari *game*. Adapun desain menu utama seperti pada gambar 4.1.



Gambar 4.1. Tampilan menu utama *Game Gunslinger*

4.1.2 Tampilan Menu Pengaturan

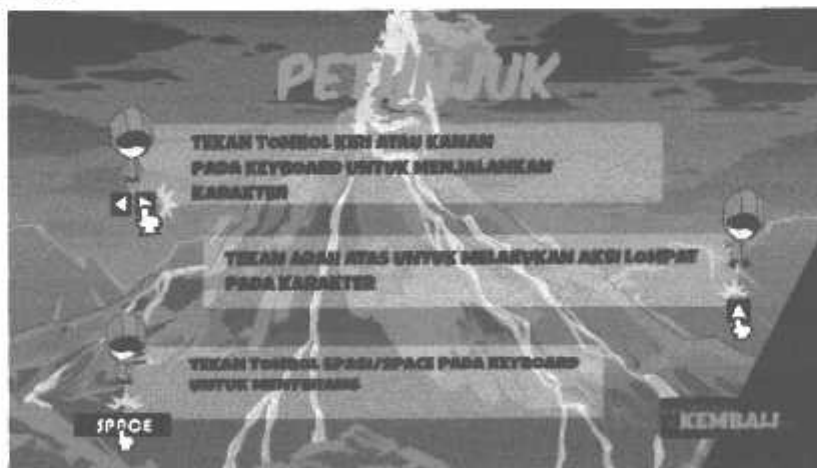
Tampilan menu pengaturan berfungsi untuk mengatur tampilan dari *Game Gunslinger*. Pada menu pengaturan terdapat 2 tombol pengaturan untuk mengatur tampilan dari *Game* antara lain tombol *on* untuk membuat fungsi tampilan *Game* menjadi *fullscreen* dan tombol *off* untuk membatalkan fungsi tampilan *fullscreen* pada *Game*. Adapun tampilan dari menu pengaturan seperti pada Gambar 4.2.



Gambar 4.2 Tampilan menu pengaturan

4.1.3 Tampilan Menu Petunjuk

Tampilan menu petunjuk berfungsi untuk memberikan penjelasan tentang cara bermain dan fungsi kontrol dalam *Game Gunslinger* dan terdapat tombol kembali untuk kembali ke menu utama. Adapun tampilan menu petunjuk seperti pada Gambar 4.3.



Gambar 4.3 Tampilan Menu Petunjuk

4.1.4 Tampilan Menu Tentang

Pada menu tentang merupakan menu yang berfungsi menampilkan profil dari penulis dan terdapat tombol kembali untuk kembali ke menu utama. Adapun tampilan menu tentang seperti pada Gambar 4.4.



Gambar 4.4 Tampilan Menu Tentang.

4.1.5 Tampilan Menu Keluar

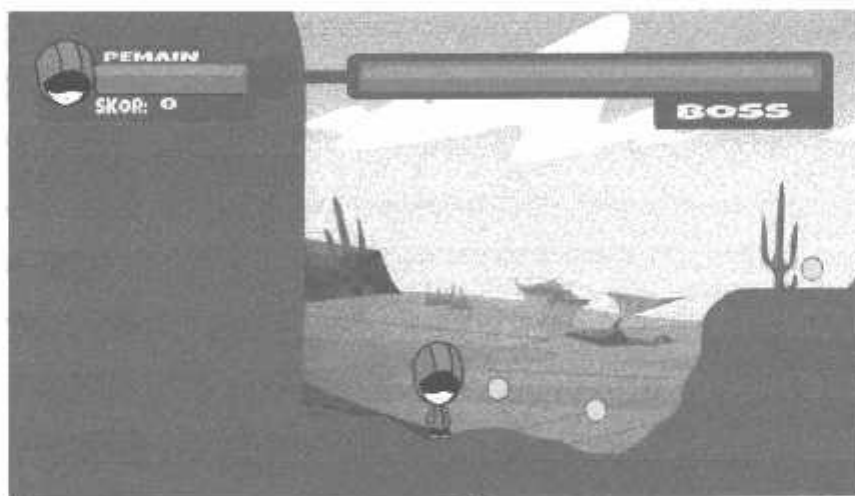
Menu keluar berfungsi untuk keluar dari *Game*. Pada menu keluar terdapat 2 tombol yaitu ya yang berfungsi sebagai konfirmasi untuk keluar dari *Game* dan tombol tidak yang berfungsi untuk kembali ke menu utama. Adapun tampilan menu keluar seperti pada gambar 4.5.



Gambar 4.5 Tampilan Menu Keluar

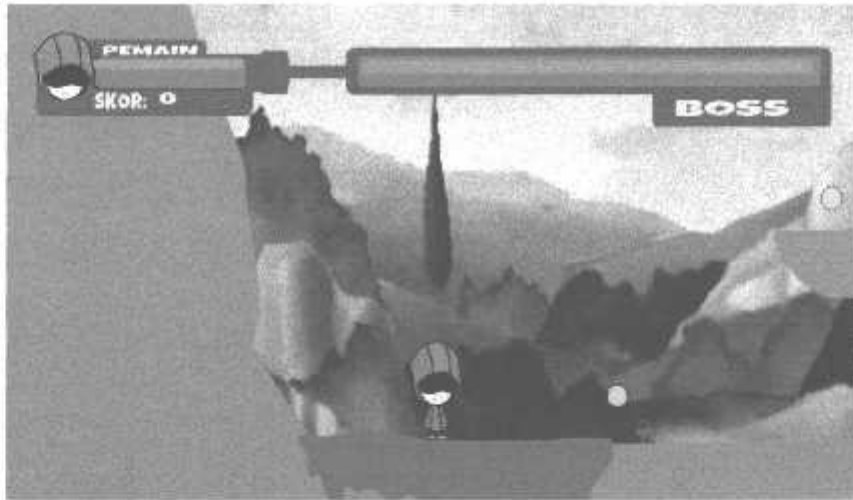
4.1.6 Tampilan *Stage*

Untuk proses pada *game* ini, pemain akan di arahkan menuju *stage* 1 diawal permainan. Setelah menyelesaikan *stage* 1, pemain akan diarahkan menuju ke *stage* 2. Untuk menyelesaikan *stage* 1, pemain diberikan misi untuk mengalahkan *boss*. Namun sebelum melawan boss, akan terdapat *minion* yang menghalangi pemain dan dituntut untuk mengalakan *minion* terlebih dahulu sebelum mengalahkan *boss*. Dalam *stage* 1 juga terdapat *item* yang dapat didapatkan oleh pemain, diantaranya *coin* untuk menambah skor dan *potion* untuk membuat *healthbar* pemain menjadi penuh. Adapun tampilan *stage* 1 terdapat pada Gambar 4.7.



Gambar 4.7 Tampilan *Stage* 1

Setelah pemain berhasil melalui *stage* 1, pemain akan diarahkan menuju *stage* 2. Jumlah musuh pada *stage* 2 akan lebih banyak dari *stage* 1 dengan tujuan mempersulit pemain untuk menuju *stage* berikutnya. Pada *stage* 2, terdapat 2 buah *minion* yang harus dihadapi dan sebuah *boss* untuk dikalahkan agar bisa menuju *stage* selanjutnya. Pada *stage* 2 juga tersedia *item* yang bisa diambil seperti *coin* dan 3 buah *potion*. Adapun tampilan *stage* 2 terdapat pada Gambar 4.8.



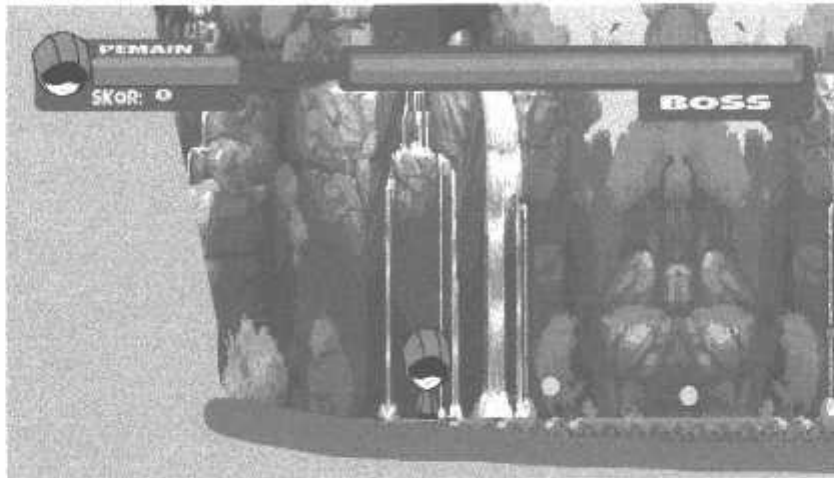
Gambar 4.8. Tampilan *Stage 2*

Setelah pemain berhasil melalui *stage 2*, pemain akan diarahkan menuju *stage 3*. Jumlah musuh pada *stage 3* akan lebih banyak dari *stage 2* untuk lebih mempersulit pemain menyelesaikan *stage*. Pada *stage 3*, terdapat 3 buah *minion* yang harus dihadapi dan sebuah *boss* untuk dikalahkan agar bisa menuju *stage* selanjutnya. Pada *stage 3* juga terdapat perangkap seperti batu jatuh yang akan mengurangi *healthbar* pemain jika menyentuh pemain. Pada *stage* ini juga tersedia *item* yang bisa diambil seperti *coin* dan 3 buah *potion*. Adapun tampilan *stage 3* terdapat pada Gambar 4.9.



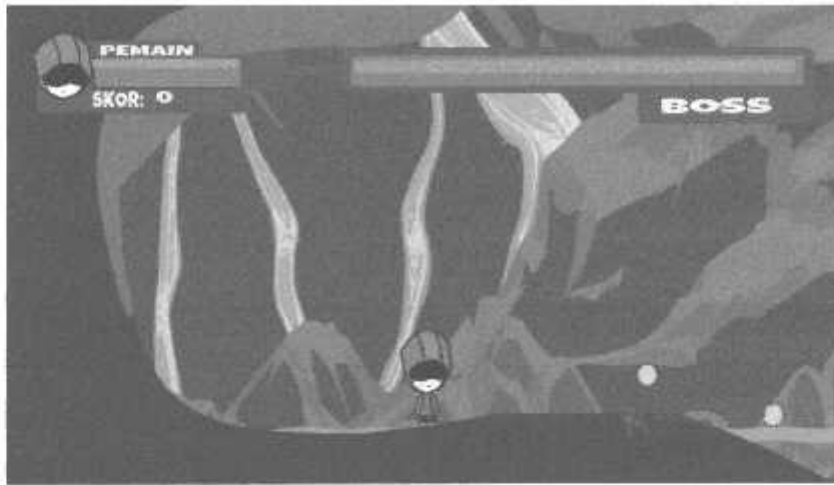
Gambar 4.9. Tampilan *Stage 3*

Setelah pemain berhasil melalui *stage* 3, pemain akan diarahkan menuju *stage* 4. Jumlah musuh pada *stage* 4 akan lebih banyak dari *stage* 3 dan terdapat *minion* jenis baru yang lebih agresif yang dinamakan *elite minion*. Pada *stage* 4, terdapat 3 buah *minion* dan sebuah *elite minion* yang harus dihadapi dan sebuah *boss* yang lebih kuat untuk dikalahkan. Pada *stage* 4 juga tersedia *item* yang bisa diambil seperti *coin* dan 3 buah *potion*. Adapun tampilan *stage* 2 terdapat pada Gambar 4.10.



Gambar 4.10. Tampilan *Stage* 4

Setelah pemain berhasil melalui *stage* 4, pemain akan diarahkan menuju *stage* terakhir yaitu *stage* 5. Jumlah musuh pada *stage* 5 adalah yang paling banyak dari *stage-stage* sebelumnya. Pada *stage* 5, terdapat 3 buah *minion*, 3 buah *elite minion* yang harus dihadapi dan sebuah *boss* yang paling untuk dikalahkan. Pada *stage* 5 juga tersedia *item* yang bisa diambil seperti *coin* dan 3 buah *potion*. Setelah pemain menyelesaikan *stage* 5, pemain bebas menentukan untuk bermain lagi dan mengulang ke *stage* 1 ataupun kembali ke menu utama jika ingin berhenti bermain. Adapun tampilan *stage* 5 terdapat pada Gambar 4.11.



Gambar 4.11. Tampilan *Stage 5*

4.1.7 Tampilan Halaman Menang

Tampilan halaman menang akan muncul jika pemain berhasil mengalahkan *boss* disetiap *stage*. Pada halaman menang, terdapat tampilan akumulasi skor yang sudah didapatkan dalam *stage*, tombol kembali ke menu untuk kembali ke menu utama, dan tombol ke *stage* selanjutnya untuk menuju ke *stage* berikutnya. Adapun desain halaman tersebut seperti pada gambar 4.12.



Gambar 4.12 Tampilan Halaman Menang.

4.1.8 Tampilan Halaman Kalah

Tampilan halaman kalah akan muncul jika pemain kalah dalam *stage*. Adapun desain halaman tersebut seperti pada gambar 4.13.



Gambar 4.13 Tampilan Halaman Kalah

4.2 Pengujian Sistem

Pengujian dilakukan aplikasi untuk memastikan bahwa aplikasi berjalan dengan benar sesuai yang diharapkan.

4.2.1 Pengujian Metode

Pengujian metode adalah pengujian metode *finite state machine* dalam *AI* pada *Game Gunslinger*. Pada *Game Gunslinger*, *AI* akan mengejar pemain pada jarak 400 *pixel* dan menyerang pada jarak 50 *pixel*. Tabel 4.1 menunjukkan pengujian metode dilakukan dengan menggunakan rumus sebagai berikut:

$$\sqrt{((sx-tx)*(sx-tx) + (sy-ty)*(sy-ty))} < 400$$

Keterangan:




sx= posisi pemain pada sumbu x

sy= posisi pemain pada sumbu y

tx= posisi musuh pada sumbu x

ty= posisi musuh pada sumbu y

Tabel 4.1. Pengujian Metode

No.	Fungsi	Pengujian	Hasil
1	Musuh tidak mengejar jika hasil kalkulasi lebih dari 400 <i>pixel</i> 	$\sqrt{(300-850)^2 + (600-450)^2} < 400$ $= 570.08 < 400 = \text{musuh diam}$	Sesuai
2	Musuh tidak mengejar jika hasil kalkulasi kurang dari 400 <i>pixel</i> 	$\sqrt{(550-850)^2 + (450-450)^2} < 400$ $300 < 400 = \text{musuh mengejar}$	Sesuai
3	<i>Health bar</i> pemain berkurang jika musuh menyerang pada jarak kurang dari 50 <i>pixel</i> 		Sesuai

Dari Tabel 4.1, disimpulkan bahwa semua fungsi berjalan sesuai dengan tujuan yang diharapkan.

4.2.2 Pengujian Fungsional

Pengujian fungsional adalah pengujian mengenai proses fungsional yang terjadi dalam *game*. Hasil dari pengujian dapat dilihat pada Tabel 4.2.

Tabel 4.1. Pengujian Fungsional

No	Fungsi	Hasil
1	Tombol pada menu utama berjalan sesuai dengan fungsi.	Sesuai
2	Musuh dapat mengejar dan menyerang karakter pada jarak serang.	Sesuai
3	Hasil akhir skor yang didapat bisa ditampilkan saat pemain menang atau kalah.	Sesuai
4	Indikator <i>health bar</i> pada karakter dan musuh berjalan sesuai dengan fungsi.	Sesuai

Dari Tabel 4.2, disimpulkan bahwa semua fungsi berjalan sesuai dengan tujuan yang diharapkan.

4.2.3 Pengujian *Performance*

Pengujian *performance* adalah pengujian yang dilakukan pada kinerja atau respon perangkat keras. Pengujian *performance* dimaksudkan untuk mengetahui apakah aplikasi berjalan pada suatu perangkat keras tertentu dengan spesifikasi perangkat keras yang berbeda-beda. Pengujian dilakukan menggunakan 4 komputer dengan spesifikasi berbeda. Hasil dari pengujian *performance* dari *Game Gunslinger* terdapat pada Tabel 4.3.

Tabel 4.3. Pengujian *Performance*

No	<i>Processor</i>	RAM	VGA	OS	Keterangan
1	Intel Celeron	2GB	718MB	Win7	Aplikasi berjalan lancar
2	Intel Core i5	4GB	1765MB	Win8	Aplikasi berjalan

					lancar
3	Intel Core 13	2GB	512MB	Win7	Aplikasi Berjalan Lancar
4	AMD Athon	2GB	512MB	WinXP	Aplikasi Berjalan Lancar
5	Intel Pentium	512MB	256MB	WinXP	Aplikasi berjalan dengan grafik yang sedikit patah-patah (23fps)

Dari Tabel 4.3, dapat disimpulkan bahwa *Game Gunslinger* dapat dijalankan pada komputer dengan RAM 512MB hingga 256MB juga dengan menggunakan Windows XP, 7, dan 8.

4.2.4 Pengujian *Control Player*

Pengujian control player adalah pengujian fungsi dari setiap tombol yang sudah diterapkan untuk menggerakkan karakter utama. Hasil pengujian *control player* dapat dilihat pada Tabel 4.4.

Tabel 4.4. Pengujian *Control Player*

No	Tombol	Fungsi	Hasil
1	←	Bergerak ke kiri	Sesuai
2	→	Bergerak ke kanan	Sesuai
3	↑	Melompat	Sesuai

4	Spasi	Menembak	Sesuai
---	-------	----------	--------

Dari Tabel 4.4 menunjukkan bahwa semua fungsi dari *control player* berjalan sesuai dengan yang diharapkan.

4.2.5 Pengujian Pada Pengguna

Pengujian pada pengguna dilakukan untuk mengetahui kepuasan pengguna dalam memainkan *Game Gunslinger* dari desain, respon karakter utama dan respon musuh. Pengujian pengguna dilakukan kepada 10 orang responden. Hasil dari pengujian pengguna dapat dilihat pada Tabel 4.4.

Tabel 4.4. Pengujian pengguna

No	Pertanyaan	Respon Pengguna			
		Sangat Baik	Baik	Kurang Baik	Buruk
1	Desain Menu Utama	30%	70%	0%	0%
2	Desain <i>Stage</i>	70%	30%	0%	0%
3	Respon Pergerakan Karakter	40%	50%	10%	0%
4	Akumulasi Skor	20%	60%	20%	0%
5	Respon Pergerakan Musuh/AI	40%	40%	20%	0%

Dari Tabel 4.4, dapat disimpulkan bahwa 7 dari 10 orang responden mengatakan desain menu utama sudah cukup menarik dengan persentase 70%, sisanya mengatakan desain sangat menarik dengan persentase 30%. 7 dari 10 orang responden mengatakan desain *stage* sangat menarik dengan persentase 70%, sisanya mengatakan desain *stage* cukup menarik dengan persentase 30%. Untuk pergerakan karakter 40% responden mengatakan sangat baik, 50% mengatakan cukup baik, dan 10% mengatakan kurang baik. Untuk sistem akumulasi skor 20% mengatakan sangat baik, 60% mengatakan cukup baik, dan 20% mengatakan kurang baik. Untuk respon musuh atau AI 40% mengatakan sangat baik, 40% mengatakan cukup baik, dan 20% mengatakan kurang baik.

BAB V

PENUTUP

5.1 Kesimpulan

Setelah penyelesaian *Game Gunslinger* maka penulis dapat menyimpulkan:

1. Implementasi *Finite State Machine* dapat diterapkan pada *game* 2D bergenre *Adventure* dengan indikasi musuh dapat mengejar dan menyerang pemain dengan kondisi tertentu.
2. Semua fungsi dari menu, pergerakan musuh, hingga kontrol karakter utama berjalan sesuai dengan yang diharapkan.
3. Dari hasil pengujian fungsional, *Game Gunslinger* dapat dijalankan pada komputer dengan RAM 512MB hingga 256MB juga dengan menggunakan Windows XP, 7, dan 8.
4. Dari hasil pengujian pengguna dapat disimpulkan bahwa 7 dari 10 orang responden mengatakan desain menu utama sudah cukup menarik dengan persentase 70%, sisanya mengatakan desain sangat menarik dengan persentase 30%. 7 dari 10 orang responden mengatakan desain *stage* sangat menarik dengan persentase 70%, sisanya mengatakan desain *stage* cukup menarik dengan persentase 30%. Untuk pergerakan karakter 40% responden mengatakan sangat baik, 50% mengatakan cukup baik, dan 10% mengatakan kurang baik. Untuk sistem akumulasi skor 20% mengatakan sangat baik, 60% mengatakan cukup baik, dan 20% mengatakan kurang baik. Untuk respon musuh atau AI 40% mengatakan sangat baik, 40% mengatakan cukup baik, dan 20% mengatakan kurang baik.

5.2 Saran

Setelah dilakukan pengujian terhadap *Game Gunslinger* maka masih ada kekurangan sehingga untuk pengembangan lebih lanjut disarankan:

1. Penambahan animasi menyerang pada musuh sehingga membuat *Game* terlihat lebih menarik.
2. Penambahan *stage* pada *Game* karena dalam *Game Gunslinger* ini hanya terdapat 5 *stage*.

3. Pada *Game Gunslinger* dapat ditambahkan alur cerita sehingga *Game* dapat lebih menarik untuk dimainkan.
 4. Karakter utama hanya memiliki 1 aksi menyerang sehingga dalam pengembangannya dapat ditambahkan variasi serangan atau *skill* dengan efek yang unik agar *Game* terlihat lebih menarik.
-

DAFTAR PUSTAKA

- [1] Hidayatullah, Priyanto. 2011. *Membuat Game Edukatif Flash*. Bandung: Penerbit INFORMATIKA.
- [2] Wibawanto, Wandah. 2013. *Memprogram Game Flash Itu Mudah*. Yogyakarta: Andi Publisher.
- [3] Kunto, Benny. 2011. *Jurus Kilat Jago CorelDRAW X5*. Bekasi: Dunia Komputer.
- [4] Wibawanto, Wandah. 2013. *Membuat Game Dengan Macromedia Flash*. Yogyakarta: Andi Publisher.
- [5] Wibawanto, Wandah. 2006. *Dasar-dasar Pemrograman Flash Game*. Yogyakarta: Andi Offset.
- [6] Syarif, Arry Maulana. 2010. *Membuat Games Seru Dengan Flash*. Yogyakarta: Andi Publisher.
- [7] Chandra. 2012. *ActionScript Flash CS5 untuk Orang Awam*. Yogyakarta: Maxikom.
- [8] Andi. 2012. *PAS: Beragam Desain Game Edukasi dengan Adobe Flash CS5*. Bandung: Wahana Komputer.
- [9] Newgrounds Studio, 2013. Platformer Tutorial AS2 <http://www.newgrounds.com/portal/view/455972> Diakses 23 Desember 2014.
- [10] Cheese, Chunky. 2012. Platformer Tutorial <http://www.newgrounds.com/portal/view/465099> Diakses tanggal 19 Desember 2014.
- [11] Bagoel. 2012. Game Adventure (Permainan Petualangan). <http://bayoekmex.blogspot.com/2012/03/game-adventure-permainan-petualangan.html> Diakses tanggal 23 Desember 2014.

LAMPIRAN

Lampiran 1. Script Pemain

```

onClipEvent(load){
    var ground: MovieClip=_root.ground;
    var grav: Number=0;
    var gravity: Number=2;
    var speed: Number=7;
    var maxJump: Number=-30;
    var touchingGround: Boolean=false;
}
onClipEvent(enterFrame){
    _y+=grav;
    grav+=gravity;
    while(ground.hitTest(_x,_y,true)){
        _y-=gravity;
        grav=0;
    }
    if(ground.hitTest(_x,_y+5,true)){
        touchingGround=true;
    }
    else{
        touchingGround=false;
    }
    if (Key.isDown(Key.UP)&& touchingGround){
        grav = maxJump;
    }
    if (Key.isDown(Key.UP)&& !touchingGround){
        this.gotoAndStop(3);
        cameraUpdate();
    }
    else if (Key.isDown(Key.LEFT)&& touchingGround){
        this._x -= speed;
        this._xscale= -120;
        this.gotoAndStop(2);
    }
    else if (Key.isDown(Key.RIGHT)&& touchingGround){
        this._x += speed;
        this._xscale= 120;
        this.gotoAndStop(2);
    }
    else if (Key.isDown(Key.LEFT)&& !touchingGround){
        this._x -= speed;
        this._xscale= -120;
        this.gotoAndStop(2);
    }
    else if (Key.isDown(Key.RIGHT)&& !touchingGround){
        this._x += speed;
        this._xscale= 120;
        this.gotoAndStop(2);
    }
    else if(Key.isDown(Key.SPACE)&&touchingGround){
        charAttack = true;
        this.gotoAndStop(4);
    }
    else if(Key.isDown(Key.SPACE)&&!touchingGround){

```

```
        charAttack = true;
        this.gotoAndStop(4);
    }
    else{
        this.gotoAndStop(1);
    }
}
```

Lampiran 2. Script Minion

```
onClipEvent(load) {
espeed = 4;
grav=0;
}
onClipEvent(enterFrame) {
if(_root.player.peluru.hitTest(this)){
this.enemyhp.nextFrame();
this.gotoAndStop(2);
}
else{
    this.gotoAndStop(1);
}
distance =400;
tx = this._x;
ty = this._y;
sx = _root.player._x;
sy = _root.player._y;
if (Math.sqrt( (sx-tx)*(sx-tx) + (sy-ty)*(sy-ty))<distance) {
if(tx < sx) {
this._x += espeed;
}
if(tx > sx) {
this._x -= espeed;
if(distance>400){
    this.gotoAndStop(1);
}
}
if(_root.player.hitTest(this)){
    _root.vcam.hpchar.nextFrame();
}
}
}
}
```

Lampiran 3: Script Elite Minion

```
onClipEvent(load) {
espeed = 4;
grav=0;
}
onClipEvent(enterFrame) {
if(_root.player.peluru.hitTest(this)){
this.minionhp.nextFrame();
this.gotoAndStop(2);
}
distance =400;
tx = this._x;
ty = this._y;
sx = _root.player._x;
sy = _root.player._y;
if (Math.sqrt( (sx-tx)*(sx-tx) + (sy-ty)*(sy-ty))<distance) {
if(tx < sx) {
this._x += espeed;
}
if(tx > sx) {
this._x -= espeed;
if(distance=500){
this.gotoAndStop(1);
}
if(ty < sy) {
this._y += espeed;
}
if(ty > sy) {
this._y -= espeed;
if(distance=500){
this.gotoAndStop(1);
}
}
if(_root.player.hitTest(this)){
_root.vcam.hpchar.nextFrame();
}
}
}
}
}
```

Lampiran 5: Script Boss

```
onClipEvent(load) {
    speed = 5;
}
onClipEvent(enterFrame) {
    if(_root.player.peluru.hitTest(this)){
        _root.vcam.hpboss1.nextFrame();
    }
    distance =1000;
    tx = this._x;
    ty = this._y;
    sx = _root.player._x;
    sy = _root.player._y;
    if (Math.sqrt( (sx-tx)*(sx-tx) + (sy-ty)*(sy-ty))<distance) {
        if(tx <= sx) {
            this._x += speed;
        }
        if(tx > sx) {
            this._x -= speed;
        }
        if(distance=900){
            this.gotoAndStop(1);
        }
        if(ty <= sy) {
            this._y += speed;
        }
        if(ty > sy) {
            this._y -= speed;
        }
        if(distance=1500){
            this.gotoAndStop(1);
        }
    }
    if(_root.player.hitTest(this)){
        _root.vcam.hpchar.nextFrame();
    }
}
}
```


Lampiran 6: Script V-Cam

```

/**
 * VCam AS2 v1.0
 *
 * VCam based on original code by Sham Bhangal and Dave Dixon
 *
 * Dynamic Registration AS2 work by Darron Schall
(www.darronschall.com)
 * and AS1 work by Robert Penner (www.robertpenner.com)
 *
 * Special Thanks to Josh Steele and Jeff Brenner
 *
 * @author Bryan Heisey
 * @version 1.0
 * @created 1-April-2008
 *
 * Requirements: Flash 8+ & Actionscript 2
 */

import flash.display.BitmapData;

addProperty("_x2",get_x2,set_x2);
addProperty("_y2",get_y2,set_y2);
addProperty("_xscale2",get_xscale2,set_xscale2);
addProperty("_yscale2",get_yscale2,set_yscale2);
addProperty("_rotation2",get_rotation2,set_rotation2);

////////////////////////////////////
////////
//          Get          stage          width          and          height
////////////////////////////////////

var oldScaleMode:String = stage.scaleMode;
stage.scaleMode = "exactFit";

var sW:Number = Stage.width;
var sH:Number = Stage.height;

stage.scaleMode = oldScaleMode;

////////////////////////////////////
////////
//          Get          Vcam          width          and          height
////////////////////////////////////

var bounds_obj:Object = this.getBounds(this);

var camH:Number = Math.abs(bounds_obj.yMax-bounds_obj.yMin);
var camW:Number = Math.abs(bounds_obj.xMax-bounds_obj.xMin);

////////////////////////////////////
////////
//          Creat          Point          for          dynamic          registration          point

```



```

////////////////////////////////////
    _parent.filters = this.filters;
    _parent.transform.colorTransform =
this.transform.colorTransform;
}

this.onUnload = reset;

function reset():Void {

    //////////////////////////////////////
    //////////////////////////////////////
    //          Resets          parent          properties
    //////////////////////////////////////

    _parent._xscale = 100;
    _parent._yscale = 100;
    _parent._x = 0;
    _parent._y = 0;
    _parent._rotation = 0;
    _parent._visible = true;

}

function set_x2(value:Number):Void {
    var a = {x:rp.x, y:rp.y};
    _parent.localToGlobal(a);
    _parent._x += value-a.x;
}

function get_x2():Number {
    var a = {x:rp.x, y:rp.y};
    _parent.localToGlobal(a);
    return a.x;
}

function set_y2(value:Number):Void {
    var a = {x:rp.x, y:rp.y};
    _parent.localToGlobal(a);
    _parent._y += value-a.y;
}

function get_y2():Number {
    var a = {x:rp.x, y:rp.y};
    _parent.localToGlobal(a);
    return a.y;
}

function get_xscale2():Number {
    return _parent._xscale;
}

function set_xscale2(value:Number):Void {
    setProperty2("_xscale",value);
}

```

```
function get_yscale2():Number {
    return _parent._yscale;
}

function set_yscale2(value:Number):Void {
    setProperty2("_yscale",value);
}

function get_rotation2():Number {
    return parent.rotation;
}
function set_rotation2(value:Number):Void {
    setProperty2("_rotation",value);
}

function setProperty2(prop:String, n:Number):Void {
    var a = {x:rp.x, y:rp.y};
    _parent.localToGlobal(a);

    _parent[prop] = n;

    var b = {x:rp.x, y:rp.y};
    _parent.localToGlobal(b);

    _parent._x -= b.x-a.x;
    _parent._y -= b.y-a.y;
}
```



FORMULIR BIMBINGAN SKRIPSI

Nama : Zulio Raynaldi Pakar Helambang
Nim : 1118001
Masa Bimbingan : 27 Oktober 2014 S/D 27 April 2015
Judul Skripsi : Rancang Bangun *Game Adventure Gunslinger* menggunakan metode *Finite State Machine*

No.	Tanggal	Uraian	Paraf Pembimbing
1	03 /12/2014	Revisi Program	
2	04/12/2014	Revisi program	
3	04/12/2014	Revisi program dan laporan	
4	28/01/2015	Demo Program	
5	02/02/2015	Revisi Tampilan Menu	
6	02/02/2015	Revisi Program	
7	02/02/2015	Revisi Laporan	
8	02/02/2015	Revisi Makalah	
9	23/02/2015	Revisi Laporan	
10	23/02/2015	Revisi Laporan	

Malang, 4 Desember 2014
Dosen Pembimbing

Ir. Yusuf Ismail Nakhoda, MT.
NIP.Y. 1018800189



INSTITUT TEKNOLOGI NASIONAL MALANG
Fakultas Teknologi Industri
Program Studi Teknik Informatika S1

FORMULIR BIMBINGAN SKRIPSI

Nama : Zulio Raynaldi Pakar Helambang
Nim : 1118001
Masa Bimbingan : 27 Oktober 2014 S/D 27 April 2015
Judul Skripsi : Rancang Bangun *Game Adventure Gunslinger* menggunakan metode *Finite State Machine*

No.	Tanggal	Uraian	Paraf Pembimbing
1	01/12/2014	Penambahan perancangan	
2	03/12/2014	Revisi bab III	
3	20/01/2015	Revisi bab IV (pengujian)	
4	20/01/2015	Demo Program	
5	24/01/2015	Revisi bab V (kesimpulan)	
6	28/01/2015	Acc bab IV dan bab V	
7	10/02/2015	Revisi makalah seminar hasil	
8	15/02/2015	Acc makalah seminar hasil	
9	20/02/2015	Revisi laporan seminar komprehensif	
10	23/02/2015	Acc laporan komprehensif	

Malang, 4 Desember 2014
Dosen Pembimbing

Sonny Prasetio, ST. MT
NIP.Y. 1031000433



INSTITUT TEKNOLOGI NASIONAL MALANG
Fakultas Teknologi Industri
Program Studi Teknik Informatika S1

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

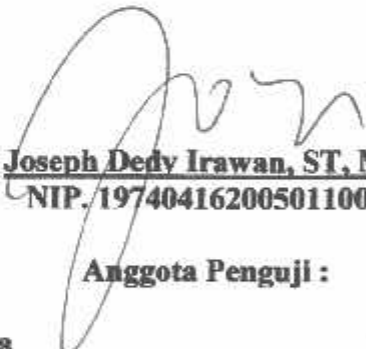
Nama : Zulio Raynaldi Pakar Herlambang
NIM : 1118001
Jurusan : Teknik Informatika S-1
Judul : Rancang Bangun Game Adventure Gunslinger Dengan Metode Finite State Machine

Dipertahankan dihadapan Majelis Penguji Skripsi Jenjang Strata Satu (S-1) pada :

Hari : Kamis
Tanggal : 23 Februari 2015
Tempat : Ruang Laboratorium Database Teknik Informatika S-1
Nilai : (A)

Panitia Ujian Skripsi :

Ketua Majelis Penguji


Joseph Dedy Irawan, ST, MT
NIP. 197404162005011002

Anggota Penguji :

Penguji Pertama


Yosep Agus Pranoto, ST, MT
NIP. 1031000435



Penguji Kedua


Ahmad Faisal, ST, MT
NIP.P 1031000431




FORMULIR PERBAIKAN SKRIPSI

Nama : Zulio Raynaldi Pakar Herlambang
NIM : 1118001
Jurusan : Teknik Informatika S-1
Judul : Rancang Bangun Game Adventure Gunslinger Dengan Metode Finite State Machine


Tanggal	Penguji	Uraian	Paraf
23 Februari 2015	I	<ul style="list-style-type: none">- Perbaiki latar belakang- Perbaiki abstraksi- Tulisan asing cetak miring- Perbaiki penomoran sub bab- Perbaiki nomor halaman- Penambahan pengujian metode- Penyesuaian metode FSM pada bab III- Laporan menggunakan spasi 1,5- Penataan laporan pada halaman yang memiliki banyak space kosong	 24/3 2015
23 Februari 2015	II	<ul style="list-style-type: none">- Perbaiki abstraksi- Perbaiki flowchart game- Penambahan efek pada setiap tembakan- Penambahan pengujian spesifikasi minimum hardware- Perbaiki nomor gambar/tabel	

Anggota Penguji :

Penguji Pertama

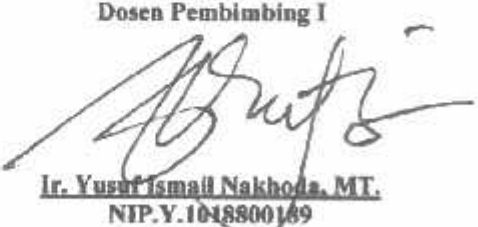

Yosep Agus Pranoto, ST, MT
NIP. 1031000432

Penguji Kedua



Ahmad Faisal, ST, MT
NIP.P 1031000431

Mengetahui

Dosen Pembimbing I


Ir. Yusuf Ismail Nakhoda, MT.
NIP.Y.1018800109

Dosen Pembimbing-II


Sonny Prasetio, ST, MT.
NIP.P 1031000429