

**APLIKASI GAME SAVE YOUR GENERATION BERBASIS
ANDROID**

LAPORAN AKHIR

Digunakan Sebagai Syarat Maju Ujian Diploma III
Politeknik Negeri Malang

Oleh :

BELLINDA NOVITA WASONO

NIM. 1431140047

EKO PUTRA SYAH WARMAN

NIM. 1431140126



**PROGRAM STUDI MANAJEMEN INFORMATIKA
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG
JUNI 2017**

HALAMAN PENGESAHAN

APLIKASI GAME SAVE YOUR GENERATION BERBASIS ANDROID

Disusun oleh :

BELLINDA NOVITA WASONO	1431140047
EKO PUTRA SYAH WARMAN	1431140126

Laporan Akhir ini telah diuji pada tanggal **2017**

Disetujui oleh:

- | | | | |
|------------------|---|--|-------|
| 1. Penguji I | : | <u>Ridwan Rismanto, SST.,M.Kom.</u>
NIP. 198603182012121001 | |
| 2. Penguji II | : | <u>Ariadi Retno Tri Hayati Ririd,</u>
<u>S.Kom.,M.Kom.</u>
NIP. 198108102005012002 | |
| 3. Pembimbing I | : | <u>Dr. Eng. Cahya Rahmad,ST.,M.Kom.</u>
NIP. 197202022005011002 | |
| 4. Pembimbing II | : | <u>Dr. Eng. Faisal Rahutomo,ST.,M.Kom.</u>
NIP. 197711162005011008 | |

Mengetahui,

Ketua Jurusan
Teknologi Informasi

Ketua Program Studi
Manajemen Informatika

Rudy Ariyanto, S.T., M.Cs.
NIP. 19711110 199903 1 002

Dr. Eng. Rosa Andrie A., S.T., M.T.
NIP. 19801010 200501 1 001

PERNYATAAN

Dengan ini saya menyatakan bahwa Laporan Akhir ini tidak terdapat karya yang pernah diajukan untuk memperoleh gelar Ahli Madya/kesarjanaan di suatu Perguruan Tinggi, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Malang,

Bellinda Novita Wasono

PERNYATAAN

Dengan ini saya menyatakan bahwa Laporan Akhir ini tidak terdapat karya yang pernah diajukan untuk memperoleh gelar Ahli Madya/kesarjanaan di suatu Perguruan Tinggi, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Malang,

Eko Putra Syah Warman

ABSTRAK

Wasono, Bellinda Novita, Warman, Eko Putra Syah. “Aplikasi Game Save Your Generation Berbasis Android”. **Pembimbing: (1) Dr. Eng. Cahya Rahmad, S.T.,M.Kom, (2) Dr. Eng. Faisal Rahutomo, S.T.,M.Kom.**

Laporan Akhir, Program Studi Manajemen Informatika, Jurusan Teknologi Informasi, Politeknik Negeri Malang, 2017.

Informasi tentang bahaya dari rokok, narkoba, dan minuman keras perlu disampaikan kepada anak-anak, karena pada usia dini lah pembentukan sikap dan karakter pada anak dimulai. Oleh sebab itu, pembentukan sikap dan karakter yang anti terhadap rokok, narkoba, dan minuman keras harus diterapkan dan ditanamkan pada anak-anak sejak dini, agar mereka menjadi generasi yang sehat. Penyuluhan tentang bahaya rokok, narkoba, dan minuman keras sudah dilakukan oleh pemerintah dan masyarakat yang berisi edukasi tentang rokok, minuman keras, dan narkoba serta bahaya dari zat-zat yang terkandung didalamnya. Game merupakan sebuah media pembelajaran untuk pentingnya peduli terhadap kesehatan diri sendiri.

Dalam game ini terdapat Menu Kuis, yang berisi Menu Belajar dan Menu Soal. Menu Belajar dan Menu Soal dapat digunakan untuk menambah pengetahuan siswa serta menguji pengetahuan mereka tentang rokok, narkoba, dan minuman keras. Diharapkan dengan adanya game ini siswa dapat memperoleh pengetahuan melalui permainan game dan dapat meningkatkan kesadaran untuk pentingnya menjaga kesehatan diri sendiri.

Kata Kunci : Game 2D, Rokok, Narkoba, Minuman Keras, Kesehatan

ABSTRACT

Wasono, Bellinda Novita, Warman, Eko Putra Syah. “Application Save Your Generation Game Based Android”. Advisor: (1) Dr. Eng. Cahya Rahmad, S.T.,M.Kom, (2) Dr. Eng. Faisal Rahutomo, S.T.,M.Kom.

Final Report, Informatics Management Program, Information Technology Department, State Polytechnic of Malang, 2017.

Information about the dangers of cigarettes, drugs, and liquor needs to be delivered to children, because at an early age the formation of attitudes and character in children begins. Therefore, the formation of anti-smoking attitudes and characters, drugs, and liquors should be applied and instilled in children from an early age, in order for them to become healthy generations. Counseling about the dangers of cigarettes, drugs, and liquor has been done by the government and society that contains education about cigarettes, liquor, and drugs and the dangers of the substances contained therein. Game is a socialization media for the importance of caring for personal health.

In this game there is Menu Kuis, which contains Menu Belajar and Menu Soal. The Belajar and Soal Menus can be used to increase students' knowledge and test their knowledge of cigarettes, drugs, and liquor. Hopefully with this game students can gain knowledge through playing games and can raise awareness for the importance of maintaining personal health.

Keyword : *2D Games, Cigarettes, Drugs, Liquor, Health*

KATA PENGANTAR

Puji Syukur kami panjatkan kehadiran Allah SWT atas segala rahmat dan hidayah-Nya penulis dapat menyelesaikan laporan akhir dengan judul “*APLIKASI GAME SAVE YOUR GENERATION* BERBASIS ANDROID”. Laporan akhir ini penulis susun sebagai persyaratan untuk menyelesaikan studi program Diploma III Program Studi Manajemen Informatika, Jurusan Teknologi Informasi, Politeknik Negeri Malang.

Kami menyadari tanpa adanya dukungan dan kerja sama dari berbagai pihak, kegiatan laporan akhir ini tidak akan dapat berjalan baik. Untuk itu, kami ingin menyampaikan rasa terima kasih kepada:

1. Bapak Rudy Ariyanto, ST., M.Cs., selaku ketua jurusan Teknologi Informasi.
2. Bapak Dr. Eng. Rosa Andrie Asmara, ST., MT., selaku ketua program studi Manajemen Informatika.
3. Bapak Dr. Eng. Cahya Rahmad, ST., M.Kom. dan Bapak Dr. Eng. Faisal Rahutomo, ST., M.Kom. selaku pembimbing laporan akhir.
4. Keluarga dan teman-teman angkatan 2014 Manajemen Infromatika Politeknik Negeri Malang.
5. Dan seluruh pihak yang telah membantu dan mendukung lancarnya pembuatan Laporan Akhir dari awal hingga akhir yang tidak dapat kami sebutkan satu persatu.

Penulis menyadari bahwa dalam penyusunan laporan akhir ini, masih banyak terdapat kekurangan dan kelemahan yang dimiliki penulis baik itu sistematika penulisan maupun penggunaan bahasa. Untuk itu penulis mengharapkan saran dan kritik dari berbagai pihak yang bersifat membangun demi penyempurnaan laporan ini. Semoga laporan ini berguna bagi pembaca secara umum dan penulis secara khusus. Akhir kata, penulis ucapkan banyak terima kasih.

Malang, Juni 2017

DAFTAR ISI

	Halaman
HALAMAN JUDUL.....	i
HALAMAN PENGESAHAN.....	ii
PERNYATAAN.....	iii
PERNYATAAN.....	iv
ABSTRAK	v
ABSTRACT.....	vi
KATA PENGANTAR	vii
DAFTAR ISI.....	viii
DAFTAR GAMBAR	xi
DAFTAR TABEL.....	xiii
BAB I. PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	2
1.3 Tujuan Penulisan	2
1.4 Batasan Masalah.....	3
1.5 Sistematika Penulisan.....	3
BAB II. LANDASAN TEORI	5
2.1 Game	5
2.2 Android.....	5
2.3 Unity	5
2.4 Adobe Photoshop	6
2.5 Adobe Illustrator.....	6
2.6 Storyboard	6
2.7 Game Play	6
2.8 Flowchart.....	7
2.9 Rokok	7
2.10 Narkoba.....	7
2.11 Minuman Keras	7
BAB III. ANALISIS DAN PERANCANGAN	8
3.1 Analisis	8
3.1.1 Analisis Masalah	8

3.1.2	Spesifikasi Perangkat Keras dan Perangkat Lunak	9
3.2	Perancangan.....	10
3.2.1	Storyboard	10
3.2.2	Perancangan Materi.....	13
3.2.3	Perancangan Karakter	13
3.2.4	Perancangan Gameplay	14
3.2.5	Flowchart	15
3.2.6	Menu Game	18
BAB IV.	IMPLEMENTASI	19
4.1	Pembuatan Sprite.....	19
4.1.1	Karakter Pemain SD.....	19
4.1.2	Karakter Pemain SMP	20
4.1.3	Karakter Pemain SMA	20
4.1.4	Karakter Monster Rokok.....	21
4.1.5	Karakter Monster Minuman Keras.....	21
4.1.6	Karakter Monster Narkoba.....	22
4.2	Pembuatan Aplikasi.....	22
4.2.1	Menu Utama.....	22
4.2.2	Arena Satu sampai Arena Tiga	23
BAB V.	PENGUJIAN DAN PEMBAHASAN	35
5.1	Hasil Perancangan	35
5.1.1	Halaman Menu Utama	35
5.1.2	Halaman Informasi.....	36
5.1.3	Halaman Bantuan	36
5.1.4	Halaman Skor Tertinggi	37
5.1.5	Halaman Arena 1.....	38
5.1.6	Halaman Arena 2.....	40
5.1.7	Halaman Arena 3.....	42
5.1.8	Halaman Kuis.....	44
5.1.9	Halaman Belajar.....	44
5.1.10	Halaman Belajar Rokok	45
5.1.11	Halaman Belajar Minuman Keras	45
5.1.12	Halaman Belajar Narkoba	46
5.1.13	Halaman Soal	46
5.1.14	Halaman Nilai Soal	47
5.2	Pengujian	47
5.2.1	Spesifikasi Perangkat Uji Coba.....	47
5.3	Pengujian Fungsional	48
5.3.1	Uji Coba Sistem Game.....	48

5.4 Uji Coba User	49
5.5 Analisis Hasil Uji Coba	54
BAB VI. KESIMPULAN.....	55
6.1 Kesimpulan.....	55
6.2 Saran	55
DAFTAR PUSTAKA	56

DAFTAR GAMBAR

Gambar 3.1 Flowchart Game	16
Gambar 3.2 Flowchart Kuis	17
Gambar 3.3 Perancangan Menu Game.....	18
Gambar 4.1 Sprite Karakter Pemain SD	19
Gambar 4.2 Sprite Karakter Pemain SMP	20
Gambar 4.3 Sprite Karakter Pemain SMA.....	20
Gambar 4.4 Sprite Karakter Monster Rokok	21
Gambar 4.5 Sprite Karakter Monster Minuman Keras	21
Gambar 4.6 Sprite Karakter Monster Narkoba	22
Gambar 4.7 Pembuatan Menu Utama	23
Gambar 4.8 Pengaturan main camera arena 1 sampai 3	23
Gambar 4.9 Hirarki karakter pemain	24
Gambar 4.10 Animator controller karakter pemain saat di ground	25
Gambar 4.11 Animator controller karakter pemain saat di jump.....	25
Gambar 4.12 Animation karakter pemain idle	26
Gambar 4.13 Animation karakter pemain run.....	26
Gambar 4.14 Animation karakter pemain attack	26
Gambar 4.15 Animation karakter pemain death	27
Gambar 4.16 Animation karakter pemain landing	27
Gambar 4.17 Animation karakter pemain take off.....	27
Gambar 4.18 Pengaturan monster biasa.....	28
Gambar 4.19 Pengaturan monster attack	29
Gambar 4.20 Hirarki monster attack	30
Gambar 4.21 Penempatan event function EnemyAttack	31
Gambar 4.22 Hirarki bos arena	31
Gambar 4.23 Pengaturan bos arena 1	32
Gambar 4.24 Pengaturan bos arena 2.....	33
Gambar 4.25 Pengaturan bos arena 3.....	33
Gambar 4.26 Pengaturan serangan.....	34
Gambar 4.27 Pengaturan health bar karakter pemain	34
Gambar 5.1 Tampilan Menu Utama Game	35
Gambar 5.2 Tampilan Menu Informasi.....	36

Gambar 5.3 Tampilan Menu Bantuan	37
Gambar 5.4 Tampilan Menu Skor Tertinggi.....	37
Gambar 5.5 Tampilan Arena 1	38
Gambar 5.6 Tampilan Pause	38
Gambar 5.7 Tampilan Percakapan Karakter dan Bos Monster Rokok	39
Gambar 5.8 Tampilan Karakter Menghadapi Bos Monster Rokok	39
Gambar 5.9 Tampilan Karakter Kalah	40
Gambar 5.10 Tampilan Arena 2	40
Gambar 5.11 Tampilan Percakapan Karakter Bos Monster Minuman Keras.....	41
Gambar 5.12 Tampilan Karakter Menghadapi Bos Monster Minuman Keras	41
Gambar 5.13 Tampilan Arena 3	42
Gambar 5.14 Tampilan Percakapan Karakter Bos Monster Narkoba.....	42
Gambar 5.15 Tampilan Karakter Menghadapi Bos Monster Narkoba	43
Gambar 5.16 Tampilan Karakter Menang	43
Gambar 5.17 Tampilan Halaman Kuis.....	44
Gambar 5.18 Tampilan Halaman Belajar.....	44
Gambar 5.19 Tampilan Halaman Belajar Rokok.....	45
Gambar 5.20 Tampilan Halaman Belajar Minuman Keras.....	45
Gambar 5.21 Tampilan Halaman Belajar Narkoba.....	46
Gambar 5.22 Tampilan Halaman Soal	47
Gambar 5.23 Tampilan Halaman Nilai Soal	47

DAFTAR TABEL

Tabel 3.1 Spesifikasi perangkat keras	9
Tabel 3.2 Spesifikasi perangkat lunak	9
Tabel 3.3 Storyboard	10
Tabel 3.4 Karakter Game Save Your Generation	13
Tabel 5.1 Spesifikasi perangkat yang dibutuhkan.....	47
Tabel 5.2 Daftar perangkat uji coba	48
Tabel 5.3 Uji coba sistem game	48
Tabel 5.4 Kuisisioner pengguna game.....	49
Tabel 5.5 Diagram hasil kuisisioner untuk pertanyaan pertama.....	50
Tabel 5.6 Diagram hasil kuisisioner untuk pertanyaan kedua.	51
Tabel 5.7 Diagram hasil kuisisioner untuk pertanyaan ketiga.	51
Tabel 5.8 Diagram hasil kuisisioner untuk pertanyaan keempat.	52
Tabel 5.9 Diagram hasil kuisisioner untuk pertanyaan kelima.....	52
Tabel 5.10 Diagram hasil kuisisioner untuk pertanyaan keenam.	53
Tabel 5.11 Diagram hasil kuisisioner untuk pertanyaan ketujuh.....	53

BAB I. PENDAHULUAN

Pada bab ini menjelaskan tentang latar belakang, permasalahan dan tujuan pembuatan *Aplikasi Game Save Your Generation Berbasis Android*.

1.1 Latar Belakang

Peringatan “Merokok Membunuhmu” terdapat pada spanduk iklan rokok, menggantikan peringatan “Merokok dapat menyebabkan kanker, serangan jantung, impotensi dan gangguan kehamilan dan janin. Bukan hanya slogannya yang berubah, melainkan untuk sekarang di setiap bungkus rokok dilengkapi dengan gambar. Pemuatan gambar tersebut diamanatkan dalam Peraturan Pemerintah (PP Nomor 109 tahun 2012) tentang pengamanan bahan yang mengandung zat adiktif berupa tembakau bagi kesehatan. Tujuannya untuk memberikan informasi yang benar dan jelas kepada masyarakat agar berfikir rasional untuk membeli rokok. [1]

Menurut data yang dikutip dari Badan Narkotika Nasional Provinsi (BNNP) Sulawesi Selatan, dampak narkoba meliputi dampak fisik, psikologis, sosial dan ekonomi. Dampak fisik misalnya gangguan pada sistem saraf (neurologis): kejang-kejang, halusinasi, dan gangguan kesadaran. Dampak psikologis berupa tidak normalnya kemampuan berpikir, berperasaan cemas, ketergantungan/selalu membutuhkan obat. [2]

Selama tujuh tahun belakangan ini terjadi peningkatan luar biasa konsumsi minuman keras (miras) di kalangan remaja. Jika pada 2007 berdasarkan Riset Kesehatan Dasar Departemen Kesehatan jumlah remaja pengonsumsi miras di Indonesia masih diangka 4,9%, tetapi pada 2014 berdasarkan hasil riset yang dilakukan Gerakan Nasional Anti Miras (GeNAM) jumlahnya melonjak drastis hingga menyentuh angka 23% dari total jumlah remaja Indonesia yang saat ini berjumlah 63 juta jiwa atau sekitar 14,4 juta orang. Menurut Ketua Umum GeNam Fahira, salah satu persoalan yang dihadapi kota-kota besar di Indonesia seperti Palembang adalah masifnya peredaran dan konsumsi miras di kalangan remaja. Sehingga tidak heran dari 18 ribu nyawa yang melayang akibat miras setiap tahun, sepertiganya atau 6.000 orang adalah remaja, baik karena miras itu sendiri maupun menjadi korban kejahatan di bawah pengaruh miras. [3]

Informasi tentang bahaya dari rokok, narkoba, dan minuman keras perlu disampaikan kepada anak-anak, karena pada usia dini lah pembentukan sikap dan karakter pada anak dimulai. Penyuluhan tentang bahaya rokok, narkoba, dan minuman keras sudah dilakukan oleh pemerintah dan masyarakat yang berisi edukasi tentang rokok, minuman keras, dan narkoba serta bahaya dari zat-zat yang terkandung didalamnya. Dalam melakukan penyuluhan dapat dilakukan dengan media yang sudah ada seperti iklan layanan masyarakat di televisi dan radio, poster-poster, baliho, video-video tentang penyuluhan, dan dapat juga melalui *game*.

Dengan semakin berkembangnya teknologi multimedia saat ini memberikan pengaruh yang besar pada perkembangan *game*. *Game* sangat disukai mulai dari anak-anak hingga orang dewasa. Penyuluhan tentang bahaya rokok, narkoba, dan minuman keras dapat disampaikan melalui *game*. Perancangan *game Save Your Generation* ini diharapkan dapat menjadi media penyuluhan untuk meningkatkan kesadaran dan menyampaikan informasi tentang bahaya rokok, narkoba, dan minuman keras kepada anak-anak agar terbebas dan terhindar dari penyakit-penyakit yang disebabkan oleh zat-zat berbahaya yang terkandung didalam rokok, narkoba, dan minuman keras. Dengan adanya *game* ini diharapkan anak-anak dapat mengerti pentingnya peduli terhadap kesehatan diri sendiri.

1.2 Rumusan Masalah

Berdasarkan latar belakang diatas, maka rumusan masalah adalah sebagai berikut:

Bagaimana merancang dan membuat *game* 2D sebagai media pembelajaran yang dapat memberikan pengetahuan pada usia dini agar mereka mengerti tentang pentingnya untuk tidak memakai dan mengkonsumsi rokok, narkoba, dan minuman keras?

1.3 Tujuan Penulisan

Tujuan pembuatan *game* ini adalah sebagai salah satu media alternatif pembelajaran yang dapat memberikan pengetahuan kepada anak-anak tentang rokok, narkoba dan minuman keras agar mereka mengetahui bahwa zat-zat yang terkandung di dalamnya berbahaya bagi kesehatan.

1.4 Batasan Masalah

Batasan masalah digunakan agar pembahasan dalam laporan akhir ini tidak menyimpang dari topik yang telah ditentukan sebelumnya. Batasan masalah dalam laporan akhir ini dapat dirumuskan sebagai berikut :

- a. *Game* berbasis Android.
- b. *Game* dirancang untuk *single player*.
- c. Diperuntukkan untuk anak usia 7-13 tahun.
- d. Pengetahuan yang ingin disampaikan melalui *game* ini adalah pentingnya mengetahui bahaya yang ditimbulkan apabila menggunakan rokok, narkoba, dan minuman keras terhadap tubuh yang menggunakannya.
- e. Aplikasi ini dibuat dengan menggunakan *software* Unity dengan grafis 2D.

1.5 Sistematika Penulisan

Sistematika penulisan dalam laporan berisi sebagai berikut :

BAB I. Pendahuluan

Pada bab ini dijelaskan mengenai hal – hal yang bersifat umum seperti latar belakang, rumusan masalah, batasan masalah, tujuan penulisan, batasan masalah, metodologi dan sistematika penulisan.

BAB II. Landasan Teori

Pada bab ini berisikan teori-teori yang mendasari dan berkaitan dengan masalah perencanaan dan pembuatan aplikasi yang digunakan acuan untuk memudahkan pemahaman dan pemecahan terhadap masalah yang ada.

BAB III. Analisis dan Perancangan

Pada bab ini berisikan tentang analisa untuk pembuatan aplikasi dan perencanaan konsep, software, dan asset-asset properti yang digunakan. Pada bab ini dijelaskan secara terperinci mengenai pengertian-pengertian apa saja yang dibutuhkan / digunakan baik mengenai software dan hal yang berkaitan dengan topik proyek Tugas Akhir.

BAB IV. Implementasi

Pada bab ini membahas tentang pengujian dan analisis dari aplikasi.

BAB V. Uji Coba

Pada bab ini berisi tentang hasil pengujian aplikasi, metode dan *game* yang sudah di uji.

BAB VI. Penutup

Membahas kesimpulan dari pembahasan pada perancangan awal serta analisis yang diperoleh. Untuk lebih meningkatkan mutu dari aplikasi yang telah dibuat maka kami memberikan saran-saran untuk perbaikan dan pengembangan *game*.

BAB II. LANDASAN TEORI

Pada bab ini akan membahas mengenai teori-teori yang relevan yang melengkapi latar belakang. Sekaligus memberi *review* tentang pustaka yang telah dibaca selama pencarian solusi terhadap masalah yang diangkat dalam tugas akhir.

2.1 Game

Permainan adalah sebuah sistem dimana pemain terlibat dalam konflik buatan, ditentukan oleh aturan yang menghasilkan hasil yang terukur. Teori tersebut dikemukakan oleh Katie Salen dan Eric Zimmerman. Sedangkan menurut David Parlett *Game* adalah sesuatu yang memiliki akhir dan cara pencapaiannya. Dalam *game* terdapat tujuan, hasil dan serangkaian peraturan untuk mencapai keduanya. Sehingga dapat disimpulkan *game* merupakan sarana yang menangani hubungan input/output, yang dikemas dalam sebuah sistem dimana pemain terlibat dalam konflik buatan yang sudah ditentukan oleh aturan untuk menghasilkan suatu tujuan tertentu. [4]

2.2 Android

Android adalah salah satu platform sistem operasi yang digemari masyarakat karena sifatnya yang open source sehingga memungkinkan pengguna untuk melakukan pengembangan. Android merupakan generasi baru platform mobile berbasis linux yang mencakup sistem operasi, middleware, dan aplikasi (Nazruddin Safaat, 2012) [5]

2.3 Unity

Unity merupakan suatu aplikasi yang digunakan untuk mengembangkan *game* multi platform yang didesain untuk mudah digunakan. Unity itu bagus dan penuh perpaduan dengan aplikasi yang profesional. Editor pada Unity dibuat dengan user interface yang sederhana. Editor ini dibuat setelah ribuan jam yang mana telah dihabiskan untuk membuatnya menjadi nomor satu dalam urutan ranking teratas untuk editor *game*. Grafis pada unity dibuat dengan grafis tingkat tinggi untuk OpenGL dan DirectX. Unity mendukung semua format file, terutamanya format umum seperti semua format dari art applications. Unity cocok dengan versi 64-bit

dan dapat beroperasi pada Mac OS x dan windows dan dapat menghasilkan *game* untuk Mac, Windows, Wii, iPhone, iPad dan Android. [6]

2.4 Adobe Photoshop

Adobe Photoshop, atau biasa disebut Photoshop, adalah perangkat lunak editor citra buatan Adobe Systems khusus untuk pengeditan foto/gambar dan pembuatan efek. Perangkat lunak ini banyak digunakan oleh fotografer digital dan perusahaan iklan sehingga dianggap sebagai pemimpin pasar (*market leader*) untuk perangkat lunak pengolah gambar/foto, dan, bersama Adobe Acrobat, dianggap sebagai produk terbaik yang pernah diproduksi oleh Adobe Systems. Versi kedelapan aplikasi ini disebut dengan nama Photoshop CS (Creative Suite), versi sembilan disebut Adobe Photoshop CS2, dan versi terbaru adalah Adobe Photoshop CC.

2.5 Adobe Illustrator

Adobe Illustrator adalah program editor grafis vektor terkemuka, dikembangkan dan dipasarkan oleh Adobe Systems. Illustrator CC merupakan versi terkini program ini, generasi kedua puluh untuk produk Illustrator.

2.6 Storyboard

Storyboard adalah visualisasi ide dari aplikasi yang akan dibangun, sehingga dapat memberikan gambaran dari aplikasi yang akan dihasilkan. Storyboard dapat dikatakan juga visual script yang akan dijadikan outline dari sebuah proyek, ditampilkan shot by shot yang biasa disebut dengan istilah scene. Storyboard sekarang lebih banyak digunakan untuk membuat kerangka pembuatan website dan proyek media interaktif lainnya seperti iklan, *game*, media pembelajaran interaktif ketika dalam tahap perancangan/desain. [7]

2.7 Game Play

Gameplay adalah alur sistem dalam sebuah *game* yaitu dimulai dari menu, area permainan, *game over*, storyline, keberhasilan misi, kegagalan misi, cara bermain, dan sistem lain yang harus ditentukan. *Gameplay* harus dirancang dengan baik, agar *gameplay* mudah untuk dimainkan dan tidak menyulitkan pemain. [8]

2.8 Flowchart

Menurut Krismiaji (2010), Bagan alir merupakan teknik analitis yang digunakan untuk menjelaskan aspek-aspek sistem informasi secara jelas, tepat dan logis. Bagan alir menggunakan serangkaian simbol standar untuk menguraikan prosedur pengolahan transaksi yang digunakan oleh sebuah perusahaan, sekaligus menguraikan aliran data dalam sebuah sistem (Pradana, 2013:12).

2.9 Rokok

Rokok adalah salah satu zat adiktif yang bila digunakan mengakibatkan bahaya bagi kesehatan individu dan masyarakat. Kemudian ada juga yang menyebutkan bahwa rokok adalah hasil olahan tembakau terbungkus termasuk cerutu atau bahan lainnya yang dihasilkan dari tanaman *Nicotiana Tabacum*, *Nicotiana Rustica* dan spesies lainnya atau sintesisnya yang mengandung nikotin dan tar dengan atau tanpa bahan tambahan. (Hans Tendra, 2003) [9]

2.10 Narkoba

Narkotika dan Obat-obatan terlarang (NARKOBA) atau Narkotik, Psikotropika, dan Zat Aditif (NAPZA) adalah bahan / zat yang dapat mempengaruhi kondisi kejiwaan / psikologi seseorang (pikiran, perasaan dan perilaku) serta dapat menimbulkan ketergantungan fisik dan psikologi. [10]

2.11 Minuman Keras

Minuman keras (MIRAS) adalah berbagai macam jenis minuman beralkohol mengandung ethanol (ethyl alcohol). (Joewana, 2001:9) Contohnya : bir, anggur, brandy, wiski, vodka, arak, tual dan lain-lain.

Alkohol menekan kerja otak (depresansia). Setelah diminum, alkohol diserap oleh tubuh dan masuk ke dalam pembuluh darah. Alkohol dapat menyebabkan mabuk, jalan sempoyongan, bicara cadel, kekerasan atau perbuatan merusak, ketidakmampuan belajar dan lain-lain. (Joewana, dkk : 13) [11]

BAB III. ANALISIS DAN PERANCANGAN

Pada bab ini diuraikan dengan jelas sistem yang akan dibuat dan kebutuhan sistem yang meliputi kebutuhan perangkat keras dan perangkat lunak. Rancangan sistem meliputi rancangan model, rancangan arsitektur sistem, rancangan proses, rancangan data dan rancangan antarmuka pengguna (*user interface*).

3.1 Analisis

Analisis bertujuan untuk mengidentifikasi permasalahan yang ada pada sistem, serta menentukan kebutuhan sistem yang akan dibangun. Analisis merupakan tahapan yang amat penting, karena kesalahan di tahap ini menyebabkan kesalahan di tahap selanjutnya. Aplikasi yang dibuat merupakan *game* 2 dimensi berbasis *sprite*, dimana obyek di dalam *game* dibentuk menggunakan gambar-gambar 2 dimensi. Aplikasi merupakan media alternatif pembelajaran tentang pengetahuan dan dampak buruk dari rokok, narkoba, dan minuman keras kepada pelajar SD.

3.1.1 Analisis Masalah

Informasi tentang bahaya rokok, narkoba, dan minuman keras terhadap kesehatan, sangat penting untuk kita ketahui terutama bagi para pelajar. Sebab mereka berada pada usia dini yang rentan terpengaruh oleh hasutan-hasutan untuk mencoba benda-benda tersebut. Selama ini, informasi tersebut disampaikan melalui baliho, poster, iklan layanan masyarakat di televisi dan radio, dan video-video tentang penyuluhan. Dimana mereka hanya membaca dan mendengarkan, sehingga kurang menarik perhatian para pelajar.

Agar solusi tersebut dapat dicapai maka dirancang *game* 2 dimensi berbasis android yang dapat memberikan pengetahuan dan pengalaman bermain *game* tentang memberantas rokok, narkoba, dan minuman keras secara lebih menyenangkan.

3.1.2 Spesifikasi Perangkat Keras dan Perangkat Lunak

a. Perangkat Keras (*Hardware*)

Perangkat keras (*hardware*) yang digunakan dalam pembuatan *game* ini adalah *Personal Computer* atau *notebook* dengan spesifikasi minimal yang dapat dilihat pada Tabel 3.1.

Tabel 3.1 Spesifikasi perangkat keras

Keterangan	Spesifikasi
Processor	Intel Core i5 2.50 GHz.
Harddisk	HDD 750 GB.
RAM	RAM 4 Gb.
Smartphone	Min Android KitKat 4.4

b. Perangkat Lunak (*Software*)

Untuk membangun *game* ini diperlukan perangkat lunak (*software*) yang mampu mendukung pengoperasian sistem. Kebutuhan perangkat lunak minimal dapat dilihat pada Tabel 3.2.

Tabel 3.2 Spesifikasi perangkat lunak

Nama	Kegunaan
Unity	Sebagai <i>software</i> untuk pembuatan <i>game</i> dan menulis kode program.
Adobe Illustrator	Sebagai <i>software</i> pembuatan desain grafis.
Adobe Photoshop	Sebagai <i>software</i> pembuatan desain grafis.
Adobe Flash	Sebagai <i>software</i> pembuatan animasi gerakan karakter dalam <i>game</i> .
MonoDevelop	Sebagai <i>text editor</i> pemrograman dengan bahasa C#.

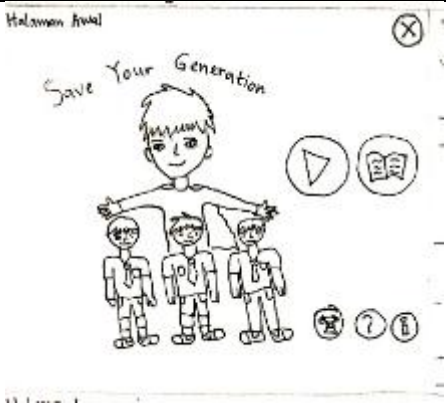

3.2 Perancangan

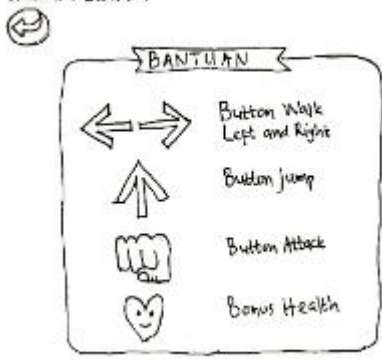



Pada tahap perancangan berisi konsep aplikasi, yang bertujuan untuk menjelaskan gambaran umum sistem, menentukan desain karakter, menentukan desain antar muka dari menu aplikasi yang akan dibangun dan rancangan proses dari aplikasi yang akan dibuat.





3.2.1 Storyboard


Storyboard game “Save Your Generation” adalah berupa konsep dasar *game* yang dibuat menggunakan gambar-gambar dan keterangan dari gambar tersebut. Dapat dijelaskan pada tabel 3.3 berikut :

Tabel 3.3 Storyboard

No	Gambar	Keterangan
1.		<p>Halaman awal, terdapat beberapa menu yaitu :</p> <ol style="list-style-type: none"> 1. Play, untuk memulai <i>game</i>. 2. Kuis, untuk belajar dan mengerjakan soal mengenai tema dari <i>game</i>. 3. Informasi 4. Bantuan 5. Skor 6. Keluar
2.		<p>Halaman Informasi tentang profil pembuat <i>game</i>.</p>

3.	<p>Halaman Bantuan</p> 	<p>Halaman Bantuan berisi tentang petunjuk bermain <i>game</i>.</p>
4.	<p>Halaman Skor</p> 	<p>Halaman Skor Tentang informasi perolehan skor tertinggi yang didapat oleh player dari <i>game</i>.</p>
5.	<p>Halaman Kuis</p> 	<p>Halaman Kuis berisi 2 macam pilihan menu, yaitu: belajar dan kuis.</p>
6.	<p>Halaman Belajar</p> 	<p>Pada halaman Belajar berisi informasi tentang rokok, narkoba, dan minuman keras.</p>

7.	<p>Halaman Soal</p> 	<p>Halaman ini berisi soal-soal yang berkaitan dengan tema <i>game</i>.</p>
8.	<p>Halaman Nilai</p> 	<p>Halaman ini berisi nilai hasil soal yang baru dikerjakan.</p>
9.	<p>ARENA 1</p> 	<p>Halaman <i>game</i> arena ke-1, karakter pada arena ini adalah pelajar SD.</p>
10.	<p>ARENA 2</p> 	<p>Halaman <i>game</i> arena ke-2, karakter pada arena ini adalah pelajar SMP.</p>

11.		<p>Halaman <i>game</i> arena ke-3, karakter pada arena ini adalah pelajar SMA.</p>
-----	---	--



3.2.2 Perancangan Materi


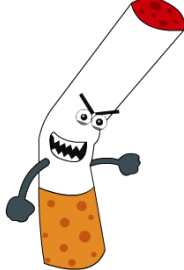


Materi yang ingin disampaikan dalam Aplikasi *Game Save Your Generation* Berbasis Android adalah pentingnya peduli terhadap kesehatan diri sendiri kepada anak sejak dini supaya menghindari dari mengkonsumsi rokok, minuman keras dan narkoba.

3.2.3 Perancangan Karakter

Pembuatan karakter *game* ini menggunakan software Adobe Photoshop dan Adobe Illustrator yang mencakup karakter pemain yang dijelaskan pada tabel 3.4 sebagai berikut :

Tabel 3.4 Karakter *Game Save Your Generation*

Gambar	Nama	Keterangan
	Anak SD	Karakter pemain yang berada di <i>game</i> arena 1.
	Anak SMP	Karakter pemain yang berada di <i>game</i> arena 2.

	Anak SMA	Karakter pemain yang berada di <i>game</i> arena 3.
	Monster Rokok	Monster ini akan muncul di semua arena <i>game</i> dan monster ini dapat mengeluarkan asap.
	Monster Minuman Keras	Monster ini akan muncul di arena dua dan tiga dan monster ini dapat mengeluarkan cairan panas dari dalam tubuhnya.
	Monster Narkoba	Monster ini akan muncul di arena tiga dan monster ini dapat mengeluarkan cairan narkoba.

3.2.4 Perancangan Gameplay

a. Arena 1

Pemain pada arena 1 adalah pelajar SD yang harus membasmi *monster* rokok, setiap *monster* yang dikalahkan akan menambah skor. Kemudian pemain akan menghadapi bos arena 1 yang sangat besar dan juga bisa membuat nyawa dari pemain menjadi berkurang.

b. Arena 2

Pemain pada arena 2 adalah pelajar SMP yang harus melewati rintangan jurang yang dalam lalu harus membasmi *monster* rokok dan

monster botol minuman keras. Jika pemain berhasil maka emain akan menghadapi bos arena 2 yaitu *monster* botol minuman keras berukuran besar yang menyerang dengan cairan panas.

c. Arena 3

Pemain pada arena 3 adalah pelajar SMA yang harus membasmi *monster* rokok, *monster* botol minuman keras, dan *monster* narkoba. Jika pemain berhasil maka pemain akan menghadapi bos arena 3 yaitu *monster* narkoba berukuran besar yang melakukan serangan tembakan langsung ke pemain.

d. Menu Belajar

Pada menu ini player bisa membaca materi mengenai rokok, narkoba dan minuman keras serta zat-zat yang terkandung di dalamnya dan dampak efek buruk yang disebabkan oleh ketiga barang tersebut.

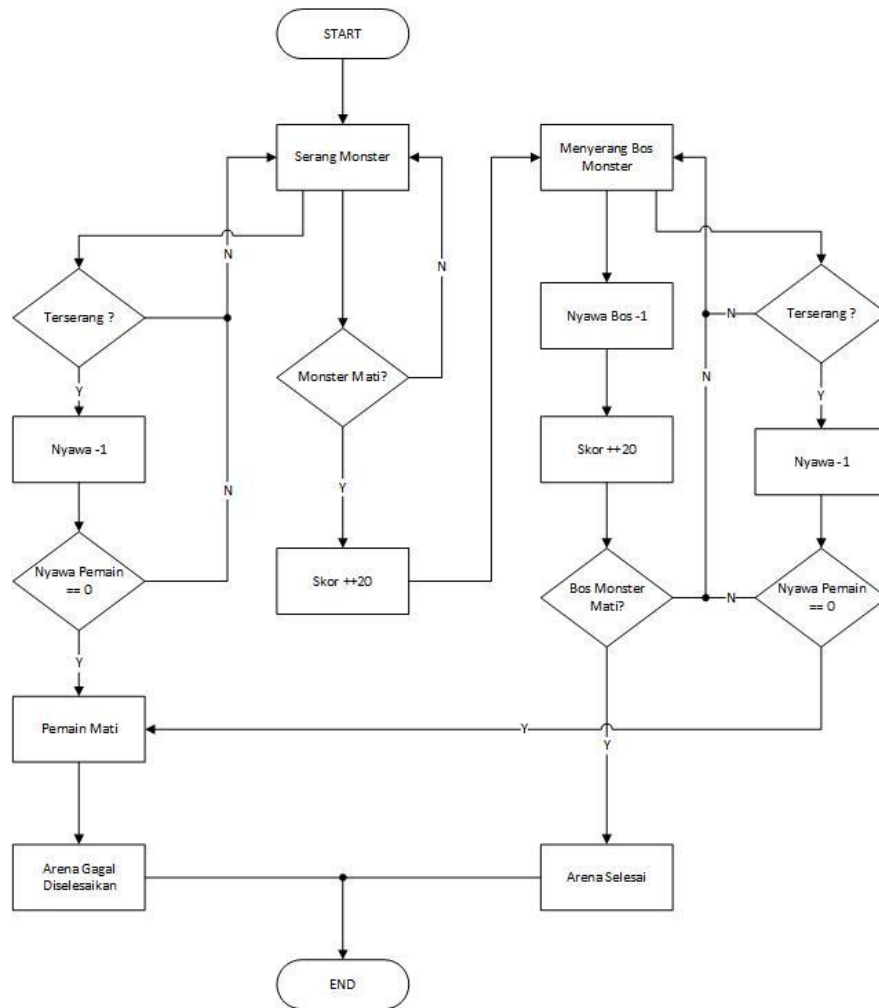
e. Menu Soal

Pada menu ini player bisa menjawab soal-soal mengenai rokok, narkoba dan minuman keras. Player akan diberikan 10 soal yang berkaitan dengan tema dari *game* dan setiap 1 soal yang benar bernilai 10 poin.

3.2.5 Flowchart

a. Flowchart Game

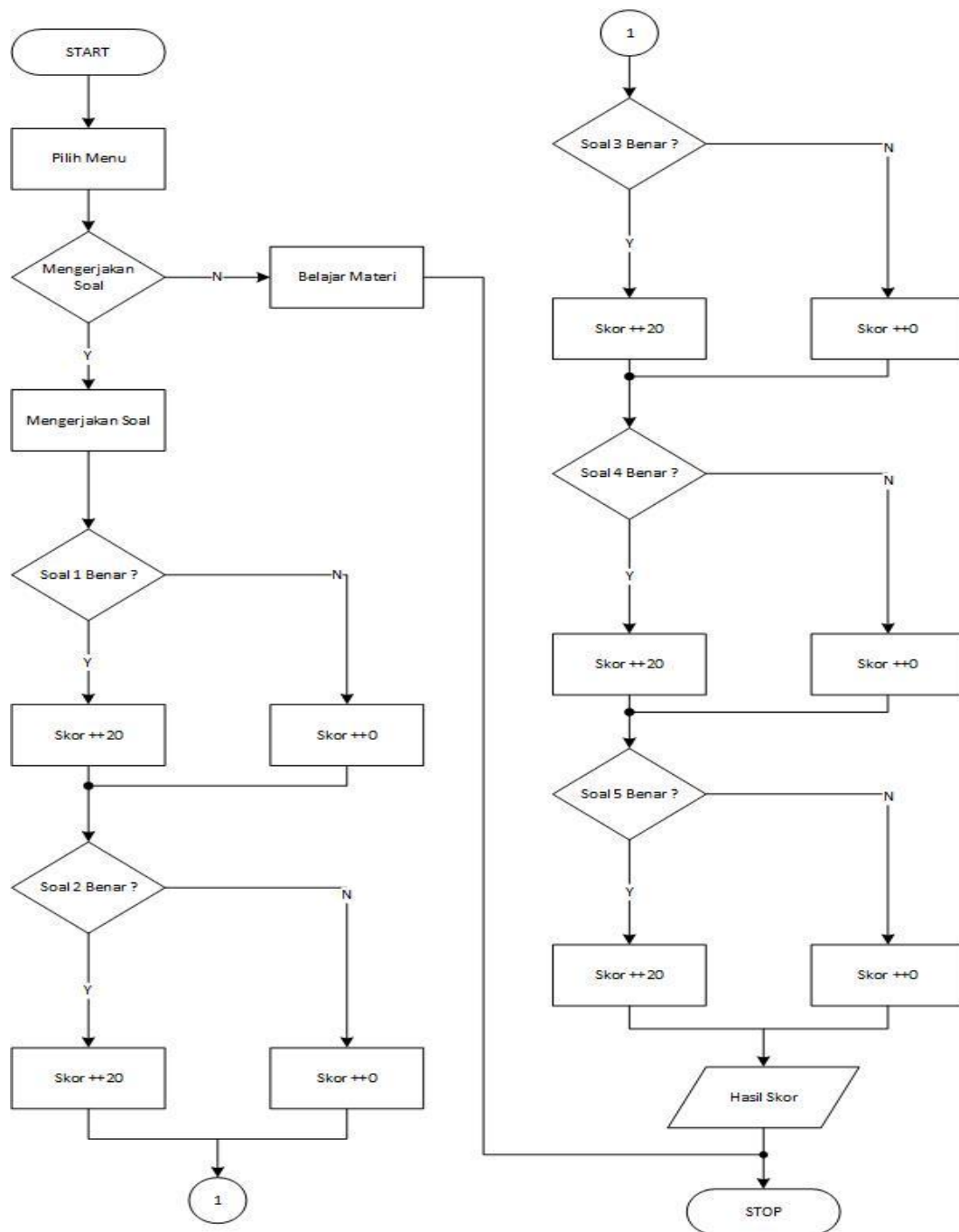
Pada menu *game*, karakter pemain akan menyerang *monster* rokok, *monster* narkoba dan *monster* minuman keras, jika karakter pemain menyerang *monster* dan *monster* tersebut mati maka skor akan bertambah 20 poin. Kemudian jika karakter pemain sudah berhadapan dengan bos *monster* maka karakter pemain harus mengalahkan bos *monster* per arena dan setiap kali serangan yang berhasil mengenai bos *monster* akan diberikan skor 20 poin. Apabila karakter pemain menyentuh *monster* atau terkena serangan oleh *monster* maka nyawa karakter pemain akan berkurang 1 dan jika nyawa karakter pemain sudah habis maka permainan akan berhenti. *Flowchart game* diterangkan pada Gambar 3.1 berikut ini :



Gambar 3.1 Flowchart Game

b. Flowchart Kuis

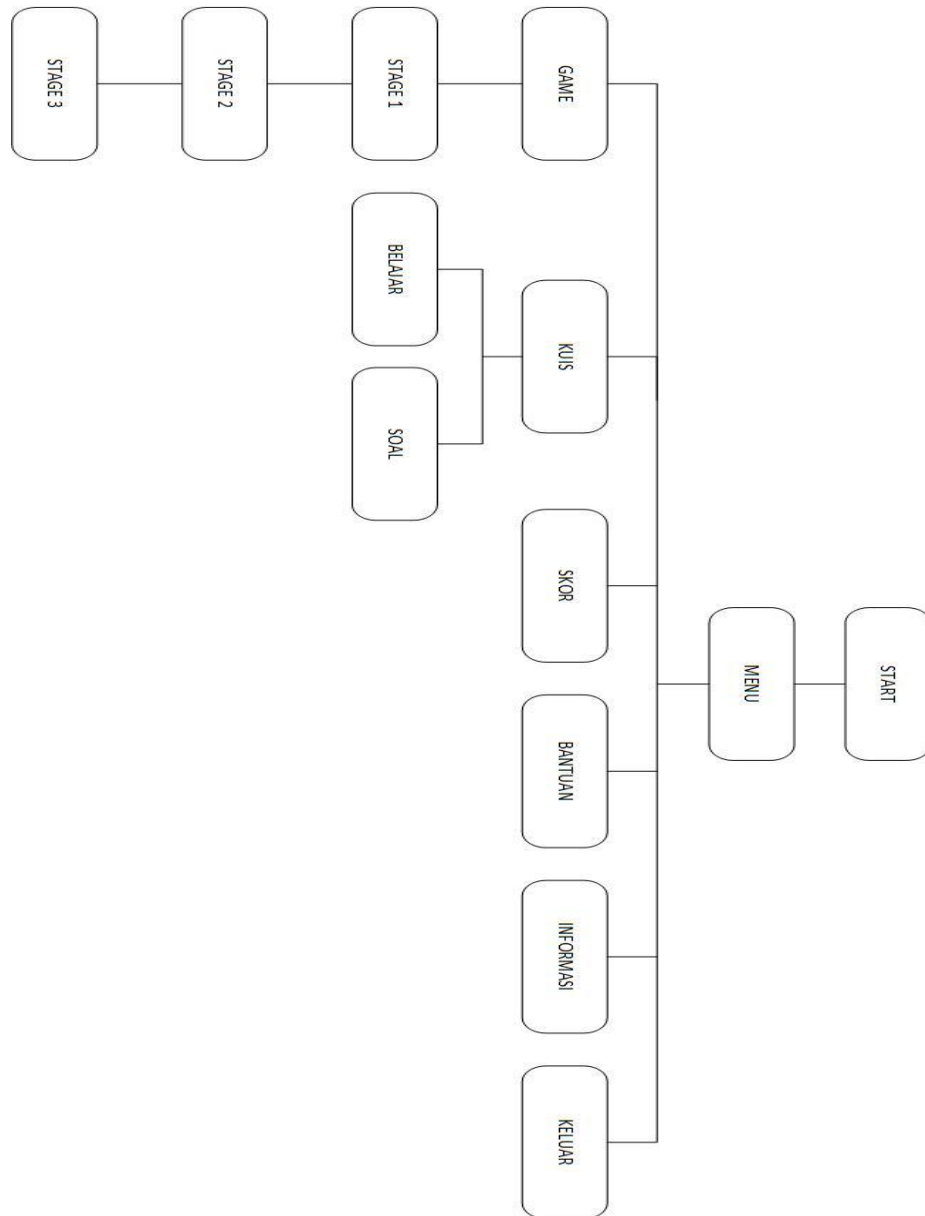
Pada menu kuis ada 2 pilihan menu yaitu menu belajar dan menu soal. Di menu belajar *user* bisa membaca materi mengenai rokok, narkoba dan minuman keras serta zat-zat yang terkandung didalamnya dan efek buruk yang disebabkan oleh ketiga benda tersebut. Di menu soal *user* akan diberikan 10 soal yang berkaitan dengan tema dari *game* dan setiap 1 soal yang benar bernilai 10 poin jika soal yang dikerjakan salah maka nilainya adalah 0 poin. Setelah mengerjakan semua soal kemudian akan muncul nilai hasil dari soal yang telah dikerjakan oleh *user*. *Flowchart* kuis diterangkan pada Gambar 3.2 berikut ini :



Gambar 3.2 Flowchart Kuis

3.2.6 Menu Game

Menu dibuat menggunakan *button* dan men-*trigger* perpindahan posisi perpindahan scene di unity. Berikut ini adalah perancangan menu yang ada dalam Aplikasi *Game Save Your Generation* Berbasis Android yang diterangkan pada Gambar 3.3 :



Gambar 3.3 Perancangan Menu *Game*

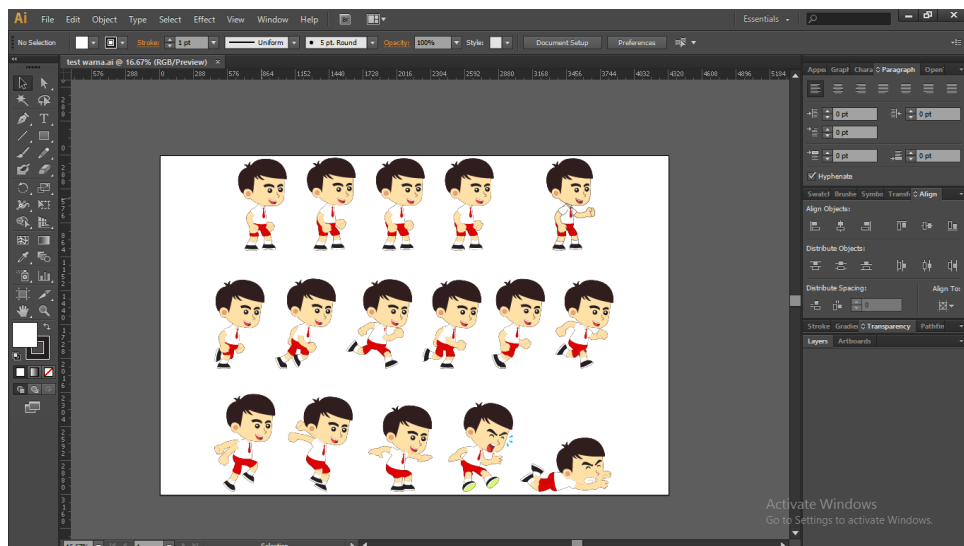
BAB IV. IMPLEMENTASI

Bab ini menjelaskan tentang implementasi dari perancangan yang dilakukan pada bab sebelumnya mengenai Aplikasi *Game Save Your Generation* Berbasis Android.

4.1 Pembuatan Sprite

Sprite adalah tekstur dua dimensi yang digunakan untuk membuat objek didalam game. Pembuatan *sprite* ini menggunakan *software* Adobe Illustrator CS6 dan Adobe Flash CS6. *Sprite* yang dibutuhkan didalam Aplikasi *Game Save Your Generation* Berbasis Android adalah sebagai berikut.

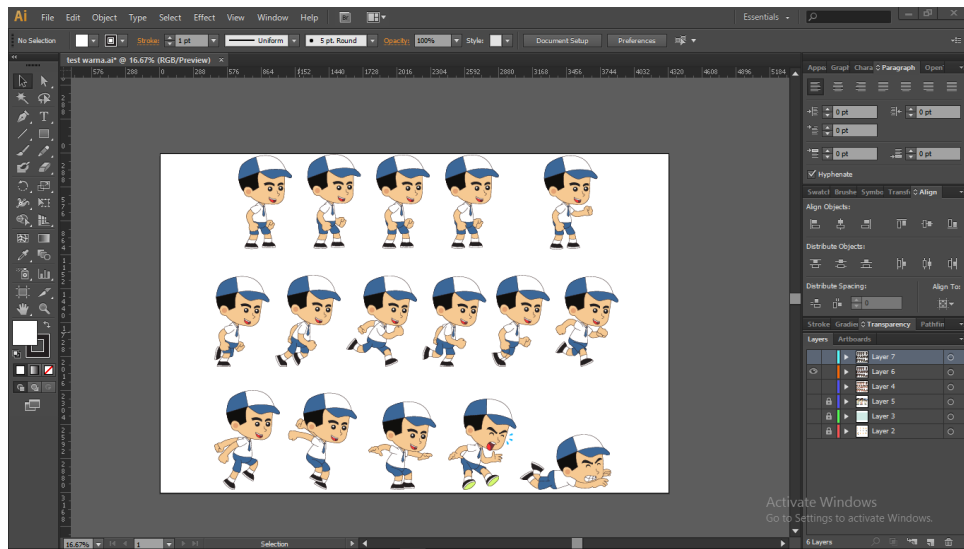
4.1.1 Karakter Pemain SD



Gambar 4.1 *Sprite* Karakter Pemain SD

Gambar 4.1 merupakan pembuatan *sprite* untuk objek karakter pemain SD yang dibuat menggunakan *software* Adobe Illustrator CS6.

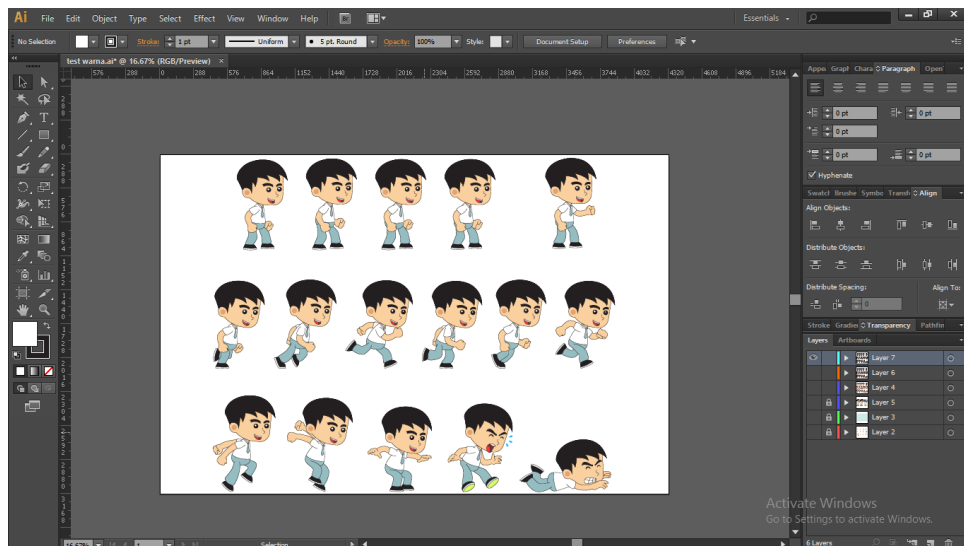
4.1.2 Karakter Pemain SMP



Gambar 4.2 *Sprite* Karakter Pemain SMP

Gambar 4.2 merupakan pembuatan *sprite* untuk objek karakter pemain SMP yang dibuat menggunakan *software* Adobe Illustrator CS6.

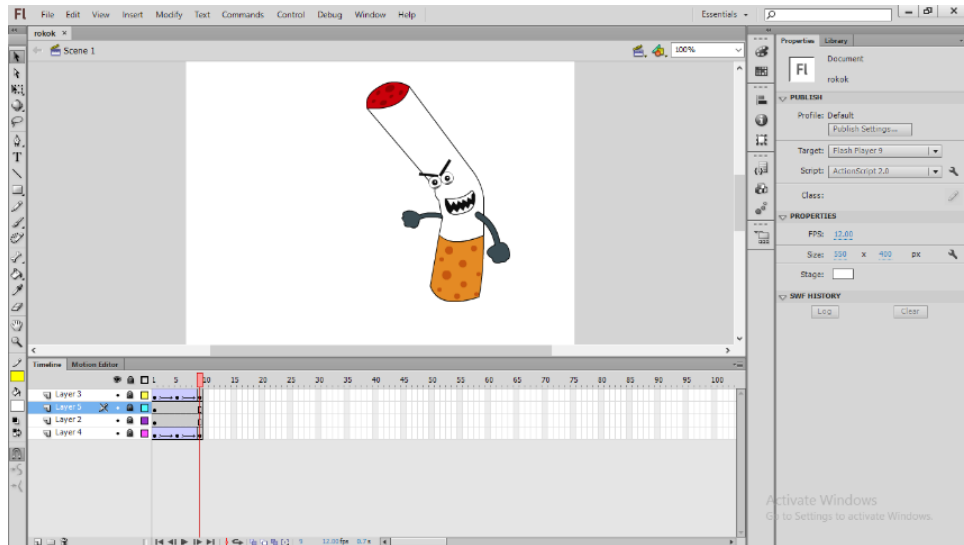
4.1.3 Karakter Pemain SMA



Gambar 4.3 *Sprite* Karakter Pemain SMA

Gambar 4.3 merupakan pembuatan *sprite* untuk objek karakter pemain SMA yang dibuat menggunakan *software* Adobe Illustrator CS6.

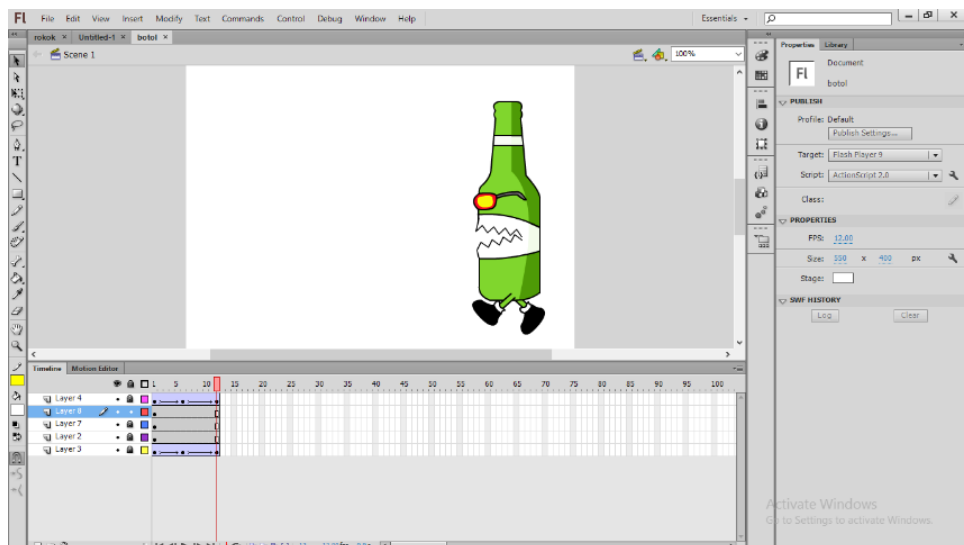
4.1.4 Karakter *Monster Rokok*



Gambar 4.4 *Sprite* Karakter *Monster Rokok*

Gambar 4.4 merupakan pembuatan *sprite* untuk objek karakter *monster* rokok yang dibuat menggunakan *software* Adobe Flash CS6.

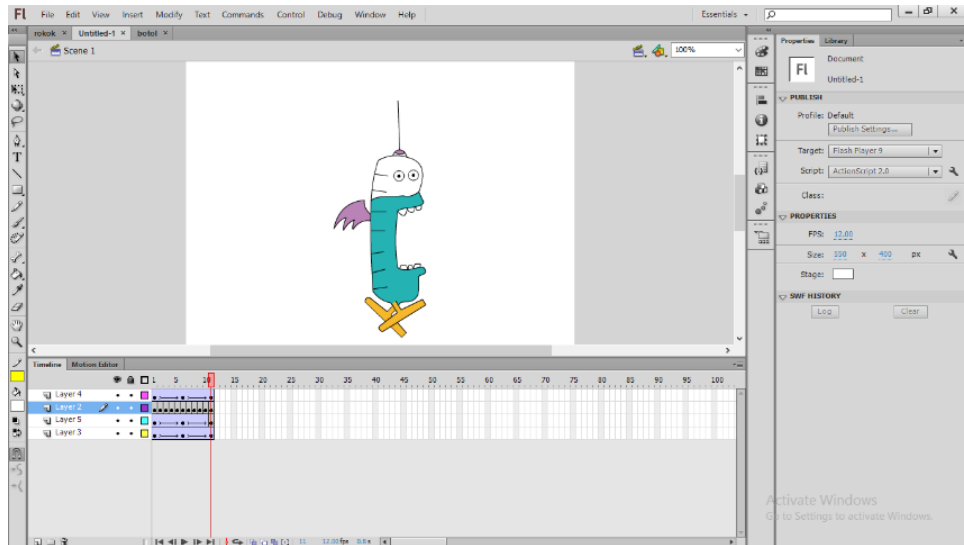
4.1.5 Karakter *Monster Minuman Keras*



Gambar 4.5 *Sprite* Karakter *Monster Minuman Keras*

Gambar 4.5 merupakan pembuatan *sprite* untuk objek karakter *monster* minuman keras yang dibuat menggunakan *software* Adobe Flash CS6.

4.1.6 Karakter *Monster Narkoba*



Gambar 4.6 *Sprite* Karakter *Monster Narkoba*

Gambar 4.6 merupakan pembuatan *sprite* untuk objek karakter *monster narkoba* yang dibuat menggunakan *software* Adobe Flash CS6.

4.2 Pembuatan Aplikasi

Tahapan selanjutnya adalah pembuatan aplikasi dengan menggunakan *software* Unity sebagai perangkat yang mendukung pembuatan Aplikasi *Game Save Your Generation* Berbasis Android.

4.2.1 Menu Utama

Menu Utama merupakan tampilan awal dan tampilan dasar dari Aplikasi *Game Save Your Generation* Berbasis Android. Pada menu utama terdapat pilihan tombol-tombol di menu utama antara lain :

- Tombol Main : tombol untuk memulai permainan.
- Tombol Kuis : tombol untuk menuju menu belajar dan menu soal.
- Tombol Informasi : tombol untuk menuju ke menu halaman informasi tentang pembuat *game*.
- Tombol Bantuan : tombol untuk menuju ke menu halaman bantuan yang berisi informasi tentang cara bermain *game*.
- Tombol Skor Tertinggi : tombol untuk melihat perolehan nilai tertinggi dari *game*.

- Tombol Keluar : tombol untuk keluar dari *game*.

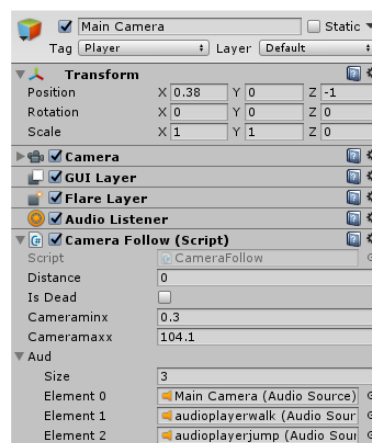


Gambar 4.7 Pembuatan Menu Utama

4.2.2 Arena Satu sampai Arena Tiga

4.2.2.1 Pengaturan *Main Camera*

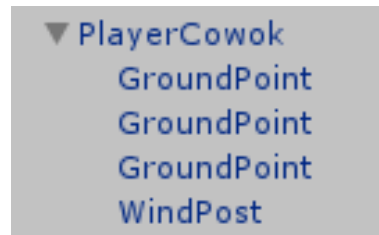
Main Camera atau kamera utama pada arena satu sampai arena tiga adalah kamera *orthographic* yang diatur untuk berada antara karakter pemain, kamera utama juga memiliki *script* *CameraFollow.cs* yang dibutuhkan di setiap arena untuk memfokuskan kamera kepada gerakana dari karakter pemain saat berlangsungnya *game*. Pada Gambar 4.8 merupakan pengaturan *main camera* pada arena 1 sampai dengan arena 3.



Gambar 4.8 Pengaturan *main camera* arena 1 sampai 3

4.2.2.2 Pengaturan Karakter Pemain

Hirarki yang dimiliki oleh objek karakter pemain dapat dilihat pada gambar 4.9.



Gambar 4.9 Hirarki karakter pemain

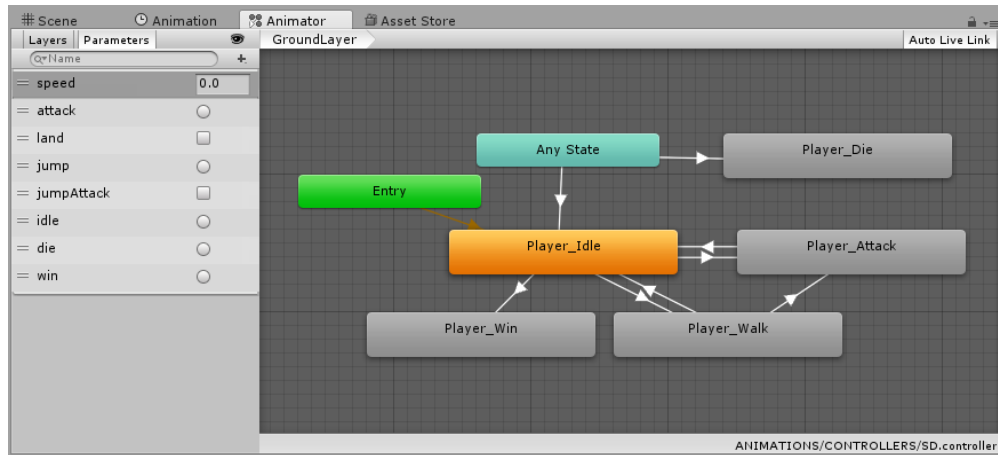
- a. *PlayerCowok* merupakan *GameObject* utama yang berisi *control* karakter, *collider* yang digunakan agar obyek *player* dapat berinteraksi dengan *collider GameObject* lain. *Script PlayerCowok.cs* merupakan *script* yang mengatur animasi dan beberapa kontrol karakter pemain.
- b. *GroundPoint* digunakan untuk mengecek apakah objek *PlayerCowok* sedang berada dalam *ground* (tidak dalam kondisi melompat diudara), jika *PlayerCowok* berada pada *ground*, maka *PlayerCowok* dapat berjalan dan melompat.
- c. *WindPost* digunakan untuk memposisikan munculnya serangan yang dilancarkan oleh obyek *PlayerCowok*.

Objek *PlayerCowok* memiliki *Animator Controller* yang digunakan untuk mengatur animasi dari karakter pemain. Di dalam *Animator Controller* terdapat dua *layers* yaitu *GroundLayer* dan *AirLayer*, di *GroundLayer* merupakan susunan animasi di saat karakter pemain berada di *ground* atau permukaan tanah dan *AirLayer* merupakan susunan animasi di saat karakter pemain berada dalam kondisi melompat. Gambar 4.10 dan gambar 4.11 merupakan animasi control pada karakter pemain dan berikut ini adalah potongan *script* untuk mengatur perpindahan *layers* di *Animator Controller* karakter pemain.

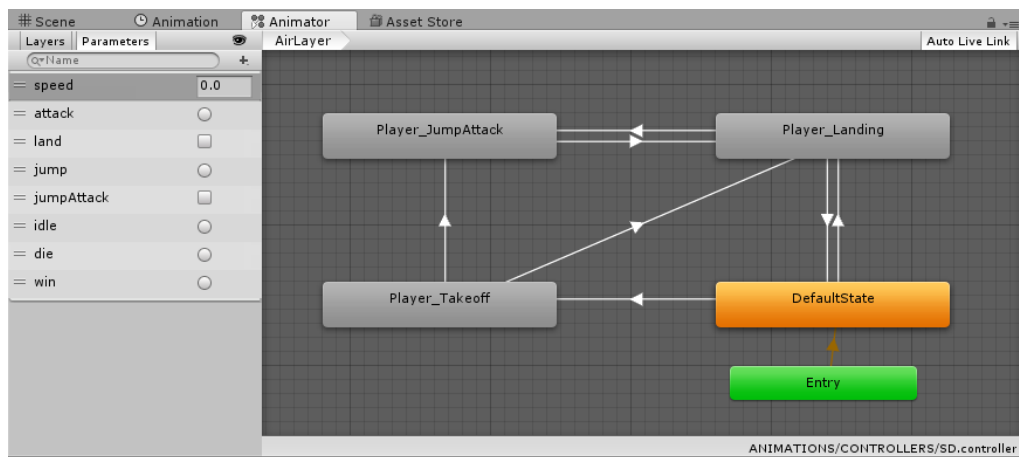
```

private void HandleLayers() {
    if (!OnGround) {
        myAnimator.SetLayerWeight(1, 1);
    } else {
        myAnimator.SetLayerWeight(1, 0);
    }
}

```



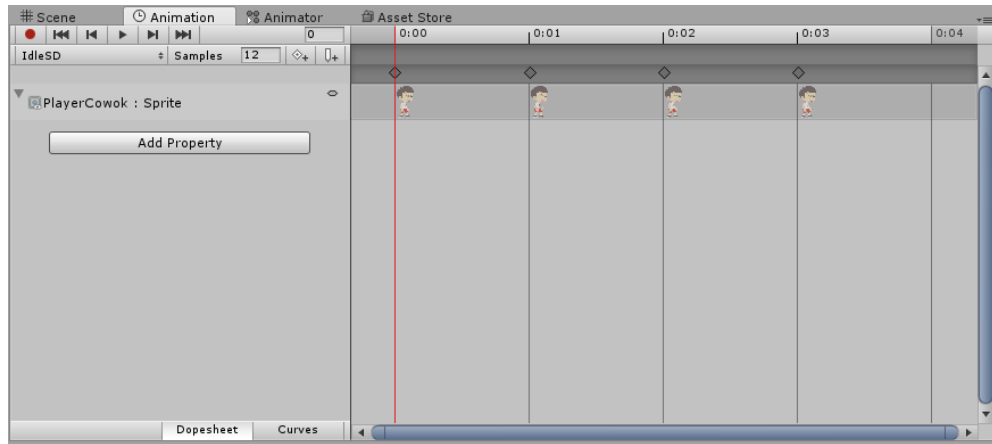
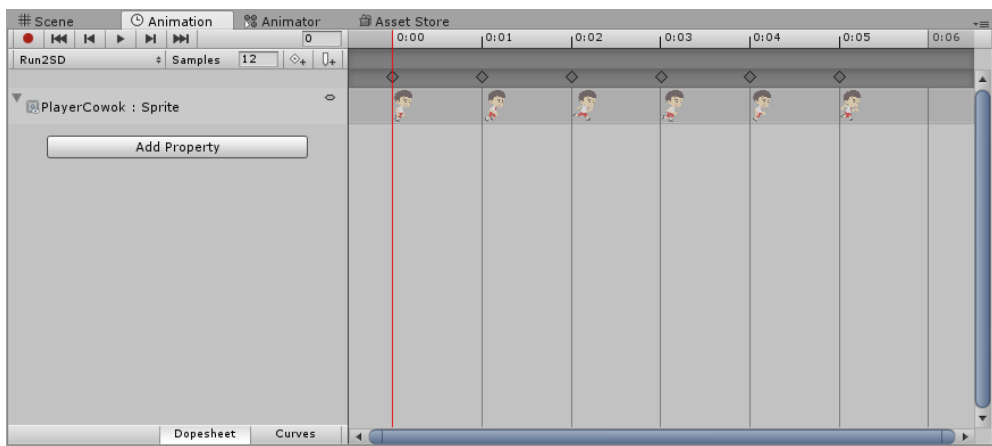
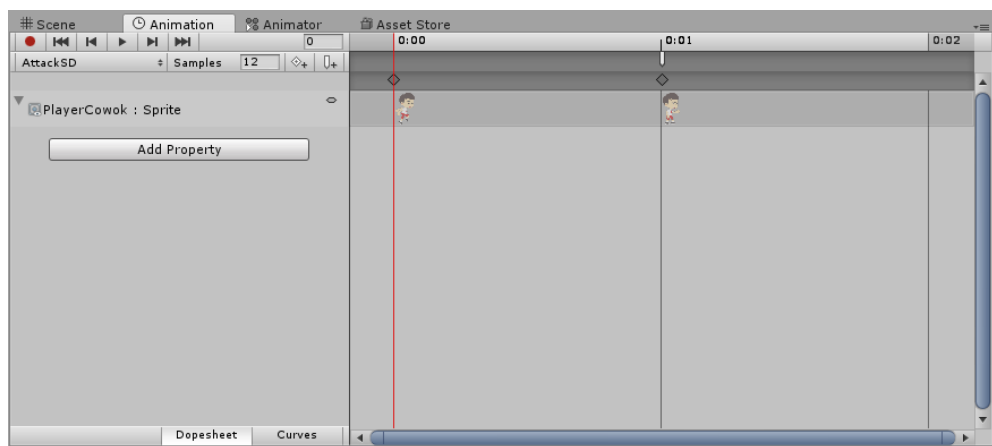
Gambar 4.10 Animator controller karakter pemain saat di ground

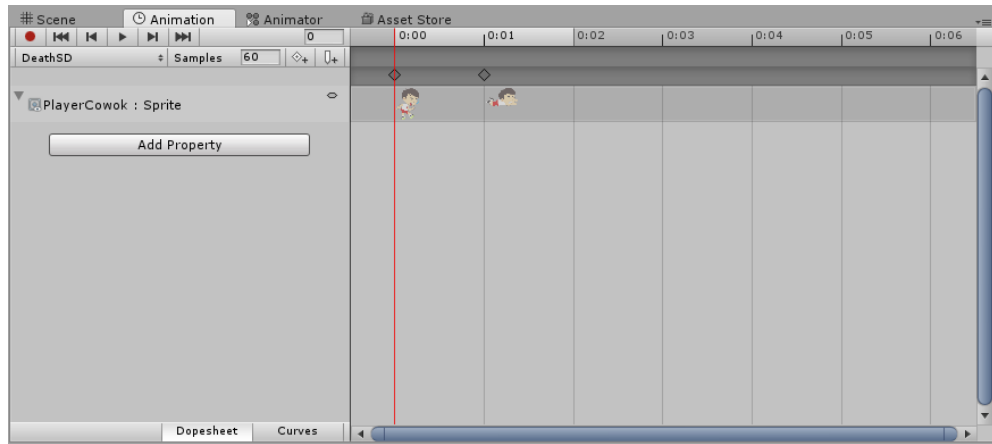


Gambar 4.11 Animator controller karakter pemain saat di jump

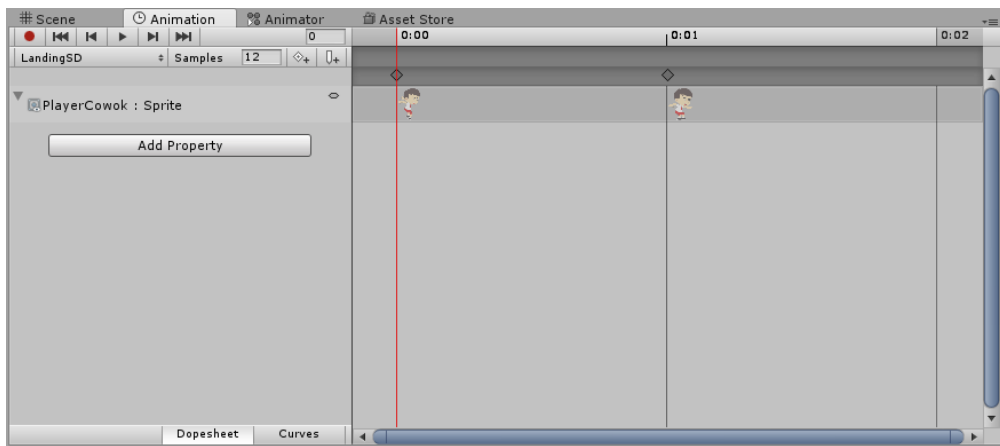
Dikarenakan Aplikasi *Game Save Your Generation* Berbasis Android menggunakan dua dimensi sebagai grafis, maka membutuhkan animasi yang dapat diatur didalam *animation* dengan memasukkan *sprite* yang dibutuhkan. Gambar 4.12 sampai Gambar 4.17 merupakan animasi yang dimiliki oleh karakter pemain dan berikut ini potongan *script* yang digunakan agar karakter pemain mengeluarkan serangan dan kemudian akan ditambahkan kedalam animasi karakter pemain dalam bentuk *event*.

```
public override void AttackWind(int value){
    if(!OnGround && value == 1 || OnGround && value == 1){
        base.AttackWind(value);
    }
}
```

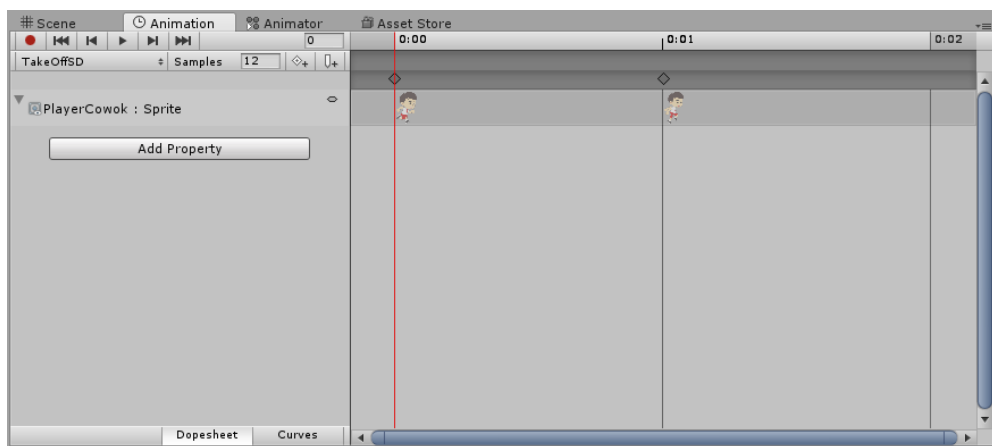
Gambar 4.12 *Animation* karakter pemain *idle*Gambar 4.13 *Animation* karakter pemain *run*Gambar 4.14 *Animation* karakter pemain *attack*



Gambar 4.15 *Animation* karakter pemain *death*



Gambar 4.16 *Animation* karakter pemain *landing*

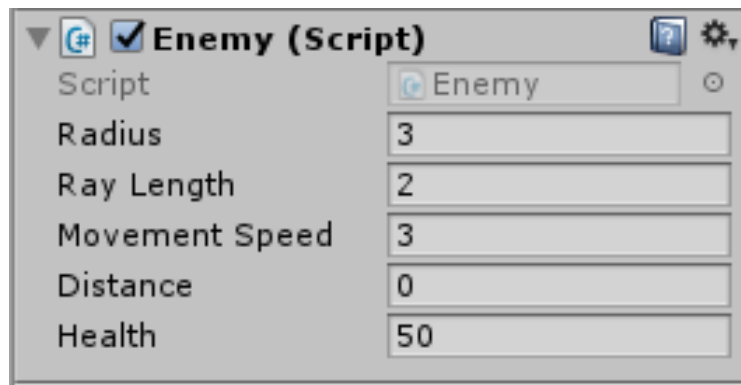


Gambar 4.17 *Animation* karakter pemain *take off*

4.2.2.3 Pengaturan Karakter *Monster Biasa* dan *Monster Attack*

Objek *monster* sama dengan objek karakter pemain, yang membedakan adalah pergerakan *monster* menggunakan algoritma dengan target adalah karakter pemain. Dalam *game monster* dibedakan menjadi 2 kategori yaitu *monster biasa* dan *monster attack*.

Monster biasa adalah *monster* yang tidak dapat mengeluarkan serangan seperti asap dan cairan, *monster* ini hanya dapat berjalan-jalan dengan radius area yang telah ditentukan, dan jika dalam radius tertentu karakter pemain tertangkap oleh radius area *monster* maka karakter pemain akan diikuti oleh *monster* jenis ini. Objek *monster* ini memiliki *script* *Monster.cs* yang mengatur animasi dan kontrol *monster*, berikut pada gambar 4.18 merupakan pengaturan *monster biasa*.



Gambar 4.18 Pengaturan *monster biasa*

Berikut merupakan potongan *script* *Enemy.cs*, dimana potongan *script* ini digunakan untuk menentukan radius wilayah *enemy* dapat patrol dan panjang radius *enemy* dapat mendeteksi keberadaan *player*.

```
void Detection() {

    RaycastHit2D isForward =
    Physics2D.Raycast(transform.position, Vector2.right,
    rayLength, 1 << LayerMask.NameToLayer("Player"));

    RaycastHit2D isBackward =
    Physics2D.Raycast(transform.position, Vector2.left,
    rayLength, 1 << LayerMask.NameToLayer("Player"));

    if(isForward) {
        Debug.Log("player di depan");
        transform.rotation = Quaternion.Euler(new Vector3(0,-
        180,0));
        dir=1;
        isDetect = true;
    }
}
```

```

    if(isBackward) {
        Debug.Log("player di belakang");
        transform.rotation = Quaternion.Euler(new
        Vector3(0,0,0));
        dir = -1;
        isDetect = true;
    }

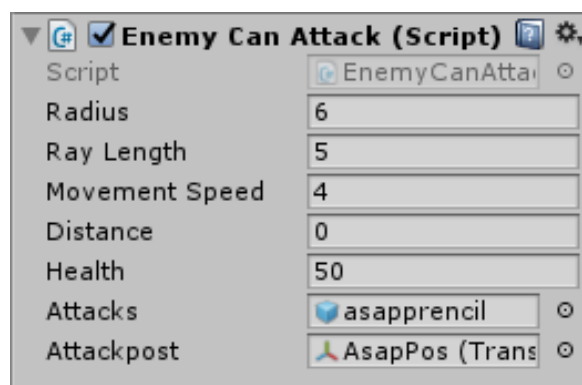
    Debug.DrawRay(transform.position, Vector2.right *
    rayLength, Color.blue);

    Debug.DrawRay(transform.position, Vector2.left *
    rayLength, Color.blue);
}

void OnDrawGizmos(){
    Gizmos.color = new Color(1f,0f,0f,0.6f);
    if(isStart){
        Gizmos.DrawCube(pos, new Vector3(2*radius, 1f,
        0f));
    }else{
        Gizmos.DrawCube(new Vector3(transform.position.x, -
        2.5f, 0f), new Vector3(2*radius, 1f, 0f));
    }
}
}

```

Monster attack adalah *monster* yang dapat mengeluarkan serangan seperti kuman, *monster* ini sama seperti *monster* biasa, dimana *monster* ini hanya dapat berjalan-jalan dengan radius area yang telah ditentukan, dan jika dalam radius tertentu *player* tertangkap oleh radius area *monster* maka *monster* akan mengeluarkan serangannya kepada karakter pemain. Obyek *monster* ini memiliki *script* *MonsterCanAttack.cs* yang mengatur animasi dan kontrol *monster*, berikut pada gambar 4.19 merupakan pengaturan *monster attack*.



Gambar 4.19 Pengaturan *monster attack*

Monster jenis ini memiliki indikator *attack post*, dimana *attack post* ini digunakan untuk menentukan posisi serangan *monster* kepada karakter pemain, berikut pada gambar 4.20 yang merupakan salah satu hirarki *monster attack*.



Gambar 4.20 Hirarki *monster attack*

Dalam *script* *EnemyCanAttack.cs*, dimana terdapat fungsi di *script* ini digunakan agar *monster* dapat mengeluarkan serangannya kepada karakter pemain, fungsi serang di *script* ini akan ditambahkan kedalam animasi *monster* dalam bentuk *event* yang digambarkan pada gambar 4.21.

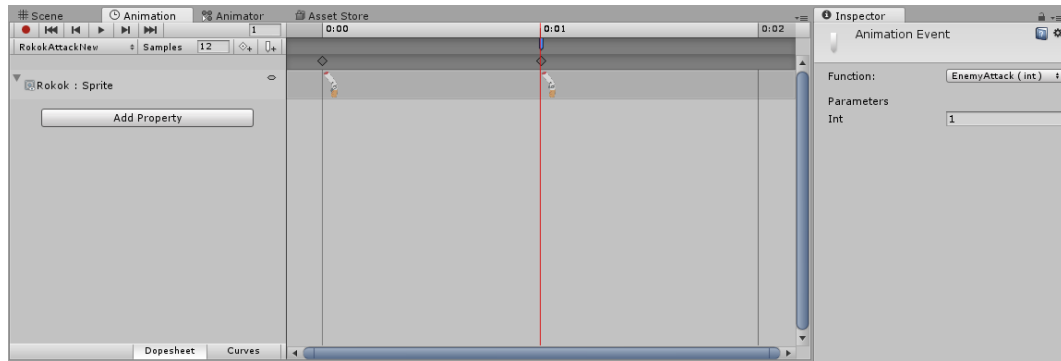
```
public void EnemyAttack(int value){
    GameObject tmp = null;

    if(dir > 0f){
        tmp = (GameObject)Instantiate(attacks, attackpost.position,
            Quaternion.Euler(new Vector3(0,0,180)));

        tmp.GetComponent<Wind>().Initialize(Vector2.right);
    }else{
        tmp = (GameObject)Instantiate(attacks, attackpost.position,
            Quaternion.Euler(new Vector3(0,0,0)));

        tmp.GetComponent<Wind>().Initialize(Vector2.left);
    }

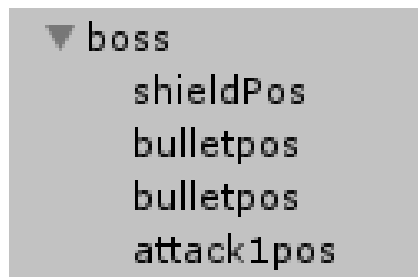
    if(tmp != null) {
        Destroy(tmp, 1f);
    }
}
```



Gambar 4.21 Penempatan *event function EnemyAttack*

4.2.2.4 Pengaturan Karakter Bos Arena

Bos Arena adalah *monster* paling besar yang menguasai setiap arena dan muncul akhir arena. Serangan yang dikeluarkan oleh setiap bos arena yang keluar berbeda-beda seperti asap rokok, cairan minuman keras dan cairan narkoba. Namun semua bos arena dapat mengeluarkan atau melakukan pertahanan seperti tameng. Objek bos arena ini memiliki *script* BosArena1.cs, BosArena2.cs, dan BosArena3.cs. *Script* ini digunakan untuk mengatur animasi dan kontrol bos arena, berikut gambar 4.22 mengenai hirarki bos arena.



Gambar 4.22 Hirarki bos arena

Gambar 4.22 merupakan hirarki dari Bos arena dan berikut keterangan hirarki tersebut.

- a. *boss* merupakan *GameObject* utama yang berisi kontrol karakter, *collider* yang digunakan agar obyek *boss* dapat berinteraksi dengan *collider GameObject* lain. *Script* BosArena1.cs, BosArena2.cs, dan BosArena3.cs merupakan *script* yang mengatur animasi dan beberapa kontrol karakter pemain.
- b. *bulletpos* digunakan memposisikan munculnya serangan yang dilancarkan oleh objek *boss*.

c. *attack1pos* digunakan untuk memposisikan munculnya serangan kedua yang dilancarkan oleh objek *boss*.

Berikut merupakan potongan BOS_Arena1.cs, dimana potongan *script* ini digunakan agar bos arena

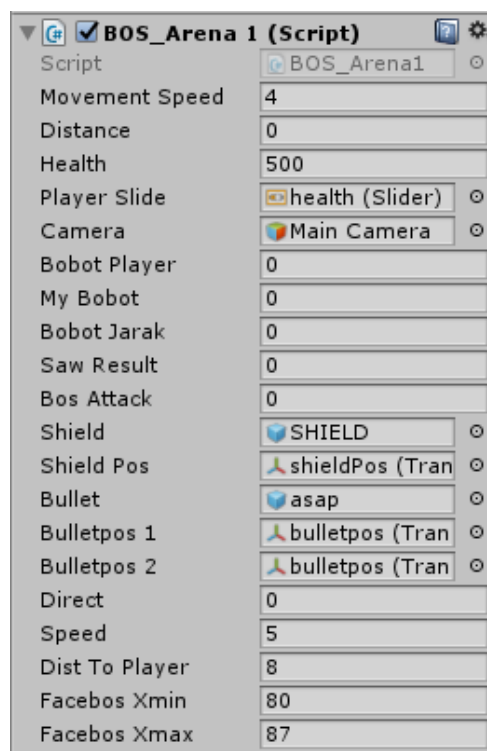
```
void move() {
    if(tempDir != direct){

        transform.Translate(Vector2.left * -1* speed
        *Time.deltaTime);

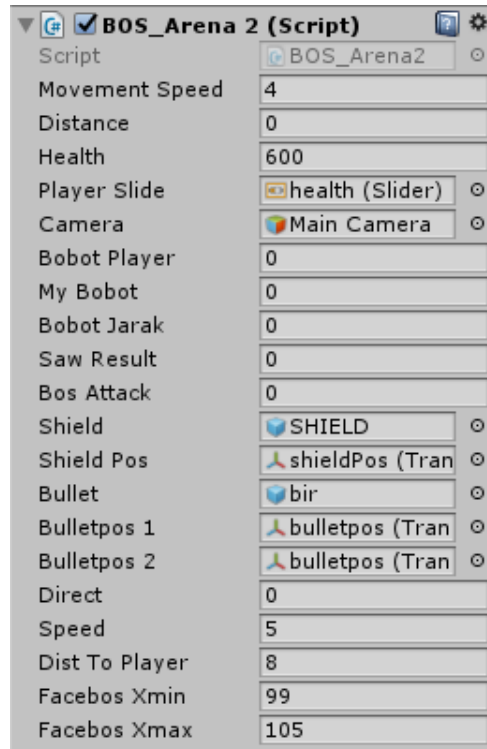
        transform.position = new
        Vector2(Mathf.Clamp(transform.position.x, facebosXmin, fa
        cebosXmax), transform.position.y);

        myAnimator.SetBool("walk",true);

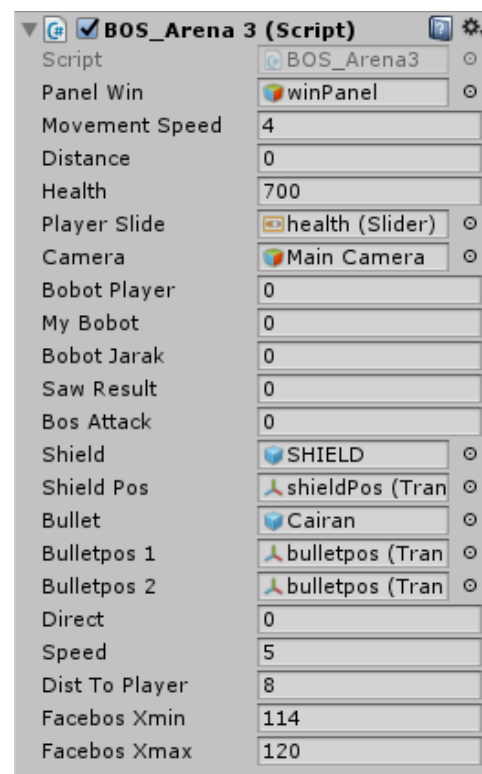
        if(transform.position.x <=facebosXmin ||
        transform.position.x >=facebosXmax)
        {
            tempDir = direct;
            myAnimator.SetBool("walk",false);
        }
    }
}
```



Gambar 4.23 Pengaturan bos arena 1



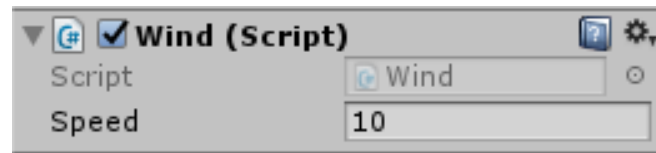
Gambar 4.24 Pengaturan bos arena 2



Gambar 4.25 Pengaturan bos arena 3

4.2.2.5 Pengaturan Serangan

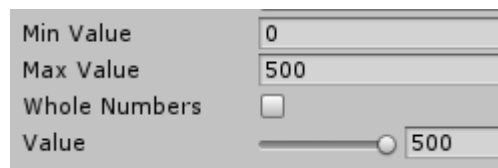
Obyek serangan pada *game* ini dimiliki oleh karakter pemain, bos arena, dan *monster*, dimana semua objek serangan ini menggunakan *script* yaitu *Wind.cs*, *script* ini digunakan untuk mengontrol kecepatan serangan.berikut pada gambar 4.26 merupakan pengaturan kecepatan serangan untuk karakter pemain dan *monster*.



Gambar 4.26 Pengaturan serangan

4.2.2.6 Pengaturan *Health Bar*

Health Bar pada karakter pemain *game* ini menggunakan UI (*user interface*) dengan jenis *slider*. Dimana pengaturan *slider* untuk darah karakter pemain ini maksimal adalah 500 dan minimal adalah 0. Pada gambar 4.27 merupakan pengaturan *Health Bar* pada karakter pemain.



Gambar 4.27 Pengaturan *health bar* karakter pemain

BAB V. PENGUJIAN DAN PEMBAHASAN

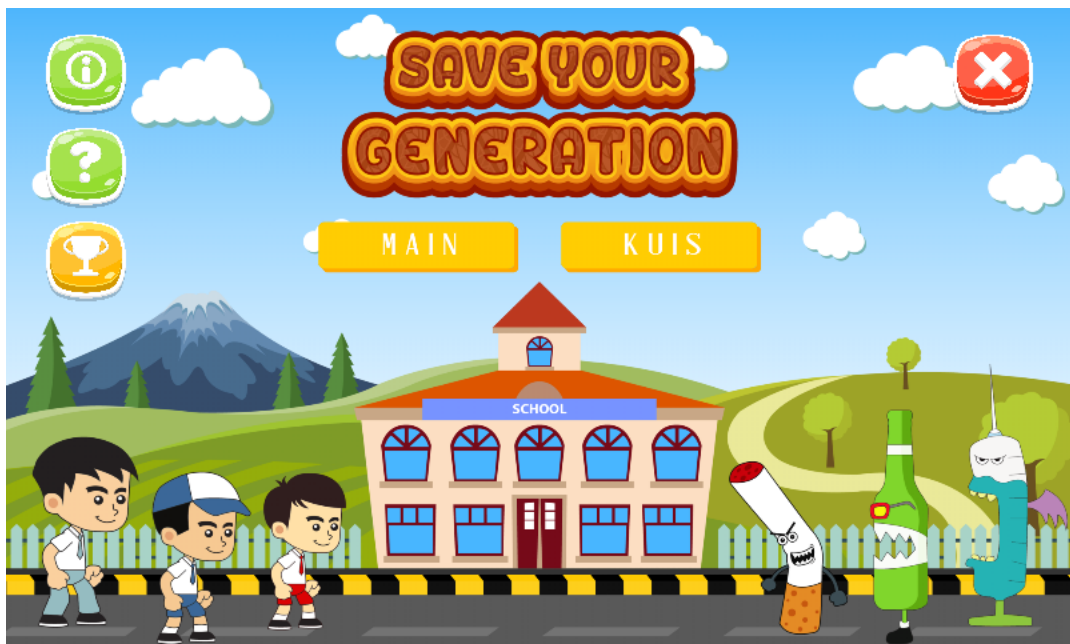
Bab ini membahas tentang proses pengujian Aplikasi *Game Save Your Generation* Berbasis Android yang sudah dibuat dengan menggunakan *Unity Game Engine* dan dengan Bahasa pemrograman *C#*.

5.1 Hasil Perancangan

Tampilan halaman utama dari aplikasi ini akan dijelaskan pada Gambar 5.1, pada menu utama terdapat tombol main, tombol kuis, tombol informasi, tombol bantuan, tombol skor tertinggi, dan tombol keluar serta logo karakter game *Save Your Generation*.

5.1.1 Halaman Menu Utama

Merupakan tampilan halaman utama dari aplikasi ini yang terdapat beberapa tombol yang tersedia dan logo karakter game *Save Your Generation*. Adapun tombol yang ada yaitu 6 tombol yang terdiri dari tombol Main, Kuis, Informasi, Bantuan, Skor Tertinggi dan Keluar.



Gambar 5.1 Tampilan Menu Utama Game

5.1.2 Halaman Informasi

Tampilan dari halaman menu informasi akan dijelaskan pada Gambar 5.2, yaitu tentang sekilas biodata pembuat *game Save Your Generation* berbasis Android.



Gambar 5.2 Tampilan Menu Informasi

5.1.3 Halaman Bantuan

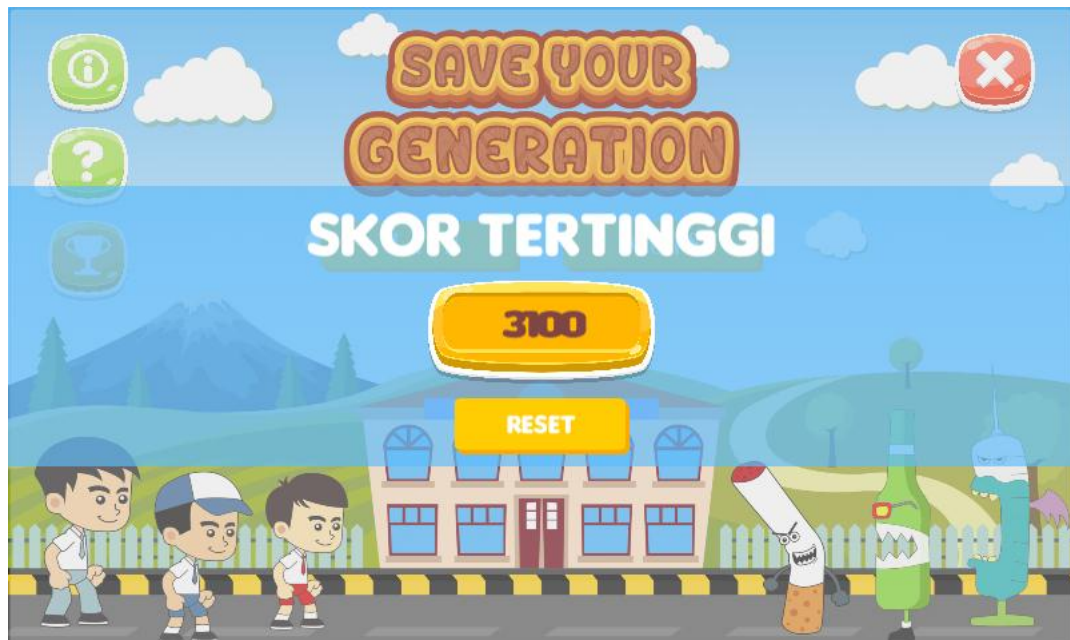
Tampilan halaman menu bantuan dijelaskan pada Gambar 5.3, pada halaman ini dijelaskan bagaimana cara menggerakkan karakter untuk melewati setiap rintangan yang ada dengan menggunakan cursor tombol 'kiri' dan 'kanan' untuk berjalan dan tombol 'atas' untuk melompat. Kemudian untuk tombol 'tangan' untuk memunculkan serangan dan gambar 'hati' yaitu untuk menambahkan darah dari *player*.



Gambar 5.3 Tampilan Menu Bantuan

5.1.4 Halaman Skor Tertinggi

Tampilan halaman menu skor tertinggi akan dijelaskan pada Gambar 5.4, yang menampilkan skor tertinggi dari permainan yang sudah pernah dilakukan sebelumnya. Kemudian di halaman ini terdapat tombol reset yang berfungsi untuk merubah nilai menjadi 0.



Gambar 5.4 Tampilan Menu Skor Tertinggi

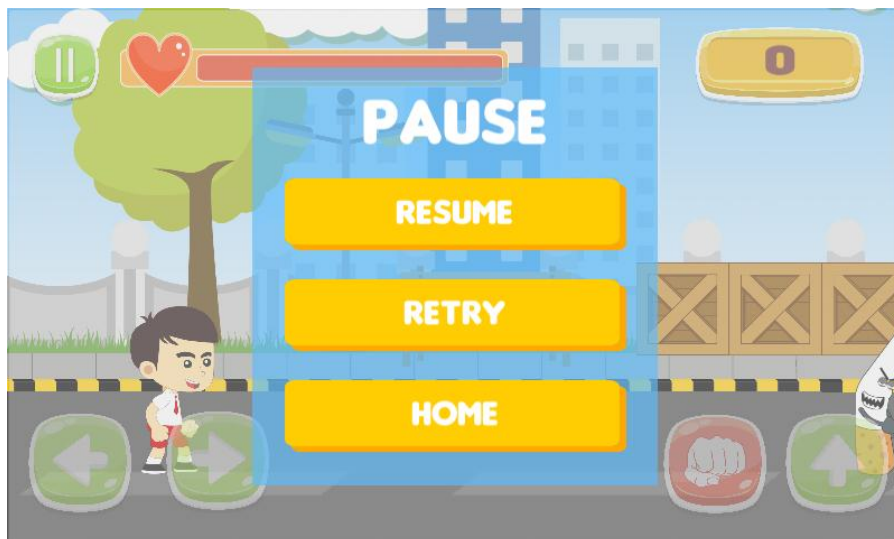
5.1.5 Halaman Arena 1

Yaitu tampilan ketika *game* pada arena satu dijalankan, disini terdapat tombol untuk menggerakkan karakter anak sd dan juga tombol untuk menyerang lawan monster rokok. Ada tombol *pause* yang digunakan menghentikan *game* sejenak juga akan menampilkan tombol untuk kembali ke beranda, melanjutkan permainan, ataupun mengulangi *game* dari awal.



Gambar 5.5 Tampilan Arena 1

Tampilan halaman tombol *pause* ditunjukkan pada Gambar 5.6. Ketika tombol *pause* di klik maka permainan game akan berhenti sementara. Pada halaman ini terdapat tombol resume, tombol retry, dan tombol home.



Gambar 5.6 Tampilan Pause

Tampilan halaman akhir game arena 1 akan dijelaskan pada Gambar 5.7. Ketika karakter sudah berada di ujung akhir arena 1 maka karakter akan bertemu dengan bos, kemudian akan muncul tampilan percakapan antara karakter dengan bos monster rokok.



Gambar 5.7 Tampilan Percakapan Karakter dan Bos Monster Rokok

Setelah percakapan antara karakter dan bos monster rokok selesai, selanjutnya karakter akan menyerang bos monster rokok seperti yang dijelaskan pada Gambar 5.8 sebagai berikut



Gambar 5.8 Tampilan Karakter Menghadapi Bos Monster Rokok

Dan apabila karakter kalah dengan bos monster rokok maka akan muncul tampilan panel seperti Gambar 5.9, di halaman ini terdapat tombol retry untuk mengulangi *game* dari awal dan tombol home untuk menuju ke menu utama. Namun apabila karakter menang, maka karakter akan melanjutkan ke arena 2.



Gambar 5.9 Tampilan Karakter Kalah

5.1.6 Halaman Arena 2

Tampilan halaman game arena 2 akan dijelaskan pada Gambar 5.10. Di arena ini karakternya adalah pelajar SMP. Pada arena ini karakter akan berhadapan dengan monster rokok dan monster botol minuman keras.



Gambar 5.10 Tampilan Arena 2

Tampilan halaman akhir game arena 2 akan dijelaskan pada Gambar 5.11. Ketika karakter sudah berada di ujung akhir arena 2, maka karakter akan bertemu dengan bos monster botol minuman keras, kemudian akan muncul tampilan percakapan antara karakter dengan bos monster botol minuman keras.



Gambar 5.11 Tampilan Percakapan Karakter Bos Monster Minuman Keras

Setelah percakapan antara karakter dan bos monster botol minuman keras selesai, selanjutnya karakter akan menyerang bos monster botol minuman keras seperti yang dijelaskan pada Gambar 5.12. Jika karakter menang, maka karakter akan melanjutkan ke arena 3.



Gambar 5.12 Tampilan Karakter Menghadapi Bos Monster Minuman Keras

5.1.7 Halaman Arena 3

Tampilan halaman game arena 3 akan dijelaskan pada Gambar 5.13. Di arena ini karakternya adalah pelajar SMA. Pada arena ini karakter akan berhadapan dengan monster rokok, monster botol minuman keras, dan monster suntikan narkoba.



Gambar 5.13 Tampilan Arena 3

Tampilan halaman akhir game arena 3 akan dijelaskan pada Gambar 5.14. Ketika karakter sudah berada di ujung akhir arena 3, maka karakter akan bertemu dengan bos monster suntikan narkoba, kemudian akan muncul tampilan percakapan antara karakter dengan bos monster suntikan narkoba.



Gambar 5.14 Tampilan Percakapan Karakter Bos Monster Narkoba

Setelah percakapan antara karakter dengan bos monster suntikan narkoba sudah selesai maka karakter akan melawan bos monster suntikan narkoba seperti pada Gambar 5.15 berikut ini.



Gambar 5.15 Tampilan Karakter Menghadapi Bos Monster Narkoba

Jika karakter berhasil mengalahkan bos monster suntikan narkoba pada arena 3, maka akan muncul pesan panel seperti pada Gambar 5.16. Player dapat meng-klik tombol home untuk kembali ke menu utama.



Gambar 5.16 Tampilan Karakter Menang

5.1.8 Halaman Kuis

Tampilan halaman menu kuis akan dijelaskan pada Gambar 5.17. Pada halaman ini terdapat sub menu yaitu Belajar dan Soal.



Gambar 5.17 Tampilan Halaman Kuis

5.1.9 Halaman Belajar

Tampilan halaman sub menu belajar akan dijelaskan pada Gambar 5.18, di sub menu ini *player* dapat memilih ingin membaca materi tentang rokok, narkoba, atau minuman keras dengan cara mengklik tombol Baca.



Gambar 5.18 Tampilan Halaman Belajar

5.1.10 Halaman Belajar Rokok

Pada Gambar 5.19 akan dijelaskan tampilan apabila player ingin membaca materi tentang pengertian dan bahaya rokok bagi tubuh. Player dapat meng-*scroll* keatas dan kebawah menggunakan scroll bar yang terletak pada bagian kanan atas.



Gambar 5.19 Tampilan Halaman Belajar Rokok

5.1.11 Halaman Belajar Minuman Keras

Pada Gambar 5.20 akan dijelaskan tampilan apabila player ingin membaca materi tentang pengertian dan bahaya minuman keras bagi tubuh. Player dapat meng-*scroll* keatas dan kebawah menggunakan scroll bar yang terletak pada bagian kanan atas.



Gambar 5.20 Tampilan Halaman Belajar Minuman Keras

5.1.12 Halaman Belajar Narkoba

Pada Gambar 5.21 akan dijelaskan tampilan apabila player ingin membaca materi tentang pengertian dan bahaya narkoba bagi tubuh. Player dapat meng-*scroll* keatas dan kebawah menggunakan scroll bar yang terletak pada bagian kanan atas.



Gambar 5.21 Tampilan Halaman Belajar Narkoba

5.1.13 Halaman Soal

Merupakan halaman soal yang muncul ketika menekan tombol 'Soal', di halaman ini terdapat 10 soal yang bisa dikerjakan dan ada waktu untuk mengerjakan soal. Jika menjawab jawaban yang benar maka skor bertambah 10 poin jika salah maka tidak mendapatkan nilai kemudian akan lanjut soal berikutnya.



Gambar 5.22 Tampilan Halaman Soal

5.1.14 Halaman Nilai Soal

Merupakan tampilan panel nilai akhir yang memunculkan nilai akhir sesudah menjawab semua soal yang diberikan. Terdapat tombol Kembali untuk kembali ke halaman kuis.



Gambar 5.23 Tampilan Halaman Nilai Soal

5.2 Pengujian

Proses pengujian Aplikasi *Game Save Your Generation* Berbasis Android dilakukan untuk mengetahui hasil dari perancangan sistem yang telah dibuat untuk menemukan kekurangan pada sistem, sehingga dapat diketahui apakah perangkat lunak tersebut telah memenuhi kriteria sesuai dengan tujuan atau tidak.

5.2.1 Spesifikasi Perangkat Uji Coba

Proses uji coba dilakukan pada anak usia sekolah dasar. Proses ini membutuhkan spesifikasi perangkat android dengan spesifikasi sebagai berikut :

Tabel 5.1 Spesifikasi perangkat yang dibutuhkan

Keterangan	Spesifikasi
Processor	Intel Core i5 2.50 GHz.
Harddisk	HDD 750 GB.
RAM	RAM 4 Gb.
Smartphone	Min Android KitKat 4.4

5.3 Pengujian Fungsional

Uji coba dilakukan pada semua menu yang ada dalam Aplikasi *Game Save Your Generation* Berbasis Android.

5.3.1 Uji Coba Sistem *Game*

Dengan media perangkat uji coba seperti yang telah dijelaskan sebelumnya, uji coba dilakukan pada semua menu yang ada dalam halaman admin. Tahap-tahap yang diuji cobakan secara fungsional dijabarkan didalam tabel berikut :

Tabel 5.2 Daftar perangkat uji coba

NO	Device	Keterangan
1	Nama Perangkat : Samsung Galaxy Ace 2 Ukuran Display : 480 x 800 pixels Processor : 800 MHz Ram : 768 gb Android : 4.4.1	- Berjalan Baik - Terinstall
2	Nama Perangkat : Smartfren Andromax A Ukuran Display : 480 x 854 pixels Processor : 1.1 GHz Ram : 1 gb Android : 5.1.1	- Berjalan Baik - Terinstall

Tabel 5.3 Uji coba sistem *game*

NO	Fitur	Hasil	
		Berhasil	Belum
1	Membuka Menu Awal	√	
2	Membuka Halaman Informasi	√	
3	Membuka Halaman Bantuan	√	
4	Memilih Halaman Skor Tertinggi	√	
5	Membuka Menu <i>Exit</i>	√	
6	Memilih Menu Belajar	√	

7	Fungsi <i>Scrollbar</i>	√	
8	Memilih <i>Button</i> Jawaban Soal	√	
9	Skor Hasil Menggerjakan Soal	√	
10	Membuka Halaman <i>Game</i>	√	
11	Menggunakan <i>Button</i> Arah dan Menyerang pada <i>Game</i>	√	
12	Menggunakan <i>Button Pause</i>	√	
13	Skor Hasil Menyerang Musuh	√	
14	Nyawa Karakter	√	
15	Menggunakan <i>Button</i> Ya atau Tidak untuk keluar dari <i>game</i>	√	

5.4 Uji Coba User

Uji coba dilakukan untuk mengukur tingkat keberhasilan sistem aplikasi yang telah dibuat. Uji coba dilakukan dengan melibatkan calon *user* dari aplikasi ini secara langsung. Dalam proses uji coba ini juga disertakan kuisisioner yang diberikan kepada *user* sebagai salah satu parameter pengukuran keberhasilan adalah menu yang berjalan sesuai dengan fungsinya, tampilan *game* dan pembuatan laporan yang sesuai. Uji coba pada Aplikasi *Game Save Your Generation* Berbasis Android dilaksanakan pada tanggal 8 Juli 2017 dengan jumlah kuisisioner sebanyak 15 kuisisioner dan mengambil sampel acak.

Pengujian aplikasi dijabarkan dalam kuisisioner sebagai berikut :

Tabel 5.4 Kuisisioner pengguna *game*

No	Pertanyaan	Tanggapan			
		Sangat Setuju	Setuju	Cukup	Tidak Setuju
1	Game berjalan baik pada perangkat				
2	Game ini mudah digunakan				
3	Komposisi warna, ukuran tulisan, dan gambar sudah bagus				
4	Menu Materi pada game sudah				

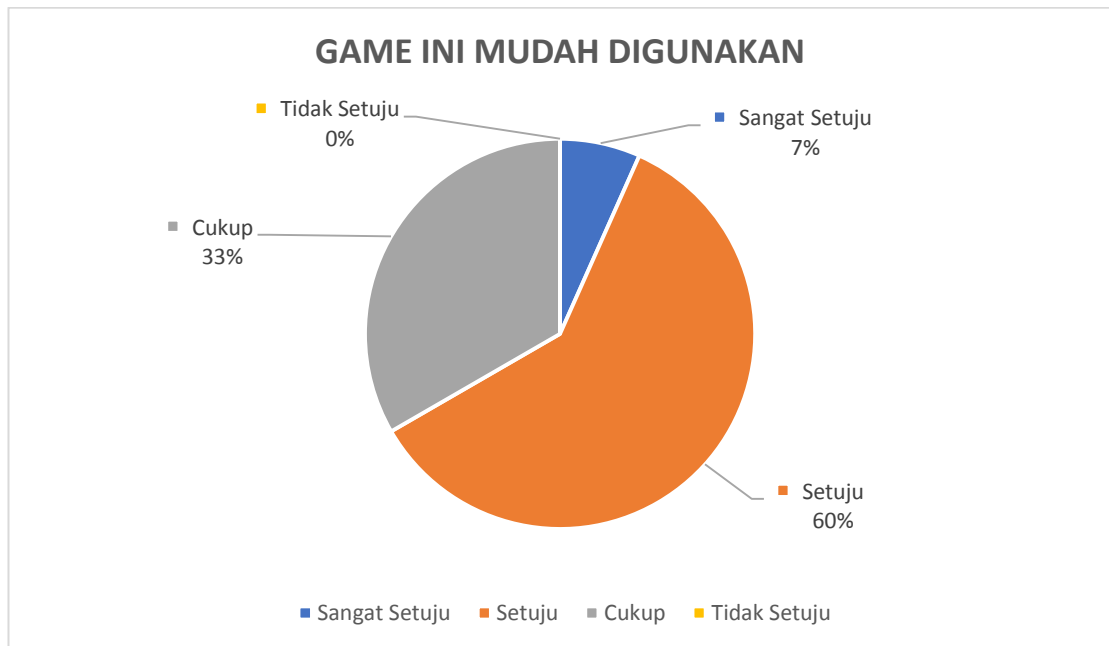
	lengkap				
5	Menu Kuis pada game untuk menguji pengetahuan sudah bagus				
6	Game ini dapat berperan sebagai media pembelajaran yang menarik				
7	Penempatan tombol pada game sudah tepat				

Dari hasil uji coba pada pengguna dapat ditarik kesimpulan sebagai berikut :

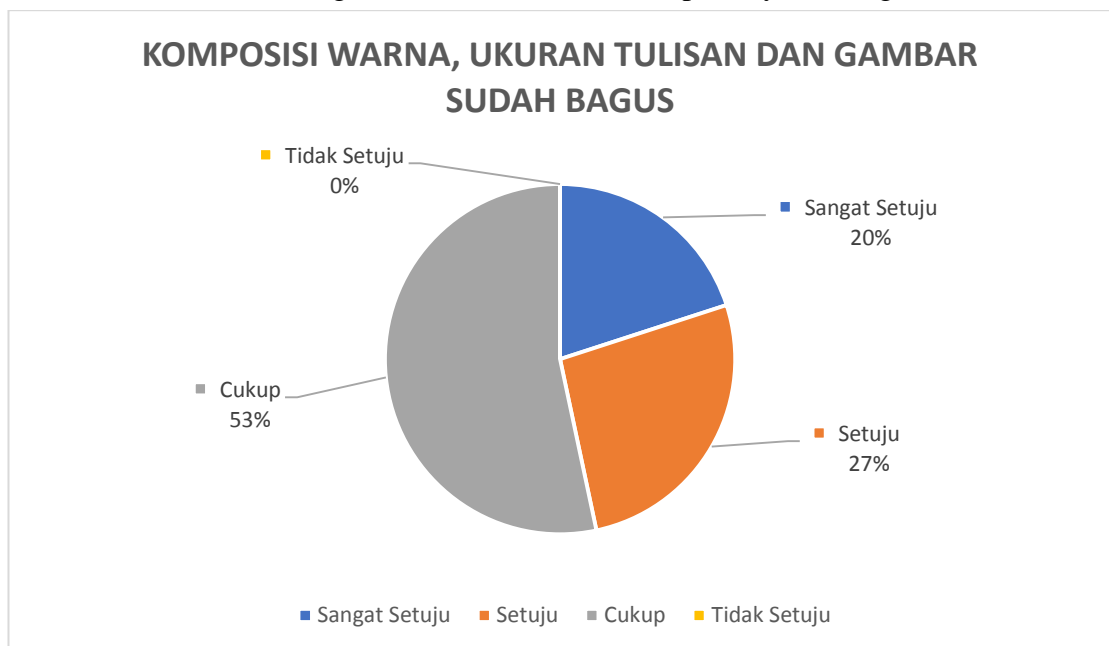
Tabel 5.5 Diagram hasil kuisisioner untuk pertanyaan pertama.



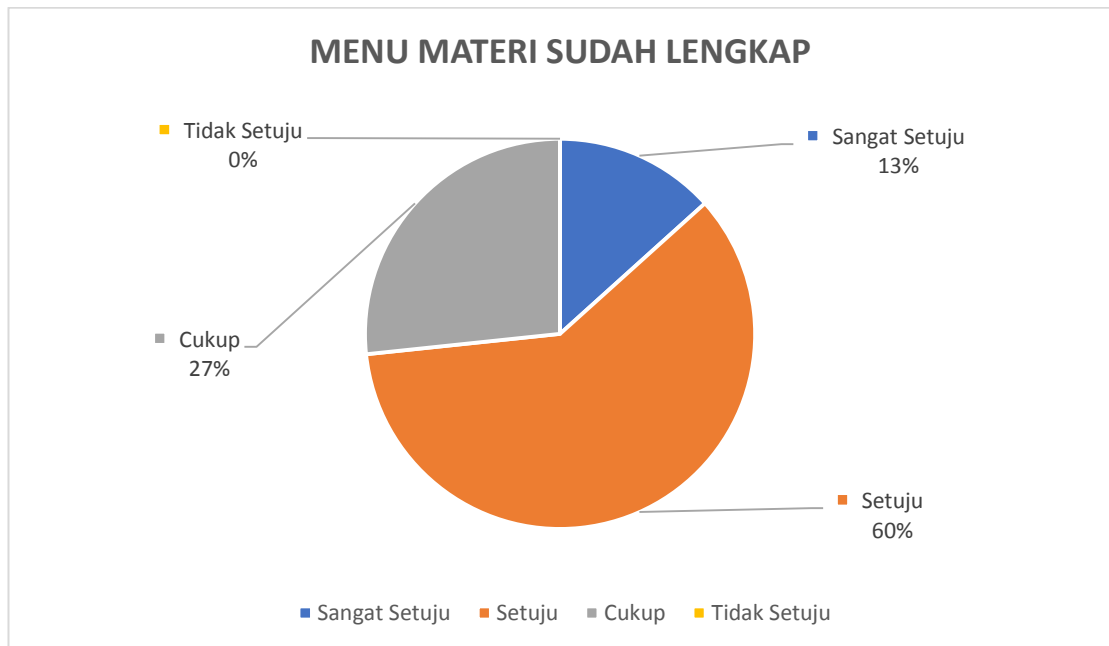
Tabel 5.6 Diagram hasil kuisioner untuk pertanyaan kedua.



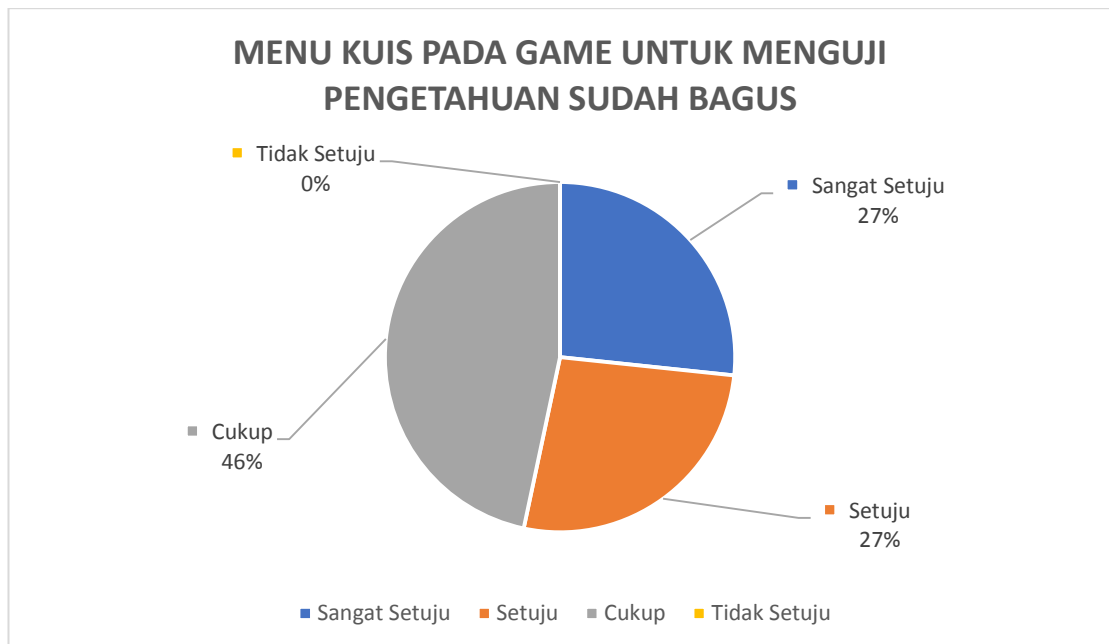
Tabel 5.7 Diagram hasil kuisioner untuk pertanyaan ketiga.



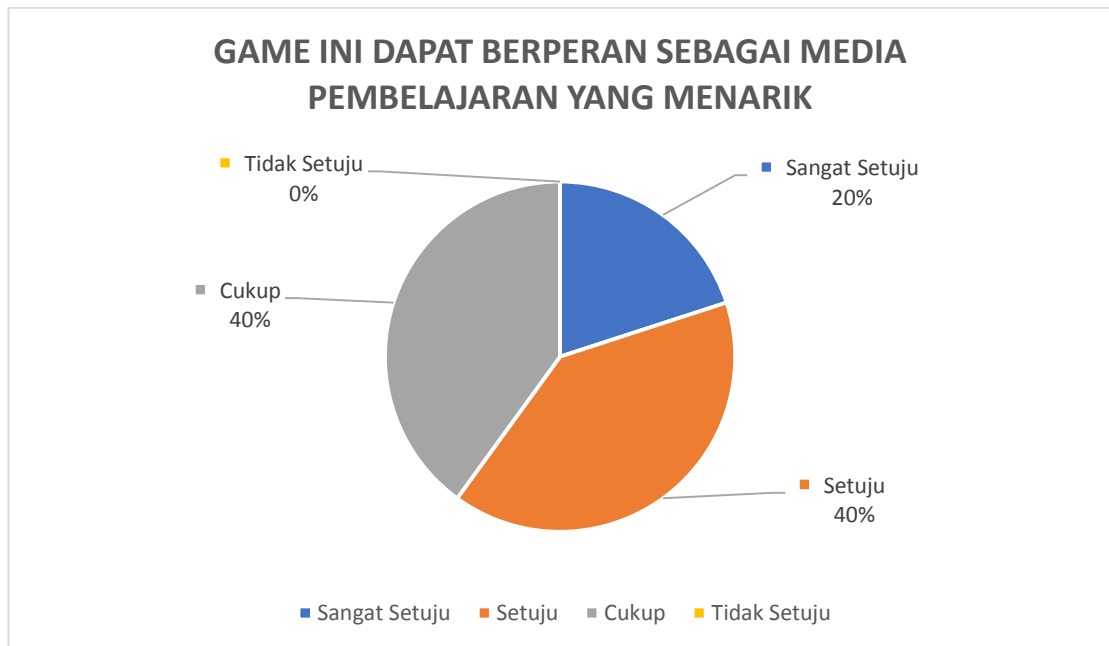
Tabel 5.8 Diagram hasil kuisioner untuk pertanyaan keempat.



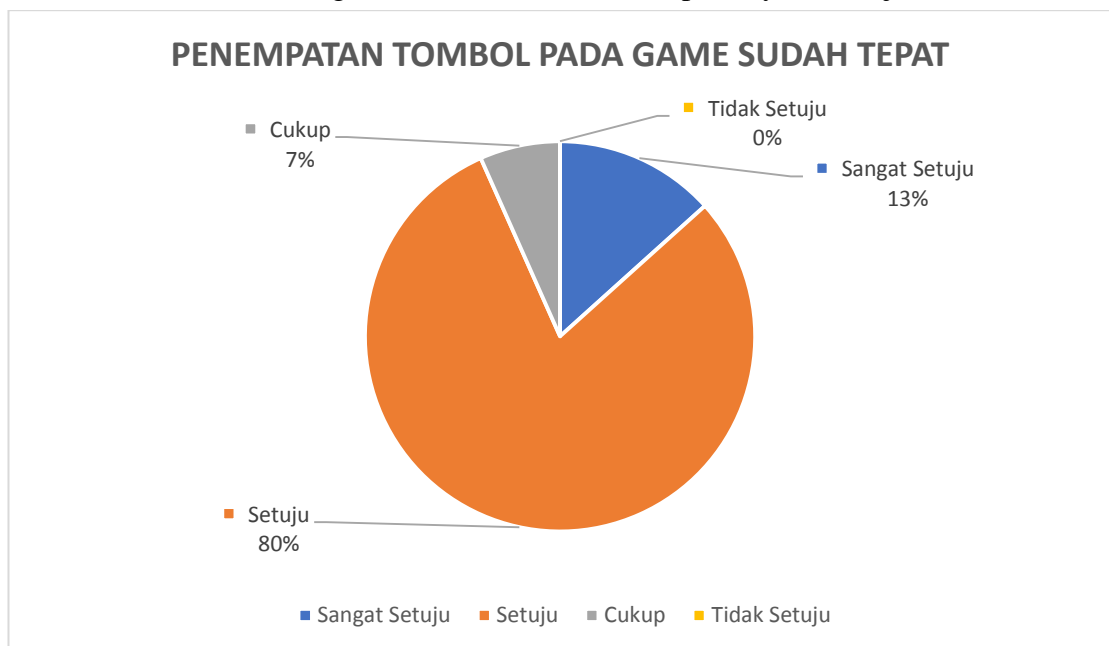
Tabel 5.9 Diagram hasil kuisioner untuk pertanyaan kelima.



Tabel 5.10 Diagram hasil kuisisioner untuk pertanyaan keenam.



Tabel 5.11 Diagram hasil kuisisioner untuk pertanyaan ketujuh.



5.5 Analisis Hasil Uji Coba

Setelah melakukan proses perencanaan dan pembuatan serta pengujian Aplikasi *Game Save Your Generation* Berbasis Android, didapatkan beberapa hal yang dapat disimpulkan sebagai berikut :

- a. Game ini mampu memberikan pengetahuan dan informasi mengenai bahaya rokok, minuman keras dan narkoba kepada generasi muda agar menghindarinya.
- b. Game ini membuat anak tertarik bermain game dengan tampilan yang menarik dan dengan *gameplay* yang mudah dimainkan.
- c. Game ini membantu pengetahuan mereka tentang rokok, narkoba dan minuman keras.

BAB VI. KESIMPULAN

6.1 Kesimpulan

Dari hasil implementasi game dan uji coba yang telah dilakukan terhadap subjek yang ditentukan yakni anak usia 7 sampai 13 tahun adalah sebagian besar pengguna merasa Aplikasi *Game Save Your Generation* Berbasis Android mampu memberikan pengetahuan dan informasi mengenai bahaya rokok, minuman keras dan narkoba kepada generasi muda dan tertarik dengan bermain *game* yang telah dibuat dengan tampilan yang menarik dan *gameplay* yang mudah dimainkan.

6.2 Saran

Aplikasi *Game Save Your Generation* Berbasis Android dapat dikembangkan dari banyak sisi. Dari sisi penyampaian pengetahuan, level *game* lebih dikembangkan kembali, keberagaman serangan *player*, *enemy* juga dapat dikembangkan dan pengembangan sistem *save game*. Dari sisi tampilan atau desain visual, lingkungan dalam game dapat dilengkapi dengan objek-objek yang menunjukkan keadaan lingkungan sekolah yang sebenarnya. Dan dari sisi kuis juga dapat dikembangkan menjadi soal puzzle yang lebih menarik.

DAFTAR PUSTAKA

- [1] http://www.kompasiana.com/dedetaufik/fenomena-rokok-di-kalangan-pelajar_54f41acf745513982b6c8777, diakses pada 13 Agustus 2017
- [2] <http://lampung.bnn.go.id/wp/2016/12/06/pengaruh-narkoba-dikalangan-remaja/>, diakses pada 13 Agustus 2017
- [3] <https://m.detik.com/news/berita/2852915/23-persen-remaja-indonesia-pernah-konsumsi-miras>, diakses pada 13 Agustus 2017
- [4] Salen, Katie Zimmerman and Eric. 2003. Rules of Play: Game Design Fundamentals. MIT Press.
- [5] Safaat, Nazruddin. (2012). “Pemrograman Aplikasi Mobile Smartphone dan Tablet PC Berbasis Android”. Bandung : Informatika.
- [6] UnityTechnologies. (2016,Des.28). Pengertian Unity.
Tersedia : <http://unity3d.com/unity>
- [7] <http://www.davidprasetyo.com/2015/11/pengertian-storyboard.html>, diakses pada 6 juli 2017
- [8] Arif, Rizqya Zakkyatul.2016.*Pengembangan Game Platformer 2D Attacking Gomi Monster*.Malang : Politeknik Negeri Malang
- [9] Hans Tendra. 2003. *Merokok dan Kesehatan*. Surabaya.
Tersedia : <http://yahoo.com>.
- [10] BNN. (2016,Des.28) *Pengertian Narkoba*. Tersedia:
<http://dedihumas.bnn.go.id/read/section/artikel/2014/03/10/929/pengertian-narkoba>.
- [11] Satya, Joewana dkk. 2001. Narkoba. Penerbit Media Pressindo : Yogyakarta.

Lampiran List Kode Program

Kode program PlayerCowok.cs

```
using UnityEngine;

using System.Collections;

using UnityEngine.UI;

public class PlayerCowok : Character
{
    private static PlayerCowok instance;

    private int jumpTick = 0;

    private bool isHitObstacle = false;

    private float direction;

    private bool move;

    [SerializeField]
    private float groundRadius;

    [SerializeField]
    private LayerMask whatIsGround;

    public GameObject panelDie;

    [SerializeField]
    private bool airControl;

    [SerializeField]
    private float JumpForce;

    [SerializeField]
    private float curhealth;

    private float calchealth;

    public Rigidbody2D MyRigidBody{get;set;}

    public bool Jump {get;set;}

    public bool OnGround {get;set;}

    public Slider health;
```

```

private bool canHit=true;

private Vector2 startPos;

private SpriteRenderer sprite;

private Color col;

public bool isFacingBoss = false;

public float facebosXmin, facebosXmax;

public int sound;

public float mincam,maxcam;

public Text scor, pScore;

public static PlayerCowok Instance

{
    get
    {
        if(instance == null)
        {
            instance =
GameObject.FindObjectOfType<PlayerCowok>();
        }
        return instance;
    }
}

public override void Start()
{
    sound = PlayerPrefs.GetInt("sound");

    health.interactable = false;

    base.Start();

    MyRigidBody = GetComponent<Rigidbody2D>();

    startPos = transform.position;

```

```

        sprite = GetComponent<SpriteRenderer>();

        col = sprite.color;
    }

    void Update()
    {
        HandleInput();

        if(checkIfFall()){

            transform.position = startPos;

            health.value -= 30f;

            StartCoroutine("blink");

        }

        pDie();
    }

    IEnumerator blink(){

        col.a = 0.4f;

        sprite.color = col;

        yield return new WaitForSeconds (0.5f);

        col.a = 255;

        sprite.color = col;

        yield return new WaitForSeconds (0.1f);

        col.a = 0.4f;

        sprite.color = col;

        yield return new WaitForSeconds (0.1f);

        col.a = 255;

        sprite.color = col;

        StopCoroutine ("blink");

    }

    void FixedUpdate()

```



```

{
    float horizontal = Input.GetAxis("Horizontal");

    OnGround = IsGrounded();

    if (move)
    {
        HandleMovement(direction);

        Flip(direction);
    }
    else
    {
        HandleMovement(horizontal);

        Flip(horizontal);
    }

    HandleLayers();
}

private void HandleMovement(float horizontal)
{
    if (MyRigidBody.velocity.y < 0)
    {
        myAnimator.SetBool("land", true);
    }

    if (!Attack && (OnGround || airControl) &&
        !isHitObstacle)
    {
        MyRigidBody.velocity = new Vector2(horizontal
        * movementSpeed, MyRigidBody.velocity.y);

        transform.position = new
        Vector2(Mathf.Clamp(transform.position.x, mincam,
        maxcam), transform.position.y);

        if (isFacingBoss)

```

```

        transform.position = new
Vector2(Mathf.Clamp(transform.position.x, facebosXmin,
facebosXmax),transform.position.y);

    }

    myAnimator.SetFloat("speed",
Mathf.Abs(horizontal));

}

void OnCollisionEnter2D(Collision2D coll) {

    //Debug.Log (coll.gameObject.layer);

    if (coll.gameObject.layer == 11)

        isHitObstacle = true;

}

void OnCollisionExit2D(Collision2D coll){

    if (coll.gameObject.layer == 11)

        isHitObstacle = false;

}

private void HandleInput()

{

    if(Input.GetKeyDown(KeyCode.Space))

    {

        myAnimator.SetTrigger("jump");

        //Jump = true;

        jumpTick++;

        if(jumpTick < 2){

            //MyRigidBody.velocity = new
Vector2(transform.position.x,14f);

            //transform.Translate(Vector2.up*100f*Time.deltaTime);

            //transform.position = new
Vector2(transform.position.x,Mathf.Clamp(transform.position.y,
-6.62f, 5.96f));

            MyRigidBody.AddForce(Vector2.up*JumpForce);

```

```

        //MyRigidBody.position = new
Vector2(transform.position.x,Mathf.Clamp(MyRigidBody.position.y
, -6.62f, 2f));

        //transform.position = new
Vector2(transform.position.x,Mathf.Clamp(transform.position.y,
-6.62f, 2f));

    }

    if(MyRigidBody.velocity.y == 0f){

        jumpTick = 0;

    }

}

if(Input.GetKeyDown(KeyCode.Q))

{

    myAnimator.SetTrigger("attack");

    AttackWind(0);

}

}

private void Flip(float horizontal)

{

    if(horizontal > 0 && !facingRight || horizontal < 0
&& facingRight)

    {

        ChangeDirection();

    }

}

private bool IsGrounded()

{

    if(MyRigidBody.velocity.y <= 0)

    {

        foreach(Transform point in groundPoints)

        {

```

```

        Collider2D[] colliders =
Physics2D.OverlapCircleAll(point.position, groundRadius,
whatIsGround);

        for (int i = 0; i < colliders.Length;
i++)

        {

            if(colliders[i].gameObject !=
gameObject)

            {

                return true;

            }

        }

    }

    return false;

}

private void HandleLayers()

{

    if(!OnGround)

    {

        myAnimator.SetLayerWeight(1, 1);

    }

    else

    {

        myAnimator.SetLayerWeight(1, 0);

    }

}

public override void AttackWind(int value)

```

```
1) {
    if(!OnGround && value == 1 || OnGround && value ==
    {
        base.AttackWind(value);
    }
}
```

```
void OnTriggerEnter2D(Collider2D other){

    if(other.gameObject.tag == "bullet")
    {
        health.value -= 20;

        if(health.value <= 0 && canHit)
        {
            myAnimator.SetTrigger("die");

            StartCoroutine(wait());
            canHit = !canHit;

        }

        Destroy(other.gameObject);
        StartCoroutine("blink");
    }

    if(other.gameObject.tag == "attack1")
    {
        health.value -= 15;
```

```

        if(health.value <= 0 && canHit)
        {
            myAnimator.SetTrigger("die");

            StartCoroutine(wait());

            canHit = !canHit;

        }

        if(Application.loadedLevel == 2)
        {
            Destroy(other.gameObject,1f);
        }else
        {
            Destroy(other.gameObject);
        }

        StartCoroutine("blink");
    }

    if(other.gameObject.tag == "kuman")
    {
        health.value -= 10;

        if(health.value <= 0 && canHit)
        {
            myAnimator.SetTrigger("die");;

            StartCoroutine(wait());

            canHit = !canHit;

        }

        Destroy(other.gameObject);
    }

```

```

        StartCoroutine("blink");
    }
}

void OnTriggerStay2D(Collider2D other)
{
    if(other.gameObject.tag == "Enemy")
    {
        health.value -= curhealth;

        if(health.value <= 0 && canHit)
        {
            myAnimator.SetTrigger("die");;

            StartCoroutine(wait());

            canHit = !canHit;

        }

        StartCoroutine("blink");
    }

    if(other.gameObject.tag == "bonushealth")
    {
        health.value += 100;

        Destroy(other.gameObject);

        StartCoroutine("blink");
    }
}

```

```

IEnumerator wait() {
    //Debug.Log(Time.time);

    yield return new WaitForSeconds(0.5f);
}

```

```

        Time.timeScale = 0;
    }

    public void BtnMove(float direction)
    {
        this.direction = direction;

        this.move = true;
    }

    public void BtnJump()
    {
        myAnimator.SetTrigger("jump");

        //Jump = true;

        jumpTick++;

        if(jumpTick < 2){

            MyRigidBody.AddForce(Vector2.up*JumpForce);

        }

        if(MyRigidBody.velocity.y == 0f){

            jumpTick = 0;

        }

    }

    public void BtnAttack()
    {
        myAnimator.SetTrigger("attack");

        AttackWind(0);

    }

    public void stop(){

        this.direction = 0f;

        this.move = false;

    }

```



```
public bool checkIfFall(){  
    bool isFall = false;  
    if(transform.position.y <= -7.2f) isFall = true;  
    else isFall = false;  
    return isFall;  
}  
  
public void faceBoss(bool val){  
    isFacingBoss = val;  
}  
  
void pDie(){  
    if(health.value <= 0){  
        pScore.text = scor.text;  
        panelDie.SetActive(true);  
    }  
}  
}
```

Kode program BosArena.cs

```
using UnityEngine;

using UnityEngine.SceneManagement;

using System.Collections;

using System.Collections.Generic;

using UnityEngine.UI;

public class BOS_Arena1 : MonoBehaviour {

    GameObject isHitWind = null;

    private float dir;

    private bool isDetect=false;

    private GameObject objPlayer;

    private PlayerCowok pc;

    private Transform player;

    public float movementSpeed = 4f;

    public float distance;

    public float health;

    private float healthStart;

    private bool isStart=false;

    private float tempSpeed;

    public Slider playerSlide;

    private float playerHealth;

    private float playerHealthStart;

    protected bool facingRight;

    private Animator myAnimator;

    private Text score;

    public GameObject camera;

    private CameraFollow cam;
```

```
float[] c = new float[3];

public float bobotPlayer;

public float myBobot;

public float bobotJarak;

public float sawResult;

public int bosAttack;

private SAW ai;

public GameObject shield;

public Transform shieldPos;

public GameObject bullet;

public Transform bulletpos1;

public Transform bulletpos2;

GameObject shld = null;

public float direct;

Vector2 v = Vector2.zero;

Vector3 pos;

private SpriteRenderer sprite;

Color col;

float tempDir;

public float speed;

public float distToPlayer;

public float facebosXmin, facebosXmax;

void Awake()

{

    ai = new SAW();

    playerHealthStart = playerSlide.value;

    healthStart = health;

    tempSpeed = movementSpeed;
```

```

        float x = transform.position.x;

        objPlayer = GameObject.Find("PlayerCowok");

        pc = objPlayer.GetComponent<PlayerCowok>();

        player = (Transform)objPlayer.transform;

        isStart = true;

        pos = new Vector3(transform.position.x, -2.5f, 0f);

        myAnimator = (Animator)GetComponent<Animator>();

        sprite =
(SpriteRenderer)GetComponent<SpriteRenderer>();

        col = sprite.color;

        score =
(Text)GameObject.Find("Score").GetComponent<Text>();

        cam = camera.GetComponent<CameraFollow>();
    }

    void Start(){

        getMagnitude();

        tempDir = direct;

    }

    void getMagnitude(){

        distance = Mathf.Abs(player.position.x -
transform.position.x);

        direct = Mathf.Sign(player.position.x -
transform.position.x);

    }

    void Update ()

    {

        getMagnitude();

        bobotPlayer = playerLife();

        myBobot = myLife();

        bobotJarak = dist();

```

```

        c[0]=bobotPlayer;

        c[1]=myBobot;

        c[2]=bobotJarak;

        sawResult = ai.V(c);

        bosAttack = bossAct();

        attack();

        flip();

        move();

    }

    float convertToPercent(float thisHealth,float
thisHealthStart){

        return (thisHealth * 100)/thisHealthStart;

    }

    void move(){

        if(tempDir != direct){

            transform.Translate(Vector2.left * -1* speed
*Time.deltaTime);

            transform.position = new
Vector2(Mathf.Clamp(transform.position.x,facebosXmin,facebosXma
x), transform.position.y);

            myAnimator.SetBool("walk",true);

            if(transform.position.x <=facebosXmin ||
transform.position.x >=facebosXmax){

                tempDir = direct;

                myAnimator.SetBool("walk",false);

            }

        }

    }

    void OnTriggerExit2D(Collider2D other){

        if(other.gameObject.tag == "PlayerCowok"){

```

```

        //transform.Translate(Vector2.left * -5f);

        //transform.position =
        Vector2.SmoothDamp(transform.position, (new
        Vector2(transform.position.x,transform.position.y))-
        (Vector2.left * -5f), ref v, 100f) ;

    }

}

void OnTriggerEnter2D(Collider2D other) {

    if(other.gameObject.tag == "wind")
    {

        addScore();

        health-=25;

        StartCoroutine("Blinker");

        isHitWind =
        GameObject.FindGameObjectWithTag("wind");

        if (isHitWind != null)
        {

            Destroy(isHitWind);

        }

        if(health <= 0)
        {

            //Debug.Log("dead");

            cam.deadBoss();

            pc.faceBoss(false);

            Destroy(this.gameObject);

            addScoreDie();

            Application.LoadLevel (11);
        }
    }
}

```

```

        }

        //sprite.color = col;

        /*addScore();

        health--;

        StartCoroutine("Blinker");

        if(health <= 0)

        {

            Destroy(this.gameObject);

        }*/
    }

}

IEnumerator Blinker(){

    col.a = 0.4f;

    sprite.color = col;

    yield return new WaitForSeconds (0.1f);

    col.a = 255;

    sprite.color = col;

    yield return new WaitForSeconds (0.1f);

    col.a = 0.4f;

    sprite.color = col;

    yield return new WaitForSeconds (0.1f);

    col.a = 255;

    sprite.color = col;

    StopCoroutine ("Blinker");

}

void addScore(){

    float val = float.Parse(score.text.ToString());

    val += 25f;

```

```

        score.text = val.ToString();
    }

    void addScoreDie()
    {
        float val = float.Parse(score.text.ToString());

        val += 200f;

        score.text = val.ToString();
    }

    float playerLife(){
        float bobot = 0;

        playerHealth = playerSlide.value;

        float percentHealth =
convertToPercent(playerHealth, playerHealthStart);

        if(percentHealth >= 0 && percentHealth <= 35) bobot
= 3;

        else if(percentHealth > 35 && percentHealth <= 70)
bobot =2;

        else if(percentHealth > 70 && percentHealth <= 100)
bobot =1;

        return bobot;
    }

    float myLife(){
        float bobot = 0;

        float percentHealth = convertToPercent(health,
healthStart);

        if(percentHealth >= 0 && percentHealth <= 35) bobot
= 3;

        else if(percentHealth > 35 && percentHealth <= 70)
bobot =2;

        else if(percentHealth > 70 && percentHealth <= 100)
bobot =1;

        return bobot;
    }

```



```

float dist(){

    float bobot=0;

    if(distance >= 8) bobot = 1;

    else if(distance >= 5 && distance < 8) bobot = 2;

    else if(distance >= 0 && distance < 5) bobot = 3;

    return bobot;

}

int bossAct(){

    int result=0;

    if(sawResult >= 0.5 && sawResult <= 0.77)

        result = 1;//attack2

    else if(sawResult >= 0.83 && sawResult <= 0.99)

        result = 2;//shield

    else if(sawResult >= 1 && sawResult <= 1.5)

        result = 3;//attack1

    return result;

}

void attack(){

    switch(bosAttack){

    case 1:

        if(shld != null){

            Destroy(shld,0.1f);

        }

        if(distance <= 10.5f){

            StartCoroutine("delayAttack2");

        }

        break;

```

```

        case 2:

            //Destroy(shld,1f);

            shieldShow();

            break;

        case 3:

            if(shld != null){

                Destroy(shld,0.1f);

            }

            if(distance <= 10.5f){

                StartCoroutine("delayAttack");

            }

            break;

        default:

            Debug.Log("No Attack");

            break;

    }

}

IEnumerator delayAttack() {

    yield return new WaitForSeconds(1f);

    myAnimator.SetTrigger("shoot");

    //BulletAttack(1);

    StopCoroutine("delayAttack");

}

IEnumerator delayAttack2() {

    yield return new WaitForSeconds(1f);

    myAnimator.SetTrigger("attack");

    //BulletAttack(1);

```

```

        StopCoroutine("delayAttack2");
    }

    IEnumerator wait() {

        //Debug.Log(Time.time);

        yield return new WaitForSeconds(5f);

        Time.timeScale = 0;

    }

    public void BulletAttack(int value)
    {

        GameObject tmp = null, tmp2=null;

        if(direct > 0f)
        {

            tmp = (GameObject)Instantiate(bullet,
bulletpos1.position, Quaternion.Euler(new Vector3(0,0,-180)));

            tmp.GetComponent<Wind>().Initialize(Vector2.right);

        }

        else

        {

            tmp = (GameObject)Instantiate(bullet,
bulletpos1.position, Quaternion.Euler(new Vector3(0,0,0)));

            tmp.GetComponent<Wind>().Initialize(Vector2.left);

        }

        if(tmp != null)
        {

            Destroy(tmp, 2f);

            Destroy(tmp2, 2f);

        }

    }

```

```

public void Attack1Bos(int value)
{
    GameObject tmp = null, tmp2=null;

    if(direct > 0f)
    {
        tmp = (GameObject)Instantiate(bullet,
bulletpos1.position, Quaternion.Euler(new Vector3(0,0,-180)));

        tmp2 = (GameObject)Instantiate(bullet,
bulletpos2.position, Quaternion.Euler(new Vector3(0,0,-180)));

        tmp.GetComponent<Wind>().Initialize(Vector2.right);

        tmp2.GetComponent<Wind>().Initialize(Vector2.right);
    }
    else
    {
        tmp = (GameObject)Instantiate(bullet,
bulletpos1.position, Quaternion.Euler(new Vector3(0,0,0)));

        tmp2 = (GameObject)Instantiate(bullet,
bulletpos2.position, Quaternion.Euler(new Vector3(0,0,0)));

        tmp.GetComponent<Wind>().Initialize(Vector2.left);

        tmp2.GetComponent<Wind>().Initialize(Vector2.left);
    }
    if(tmp != null)
    {
        Destroy(tmp, 2f);

        Destroy(tmp2, 2f);
    }
}

public void shieldShow(){

```

```

        if(shld==null)
        {
            if(direct > 0f)
            {
                shld =
                (GameObject)Instantiate(shield, shieldPos.position,
                Quaternion.Euler(new Vector3(0,180,0)));
            }else{
                shld =
                (GameObject)Instantiate(shield, shieldPos.position,
                Quaternion.Euler(new Vector3(0,0,0)));
            }
        }

        public void flip(){
            if(direct > 0f)
            {
                transform.localRotation =
                Quaternion.Euler(new Vector3(0,180,0));
            }else{
                transform.localRotation =
                Quaternion.Euler(new Vector3(0,0,0));
            }
        }

        void limitMove(){
        }
    }

```

Kode program Enemy.cs

```
using UnityEngine;

using System.Collections;

using UnityEngine.UI;

public class Enemy : MonoBehaviour

{

    [SerializeField]

    private float radius, rayLength;

    private float radiusLeft, radiusRight;

    GameObject isHitWind = null;

    private float dir;

    private bool isDetect=false;

    private Transform player;

    public float movementSpeed = 4f;

    public float distance;

    public float health = 2f;

    private bool isStart=false;

    private float tempSpeed;

    Vector3 pos;

    private SpriteRenderer sprite;

    Color col;

    private Text score;

    BoxCollider2D myCollider;

    void Awake()

    {

        myCollider = GetComponent<BoxCollider2D>();

        tempSpeed = movementSpeed;

        dir=1;
```

```

        float x = transform.position.x;

        radiusLeft = x - radius;

        radiusRight = x + radius;

        player =
(Transform)GameObject.Find("PlayerCowok").transform;

        isStart = true;

        pos = new Vector3(transform.position.x, -2.5f, 0f);

        sprite =
(SpriteRenderer)GetComponent<SpriteRenderer>();

        col = sprite.color;

        score =
(Text)GameObject.Find("Score").GetComponent<Text>();
    }

    void Update ()
    {

        Detection();

        Movement(isDetect);

        Magnitude();

        myGrav();

    }

    void Magnitude(){

        distance = Mathf.Abs(player.position.x -
transform.position.x);

    }

    void Chase(){

        transform.Translate(dir*transform.right *
movementSpeed * Time.deltaTime);

    }

    void OnTriggerEnter2D(Collider2D other) {

        if(other.gameObject.tag == "PlayerCowok")

        {

```

```

        movementSpeed -= tempSpeed/2f;
    }

    if(other.gameObject.tag == "wind")
    {
        addScore();

        health-=25;

        StartCoroutine("Blinker");

        isHitWind =
GameObject.FindGameObjectWithTag("wind");

        if (isHitWind != null)
        {
            Destroy(isHitWind);
        }

        if(health <= 0)
        {
            Destroy(this.gameObject);
        }
    }
}

void addScore() {
    float val = float.Parse(score.text.ToString());

    val += 25f;

    score.text = val.ToString();
}

IEnumerator Blinker() {
    col.a = 0.4f;

    sprite.color = col;

    yield return new WaitForSeconds (0.1f);

    col.a = 255;
}

```



```

        sprite.color = col;

        yield return new WaitForSeconds (0.1f);

        col.a = 0.4f;

        sprite.color = col;

        yield return new WaitForSeconds (0.1f);

        col.a = 255;

        sprite.color = col;

        StopCoroutine ("Blinker");
    }

    void OnTriggerExit2D(Collider2D other) {

        movementSpeed = tempSpeed;

    }

    void Movement(bool isChase)

    {

        float x = transform.position.x;

        if(!isChase){

            if(x >= radiusRight)

            {

                transform.rotation =
Quaternion.Euler(new Vector3(0,0,0));

                //ChangeDirection();

                dir = -1;

            }

            if(x <= radiusLeft)

            {

                transform.rotation =
Quaternion.Euler(new Vector3(0,-180,0));

                //ChangeDirection();

```

```

        dir = 1;

    }

}

transform.Translate(dir*transform.right *
movementSpeed * Time.deltaTime);

}

void Detection()

{

    RaycastHit2D isForward =
Physics2D.Raycast(transform.position, Vector2.right, rayLength,
1 << LayerMask.NameToLayer("Player"));

    RaycastHit2D isBackward =
Physics2D.Raycast(transform.position, Vector2.left, rayLength,
1 << LayerMask.NameToLayer("Player"));

    if(isForward)

    {

        Debug.Log("player di depan");

        transform.rotation = Quaternion.Euler(new
Vector3(0,-180,0));

        dir=1;

        isDetect = true;

    }

    if(isBackward)

    {

        Debug.Log("player di belakang");

        transform.rotation = Quaternion.Euler(new
Vector3(0,0,0));

        dir = -1;

        isDetect = true;

    }

    Debug.DrawRay(transform.position, Vector2.right *
rayLength, Color.blue);

    Debug.DrawRay(transform.position, Vector2.left *

```

```

rayLength, Color.blue);

    }

    void OnDrawGizmos() {

        Gizmos.color = new Color(1f,0f,0f,0.6f);

        if(isStart)

            Gizmos.DrawCube(pos, new Vector3(2*radius,
1f, 0f));

        else

            Gizmos.DrawCube(new
Vector3(transform.position.x, -2.5f, 0f), new Vector3(2*radius,
1f, 0f));

    }

    Vector3 calcCollider() {

        Bounds bound = myCollider.bounds;

        Vector3 pos = bound.center - new Vector3(0f,
bound.extents.y,0f);

        return pos;

    }

    void myGrav() {

        RaycastHit2D hit =
Physics2D.Raycast((Vector2)calcCollider(),Vector2.down,0.1f,
1<<LayerMask.NameToLayer("ground"));

        Debug.DrawRay(calcCollider(),Vector3.down*0.1f,Color.red)
;

        if(!hit)

            transform.Translate(Vector2.down * 10f *
Time.deltaTime);

    }

}

```

Kode program EnemyCanAttack.cs

```
using UnityEngine;

using System.Collections;

using UnityEngine.UI;

public class EnemyCanAttack : MonoBehaviour {

    [SerializeField]

    private float radius, rayLength;

    private float radiusLeft, radiusRight;

    GameObject isHitWind = null;

    private float dir;

    private bool isDetect=false;

    private Transform player;

    public float movementSpeed = 4f;

    public float distance;

    public float health = 2f;

    private bool isStart=false;

    private float tempSpeed;

    private Animator myAnimator;

    [SerializeField]

    protected GameObject attacks;

    [SerializeField]

    protected Transform attackpost;

    private bool OnGround;

    protected bool facingRight;

    private Text score;

    Vector3 pos;

    private SpriteRenderer sprite;

    Color col;
```

```

void Awake()
{
    myAnimator = (Animator)GetComponent<Animator>();

    tempSpeed = movementSpeed;

    dir=1;

    float x = transform.position.x;

    radiusLeft = x - radius;

    radiusRight = x + radius;

    player =
(Transform)GameObject.Find("PlayerCowok").transform;

    isStart = true;

    pos = new Vector3(transform.position.x, -2.5f, 0f);

    sprite =
(SpriteRenderer)GetComponent<SpriteRenderer>();

    col = sprite.color;

    score =
(Text)GameObject.Find("Score").GetComponent<Text>();

    //col.a = 0.5f;
}

void Update ()
{
    Detection();

    Movement(isDetect);

    Magnitude();
}

void Magnitude(){

    distance = Mathf.Abs(player.position.x -
transform.position.x);

}

```

```

void OnTriggerEnter2D(Collider2D other) {

    if(other.gameObject.tag == "PlayerCowok")

        movementSpeed -= tempSpeed/2f;

    if(other.gameObject.tag == "wind")
    {

        //sprite.color = col;

        addScore();

        health-=25;

        StartCoroutine("Blinker");

        isHitWind =
GameObject.FindGameObjectWithTag("wind");

        if (isHitWind != null)
        {

            Destroy(isHitWind);

        }

        if(health <= 0)
        {

            Destroy(this.gameObject);

        }

    }

}

void addScore() {

    float val = float.Parse(score.text.ToString());

    val += 25f;

    score.text = val.ToString();

}

IEnumerator Blinker() {

    col.a = 0.4f;

    sprite.color = col;

```

```

        yield return new WaitForSeconds (0.1f);

        col.a = 255;

        sprite.color = col;

        yield return new WaitForSeconds (0.1f);

        col.a = 0.4f;

        sprite.color = col;

        yield return new WaitForSeconds (0.1f);

        col.a = 255;

        sprite.color = col;

        StopCoroutine ("Blinker");
    }

    void OnTriggerExit2D(Collider2D other) {

        movementSpeed = tempSpeed;

    }

    void Movement(bool isChase)

    {

        float x = transform.position.x;

        if(!isChase){

            if(x >= radiusRight)

            {

                transform.rotation =
Quaternion.Euler(new Vector3(0,0,0));

                //ChangeDirection();

                dir = -1;

            }

            if(x <= radiusLeft)

            {

                transform.rotation =
Quaternion.Euler(new Vector3(0,-180,0));

```

```

        //ChangeDirection();

        dir = 1;

    }

    } else{

        StartCoroutine("delayAttack");

        movementSpeed=0f;

        if(distance>radius){

            isDetect = false;

            movementSpeed=tempSpeed;

        }

    }

    transform.Translate(dir*transform.right *
movementSpeed * Time.deltaTime);

}

IEnumerator delayAttack(){

    yield return new WaitForSeconds(10);

    EnemyAttack(1);

    StopCoroutine("delayAttack");

}

void Detection()

{

    RaycastHit2D isForward =
Physics2D.Raycast(transform.position, Vector2.right, rayLength,
1 << LayerMask.NameToLayer("Player"));

    RaycastHit2D isBackward =
Physics2D.Raycast(transform.position, Vector2.left, rayLength,
1 << LayerMask.NameToLayer("Player"));

    //Debug.Log(LayerMask.NameToLayer("Player"));

    if(isForward)

    {

```



```

        Debug.Log("player di depan");

        transform.rotation = Quaternion.Euler(new
Vector3(0,-180,0));

        myAnimator.SetTrigger("attack");

        dir=1;

        isDetect = true;

    }

    if(isBackward)

    {

        Debug.Log("player di belakang");

        transform.rotation = Quaternion.Euler(new
Vector3(0,0,0));

        myAnimator.SetTrigger("attack");

        dir = -1;

        isDetect = true;

    }

    Debug.DrawRay(transform.position, Vector2.right *
rayLength, Color.blue);

    Debug.DrawRay(transform.position, Vector2.left *
rayLength, Color.blue);

}

public void EnemyAttack(int value)

{

    GameObject tmp = null;

    //Debug.Log("facing="+facingRight.ToString());

    if(dir > 0f)

    {

        tmp = (GameObject)Instantiate(attacks,
attackpost.position, Quaternion.Euler(new Vector3(0,0,180)));

        tmp.GetComponent<Wind>().Initialize(Vector2.right);

```

```

    }

    else

    {

        tmp = (GameObject) Instantiate(attacks,
attackpost.position, Quaternion.Euler(new Vector3(0,0,0)));

        tmp.GetComponent<Wind>().Initialize(Vector2.left);

    }

    if(tmp != null)

    {

        Destroy(tmp, 1f);

    }

}

void OnDrawGizmos() {

    Gizmos.color = new Color(1f,0f,0f,0.6f);

    if(isStart)

        Gizmos.DrawCube(pos, new Vector3(2*radius,
1f, 0f));

    else

        Gizmos.DrawCube(new
Vector3(transform.position.x, -2.5f, 0f), new Vector3(2*radius,
1f, 0f));

}

}

```

PROFIL PENULIS



DATA PRIBADI

Nama : Bellinda Novita Wasono
Tempat, Tanggal Lahir : Bekasi, 29 November 1995
Alamat : Jl. Danau Maninjau Barat IV B2 C3 Sawojajar –
Malang
Usia : 21 Tahun
Agama : Islam
Kewarganegaraan : Indonesia
Nomor Telepon : 085259484307
E-mail : novitabellinda@gmail.com

RIWAYAT PENDIDIKAN

2002 – 2003 : SD Negeri Purwantoro 2 Malang
2003 – 2007 : SD Negeri Sumbersari I Malang
2007 – 2008 : SD Negeri Lowokwaru I Malang
2008 – 2011 : SMP Negeri 4 Malang
2011 – 2014 : SMA Negeri 10 Malang
2014 – 2017 : Program Studi Manajemen Informatika, Jurusan Teknologi
Informasi Politeknik Negeri Malang

PROFIL PENULIS



DATA PRIBADI

Nama : Eko Putra Syah Warman
Tempat, Tanggal Lahir : Surabaya, 26 Mei 1996
Alamat : Jl. Petemon 3 No. 200 F - Surabaya
Usia : 21 Tahun
Agama : Islam
Kewarganegaraan : Indonesia
Nomor Telepon : 089677033454
E-mail : ekoputrasw@gmail.com

RIWAYAT PENDIDIKAN

2002 – 2008 : SD Negeri Petemon 11 Surabaya
2008 – 2011 : SMP Giki 1 Surabaya
2011 – 2014 : SMK Negeri 2 Surabaya
2014 – 2017 : Program Studi Manajemen Informatika, Jurusan Teknologi
Informasi Politeknik Negeri Malang