GAME PENGENALAN HEWAN LAUT BERBASIS ANDROID MENGGUNAKAN ACCELEROMETER

Disusun Oleh:

N a m a : Dadan Mahardhika Siagian

NIM : 12523055

JURUSAN TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

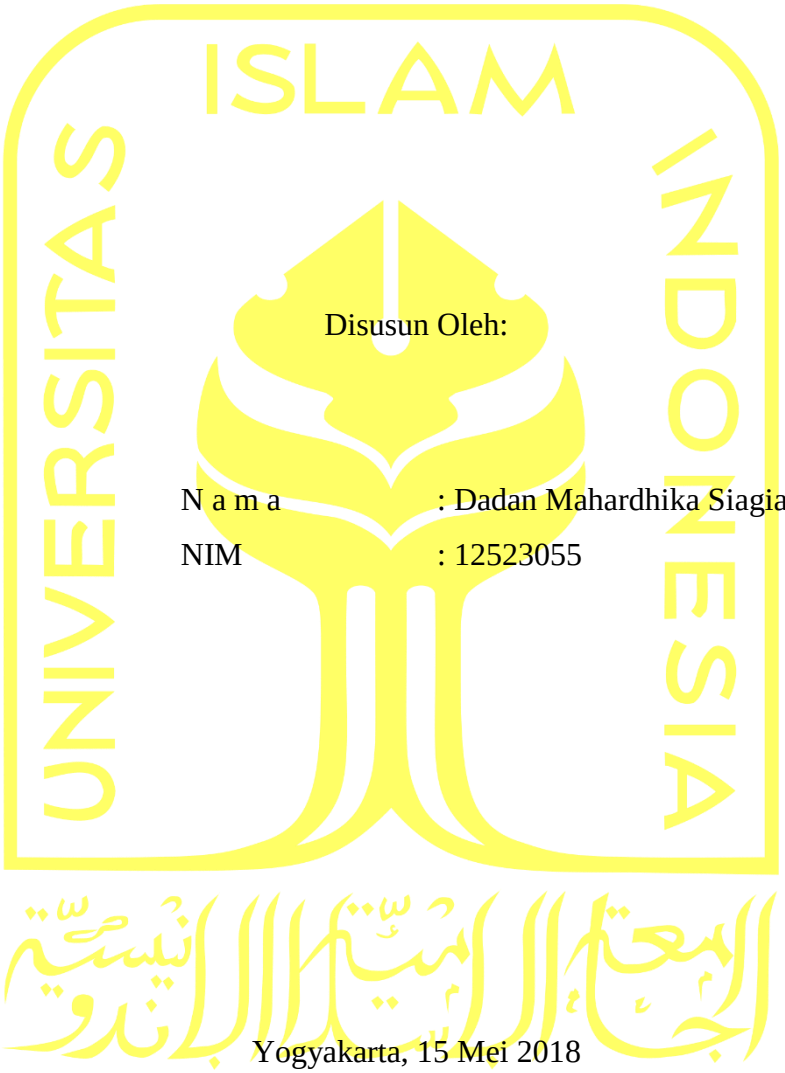
UNIVERSITAS ISLAM INDONESIA

2018

HALAMAN PENGESAHAN DOSEN PEMBIMBING

GAME PENGENALAN HEWAN LAUT BERBASIS ANDROID MENGGUNAKAN ACCELEROMETER

TUGAS AKHIR



Disusun Oleh:

N a m a : Dadan Mahardhika Siagian

NIM : 12523055

Yogyakarta, 15 Mei 2018

Pembimbing,

(Chandra Kusuma Dewa, S.Kom., M.Cs.)

HALAMAN PENGESAHAN DOSEN PENGUJI

GAME PENGENALAN HEWAN LAUT BERBASIS ANDROID MENGGUNAKAN ACCELEROMETER

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik Informatika
di Fakultas Teknologi Industri Universitas Islam Indonesia
Yogyakarta, 2 April 2018

Tim Penguji

Chandra Kusuma Dewa, S.Kom., M.Cs. _____

Anggota 1

Sheila Nurul Huda, S.Kom., M.Cs. _____

Anggota 2

Septia Rani, S.T., M.Cs. _____

Mengetahui,
Ketua Jurusan Teknik Informatika
Fakultas Teknologi Industri
Universitas Islam Indonesia

(Hendrik, S.T., M.Eng.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Dadan Mahardhika Siagian

NIM : 12523055

Tugas akhir dengan judul:

GAME PENGENALAN HEWAN LAUT BERBASIS ANDROID MENGGUNAKAN ACCELEROMETER

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila dikemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung resiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 22 Mei 2018

(Dadan Mahardhika Siagian)

HALAMAN PERSEMBAHAN

Segala puji dan syukur saya panjatkan kepada Allah SWT karena limpahan rahmat dan hidayah-Nya, dan tak lupa shalawat dan salam kita curahkan kepada junjungan kita Nabi Muhammad SAW.

Tugas Akhir ini saya persembahkan kepada kedua orang tua saya yang selalu mendo'akan yang terbaik untuk saya. Kepada kakak, mas, keponakan, dan keluarga besar saya yang selalu memberikan dukungan untuk saya untuk segera lulus.

Terima kasih juga untuk sahabat-sahabat dan teman seperjuangan yang selalu memberi dukungan dan saran demi kelancaran dalam penyusunan tugas akhir ini.

HALAMAN MOTO

"Learning never exhausts the mind."
(Leonardo Da Vinci)

"The noblest pleasure is the joy
of understandimg."
(Leonardo Da Vinci)

"When you give up, that's when the game ends."
(Slam Dunk / Mitsuyoshi Anzai)

"Stop counting only those things you have lost! What is gone, is gone! So ask yourself this. What is there... that still remains to you?!"
(One Piece / Jinbei)

KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabarakatuh

Alhamdulillahirabbil'alamin, segala puji bagi Allah SWT, karena limpahan rahmat dan hidayah-Nya, sehingga Tugas Akhir dengan judul "Game Pengenalan Hewan Laut" dapat penulis selesikan. Tak lupa shalawat dan salam kita curahkan kepada junjungan kita Nabi Muhammad SAW. Laporan Tugas Akhir ini disusun sebagai salah satu syarat untuk memperoleh gelar Sarjana Strata-1 (S1) Informatika di Universitas Islam Indonesia.

Dalam penyusunan Tugas Akhir ini, banyak sekali dukungan, bimbingan, inspirasi dan doa yang selalu menyertai penulis baik itu keluarga, dosen, dan teman-teman penulis. Dalam kesempempatan, penulis ingin mengucapkan terima kasih kepada:

1. Allah SWT yang telah memberikan kesehatan, kekuatan dan kesabaran sehingga penulis bisa menyelesaikan tugas akhir ini..
2. Bapak, Ibu, dan keluarga besar tercinta atas dukungan dan do'a.
3. Bapak Hendrik, ST., M.Eng.Sc. selaku Ketua Jurusan Informatika Fakultas Teknologi Industri Universitas Islam Indonesia.
4. Bapak Chandra Kusuma Dewa, S.Kom., M.Cs. selaku dosen pembimbing dalam penyusunan tugas akhir ini yang telah memberikan bimbingan, ilmu, dan waktunya untuk membimbing penulis dalam menyusun tugas akhir ini.
5. Segenap dosen Informatika Universitas Islam Indonesia yang telah mengajarkan saya banyak hal dan ilmu yang sangat bermanfaat.
6. Teman-teman kost Dwi, Fandra, Iqbal, Irfan, Sulaiman, Prasetyo, Cahyo, Wawan, Ipin, Alfi, dan Wawan yang selalu membantu disaat saya sedang kesulitan, dan mengajarkan ilmunya.

Masih banyak kekurangan dalam penyusunan tugas akhir ini. Oleh karena itu, penulis mengharapkan saran dan kritik yang membangun sebagai bekal di masa depan. Semoga apa yang telah disusun dapat memberikan manfaat bagi pembaca.

Yogyakarta, 22 Mei 2018

(Dadan Mahardhika Siagian)

SARI

Pesatnya perkembangan teknologi seharusnya dapat mempermudah seseorang untuk belajar. Akan tetapi, tidak dapat dipungkiri bahwa teknologi tersebut dapat mengurangi motivasi seseorang untuk belajar. Hal ini dikarenakan banyak hal yang lebih menarik yang dapat dilakukan dengan memanfaatkan teknologi tersebut, seperti halnya bermain *game*. *Game* merupakan salah satu media yang dapat digunakan untuk membantu dalam mengenal hewan-hewan laut. *Game* pengenalan hewan laut yang termasuk dalam genre *racing game* memasukkan metode belajar dan bermain di dalamnya, yang dapat membuat seseorang mengetahui informasi mengenai hewan laut dan klasifikasi ilmiahnya.

Game pengenalan hewan laut ini dibangun menggunakan Unity, dan bahasa C#. *Game* ini memanfaatkan sensor *accelerometer* yang ada pada *smartphone* untuk menggerakkan karakternya. *Accelerometer* adalah sensor yang digunakan oleh sistem untuk mendeteksi orientasi suatu perangkat berdasarkan gerakan ke segala arah yang memungkinkan fitur untuk bertindak. Untuk proses merancang *game* ini, digunakan diagram HIPO (*Hierarchy plus Input Process Output*), dimana di dalamnya terdapat VTOC, Overview Diagram, dan Detail Diagram. Adobe Photoshop digunakan untuk proses pembuatan dan manipulasi gambar. Untuk membuat objek lintasan balap dan ikan, digunakan *software* Blender. Untuk membuat musik dan suara tombol, digunakan *software* FL Studio.

Tujuan dibangunnya *game* pengenalan hewan laut ini adalah untuk menyediakan media belajar alternatif yang menarik, dan membuat seseorang dapat belajar mengenal hewan laut secara tidak langsung. Dengan antarmuka yang sederhana dan *gameplay* yang tidak terlalu sulit sehingga dapat dimainkan oleh pengguna semua umur, *game* pengenalan hewan laut diharapkan dapat mencapai tujuan dari penelitian ini.

Untuk mencapai tujuan dari penelitian, dilakukanlah pengujian. Pengujian dilakukan dengan dua cara, yaitu pengujian dengan kuesioner dan pengujian perangkat. Dari pengujian yang telah dilakukan, *game* pengenalan hewan laut ini dapat menjadi media belajar alternatif yang efektif dan menarik bagi para pengguna.

Kata kunci: *accelerometer*, android, *game*, hewan laut, HIPO, racing, unity.

GLOSARIUM

Smartphone	Ponsel pintar.
Game	Permainan.
Racing Game	Permainan balapan.
Fighting Game	Permainan pertarungan.
Shooting Game	Permainan tembak-menembak.
Role-Playing	Permainan memainkan peran.
Gamer	Seseorang yang memainkan gim.
Developer	Pengembang perangkat lunak.
Software	Perangkat lunak
Game Engine	Perangkat lunak yang digunakan untuk membangun sebuah gim.
3D Modelling	Membangun objek 3 dimensi.
Platform	Perangkat sistem operasi tertentu.
Cross-Platform	Sistem yang dapat dijalankan pada berbagai sistem operasi.
Update	Mermperbarui.
Project Name	Nama proyek.
Location	Lokasi.
Scene	Adegan, panggung, layar.
Hierarchy	Hirarki.
Project	Proyek.
Inspector	Penilik, pemeriksa.
Web	Jaringan yang saling berhubungan.
Programmer	Seseorang yang menulis atau membangun program.
Framework	Software yang memudahkan programmer agar sistem lebih terstruktur.
Rigging	Proses penulangan dalam animasi.
Rendering	Proses akhir dari keseluruhan proses pemodelan ataupun animasi.
Compositing	Membuat komposisi.
Motion Tracking	Melacak gerak.
Shareware	Perangkat lunak yang dibagikan secara gratis.
Gameplay	Teknis permainan.
Puzzle	Teka-teki.
Open Scource	Bersifat terbuka.
Accelerometer	Sensor accelerometer.

Workstation	Komputer yang digunakan untuk perhitungan ilmiah atau teknis, atau keperluan tingkat tinggi lainnya.
Input	Masukan
Output	Keluaran
Background	Latar
Home	Halaman utama.
Splashscreen	Halaman yang muncul sebelum halaman utama
Storyboard	Alur cerita atau alur penggunaan sebuah aplikasi.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pesatnya perkembangan teknologi seharusnya dapat mempermudah seseorang untuk belajar. Akan tetapi, tidak dapat dipungkiri bahwa teknologi tersebut dapat mengurangi motivasi sesorang untuk belajar. Hal ini dikarenakan banyak hal yang lebih menarik yang dapat dilakukan dengan memanfaatkan teknologi tersebut, seperti halnya bermain *game*. Tidak sedikit dari kita yang memiliki pengetahuan minim tentang hewan-hewan laut, terutama hewan-hewan laut yang jarang dilihat pada kehidupan sehari-hari. Selain itu, masih banyak dari kita yang tidak mengetahui apakah spesies tertentu dari hewan tersebut merupakan spesies yang terancam punah atau tidak. Sehingga, terkadang ada beberapa tindakan berbahaya yang dilakukan berdasarkan ketidaktahuan tersebut, seperti menangkap ikan yang terancam punah dengan tujuan untuk dikonsumsi atau dipelihara. Banyak buku dan artikel di internet yang berisi informasi mengenai hewan laut. Namun buku-buku dan artikel tersebut tidak mampu menarik minat sesorang untuk membaca, meskipun buku-buku dan artikel sudah disusun dan dikemas dengan baik. Hal ini dikarenakan seseorang cenderung lebih tertarik dengan sesuatu yang sifatnya menyenangkan dan memilih bermain *game* dibanding membaca artikel atau buku.

Game adalah sebuah sistem dimana pemain terlibat dengan konflik buatan, yang didefinisikan oleh peraturan, yang menghasilkan hasil yang dapat dihitung (Salen & Zimmerman, 2003). Dengan kepemilikan *smartphone* telah menjadi sesuatu yang lazim, *game* sangat mudah untuk diakses dan sudah menjadi bagian dari aktifitas kehidupan sehari-hari. Salah satu dari sekian banyak kategori *game* yang populer adalah *Racing Game*. Para *developer* saling berlomba untuk menjadi yang terbaik dari sekian banyak *game developer* yang ada. Hal itu tentu saja merupakan hal yang positif, akan tetapi, dari sekian banyak game yang tersedia, masih dapat dikatakan sedikit *game* yang memiliki nilai edukasi di dalamnya.

Berdasarkan permasalahan itulah penulis berkeinginan untuk membangun sebuah *game* berbasis android sebagai media edukasi untuk mengenalkan nama-nama hewan laut, khususnya ikan dan memberikan informasi status spesies ikan tersebut. *Game* ini termasuk ke dalam kategori *Racing Game*, dimana pemain akan diminta mengumpulkan ikan sebanyak mungkin untuk di letakkan di akuarium dengan cara mengalahkan ikan tersebut melalui sebuah balapan.

Judul yang diangkat pada tugas akhir ini adalah "Game Pengenalan Hewan Laut Berbasis Android Menggunakan Accelerometer".

Game yang akan dibangun dengan menggunakan aplikasi Unity ini diharapkan dapat digunakan sebagai media alternatif untuk mempelajari nama hewan laut bagi anak-anak, remaja dan orang dewasa yang tidak tertarik untuk membaca buku maupun artikel di internet.

1.2 Rumusan Masalah

Berdasarkan dari masalah yang dijelaskan pada latar belakang, maka rumusan masalah yang diambil adalah bagaimana membangun sebuah "Game Pengenalan Hewan Laut Berbasis Android Menggunakan Accelerometer" agar dapat mempermudah dan mampu menarik perhatian seseorang untuk mengenal ataupun mempelajari hewan-hewan laut dengan cara yang lebih menyenangkan.

1.3 Batasan Masalah

Untuk membatasi pengerjaan yang akan dilakukan penulis, maka disusunlah batasan malasalah sebagai berikut:

a. Hewan laut yang digunakan dalam game ini hanya hewan yang termasuk di dalam spesies ikan.
b. Hanya 5 area balapan atau sirkuit yang diimplementasikan dalam game ini.
c. Hanya 10 jenis ikan yang digunakan.

1.4 Tujuan Penelitian

Membangun sebuah game berbasis android untuk mengenalkan nama-nama hewan laut, khususnya ikan sebagai media edukasi yang mampu menarik minat seseorang untuk belajar.

1.5 Manfaat Penelitian

Adapun manfaat yang diharapkan pada penyusunan tugas akhir ini adalah sebagai berikut:

a. Pengguna:
    1. Memberikan pengetahuan mengenai hewan laut kepada pengguna.
    2. Menyediakan media belajar alternatif yang menarik dan menyenangkan.
    3. Sebagai referensi bagi pengembang game sejenis.

b. Penulis:

1. Mem ahami proses pembuatan sebuah game atau sistem.
2. Pengetahuan mengenai hewan laut yang didapatkan ketika mengumpulkan data.
3. Memahami software yang digunakan untuk membuat sistem ini lebih dalam.

1.6 Metodologi Pembuatan *Game*

Dalam pembuatan aplikasi ini, tahap yang digunakan adalah sebagai berikut:

a. Pengumpulan Data

Pada tahap ini, metode yang digunakan adalah kepustakan. Metode kepustakaan digunakan untuk mengumpulkan informasi yang diperlukan terkait dengan masalah yang dihadapi, baik itu dari buku – buku, makalah, maupun data – data dari internet.

b. Tahap Pembangunan *Game*

1. Analisis Kebutuhan

   Tahap analisis digunakan digunakan untuk mengetahui dan menerjemahkan semua permasalahan serta kebutuhan perangkat lunak dan kebutuhan sistem yang akan dibangun. Metode ini menganalisis proses dan antarmuka yang dibutuhkan dalam pengembangan game yang dibuat dengan teratur sehingga game yang akan dibuat dapat didefinisikan dengan baik dan jelas.

2. Perancangan

   Tahap ini merupakan tahap di mana dilakukannya perancangan sistem, baik itu antarmuka, struktur, dll. Pada perancangan sistem ini, penulis menggunakan model HIPO (*Hierarchy Plus Input-Process-Output*). Model HIPO digunakan untuk mewakili modul-modul yang ada pada sistem sistem sebagai hierarki dan untuk mendokumentasikan setiap modul.

3. Implementasi

   Pada tahap ini, hasil dari rancangan sistem akan diimplementasikan menggunakan *software game engine*, 3D *modelling*, bahasa pemrograman dan *software* lain yang diperlukan dalam membangun sistem ini.

4. Pengujian

   Tahap pengujian terhadap game yang dibangun agar jika terjadi kesalahan maka dapat langsung diperbaiki. Tahap pengujian dapat dilakukan dengan mengujikan *game* yang telah dibangun kepada beberapa pengguna.

1.7 Sistematika Penulisan

Sistematika Penulisan dalam penyusunan tugas akhir ini terdiri dari 5 bab yang isinya adalah :

BAB I PENDAHULUAN

Pada bab ini, dijelaskan latar belakang dari judul yang telah dipilih oleh penulis, yaitu "Game Pengenalan Hewan Laut Berbasis Android Menggunakan Accelerometer" serta menjelaskan rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi pembuatan sistem, serta sistematika penulisan laporan.

BAB II LANDASAN TEORI

Pada bab ini dijelaskan tentang landasan-landasan teori yang mendukung dalam pembuatan *game* "Game Pengenalan Hewan Laut Berbasis Android Menggunakan Accelerometer" ini, seperti penjelasan mengenai *game*, android, serta penjelasan mengenai software-software yang akan digunakan.

BAB III METODOLOGI

Pada bab ini dijelaskan tentang langkah-langkah yang digunakan dalam pembuatan sistem, seperti analisis, metode yang digunakan, serta perancangan *game* itu sendiri.

BAB IV HASIL DAN PEMBAHASAN

Bab ini berisi penjelasan mengenai hasil implementasi dari *game* yang telah dirancang pada bab sebelumnya, pengujian terhadap *game*, dan penjelasan dari fungsi-fungsi yang ada pada *game*.

BAB V SARAN DAN KESIMPULAN

Pada bab ini akan dijelaskan rangkuman dari kesimpulan-kesimpulan yang didapat dari analisis yang telah dilakukan sebelumnya. Bab ini juga berisikan saran dari penulis mengenai *game* "Game Pengenalan Hewan Laut Berbasis Android Menggunakan Accelerometer" untuk dikembangkan lebih lanjut agar menjadi lebih baik.

BAB II
LANDASAN TEORI

2.1 *Game*

Game adalah sebuah sistem dimana pemain terlibat dengan konflik buatan, yang didefinisikan oleh peraturan, yang menghasilkan hasil yang dapat dihitung (Salen & Zimmerman, 2003) . Pada permainan atau *game*, terdapat berbagai aturan yang harus dipahami oleh penggunanya. *Game* juga memiliki skenario agar alur permainan jelas dan terarah. Skenario disini bisa meliputi level, alur cerita, bahkan efek yang ada didalam *game* tersebut.

2.1.1 Klasifikasi Game

Indonesia memiliki sistem yang akan membantu orang tua atau bahkan *gamer* itu sendiri, untuk mendapatkan konten *game* yang sesuai dengan usia mereka. Klasifikasi usia di Indonesia ditentukan oleh IGRS (Indonesia *Game Rating System*). Berdasarkan data yang ditampilkan pada halaman resmi milik IGRS (IGRS.ID, 2016), klasifikasi usia yang ada pada Indonesia adalah sebagai berikut:

a. Kelom pok usia pengguna 3 (tiga) tahun atau lebih,
b. Kelompok usia pengguna 7 (tujuh) tahun atau lebih,
c. Kelompok usia pengguna 13 (tiga belas) tahun atau lebih,
d. Kelompok usia pengguna 18 (delapan belas) tahun atau lebih, dan
e. Kelompok pengguna semua usia. Kelompok pengguna usia yang dimulai dari usia 7 (tujuh) tahun.

Berdasarkan klasifikasi yang telah ditentukan oleh IGRS, *game* Pengenalan Hewan Laut ini termasuk ke dalam kategori kelompok pengguna semua umur. Hal ini dikarenakan *game* Pengenalan Hewan Laut tidak memiliki konten yang berkaitan dengan zat adiktif, kekerasan, penggunaan bahasa yang kasar, hal-hal yang berbau seksual, dan simulasi judi.

2.1.2 Genre

Game dapat dikategorikan berdasarkan *genre*. Beberapa *genre game* yaitu:

a. Aksi

Game aksi memerlukan refleks, akurasi, dan waktu yang tepat untuk menyelelesaikan sebuah tantangan. Terdapat beberapa sub-genre yang popular dari permainan aksi, seperti *Fighting Games* dan *Shooting Games*.

b. Petualangan-Aksi

Petualangan-aksi menggabungkan elemen dari dua genre, biasanya menampilkan hambatan jangka panjang yang harus diatas dengan menggunakan alat atau item yang telah dikumpulkan sebelumnya. Untuk menyelesaikan hambatan atau rintangan yang ada, dibutuhkan sedikit unsur dari *game* aksi. Genre ini juga berfokus pada eksplorasi, pengumpulan barang, dan pertarungan.

c. Petualangan

Game petualangan menggambarkan permainan yang tidak membutuhkan refleks yang tinggi ataupun aksi. Genre ini biasanya meminta pemain memecahkan berbagai teka-teki atau masalah dengan cara berinteraksi dengan orang-orang atau lingkungan sekitar.

d. Role-Playing

Game Role-Playing adalah *game* bermain peran. Pemain memiliki peran sebagai tokoh utama dari sebuah cerita. Unsur yang paling sering dikaitkan dengan genre ini adalah pengembangan karakter melalui poin pengalaman yang diperoleh ketika bermain.

e. Simulasi

Game simulasi didesain untuk mensimulasikan aspek-aspek yang ada di dunia nyata, seperti simulasi membangun kota, simulasi kehidupan sehari-hari, dll.

f. Strategi

Game strategi fokus pada permainan yang membutuhkan pemikiran dan perencanaan yang hati-hati untuk meraih sebuah kemenangan.

g. Olahraga

Game olahraga merupakan *game* yang mensimulasikan olahraga yang ada di dunia nyata, seperti sepak bola, basket, balap, dll.

Game Pengenalan Hewan Laut ini termasuk ke dalam genre olahraga. Hal ini dikarenakan *game* Pengenalan Hewan Laut mensimulasikan olahraga balap. Hanya saja, karakter yang melakukan balapan merupakan objek yang berbentuk ikan.

2.2 Multimedia

Multimedia adalah sebuah konten yang menggunakan kombinasi dari berbagai jenis konten seperti teks, suara, gambar, animasi, video, dan konten interaktif. Multimedia juga dapat

di artikan sebagai penggunaan komputer untuk menyajikan dan menggabungkan teks, suara, gambar, animasi, video dengan alat bantu (*tool*) dan koneksi (*link*) sehingga pengguna dapat melakukan navigasi, berinteraksi, berkarya, dan berkomunikasi (Bagaimanamultimedia, 2013). Berikut adalah definisi multimedia menurut beberapa ahli:

a. Menurut Steinmetz (1995), multimedia adalah gabungan dari seminimalnya sebuah media diskrit dan sebuah media kontinu. Media diskrit adalah sebuah media dimana validitas datanya tidak tergantung dari kondisi eaktu, termasuk di dalamnya teks dan grafik. Sedangkan yang dimaksud dengan media kontinu adalah sebuah media dimana validitas datanya tergantung dari kondisi waktu, termasuk di dalamnya suara dan video.
b. Menurut Rosch (1996) , multimedia adalah kombinasi dari komputer dan video.
c. Menurut McComick (1996), multimedia adalah kombinasi dari tiga elemen: suara, gambar, dan teks.
d. Menurut Najjar (1996), multimedia adalah penyampaian informasi meggunakan gabungan dari teks, grafik, suara, video, dan animasi.
e. Robin dan Linda (2001), multimedia adalah alat yang dapat menciptakan presentasi yang dinamis dan interaktif yang mengkombinasikan teks, grafik, animasi, audio, dan video.
f. Menurut Vaughan (2004), multimedia adalah beberapa kombinasi dari teks, gambar, suara, animasi dan video dikirim ke anda melalui komputer atau alat elektronik lainnya atau dengan manipulasi digital.

2.3 Android

Android merupakan sistem operasi berbasis Linux yang didesain khususnya untuk perangkat yang memiliki fitur layar sentuh seperti *smartphone* dan komputer tablet (Wikipedia, 2017b). Android menyediakan *platform* yang bersifat *open source* bagi para pengembang untuk membangun aplikasi mereka sendiri untuk digunakan pada *mobile device*.

Setiap sistem operasi memiliki kelebihan dan kekurangan jika dibandingkan dengan sistem operasi lainnya. Berikut adalah beberapa kelebihan dan kekurangan yang ada pada sistem operasi Android:

a. Kelebihan :
    1. Bersifat open source, sehingga mudah dikembangkan oleh developer.
    2. Mudah untuk di kustomisasi oleh para pengguna yang mengerti.
    3. Sistem operasi android dapat dijalankan pada banyak pilihan dan spesifikasi perangkat yang tersedia.

4. Dukungan aplikasi yang banyak dan beragam.

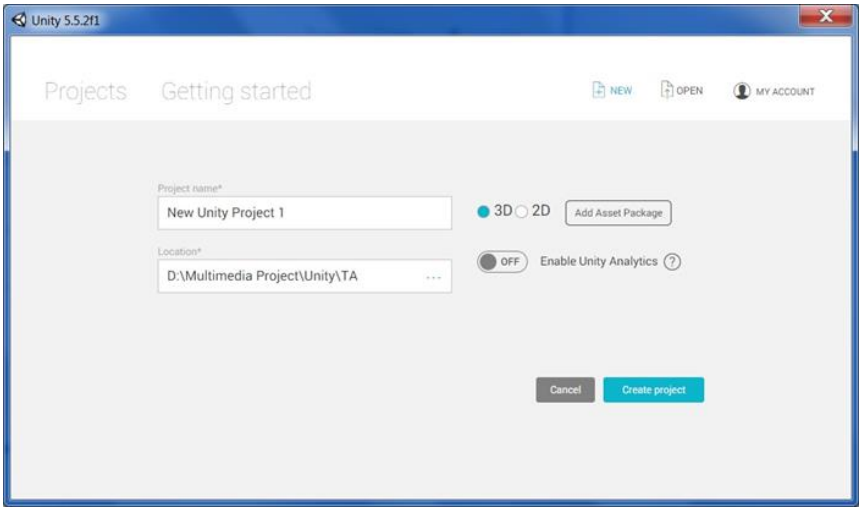
b. Kekurangan

1. Karena sifatnya open source, maka banyak yang melakukan kustomisasi pada sistem operasi ini, sehingga sistem operasi sering tidak stabil dan kurang optimal.
2. Tersedianya update sistem operasi pada perangkat yang digunakan bergantung pada developer yang mengembangkan, sehingga jangka waktu tersedianya update sistem untuk perangkat yang dikembangkan oleh pengembang tertentu akan berbeda dengan perangkat yang di kembangkan oleh pengembang lainnya.

Tingkat respon bergantung pada spesifikasi perangkat yang digunakan. Oleh karena itu, jika disandingkan dengan perangkat yang memiliki kapasitas memori yang cukup kecil dan prosesor yang kurang mumpuni, maka sistem operasi akan terasa lambat.

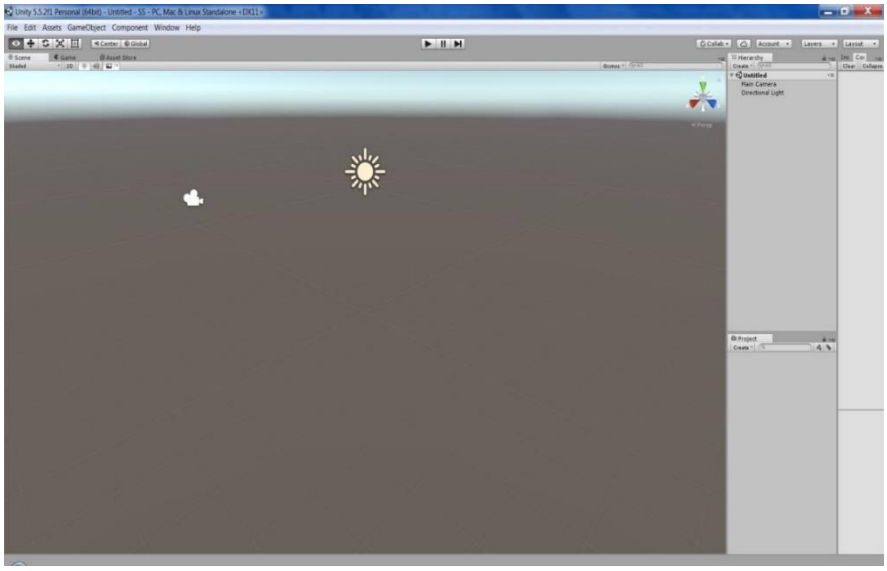
2.4 Unity

Unity 5 merupakan versi terbaru dari *cross-platform game engine* Unity yang dikembangkan oleh Unity Technologies yang khususnya digunakan untuk membuat *game* dan simulasi pada komputer (Wikipedia, 2017d). Unity pertama kali diumumkan atau diluncurkan pada tahun 2005. Saat itu Unity hanya bisa digunakan untuk OS X milik Apple. Unity terus dikembangkan hingga akhirnya saat ini dapat digunakan di berbagai platform dan mampu menghasilkan *game* yang dapat digunakan oleh 27 platform. *Game* engine ini mendukung grafis 2D dan 3D, fungsi drag and drop, dan 3 script khusus Unity, yaitu C#, UnityScript dan Boo. Unity memiliki antarmuka yang mudah digunakan. Meskipun mudah digunakan, bukan berarti *game engine* ini dapat diremehkan. Sehingga, dapat dikatakan bahwa Unity cukup mudah digunakan bagi pemula dalam membangun *game* yang sederhana dan cukup kuat untuk bagi para ahli yang ingin membangun *game* yang memiliki tingkat kompleksitas yang tinggi.



Gambar 2.1 Jendela awal Unity

Halaman awal adalah tampilan yang pertama kali ditampilkan ketika pertama kali menjalankan aplikasi Unity 5. Pada halaman ini terdapat 2 pilihan, yaitu membuat proyek baru dan membuka proyek yang pernah disimpan sebelumnya. Ketika membuat proyek baru, pengguna dapat mengetikkan nama proyek yang akan dibuat pada kolom *Project name* dan menentukan dimana lokasi proyek tersebut disimpan pada kolom *Location*. Selain itu, pengguna juga dapat memilih proyek yang akan dibuat bersifat 2D atau 3D dan menambahkan paket aset yang disediakan oleh Unity 5 ke dalam proyeknya. Tampilan jendela awal Unity dapat dilihat pada Gambar 2.1.



Gambar 2.2 Jendela *scene* Unity

Gambar 2.2 diatas adalah jendela *scene* Unity 5 dimana jendela tersebut akan muncul ketika pengguna membuat proyek baru. Jendela *scene* adalah area pandang 3D dimana pengguna dapat mengatur aset secara fisik dengan memindahkannya di dalam area ruang 3D. Panel *Hierarchy* merupakan panel tempat aset yang digunakan di dalam *scene* dikelola. Aset dari panel *Project* dapat diseret ke panel Hierarchy untuk menambahkannya pada scene yang sedang dikerjakan oleh pengguna. Panel *Project* merupakan tempat semua aset dalam sebuah proyek disimpan. Saat aset diimpor, aset-aset tersebut akan muncul di panel ini. Panel *Inspector* digunakan untuk melihat dan mengubah atribut dari aset yang dipilih, seperti posisi, rotasi, apakah aset tersebut dipengaruhi gravitasi, dan bayangan.

2.5 Accelerometer

Accelerometer adalah sensor yang digunakan untuk mengukur percepatan suatu objek. *Accelometer* mengukur percepatan dinamis dan statis. Pengukuran dinamis adalah pengukuran percepatan pada objek bergerak, sedangkan pengukuran statis adalah pengukuran terhadap gravitasi bumi, atau dapat disebut untuk mengukur sudut kemiringan. (Oktriaviani, 2012)

Sensor *accelerometer* merupakan sebuah fitur yang ditanam pada perangkat *smartphone* yang biasanya digunakan untuk mengukur sudut kemiringan dari *smartphone* itu sendiri. Pada dasarnya, fungsi sensor ini untuk mengubah orientasi layar dari posisi *landscape* menjadi *potrait*, ataupun sebaliknya sehingga tampilan antarmuka akan menyesuaikan posisi dari *smartphone*. Selain itu, sensor *accelerometer* ini juga banyak digunakan pada *game* berbasis android, seperti pada *game* balapan, sensor ini digunakan untuk membelokkan mobil. Berikut adalah beberapa jenis dari accelerometer:

a. Piezoelektrik *accelerometer*
b. Modus geser *accelerometer*
c. Permukaan *micromachined* kapasitif (MEMS)
d. Termal (*submicrometre* CMOS proses)
e. *Micromachined* Massal kapasitif
f. *Micromachined* Massal resistif piezoelektrik
g. Kapasitif pegas massa dasar
h. Elektromekanis Servo (Servo Angkatan Saldo)
i. *Null-balance*
j. Resonansi
k. Induksi magnetik

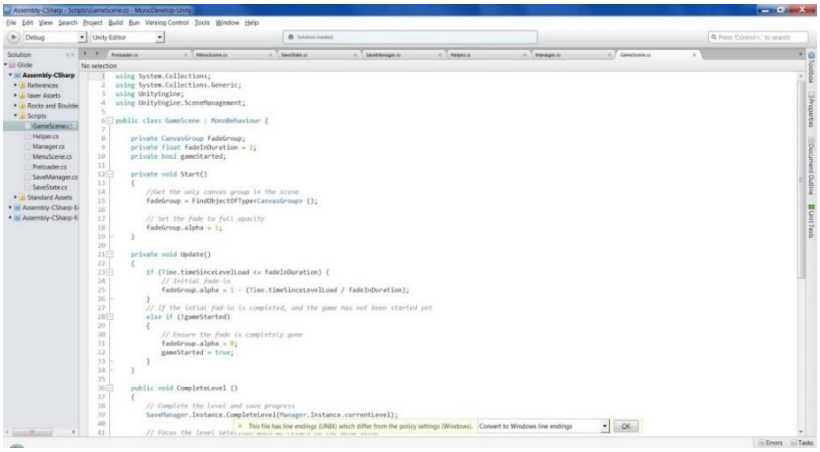
l. Optik
m. Permukaan gelombang akustik (SAW)
n. Laser *accelerometer*
o. DC respons
p. Suhu tinggi
q. Frekuensi rendah
r. Tinggi gravitasi

Sensor *accelerometer* yang ada pada sebuah *smartphone* merupakan sensor permukaan *micromachined* kapasitif atau biasa disebut dengan MEMS. *Micro Electro Mechanical Systems* (MEMS) adalah struktur perangkat elektro-mekanik terdiri dari sensor mikro, aktuator mikro, dan struktur lainnya di dalam ukuran miniatur seukuran rangkaian sirkuit terpadu.

2.6 C# 6.0

C# 6.0 merupakan salah satu versi dari bahasa pemrograman C#. C# adalah bahasa pemrograman berorientasi objek yang dikembangkan oleh Microsoft yang bertujuan menggabungkan kekuatan komputasi bahasa pemorgraman C++ dengan kemudahan pemrograman Visual Basic. (Wikipedia, 2017c). C# didasarkan pada C++ dan berisi fitur yang serupa dengan Java. C# dirancang untuk mampu bekerja dengan *platform* .Net milik Microsoft. Tujuan dari dikembangkannya C# adalah untuk memfasilitasi pertukaran informasi dan layanan melalui *Web* dan untuk memungkinkan *developer* membangun aplikasi yang sangat portabel.

C# menyederhanakan pemrograman melalui penggunaan *Extensible Markup Language* (XML) dan *Simple Object Access Protocol* (SOAP) yang memungkinkan akses ke metode atau objek pemrograman tanpa mengharuskan *programmer* menulis kode tambahan untuk setiap langkah. Hal itu dikarenakan *programmer* dapat membangunnya menggunakan kode yang sudah ada tanpa harus menduplikatnya berulang kali. Selain itu, Microsoft juga berkolaborasi dengan ECMA, badan standar internasional untuk menciptakan standar untuk bahasa pemrograman C#.



Gambar 2.3 Kode C# yang ditulis menggunakan MonoDevelop

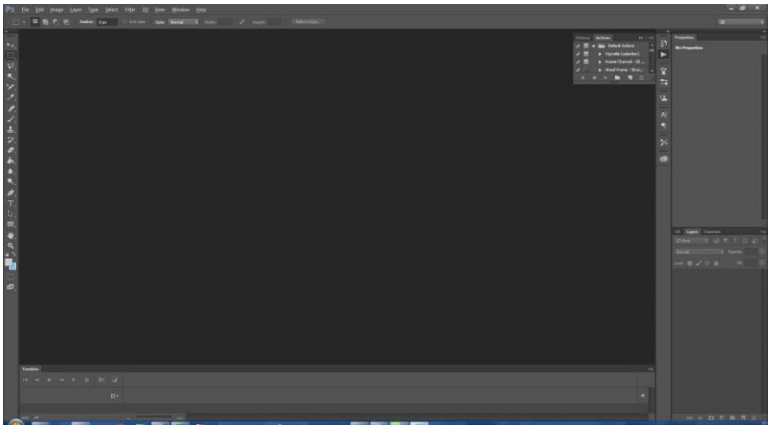
Pada aplikasi Unity 5, kode C# ditulis menggunakan MonoDevelop yang merupakan IDE bawaan dari Unity 5. MonoDevelop merupakan IDE (*Integrated Development Environtment*) yang dapat digunakan pada Windows, Linux, dan Mac OS. Fokus utama dari penggunaan MonoDevelop adalah pengembangan proyek yang menggunakan *framework* Mono dan .Net. Tampilan MonoDevelop dapat dilihat pada Gambar 2.3.

2.7 Adobe Photoshop CC 2015

Photoshop CC 2015 adalah salah satu produk unggulan dari seri Adobe Photoshop milik Adobe Systems. Adobe Photoshop adalah editor grafis yang dikembangkan dan di publikasikan oleh Adobe Systems untuk MacOS dan Windows.(Wikipedia, 2017a) Photoshop pertama kali dikembangkan pada 1987 oleh Thomas Knoll untuk menampilkan gambar *grayscale* pada layar monokrom. Program ini awalnya bernama "Display", sebelum akhirnya John Knoll menyarankan Thomas untuk mengubah program tersebut menjadi program untuk mengolah gambar. Thomas berkolaborasi dengan Knoll membangun program tersebut dan berencana mengubah nama program menjadi ImagePro. Akan tetapi, nama tersebut telah digunakan. Tahun berikutnya, Thomas mengubah nama program tersebut menjadi Photoshop. Pada tahun 1988, lisensi distribusi Photoshop dibeli oleh Adobe Systems dan terus dikembangkan hingga menjadi salah satu program untuk mengolah gambar yang sering digunakan. Berkas yang dihasilkan oleh Adobe Photoshop memiliki tipe file extension berupa .PSD dan terkadang .PSB.

Adobe Photoshop merupakan program yang digunakan untuk mengolah gambar yang umumnya berjenis bitmap. Selain untuk mengedit, Photoshop juga dapat digunakan untuk

membuat gambar dan animasi sederhana. Photoshop dapat digunakan pada sistem operasi Microsoft Windows, Mac OS, dan Linux



Gambar 2.4 Jendela utama Adobe Photoshop CC 2015

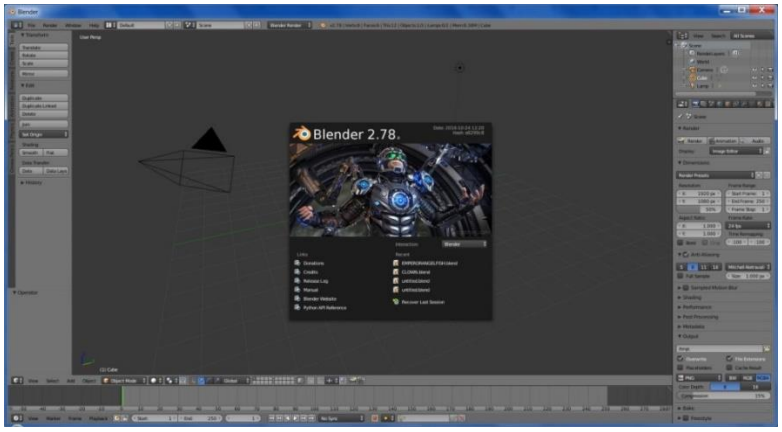
Ketika pertama kali menjalankan Adobe Photoshop, akan ditampilkan jendela utama dimana pengguna dapat memulai untuk mengolah gambar. Seperti yang dapat dilihat pada Gambar 2.4, pengguna dapat menekan menu “File” dan selanjutnya memilih “New” untuk membuat dokumen baru.

2.8 Blender

Blender merupakan software grafis komputer 3D komputer yang bersifat *open source*. (Blender.org., 2017). Software ini mendukung pemodelan 3D, *rigging*, animasi, simulasi, *rendering*, *compositing*, *motion tracking*, bahkan mengedit video dan pembuatan *game*. Studio animasi Belanda Neo Geo mengembangkan Blender pada bulan Januari 1995, dengan pencipta utama yang merupakan seorang *software developer* yang bernama Ton Roosendaal. Nama “Blender” terinspirasi oleh sebuah lagu milik Yello dari album Baby. Ketika Neo Geo diakuisisi oleh perusahaan lain, Ton Roosendaal dan Frank van Beek mendirikan Not a Number Technologies (NaN) pada Juni 1998 untuk mengembangkan Blender lebih lanjut, dengan tujuan untuk mendistribusikan Blender sebagai *shareware* sebelum akhirnya NaN bangkrut pada tahun 2002.

Karena sifat *open source* dari Blender, beberapa program lain memanfaatkan kesuksesan yang diperoleh Blender dengan mengemas ulang dan menjual versi modifikasinya. Blender memiliki sistem file internal yang dapat mengemas beberapa *scene*, objek, material, tekstur, suara, gambar, dan efek ke dalam sebuah berkas dengan ektensi .blend. Selain itu, berbagai

macam skrip impor / ekspor memungkinkan Blender untuk melakukan inter-operasional dengan *software* 3D lainnya.



Gambar 2.5 Jendela awal Blender

Gambar 2.5 merupakan tampilan awal ketika pengguna pertama kali menjalankan Blender. Terdapat beberapa tautan yang dapat dilihat pengguna seperti donasi, *website* Blender, petunjuk pengguna, dan lain-lain. Pada bagian *recent,* terdapat beberapa berkas yang sebelumnya pernah dibuat oleh pengguna.

2.9 FL Studio

FL Studio (yang sebelumnya diketahui sebagai FruityLoops) merupakan *Digital Audio Workstation* yang dikembangkan oleh perusahaan Image-Line asal Belgia (Wikipedia, 2016). Versi pertama dari FruityLoops dikembangkan oleh Didier Dambrin untuk Image-Line dan dirilis sebagian pada Desember 1997. Peluncuran resmi FruityLoops dilakukan pada tahun 1998. Dambrin selaku pengembang software tersebut menjadi kepala arstitek perangkat lunak bagi Image-Line dan bertugas mengembangkan software tersebut hingga akhirnya menjadi Digital Audio Workstation yang kompleks dan popular. FL Studio telah mengalami 12 peningkatan besar sejak pertama kali dikembangkan. Pada Juni 2015, telah diluncurkan FL Studio untuk sistem operasi MacOS.

FL Studio memiliki fitur antarmuka grafis berdasarkan pada musik *sequencer* berbasis pola. Untuk perangkat yang menajalankan sistem operasi Windows, program ini memiliki tiga edisi berbeda, yaitu Fruity Edition, Producer Edition, dan Signature Bundle. Diantara ketiga edisi tersebut, hanya Fruity Edition yang bisa diapatkan secara gratis. Image-Line menawarkan gratis update seumur hidup yang berarti pelanggan dapat menerima semua update yang akan

datang secara gratis. FL Studio juga dapat diguakan sebagai instrumen virtual dan klien ReWire. Tampilan antarmuka awal ketika FL Studio 11 dijalankan dapat dilihat pada Gambar 2.6.

Gambar 2.6 Antarmuka halaman awal FL Studio

2.10 Review Aplikasi Sejenis

Terdapat beberapa aplikasi yang memiliki *gameplay* sejenis. Oleh karena itu, akan dilakukan perbandingan terhadap aplikasi-aplikasi tersebut. Dari perbandingan yang ada, akan diteliti setiap kekurangan dan kelebihan yang dimiliki oleh masing-masing aplikasi. Data yang diperolah akan menjadi acuan dalam pengembangan *game* yang akan penulis bangun. Data tersebut dapat dilihat pada Tabel 2.1.

Tabel 2.1 Aplikasi sejenis

Nama	Objek *Game*	Bahasa	Platform	Kelebihan	Kekurangan
Dolphin Racing 3D	Permainan balap ikan dengan menggunakan ikan lumba-lumba.	Bahasa Inggris	Android	- Aplikasi yang ringan. - Grafis sudah cukup baik. - Mudah untuk dimainkan. - Menggunakan accelerometer.	- *Gameplay* hanya berupa balapan ikan. - Hanya satu jenis ikan yang ada pada permainan.
Fish Adventu-res	Memelihara ikan.	Bahasa Inggris	Android	- Aplikasi yang ringan. - Grafis lebih baik jika dibandingkan dengan Dolphin Racing 3D.	- *Gameplay* hanya berupa memelihara ikan. - Untuk menambahkan ikan jenis lain, pemain diminta

				- Banyak fitur menarik seperti membersihkan akuarium, memberi makan ikan, dan mengembangb-iakan ikan. - Jenis ikan yang disediakan sangat banyak disertai nama ikan tersebut.	untuk membeli ikan tersebut melalui toko yang ada di dalam *game*. - Harga ikan di dalam sangat mahal dibandingkan jumlah koin yang didapat ketika merawat ikan.
Fishdom	*Puzzle* dan memelihara ikan.	Bahasa Inggris	Android	- Memiliki grafis yang sangat bagus. - Terkadang ikan akan berbicara kepada pemain melalui teks. - Disediakan fitur memberi makan ikan dan membersihkan akuarium. - Akuarium dapat didekorasi sesuai keinginan pemain dengan menggunakan benda-benda yang disediakan di toko.	- *Gameplay Puzzle* yang cukup sulit dan membutuhkan waktu berhari-hari untuk menyelesaikan satu level jika pemain sudah mencapai level yang cukup tinggi. - Ikan didapatkan dengan cara dibeli di toko yang ada di dalam *game*. - Koin yang didapat ketika menyelesaikan *puzzle* sangat sedikit jika dibandingkan dengan harga ikan dan benda-benda yang digunakan sebagai dekorasi akuarium.

a. Dolphin Racing 3D

Game yang berjudul Dolphin Racing 3D ini termasuk ke dalam *Racing Game* dimana pemain diminta untung menggerakkan lumba-lumba dalam sebuah balapan. Pemain menggerakkan lumba-lumba ke kanan atau kiri dengan menggunakan accelerometer, *drag* ke

atas untuk melompat, *drag* ke bawah untuk menyelam, dan dan menyentuh dan menahan bagian bawah layar sebelah kiri untuk memperlambat ikan berenang. Tampilan *game* Dolphin Racing 3D dapat dilihat pada Gambar 2.7.

Gambar 2.7 *Gameplay* Dolphin Racing 3D

b. Fish Adventures

Pada *game* ini, pemain diminta untuk mengelola ikan dan akuarium yang disediakan, baik itu menjaga kebersihan akuarium, maupun memberi makan ikan. Untuk mendapatkan ikan, pemain harus membeli di toko yang ada di dalam game menggunakan koin yang didapat dengan cara memelihara ikan dan akuarium. Selain membeli, ikan juga dapat diperoleh dengan cara mengembang biakan ikan yang sejenis. Tampilan game Fish Adventures dapat dilihat pada Gambar 2.8.

Gambar 2.8 *Gameplay* Fish Adventures

c. Fishdom

Fishdom merupakan sebuah *game puzzle* dengan fitur tambahan, yaitu memelihara ikan. Pemain diminta untuk menyelesaikan level dengan cara memindahkan bentuk-bentuk yang

sama agar sebaris dengan jumlah pergerakan yang terbatas. Semakin tinggi level yang dimainkan, akan semakin banyak pula kondisi yang harus dipenuhi untuk menyelesaikan sebuah level. Seperti yang dapat dilihat pada Gambar 2.10, pemain diminta untuk mengumpulkan cumi sebanyak 12 ekor dan mengumpulkan emas sebanyak 6 buah dalam 33 pergerakan. Setelah berhasil menyelesaikan level, pemain akan mendapatkan koin. Koin tersebut digunakan untuk membeli ikan, dan benda-benda yang digunakan sebagai dekorasi akuarium. Selain menyelesaikan level, koin juga dapat diperoleh dengan cara memberi makan ikan dan membersihkan akuarium. Akan tetapi, jumlah koin yang diperoleh dengan merawat ikan dan akuarium dapat dikatakan sangat sedikit jika dibandingkan dengan koin yang diperoleh dengan cara menyelesaikan level. Tampilan akuarium pada game Fishdom dapat dilihat pada Gambar 2.9.

Gambar 2.9 Akuarium di dalam *game* Fishdom

Gambar 2.10 *Gameplay* puzzle pada *game* Fishdom

BAB III
METODOLOGI

3.1 Gambaran Umum *Game*

Game Pengenalan Hewan Laut ini merupakan *game* bergenre *Racing Game* dimana pemain diminta untuk menggerakkan karakter yang berupa ikan untuk memenangkan balapan melawan ikan lainnya yang digerakkan oleh komputer. Dalam permainan ini, pemain akan balapan melawan ikan lain pada sirkuit yang diacak oleh sistem. Ketika pemain memenangkan balapan, pemain akan mendapatkan koin. Koin ini akan digunakan untuk membeli ikan yang nantinya akan ditempatkan pada akuarium. Jika pemain kalah, pemain tidak akan mendapatkan apapun. Permainan ini juga memiliki halaman ensiklopedia yang berisi informasi mengenai ikan yang ada di dalam *game*.

3.2 Analisis

Sebelum melakukan perancangan *game*, akan dilakukan analisis terlebih dahulu. Analisis dilakukan dengan tujuan untuk memahami *game* yang akan dibangun, sehingga penulis dapat mempersiapkan segala kebutuhan yang nantinya akan dibutuhkan dalam proses membangun *game*. Dalam hal ini, *game* yang akan dibangun berjudul "Game Pengenalan Hewan Laut".

3.2.1 Analisis Kebutuhan *Game*

Analisis kebutuhan mencakup penentuan kebutuhan atau kondisi yang harus dipenuhi dalam membangun suatu sistem dengan mempertimbangkan apa saja yang dibutuhkan untuk membangun sistem tersebut. Kebutuhan yang dimaksud seperti apa yang harus dilakukan oleh sistem, perangkat yang digunakan untuk membuat, dan perangkat untuk menjalankan sistem.

Analisis kebutuhan sangat diperlukan untuk mendukung kinerja dari *game* yang akan dibangun. Untuk mempermudah menentukan kebutuhan secara keseluruhan, maka kebutuhan sistem dapat dibagi menjadi dua jenis, yaitu kebutuhan fungsional dan kebutuhan non-fungisonal.

3.2.2 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional merupakan jenis kebutuhan yang berisis proses-proses apa saja yang nantinya akan dilakukan oleh sistem. Selain itu, kebutuhan fungsional juga berisi

informasi mengenai apa yang harus dihasilkan sistem dalam kondisi tertentu. Berikut adalah kebutuhan fungsional pada *Racing Game* Pengenalan Hewan Laut:

a. *Game* dapat menampilkan menu utama yang di dalamnya terdapat beberapa pilihan menu, yaitu Main, Akuarium, Cara Bermain, dan Keluar.
b. Saat pemain memilih menu Main, pemain akan memulai balapan pada sirkuit yang diacak oleh sistem, dan pemain akan mendapatkan koin jika pemain berhasil memenangkan balapan. Koin tersebut nantinya akan digunakan untuk membeli ikan.
c. Pada saat pemain memilih menu Akuarium, *game* akan menampilkan akuarium 3D yang di dalamnya terdapat ikan-ikan yang telah dibeli oleh pemain. Di dalam menu ini terdapat sub menu toko dan ensiklopedia. Pada sub menu toko, pemain dapat membeli dan memilih karakter yang akan digunakan ketika balapan. Pada sub menu ensiklopedia, terdapat beberapa informasi mengenai ikan, seperti nama ilmiah, dan nama umum dari ikan tersebut.
d. Jika pemain memilih menu Cara Bermain, *game* akan menampilkan informasi mengenai bagaimana cara memainkan *game* tersebut.
e. Ketika pemain memilih menu keluar, *game* akan ditutup.

3.2.3 Analisis Kebutuhan Non-Fungsional

Analisis kebutuhan non-fungsional merupakan analisis yang berisi informasi mengenai perangkat yang akan digunakan untuk membangun dan pengujian *game* untuk mendukung kelancaran saat proses tersebut dilakukan. Perangkat tersebut antara lain :

a. Perangkat keras yang digunakan dalam pembuatan *game* :
    1. Prosesor AMD FX™ Vishera 6300 6-core 3.5 Ghz.
    2. Memori 8 GB DDR 3.
    3. Kapasitas penyimpanan 1 TB.
    4. VGA Radeon HD RX 560 4 GB.
b. Perangkat lunak yang digunaka dalam pembuatan *game* :
    1. Sistem operasi Windows 7 SP 1.
    2. Unity 5 untuk membangun *game*.
    3. Bahasa pemrograman C#.
    4. Adobe Photoshop CC 2015 untuk mendesain tombol dan antarmuka lainnya.
    5. Blender 2.78 untuk membuat objek ikan 3D.
    6. FL Studio 11 untuk membuat musik yang akan dimainkan ketika *game* dijalankan.

c. Perangkat keras yang digunakan untuk pengujian *game* :
   1. Prosesor MediaTek MT6592 8-core 1.4 GHz Cortex-A7.
   2. Kapasitas penyimpanan 16 GB.
   3. 2 GB RAM.
   4. GPU Mali-450.

d. Perangkat lunak yang digunakan untuk pengujian *game* :
   1. Sistem Operasi Android versi 5.1 (Lollipop).

3.2.4 Analisa Kelayakan

Pada bagian ini, penulis membagi analisis kelayakan menjadi 4 bagian, yaitu :

a. Kelayakan Teknologi

   Pada segi teknologi, *game* ini dapat dikatakan layak. Hal ini dikarenakan *smartphone* yang beredar di pasaran sudah memiliki spesifikasi yang cukup tinggi dengan harga yang terjangkau.

b. Kelayakan Operasional

   Dalam segi operasional, *game* ini dapat dikatakan layak karena sebagian besar masyarakat sudah memiliki *smartphone* dan mampu mengoperasikan *smartphone* dengan baik.

c. Kelayakan Hukum

   Game ini tidak mengandung unsur SARA dan pornografi sehingga *game* yang akan dibangun dapat dikatakan layak.

d. Kelayakan Ekonomi

   Dalam segi ekonomi, *game* ini dapat dikatakan layak. Hal ini karena untuk membangun *game* ini tidak dibutuhkan dana yang cukup besar. Selain itu, *game* ini dapat juga digunakan sebagai media promosi dengan cara memasukkan iklan ke dalam *game*.

3.3 Konsep *Game*

Game yang berjudul "Game Pengenalan Hewan Laut" ini dirancang sebagai media pembelajaran alternatif untuk mengingat nama-nama hewan laut. *Game* ini adalah sebuah *game* balapan atau *Racing Game* dengan menggunakan ikan laut sebagai karakter yang dimainkan oleh pemain. Ikan akan bergerak maju secara otomatis, sehingga pada saat balapan, pemain hanya diminta untuk membelokkan ikan tersebut dengan cara memiringkan *smartphone* ke kanan untuk berbelok ke kanan dan sebaliknya atau dengan menggunakan *touch input* dengan menyentuh layar smartphone dan menggeserkan ke kiri atau ke kanan.

Ketika memilih main, pemain akan diarahkan ke halaman dimana balapan akan dilakukan. Sirkuit yang digunakan untuk balapan akan diacak setiap kali pemain melakukan balapan. Dalam hal ini, penulis hanya akan menyediakan lima arena balapan. Setiap kali pemain memenangkan pertandingan, pemain akan mendapatkan koin. Koin inilah yang nantinya digunakan untuk membeli ikan. Ikan yang telah dibeli oleh pemain akan ditempatkan di dalam akuarium. Setelah menyelesaikan balapan, sistem akan menampilkan hasil balapan apakah pemain menang atau kalah dan menampilkan jumlah koin yang diperoleh pemain.

Ikan yang digunakan pemain untuk balapan dan ikan yang telah dibeli oleh pemain ditempatkan akan ditempatkan di akuarium. Pemain dapat masuk ke halaman akuarium melalui menu akuarium. Pada halaman ini, sistem akan menampilkan akuarium 3D. Selain itu, terdapat tombol toko yang jika disentuh akan menampilkan kotak dialog toko, dimana akan ditampilkan ikan apa saja yang dapat dibeli oleh pemain. Selain tombol toko, di menu akuarium ini terdapat tombol ensiklopedia. Ketika pemain memilih tombol ensiklopedia, akan ditampilkan informasi mengenai ikan-ikan yang ada di dalam *game*, seperti nama ilmiah, nama umum, ordo, kelas, dan famili.

3.4 Perancangan Game

Tahap perancangan merupakan tahapan dimana data-data yang sudah dikumpulkan sebelumnya pada tahap analisis dituang ke sebuah gambaran awal dari *game* yang akan dibangun sebelum diimplementasikan. Metode yang digunakan pada tahap ini adalah metode *Hierarchy Plus Input Process Output* atau yang biasa disingkat menjadi HIPO. HIPO bertujuan untuk menunjukkan hubungan antara modul dan fungsi pada *game* yang akan dibangun dan memberikan gambaran dari struktur *game* itu sendiri.

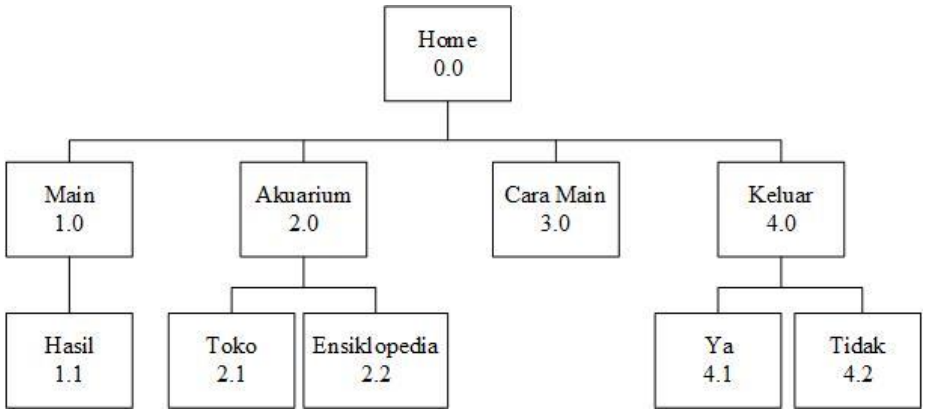
3.4.1 HIPO

Metode perancangan yang digunakan dalam mebangun *game* ini ada metode HIPO (*Hierarchy Plus Input Process Output)*. Metode ini sering digunakan sebagai alat desain dan teknik dokumentasi dalam siklus pengenembangan sistem. Metode ini menjelaskan bagaimana sebuah data melalui proses-proses sehingga akhirnya menjadi sebuah informasi. Selain itu, tujuan dari digunakannya diagram ini adalah menciptakan struktur yang menggambarkan hubungan antara fungsi dalam ke dalam sebuah hirarki. HIPO juga menyediakan penjelasan yang tepat dari *input* yang akan digunakan, proses yang akan dilakukan, serta *output* yang akan

dihasilkan oleh sistem. Diagram HIPO yang akan dijelaskan akan dibagi menjadi tiga jenis diagram, yaitu:

a. Visual Table of Contents (VTOC)

Diagram ini digunakan untuk menggambarkan hubungan dari modul-modul yang ada pada suatu sistem secara berjenjang. Diagram VTOC untuk "Game Pengenalan Hewan Laut" dapat dilihat pada Gambar 3.1.



Gambar 3.1 Diagram VTOC

b. Overview Diagram (OD)

Overview Diagram digunakan untuk menunjukkan secara garis besar hubungan dari *input*, proses, dan *output*. Diagram ini merupakan penjelasan dari diagram VTOC. Diagram *overview* sendiri dapat dilihat pada Tabel 3.1.

Tabel 3.1 Tabel diagram *overview Game* Pengenalan Hewan Laut

Modul	Input	Proses	Output
Home 0.0	- Tombol kembali. - Tombol "Tidak".	- Memuat halaman *Home*.	- Halaman *Home*
Main 1.0	- Tombol "Main".	- Memuat halaman Main.	- Halaman Main
Hasil 1.1	- Hasil balapan.	- Memuat halaman hasil.	- Halaman hasil
Akuarium 2.0	- Tombol "Akuarium".	- Memuat halaman Akuarium	- Halaman Akuarium
Toko 2.1	- Tombol "Toko".	- Memuat panel oko	- Panel toko
Ensiklopedia 2.2	- Tombol "Ensiklopedia".	- Memuat halaman ensiklopedia	- Halaman ensiklopedia
Cara Main 3.0	- Tombol "Cara Main".	- Memuat halaman cara main	- Halaman cara main

Keluar 4.0	- Tombol "Keluar".	- Memuat kotak dialog konfirmasi	- Kotak dialog konfirmasi
Ya 4.1	- Tombol "Ya".	- Menutup *game*.	- Keluar dari *game*.
Tidak 4.2	- Tombol "Tidak" .	- Memuat halaman *Home*.	- Halaman *Home*.

c. Detail Diagram (DD)

Detail Diagram ini berisi elemen-elemen dasar dari paket yang menggambarkan secara rinci kerja dari fungsi atau modul. Setelah Diagram VTOC pada Gambar 3.1 dijelaskan melalui tabel *overview* yang dapat dilihat pada Tabel 3.1, penulis akan menjelaskan lebih detail mengenai *input*, proses, dan *output* dari diagram VTOC tersebut dalam sebuah diagram detail. Diagram detail dapat dilihat pada Tabel 3.2.

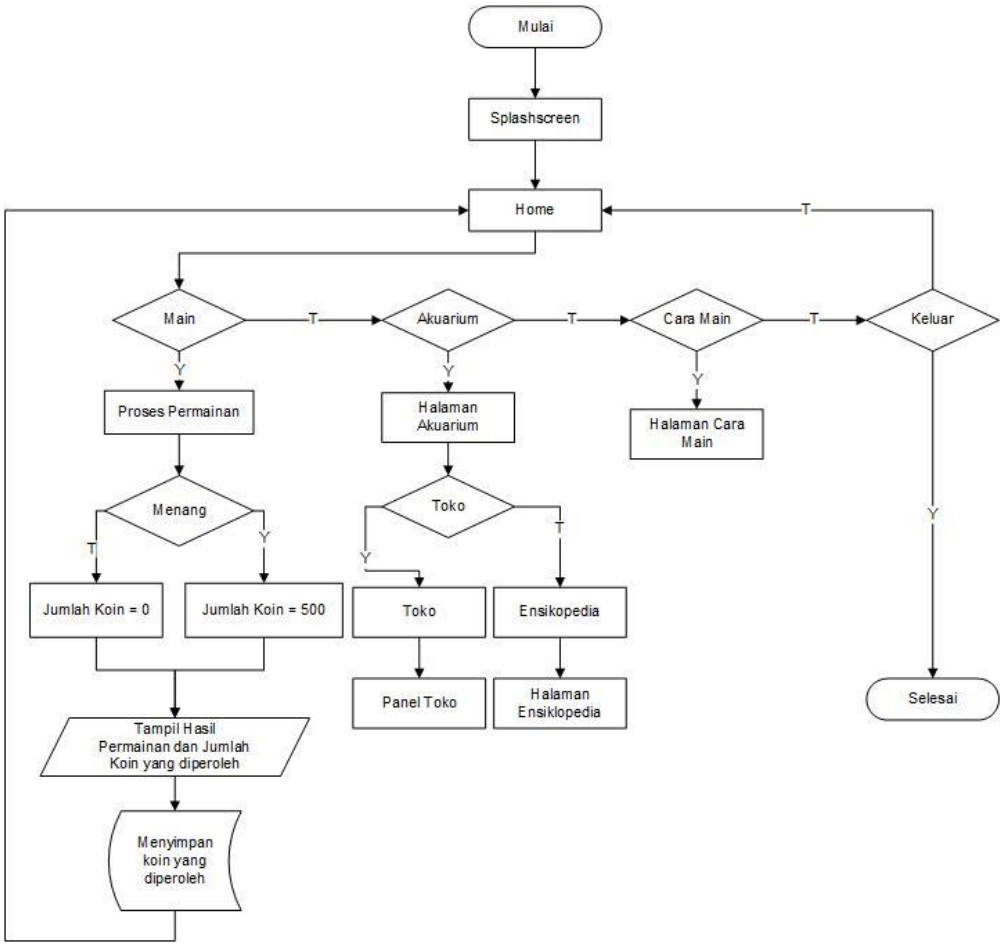
Tabel 3.2 Tabel Diagram Detail

Modul	**Input**	**Proses**	**Output**
Home 0.0	- Tombol kembali. - Tombol "Tidak".	- Memuat suara tombol. - Memuat halaman *Home*.	- Suara tombol. - Halaman *Home* yang berisi menu "Main", "Akuarium", "Cara Bermain", dan "Keluar". - Musik *background.*
Main 1.0	- Tombol "Main".	- Memuat suara tombol. - Memuat halaman main.	- Suara tombol. - Musik *background.* - Tampilan halaman main. - Menampilkan tombol "Pause" untuk menghentikan sementara balapan yang sedang berlangsung dan menampilkan tombol "Resume" untuk melanjutkan balapan. - Menampilkan tombol "Keluar" untuk keluar dari balapan.
Hasil 1.1	- Hasil balapan.	- Memuat halaman hasil.	- Musik *background.* - Hasil balapan. - Jumlah koin yang diperoleh pemain. - Menampilkan tombol "Kembali" untuk kembali ke halaman *Home*.
Akuarium 2.0	- Tombol "Akuarium".	- Memuat suara tombol.. - Memuat halaman Akuarium.	- Suara tombol. - Musik *background.* - Menampilkan halaman Akuarium. - Menampilkan tombol "Toko".

			- Menampilkan tombol "Ensiklopedia". - Menampilkan tombol "Kembali" untuk kembali ke halaman *Home*.
Toko 2.1	- Tombol "Toko".	- Memuat suara tombol - Memuat panel toko.	- Suara tombol. - Musik *background.* - Tampilan kotak dialog Toko. - Menampilkan tombol harga ikan pilih ikan sebagai karakter yang digunakan. - Menampilkan tombol untuk menutup kotak dialog Toko.
Ensikloped ia 2.2	- Tombol "Ensiklopedi a".	- Memuat suara tombol. - Memuat halaman ensiklopedia.	- Suara tombol. - Musik *background.* - Menampilan halaman ensiklopedia. - Menampilkan tombol untuk menutup halaman ensiklopedia.
Cara Main 3.0	- Tombol "Cara Main".	- Memuat suara tombol. - Memuat halaman Cara Main.	- Suara tombol. - Musik *background.* - Menampilkan halaman Cara Main. - Menampilkan tombol untuk kembali ke halaman Home.
Keluar 4.0	- Tombol "Keluar".	- Memuat suara tombol. - Memuat kotak dialog konfirmasi.	- Suara tombol. - Musik *background.* - Menampilkan pilihan "Ya" untuk keluar dari *game* dan pilihan "Tidak" untuk kembali ke halaman "Home".
Ya 4.1	- Tombol "Ya".	- Memuat suara tombol. - Menutup *game*.	- Suara tombol. - Musik *background.* - Keluar dari *game*.
Tidak 4.2	- Tombol "Tidak".	- Memuat suara tombol. - Memuat halaman "Home".	- Suara tombol. - Musik *background.* - Menampilkan halaman "Home".

3.4.2 Flowchart

Flowchart adalah bagan yang digunakan untuk menggambarkan alur dari sebuah program. Flowchart dari "Game Pengenalan hewan Laut" dapat dilihat pada Gambar 3.2.



Gambar 3.2 Flowchart Game Pengenalan Hewan Laut

3.4.3 Perancangan Antarmuka

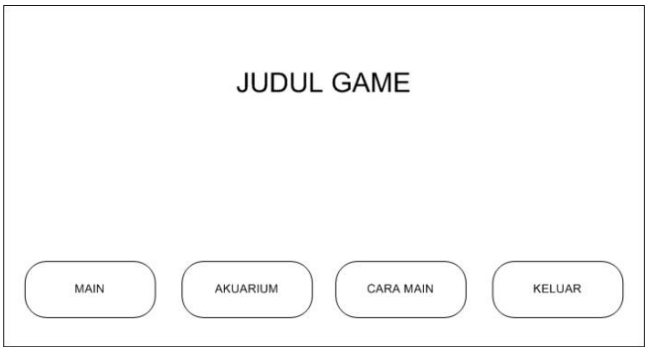
Antarmuka merupakan mekanisme interaksi antara manusia dan sistem. Antarmuka yang baik adalah antarmuka yang dapat mempermudah interaksi antara manusia dan sistem. Rancangan antarmuka merupakan rancangan tampilan yang dibuat yang nantinya akan diaplikasikan pada sistem. Perancangan antarmuka dilakukan dengan tujuan untuk mempermudah implementasi *game* yang akan dibangun. Berikut adalah rancangan antarmuka "Game Pengenalan Hewan Laut":

a. Rancangan Halaman *Splashcreen*

Gambar 3.3 Rancangan halaman *Splashcreen*

Gambar 3.3 merupakan rancangan halaman *splashcreen.* Halaman *splashscreen* merupakan halaman yang pertama kali ditampilkan ketika *game* dijalankan sebelum masuk ke halaman *home*.

b. Rancangan Halaman *Home*

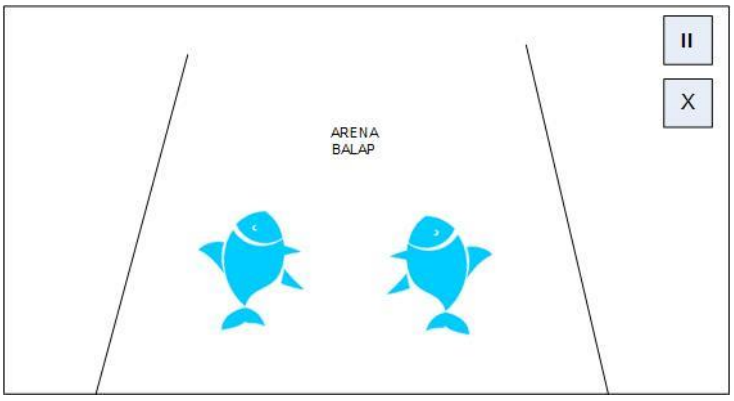


Gambar 3.4 Rancangan halaman *Home*

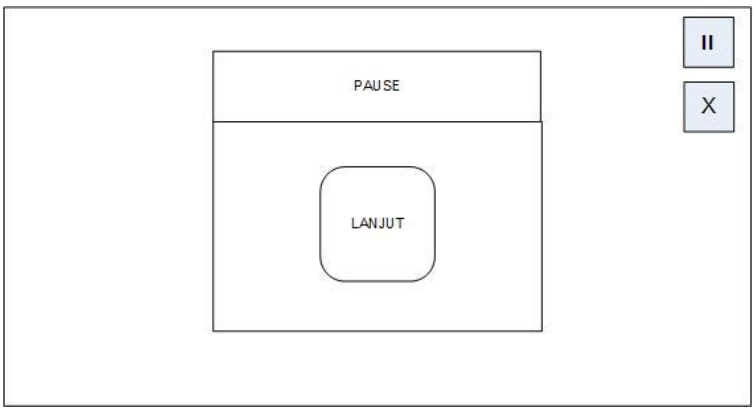
Gambar 3.4 merupakan rancangan halaman Home yang merupakan halaman yang akan muncul setelah *splashscreen*. Pada halaman ini, terdapat 4 tombol menu, yaitu "Main", "Akuarium", "Cara Bermain", dan "Keluar". Tombol "Main" akan mengarahkan pemain untuk mulai balapan, tombol "Akuarium" akan mengarahkan pemain ke halaman akuarium, tombol "Cara Bermain" akan mengarahkan pemain ke halaman cara bermain, dan tombol "Keluar" akan memunculkan kotak dialog yang berisi pilihan untuk keluar dari *game* atau tidak.

c. Rancangan Halaman Main

Halaman main merupakan halaman dimana pemain akan melakukan balapan melawan komputer. Halaman ini menampilkan arena balap atau sirkuit yang digunakan untuk balapan, karakter pemain, karakter lawan, tombol "Keluar"dan tombol "pause". Rancangan halaman main dapat dilihat pada Gambar 3.5 . Ketika pemain menekan tombol "Pause", akan muncul kotak dialog dimana akan ditampilkan sebbuah tombol, yaitu tombol "Resume". Rancangan kotak dialog tombol "Pause" dapat dilihat pada Gambar 3.6. Ketika pemain memilih tombol "Resume", maka pemain akan melanjutkan permainan. Jika pemain memilih tombol "Keluar", maka sistem akan mengarahkan pemain ke halaman *home*. Setelah pemain menyelesaikan permainan, akan muncul kotak dialog yang akan menampilkan hasil balapan, jumlah koin yang didapatkan, dan tombol "Kembali". Ketika pemain memilih tombol "Kembali", maka sistem akan mengarahkan pemain ke halaman *home*. Rancangan kotak dialog hasil balapan dapat dilihat pada Gambar 3.7.



Gambar 3.5 Rancangan halaman Main

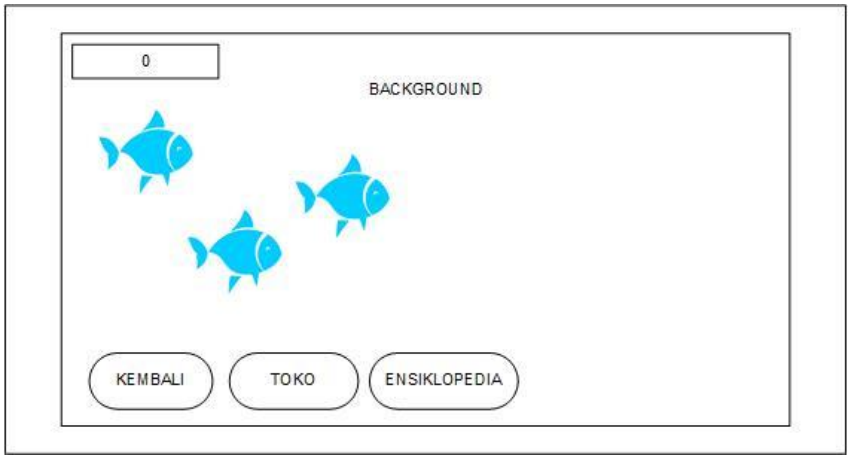


Gambar 3.6 Rancangan Kotak Dialog Tombol *Pause*

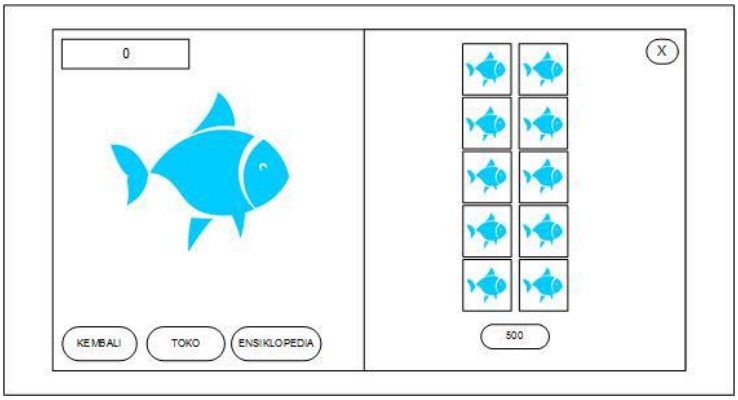
Gambar 3.7 Rancangan Kotak Dialog Hasil Balap

d. Rancangan Halaman Akuarium

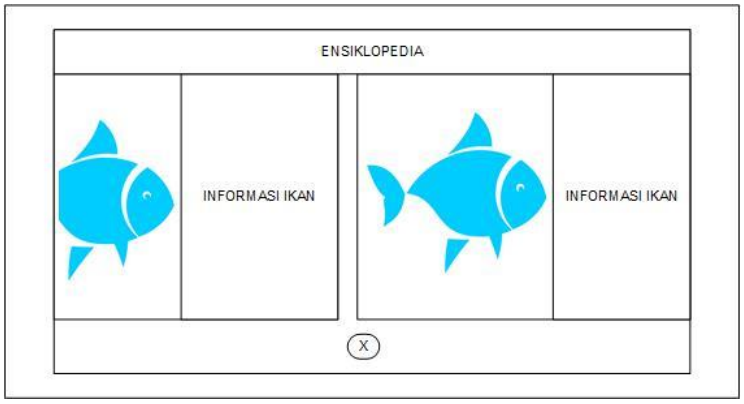
Halaman akuarium merupakan halaman dimana ikan-ikan yang telah dikumpulkan oleh pemain ditempatkan. Halaman ini menampilkan objek ikan 3D. Pada halaman ini terdapat tiga buah tombol, yaitu tombol "Kembali", tombol "Toko" dan tombol "Ensiklopedia". Saat pemain menyentuh tombol "Kembali", maka pemain akan kembali ke halaman *home*. Jika pemain menyentuh tombol "Toko", maka akan muncul kotak dialog yang berisi ikan-ikan yang dapat dibeli. Jika pemain membeli salah satu ikan, maka Selain itu, pada kotak dialog toko juga terdapat tombol "Tutup" untuk menutup kotak dialog toko. Rancangan halaman akuarium dapat dilihat pada Gambar 3.8 , rancangan kotak dialog toko dapat dilihat pada Gambar 3.9, dan rancangan kotak dialog informasi ikan dapat dilihat pada Gambar 3.10.



Gambar 3.8 Rancangan halaman Akuarium



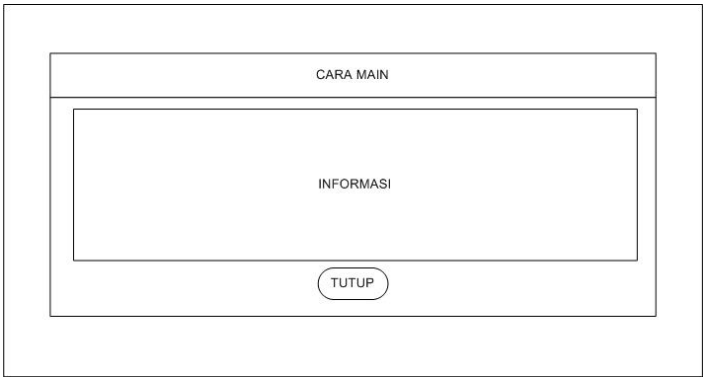
Gambar 3.9 Rancangan Kotak Dialog Toko



Gambar 3.10 Rancangan halaman Ensiklopedia

e. Rancangan Halaman Cara Main

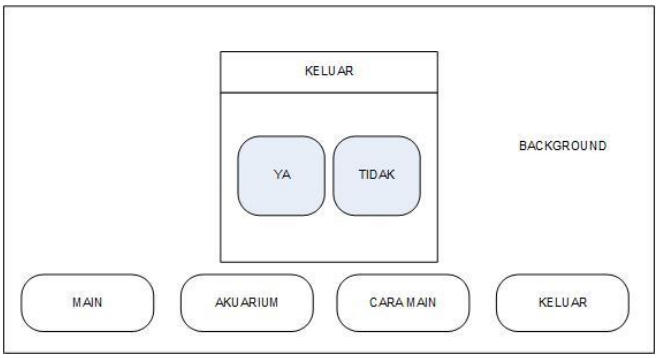
Halaman cara bermain merupakan halaman dimana akan ditampilkan informasi mengenai cara memainkan *game* dan bagaimana cara mendapatkan ikan. Halaman ini juga akan menampilkan tombol "Tutup" untuk menutup halaman cara bermain dan kembali ke halaman *home*. Rancangan halaman cara bermain dapat dilihat pada Gambar 3.11.



Gambar 3.11 Rancangan halaman Cara Bermain

f. Rancangan Kotak Dialog Keluar

Ketika pemain menekan tombol keluar yang dapat dilihat pada Gambar 3.4, akan muncul sebuah kotak dialog konfirmasi yang di dalamnya terdapat dua tombol, yaitu tombol "Ya" dan "Tidak". Ketika pemain memilih tombol "Ya", maka pemain akan keluar dari *game*. Jika pemain milih tombol "Tidak", maka kotak dialog tersebut akan menghilang dan pemain akan kembali ke halaman *home*.



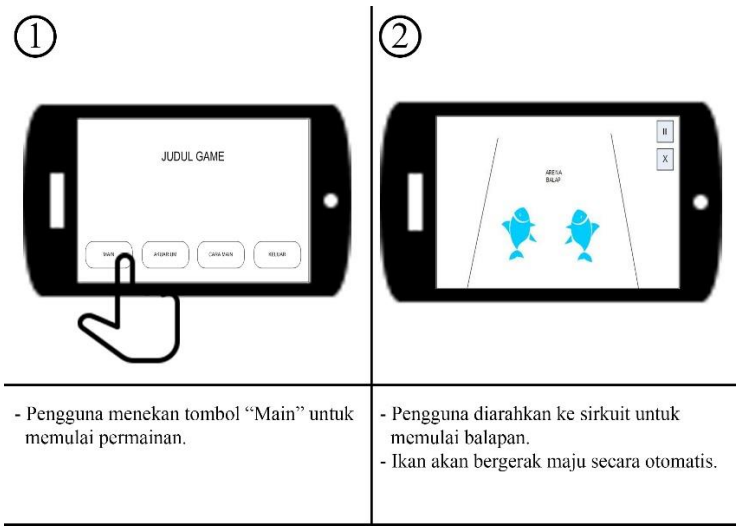
Gambar 3.12 Rancangan Kotak Dialog Keluar

3.4.4 Rancangan *Storyboard*

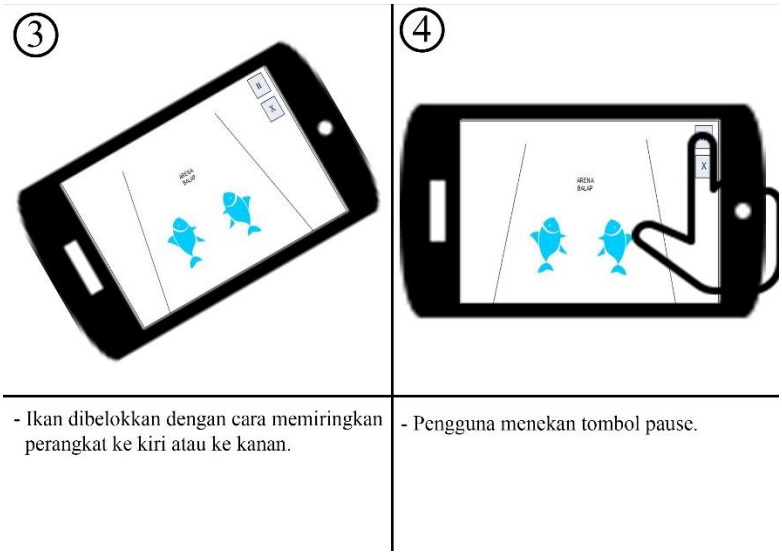
Storyboard merupakan visualisasi ide dari aplikasi yang akan dibangun, sehingga dapat memberikan gambaran dari aplikasi yang akan dihasilkan. *Storyboard* dari "Racing Game Pengenalan Hewan Laut" akan dibagi berdasarkan menu utama yang ada pada halaman *home.*

a. *Storyboard* Main

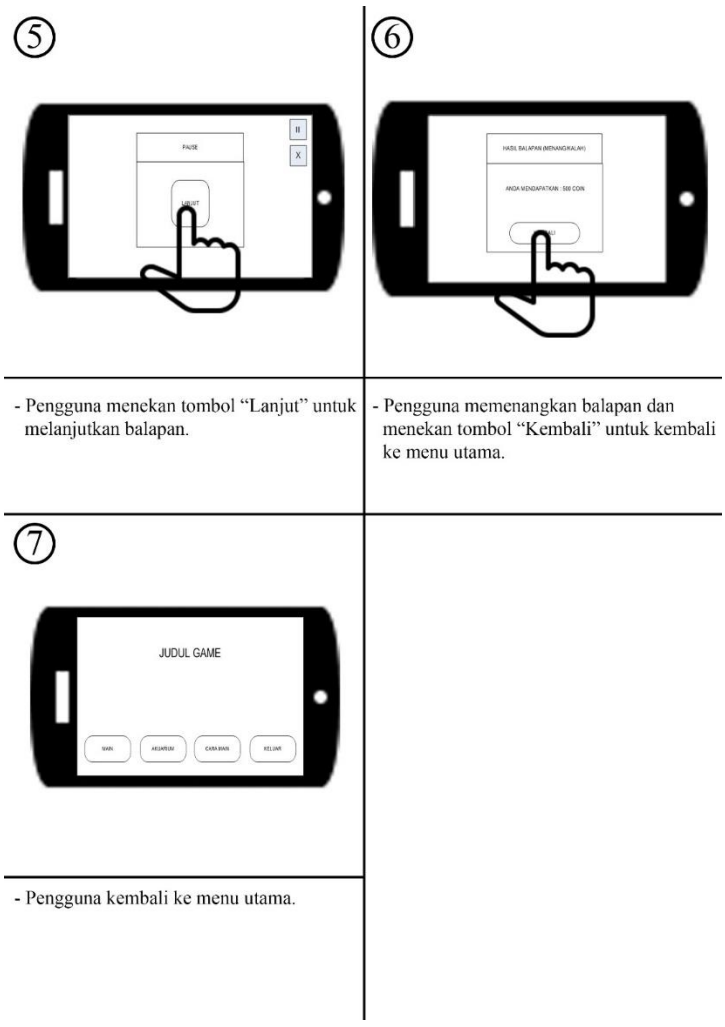
Storyboard main merupakan alur ketika pemain mengakses menu main dan melakukan balapan. Storyboard main dapat dilihat pada Gambar 3.13, Gambar 3.14, dan Gambar 3.15.



Gambar 3.13 *Storyboard* Main (1 - 2)



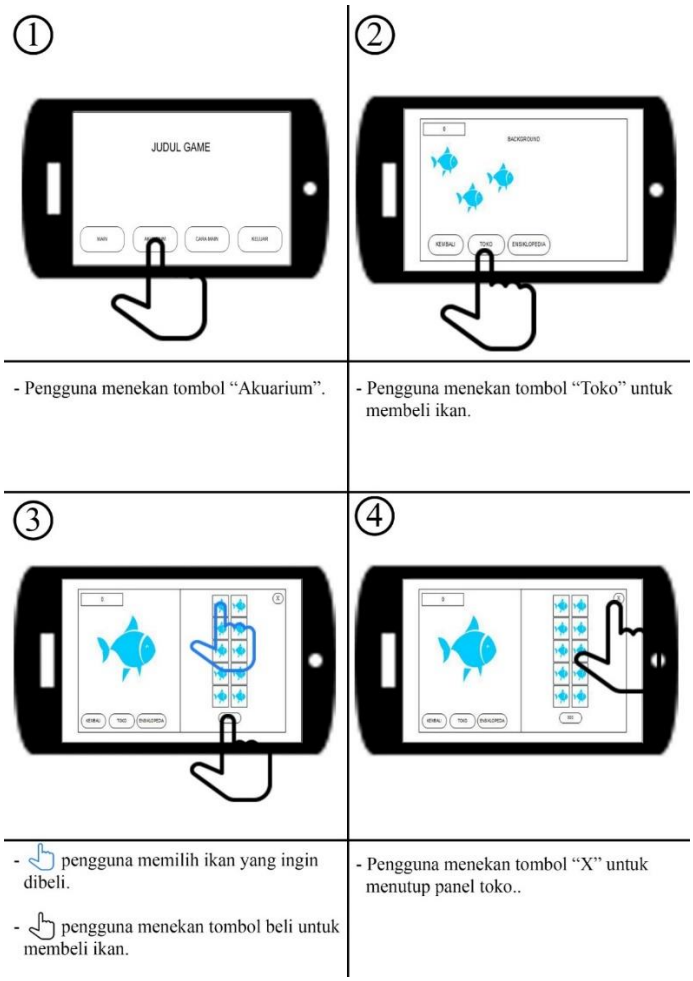
Gambar 3.14 Storyboard Main (3 - 4)



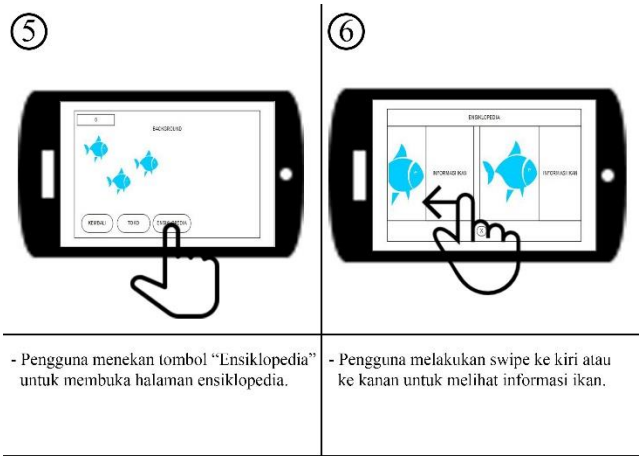
Gambar 3.15 *Storyboard* Main (5 dan 6)

b. *Storyboard* Akuarium

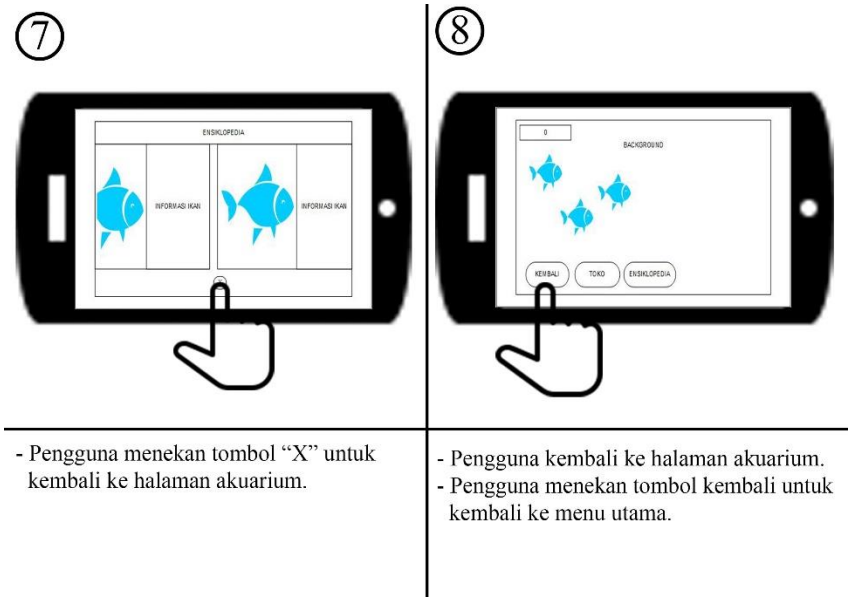
Storyboard akuarium merupakan alur ketika pemain mengakses menu akuarium. Storyboard akuarium dapat dilihat pada Gambar 3.16, Gambar 3.17, dan Gambar 3.18.



Gambar 3.16 *Storyboard* Akuarium (1 - 4)



Gambar 3.17 *Storyboard* Akuarium (5 - 6)

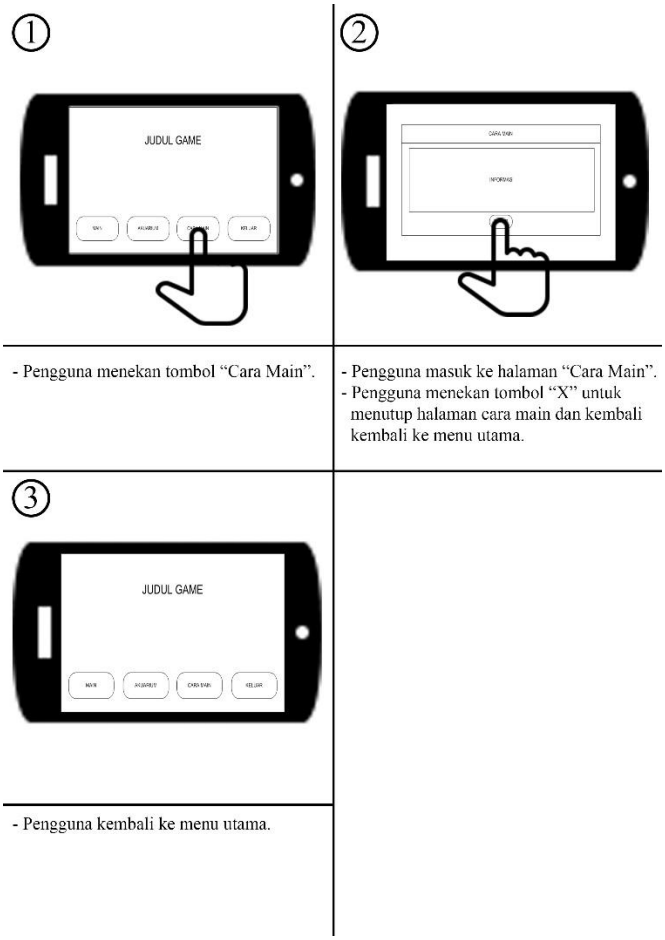


Gambar 3.18 *Storyboard* Akuarium (7 - 8)

Gambar 3.19 *Storyboard* Akuarium 9

c. *Storyboard* Cara Main

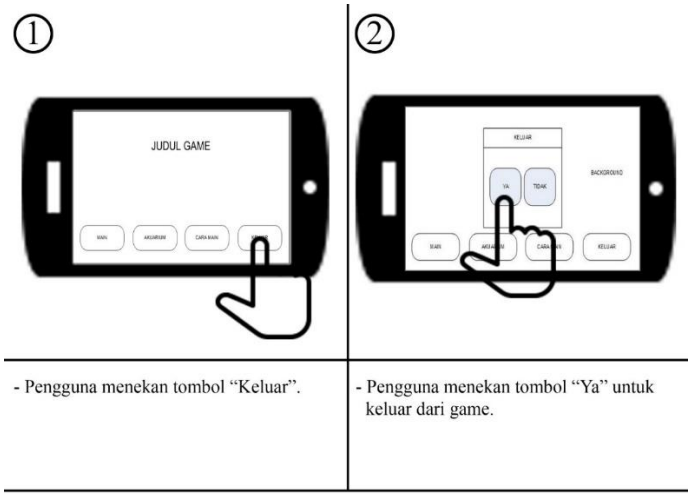
Storyboard cara main merupakan alur ketika pemain mengakses menu cara main. Storyboard cara main dapat dilihat pada Gambar 3.20.



Gambar 3.20 *Storyboard* Cara Main

d. *Storyboard* Keluar

Storyboard keluar merupakan alur ketika pemain mengakses menu keluar. Storyboard keluar dapat dilihat pada Gambar 3.21.



Gambar 3.21 *Storyboard* Keluar

3.5 Rancangan Pengujian

Rancangan pengujian adalah tahap dimana dilakukannya pengujian dari game yang telah dibangun apakah sesuai dengan yang diharapkan atau tidak. Pada kasus ini, penulis melakukan dua jenis pengujian. Pengujian yang yang pertama adalah dengan menggunakan kuesioner, dan pengujian kedua adalah pengujian perangkat.

3.5.1 Pengujian Menggunakan Kuesioner

Responden pada tahap pengujian game "Racing Game Pengenalan Hewan Laut" ini dipilih secara acak dan tidak terfokus di suatu daerah atau di usia tertentu. Sasaran responden bisa mencakup karyawan, mahasiswa, siswa SMA, siswa SMP dan siswa SD. Responden akan diberikan kuesioner setelah diminta untuk memainkan game ini. Poin-poin pernyataan kuesioner dapat dilihat pada Tabel 3.3.

Tabel 3.3 Poin pernyataan kuesioner

No.	Pernyataan
1.	Topik pengenalan hewan laut yang diangkat penulis cukup menarik.
2.	Informasi mengenai klasifikasi ilmiah ikan yang diberikan sudah akurat.
3.	Topik pengenalan laut yang diangkat sesuai untuk pengguna dengan kategori semua umur.
4.	Aplikasi Game Pengenalan Hewan Laut merupakan media pembelajaran alternatif yang efektif.
5.	Media pembelajaran alternatif ini mampu untuk meningkatkan motivasi belajar.
6.	Kemudahan penggunaan aplikasi.
7.	Antarmuka aplikasi.
8.	Kualitas objek yang digunakan pada aplikasi ini, seperti ikan, sirkuit, dan batu.
9.	Gameplay aplikasi Game Pengenalan Hewan Laut cukup menarik.
10.	Tingkat kesulitan gameplay.

Untuk memudahkan penulis dalam menghitung hasil kuesioner, maka setiap jawaban akan diberikan nilai sebagai berikut:

a. Nilai 5 untuk jawaban sangat baik (SB).
b. Nilai 4 untuk jawaban baik (B).
c. Nilai 3 untuk jawaban cukup (C).
d. Nilai 2 untuk jawaban kurang (K).
e. Nilai 1 untuk jawaban sangat kurang (SK).

Nilai-nilai tersebut kemudian digunakan untuk menghitung nilai akhir dari jawaban responden, sehingga akan menghasilkan tingkat kepuasan responden terhadap aplikasi yang dibuat.

3.5.2 Pengujian Perangkat

Pengujian perangkat ini dilakukan untuk menguji apakah *game* ini bisa dijalankan di beberapa berangkat yang berbeda. Selain itu, pengujian ini juga dapat memberikan gambaran spesifikasi minimal yang dibutuhkan untuk menjalankan "Game Pengenalan Hewan Laut".

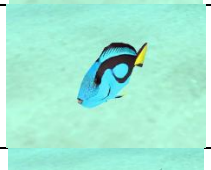
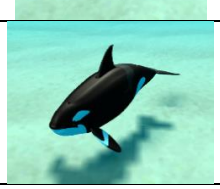
3.6 Aset

Untuk membangun "Game Pengenalan Hewan Laut", terdapat beberapa aset yang dibutuhkan. Jika penulis membuat semua aset tersebut sendiri, akan membutuhkan waktu yang sangat lama. Oleh karena itu, penulis menggunakan aset-aset yang disediakan di beberapa halaman web yang ada di internet.

3.6.1 Objek Ikan 3D

Objek ikan 3D ini merupakan objek yang digunakan sebagai karakter, baik itu karakter untuk pemain, ataupun karakter musuh. Objek ikan yang digunakan dapat dilihat pada Tabel 3.4.

Tabel 3.4 Objek ikan 3D

No.	Gambar	Nama	Tautan
1.		Tropical Fish Pack	https://www.turbosquid.com/3d-models/free-tropical-fish-pack-3d-model/652729
2.		Tropical Fish Pack	https://www.turbosquid.com/3d-models/free-tropical-fish-pack-3d-model/652729
3.		Yellowfin tuna fish 3d model	http://www.cadnav.com/3d-models/model-578.html
4.		Atlantic blue marlin 3d model	http://www.cadnav.com/3d-models/model-29489.html
5.		Orcinus orca killer whale 3d model	http://www.cadnav.com/3d-models/model-10290.html

6.		California Sheephead 3d model	http://www.cadnav.com/3d-models/model-35214.html
7.		Yellowtail clownfish 3d model	http://www.cadnav.com/3d-models/model-35203.html
8.		Leopard shark 3d model	http://www.cadnav.com/3d-models/model-35279.html
9.		Brown trout 3d model	http://www.cadnav.com/3d-models/model-35284.html

3.6.2 Tekstur

Tekstur merupakan gambar yang digunakan sebagai tekstur pada suatu objek. Dalam kasus ini, penulis menggunakan gambar pasir sebagai tekstur dari sirkuit dan dasar akuarium. Tekstur yang digunakan dapat dilihat pada Tabel 3.5.

Tabel 3.5 Tekstur

No.	Gambar	Nama	Tautan
1.		3D Scanned Wet Sand 8 – 3x3 meters	https://www.textures.com/download/3dscans0018/126510?q=sand
2.		Sand_texture 1010	http://bgfons.com/download/1010/

3.6.3 Bebatuan

Objek batu 3D digunakan sebagai penghalang pada sirkuit. Selain itu, objek batu ini kiha digunakan untuk menghias akuarium. Objek batu yang digunakan dapat dilihat pada Tabel 3.6.

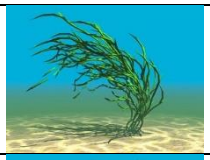
Tabel 3.6 Bebatuan

No.	Gambar	Nama	Tautan
1.		LOW POLY STONE ROCK Free VR / AR / low-poly 3D model	https://www.cgtrader.com/free-3d-models/exterior/landscape/low-poly-stone-rock

3.6.4 Tanaman

Objek tanaman 3D merupakan objek yang digunakan untuk menghiasa akuarium. Objek tanaman 3D yang digunakan dapat dilihat pada Tabel 3.7.

Tabel 3.7 Tanaman

No.	Gambar	Nama	Tautan
1.		Aquatic Plant 04	http://www.3dxo.com/models/3628_aquatic_plant_04
2.		Aquatic Plant 06	http://www.3dxo.com/models/3630_aquatic_plant_06
3.		Aquatic Plant 08	http://www.3dxo.com/models/3632_aquatic_plant_08
4.		Aquatic Plant 10	http://www.3dxo.com/models/3634_aquatic_plant_10

BAB IV
HASIL DAN PEMBAHASAN

4.1 Implementasi Sistem

Implementasi sistem adalah tahap penerapan sistem yang telah dirancang pada tahap perancangan sistem agar siap untuk dioperasikan. Tujuan dari implementasi sistem ini adalah agar sistem dapat diuji untuk mencari kesalahan yang ada pada sistem dan diperbaiki kesalahannya. Kesalahan yang mungkin terjadi antara lain kesalahan menulis kode, kesalahan pada antarmuka, atau kesalahan proses.

4.2 Implementasi Antarmuka

Implementasi antarmuka ini merupakan hasil implementasi dari rancangan antarmuka yang telah dirancang sebelumnya.

4.2.1 Halaman Splashscreen

Halaman splashscreen merupakan halaman yang muncul sebelum sistem masuk ke halaman home. Halaman ini umumnya digunakan untuk menampilkan nama atau logo developer yang mengembangkan sistem tersebut. Dalam kasus ini penulis menggunakan logo milik Unity, yang merupakan software yang penulis gunakan untuk mengembangkan sistem ini. Hasil implementasi halaman splashscreen dapat dilihat pada Gambar 4.1.

Gambar 4.1 Halaman *Splashscreen*

4.2.2 Halaman *Home*

Halaman home merupakan halaman menu utama. Pemain dapat mengakses halaman-halaman yang tersedia di dalam sistem melalui halaman home. Pada halaman ini, terdapat 5 buah tombol. Tombol-tombol tersebut adalah tombol "Main", "Akuarium", "Cara Main", "Keluar", dan "Tilt Control". Tombol "Main" digunakan untuk mengakses halaman dimana pemain melakukan balapan. Tombol "Akuarium" digunakan untuk mengakses halaman akuarium. Tombol "Cara Main" merupakan tombol yang digunakan untuk mengakses halaman cara main. Tombol "Keluar" merupakan tombol yang digunakan untuk keluar dari sistem, dan tombol "Tilt Control" merupakan tombol yang digunakan untuk mengganti tipe kontrol apakah pemain ingin menggunakan accelerometer atau touch input untuk menggerakkan karakternya ketika balapan. Halaman home dapat dilihat pada Gambar 4.2

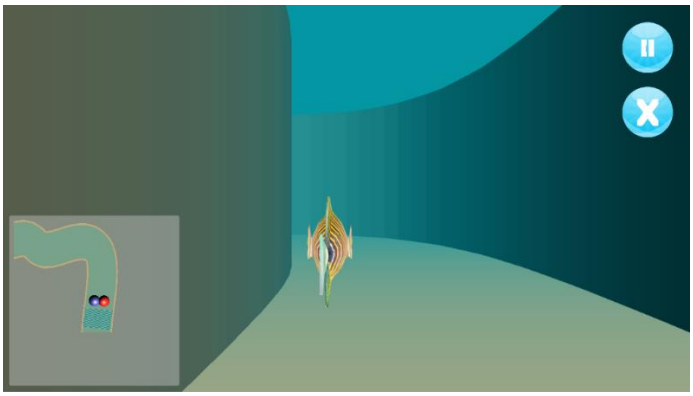
Gambar 4.2 Halaman *Home*

4.2.3 Halaman Main

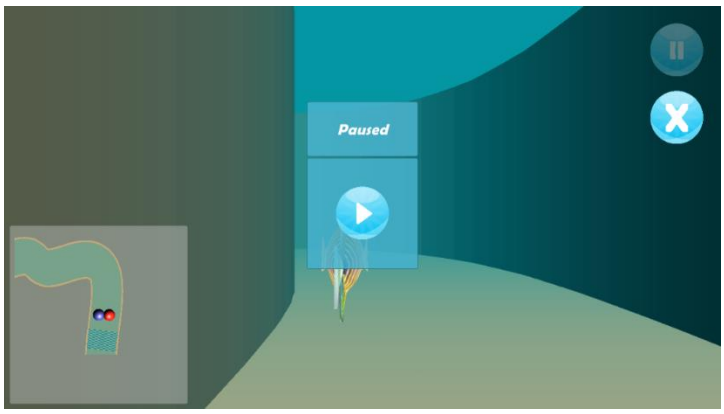
Halaman main merupakan halaman dimana balapan dilakukan. Sesaat sebelum memulai balapan, akan ditampilkan informasi klasifikasi ilmiah karakter yang digunakan pemain. Pada balapan ini, pemain akan menggunakan karakter ikan yang telah dipilih sebelumnya. Jika pemain memenangkan balapan, akan muncul halaman menang, yang merupakan halaman yang menyatakan bahwa pemain berhasil menggunakan balapan. Jika pemain kalah, akan muncul halaman kalah yang akan menyatakan bahwa pemain kalah dalam balapan. Selain itu, pada halaman main terdapat dua buah tombol. Tombol-tombol tersebut adalah tombol "Pause" dan tombol "Keluar". Tombol "Pause" akan menghentikan balapan sementara dan menampilkan panel pause. Tombol "Keluar" akan membuat balapan dihentikan dan akan mengarahkan

pemain ke halaman *home*. Pada panel pause, terdapat tombol "Resume". Tombol ini digunakan untuk melanjutkan balapan yang dihentikan sementara. Tampilan informasi karakter pemain dapat dilihat pada Gambar 4.3, halaman main dapat dilihat pada Gambar 4.4, panel pause pada Gambar 4.5, halaman menang pada Gambar 4.6, dan halaman kalah pada Gambar 4.7.

Gambar 4.3 Tampilan Informasi Karakter

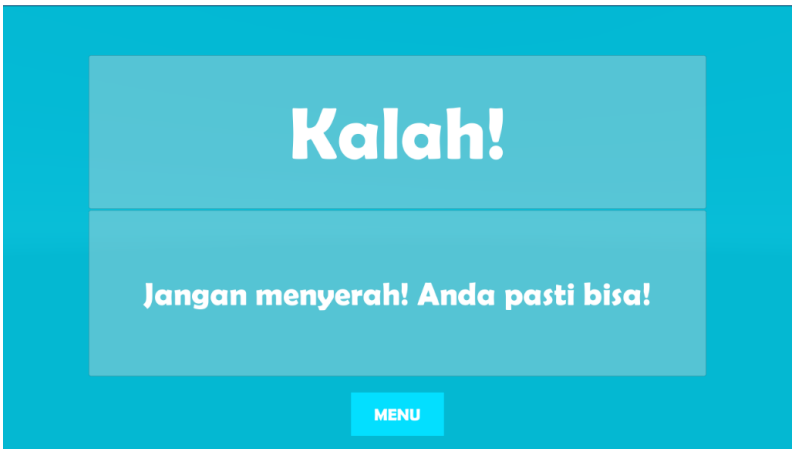


Gambar 4.4 Halaman Main.



Gambar 4.5 Panel *Pause*

Gambar 4.6 Halaman Menang.



Gambar 4.7 Halaman Kalah

Pada halaman menang yang dapat dilihat pada Gambar 4.6 dan halaman kalah yang dapat dilihat pada Gambar 4.7, masing-masing terdapat sebuah tombol. Tombol tersebut merupakan tombol "Menu". Tombol tersebut digunakan untuk menutup masing-masing halaman dan mengarahkan pemain kembali ke halaman *home*.

4.2.4 Halaman Akuarium

Halaman akuarium merupakan halaman dimana ikan yang telah dibeli oleh pemain diletakkan. Pada halaman ini, terdapat panel koin, tombol "Kembali", tombol "Toko", dan tombol "Ensiklopedia". Panel koin merupakan panel yang berisi informasi mengenai berapa banyak koin yang dimiliki oleh pemain. Tombol "Kembali" berfungsi untuk kembali ke halaman *home*. Tombol "Toko" akan menampilkan panel toko dimana di dalamnya terdapat ikan yang dapat dibeli oleh pemain, sekaligus tempat dimana pemain memilih karakter atau ikan yang akan digunakan. Jika pemain menekan tombol "Ensiklopedia", maka pemain akan

diarahkan menuju halaman ensiklopedia dimana di halaman tersebut terdapat informasi mengenai ikan yang terdapat di dalam game ini. Halaman akuarium dapat dilihat pada Gambar 4.8.

Gambar 4.8 Halaman Akuarium.

Gambar 4.9 Panel Toko

Gambar 4.10 Pembelian Gagal

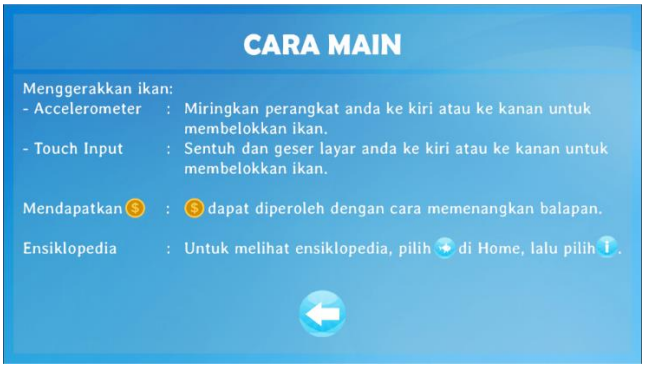
Pada panel toko yang dapat dilihat pada Gambar 4.9, terdapat dua buah tombol, yaitu tombol “Tutup” dan tombol “Beli”. Tombol “Tutup” digunakan untuk menutup panel toko, dan tombol “Beli” digunakan untuk membeli ikan yang dipilih. Teks pada tombol “Beli” dapat berubah tergantung pada ikan yang dipilih. Jika ikan yang dipilih belum dimiliki pemain, maka teks tersebut akan bertuliskan “Beli : Harga Ikan”. Jika ikan yang dipilih telah dibeli, teks akan berubah menjadi “Pilih”. Jika pemain menekan tombol tersebut ketika tombol menampilkan teks “Pilih”, maka pemain akan memilih ikan tersebut sebagai karakter yang akan digunakan ketika balapan. Jika tombol menampilkan teks “Aktif”, maka ikan yang dipilih merupakan ikan yang sedang digunakan oleh pemain. Saat pemain membeli ikan, ikan tersebut secara otomatis akan dijadikan sebagai karakter yang digunakan pemain. Jika koin pemain tidak cukup untuk membeli ikan, akan muncul notifikasi bahwa pembelian gagal. Notifikasi ketika gagal membeli ikan dapat dilihat pada Gambar 4.10.

Gambar 4.11 Halaman Ensiklopedia.

Halaman ensiklopedia merupakan halaman dimana informasi mengenai ikan-ikan yang ada pada *game* ini ditampilkan. Informasi tersebut berupa nama umum, dan kasifikasi ilmiahnya. Pada halaman ensiklopedia yang dapat dilihat pada Gambar 4.11, terdapat tiga buah tombol. Tombol tersebut adalah tombol “Kembali”, tombol “Prev”, dan “Next”. Jika pemain menekan tombol “Kembali”, maka pemain akan kembali ke halaman akuarium. Jika pemain menekan tombol “Prev”, maka akan ditampilkan informasi ikan pada indeks sebelumnya. Jika pemain menekan tombol “Next”, maka akan ditampilkan informasi ikan pada indeks selanjutnya.

4.2.5 Halaman Cara Main

Halaman cara main merupakan halaman dimana informasi untuk memainkan "Game Pengenalan Hewan Laut" ditampilkan. Informasi yang disediakan pada halaman ini berupa informasi tentang bagaimana cara menggerakkan karakter pemain ketika balapan, cara mendapatkan koin, dan cara melihat informasi ikan. Pada halaman ini terdapat sebuah tombol. Tombol tersebut adalah tombol "Kembali". Saat pemain menekan tombol "Kembali", pemain akan diarahkan ke halaman *home*. Halaman cara main dapat dilihat pada Gambar 4.12.



Gambar 4.12 Halaman Cara Main

4.2.6 Kotak Dialog Keluar

Kotak dialog keluar merupakan panel konfirmasi yang muncul ketika pemain menekan tombol "Keluar" pada halaman *home* yang dapat dilihat pada Gambar 4.4. Pada panel ini terdapat dua buah tombol. Tombol-tombol tersebut adalah tombol "YA" dan tombol "TIDAK". Jika pemain menekan tombol "YA", maka pemain akan keluar dari *game*. Jika pemain menekan tombol "TIDAK", maka panel konfirmasi akan ditutup dan pemain akan kembali ke halaman *home*. Kotak dialog keluar dapat dilihat pada Gambar 4.13.

Gambar 4.13 Kotak Dialog Keluar.

4.3 *Script*

Script merupakan kode-kode pemrograman yang penulis gunakan untuk membangun "Game Pengenalan Hewan Laut". Tanpa *script,* antarmuka yang telah dibuat tidak akan berfungsi.

4.3.1 *Script* Preloader

Script preloader merupakan *script* yang mengatur waktu lamanya waktu halaman splashscreen ditampilkan. Fungsi *Start()* adalah fungsi yang pertama kali dijalankan disaat mengakses sebuah *scene*, dan dalam kasus ini, scene tersebut adalah *preloader*. Pada baris pertama hingga baris ketiga merupakan *library* bahasa C# dan Unity yang digunakan pada *script* ini. Pada baris kedelapan hingga kesepuluh merupakan deklarasi variabel yang digunakan. Variabel *fadeGroup* dengan tipe data *CanvasGroup* untuk menyimpan objek bertipe *CanvasGroup*, variabel *loadTime* dengan tipe data *float* untuk menyimpan berapa lama waktu yang diperlukan untuk memuat halaman *home*. Variabel *minimumLogoTime* merupakan variabel yang digunakan untuk menyimpan nilai yang menentukan berapa lama logo pada halaman *spashscreen* ditampilkan.

Pada baris 15 merupakan kode untuk mengidentifikasi objek bertipe *CanvasGroup* yang ada pada scene yang sedang dibuka. Pada baris 18 merupakan kode untuk mengubah nilai properti *alpha* yang ada pada *fadeGroup* menjadi 1. Pada baris 21 hingga baris 24 merupakan kode yang digunakan untuk menentukan apabila waktu yang terhitung lebih kecil dari *minimumLogoTime*, maka *loadTime* memiliki nilai yang sama dengan *minimumLogoTime*. Jika waktu yang terhitung tidak lebih kecil dari nilai waktu *minimumLogoTime*, maka lamanya halaman *splashscreen* ditampilkan akan sama dengan waktu yang terhitung.

Pada baris 30 hingga baris 33 digunakan untuk menentukan berapa lama waktu yang dibutuhkan sebelum menampilkan logo yang ada pada halaman *splashscreen*. Jika waktu yang terhitung lebih kecil dari *minimumLogoTime*, maka alpha pada *fadeGroup* akan berkurang. Pada baris 36 hingga baris 43, jika waktu yang terhitung lebih besar dari *minimumLogoTime* dan *loadTime* tidak sama dengan 0, maka nilai dari alpha adalah waktu yang terhitung dikurang *minimumLogoTime*, dan jika *alpha* lebih besar sama dengan 1, maka sistem akan memuat halaman *home. Script Preloader* dapat dilihat pada Gambar 4.14.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.SceneManagement;
```

```
5
6 public class Preloader : MonoBehaviour {
7
8   private CanvasGroup fadeGroup;
9   private float loadTime;
10  private float minimumLogoTime = 3.0f; // Minimum time of that scene
11
12  private void Start()
13  {
14      // Grab the only CanvasGroup in the scene
15      fadeGroup = FindObjectOfType<CanvasGroup>();
16
17      //Start with a white screen;
18      fadeGroup.alpha = 1;
19
20      //Get a timestamp of the completion time
21      if (Time.time < minimumLogoTime)
22          loadTime = minimumLogoTime;
23      else
24          loadTime = Time.time;
25  }
26
27  private void Update()
28  {
29      // fade in
30      if (Time.time < minimumLogoTime)
31      {
32          fadeGroup.alpha = 1 - Time.time;
33      }
34
35      // fadeout
36      if (Time.time > minimumLogoTime && loadTime != 0)
37      {
38          fadeGroup.alpha = Time.time - minimumLogoTime;
39          if (fadeGroup.alpha >= 1)
40          {
41              SceneManager.LoadScene("Menu");
42          }
43      }
44  }
45}
```

Gambar 4.14 *Script Preloader*

4.3.2 *Script* MenuScene

Script menuscene merupakan script yang menangani fungsi-fungsi yang digunakan pada halaman *home.* Selain fungsi yang digunakan untuk berpindah halaman dari *home* ke halaman lain, fungsi yang digunakan pada halaman akuarium tergabung di dalam script ini, seperti fungsi toko, fungsi membuat ikan berenang di akuarium, dan fungsi untuk mengakses halaman ensiklopedia.

Pada baris 10 hingga baris 22 merupakan deklarasi variabel dari objek yang digunakan. Pada baris 24 hingga baris 26 merupakan deklarasi variabel untuk memanipulasi objek yang digunakan. Pada baris 28 hingga 29 merupakan deklarasi variabel untuk menyimpan teks yang digunakan pada tombol Beli, dan panel yang menampilkan jumlah koin yang dimiliki. Pada baris 31 merupkan deklarasi *array* yang meyimpan nilai harga dari masing-masing ikan yang

dijual pada panel Toko. Pada baris 32 merupakan deklarasi variabel yang digunakan untuk menyimpan indeks ikan yang sedang dipilih. Pada baris 33 merupakan deklarasi variabel yang digunakan untuk menyimpan indeks ikan yang sedang aktif. Pada baris 35 hingga 37 merupakan deklarasi tombol untuk menggunakan *accelerometer* dan warna tombol tersebut ketika pemain menggunakan *accelerometer* atau tidak. Pada baris 43 hingga baris 49 merupakan kode yang digunakan untuk mengidentifikasi apakah pemain perangkat pemain memiliki sensor *accelerometer* atau tidak. Jika perangkat pemain memiliki sensor *accelerometer*, maka tombol untuk mengaktifkan dan mematikan fitur *accelerometer* pada sistem akan ditampilkan dengan warna yang sesuai dengan status fitur *accelerometer* pada sistem. Jika perangkat pemain tidak memiliki sensor *accelerometer*, maka tombol tersebut tidak akan ditampilkan. Pada baris 52 merupakan kode yang digunakan untuk memanggil fungsi *UpdateGoldText()*.

Pada baris 94 merupakan kode yang digunakan untuk menentukan panel yang akan ditampilkan pada kamera. Pada baris 96 hingga baris 109 merupaka kode yang digunakan untuk membuat pemain tidak dapat melakukan interaksi pada tombol Main, Akuarium, CaraMain dan Keluar jika PanelKonfirmasi sedang aktif. Pada baris 112 hingga baris 117 merupakan kode yang digunakan untuk mematikan kapabilitas interaksi pada tombol Kembali, *Shop*, dan Ensiklopeda jika panel *ShopPop* sedang aktif.

Pada baris 141 hingga 163 merupakan prosedur untuk panel toko. Pada baris 149 hingga baris 162 merupakan kode yang digunakan untuk menambahkan fungsi *onClick.AddListener* pada setiap objek *children* dari PanelIsi dan meyimpannya pada prosedur *OnFishSelect* dengan parameter indeks yang terpilih, lalu mengubah gambar masing-masing *children* berdasarkan apakah ikan tersebut sudah dimiliki atau tidak. Pada baris 165 hingga 177 merupakan kode prosedur yang digunakan untuk menentukan posisi panel menu yang akan ditampilkan dengan paramater *menuIndex*. Pada prosedur ini, terdapat fungsi *switch* dengan *case* 0 menampilkan halaman home, dan *case* 1 menampilkan halaman akuarium.

Pada baris 179 hingga 191 merupakan prosedur yang digunakan untuk menentukan ikan yang aktif yang menyimpan pilihan pemain pada sistem. Pada baris 182 berfungsi menyimpan indeks yang sedang aktif pada variabel *activeFishIndex*. Pada baris 183 berfungsi menyimpan status ikan yang sedang aktif pada prosedur *activeFish* yang terletak pada *script SaveManager*. Pada baris 186 berfungsi untuk mengubah teks pada tombol Beli menjadi “Aktif”. Pada baris 190 berfungsi untuk memanggil prosedur *Save()* yang terletak pada *script SaveManager* yang

digunakan untuk meyimpan data pemain. Pada baris 193 hingga 196 merupakan prosedur yang digunakan untuk mengubah teks pada panel koin yang menampilkan jumlah koin yang dimiliki.

Pada baris 199 hingga 235 merupakan prosedur yang menangani animasi pada indeks yang dipilih pemain dan mengubah teks pada tombol Beli. Jika ikan sudah dimiliki pemain, maka teks yang akan ditampilkan adalah "Pilih". Jika pemain menekan tombol pilih, maka teks akan berubah menjadi "Aktif" menandakan ikan tersebut merupakan ikan yang sedang digunakan pemain. Jika pemain belum memiliki ikan tersebut, maka teks yang ditampilkan adalah "Beli" serta harga ikan tersebut.

Pada baris 237 hingga 240 merupakan prosedur yang berfungsi untuk kembali ke halaman home. Pada baris 242 hingga 245 merupakan prosedur yang berfungsi untuk menampilkan panel toko. Pada baris 247 hingga 250 merupakan prosedur yang berfungsi untuk membuka halaman ensiklopedia. Pada baris 252 hingga 281 prosedur yang berfungsi untuk menangani proses ketika pemain membeli ikan. Saat membeli ikan, ikan tersebut secara otomatis menjadi ikan yang dipilih untuk digunakan saat balapan. Lalu, ikon dari ikan tersebut akan berubah warna untuk menandai bahwa ikan tersebut telah dimiliki pemain. Setelah membeli ikan, sistem secara otomatis akan mengurangi koin yang dimiliki pemain berdasarkan harga ikan harga ikan yang dibeli dan menampilkan teks "Berhasil". Jika koin pemain tidak cukup untuk membeli ikan, maka akan ditampilkan notifikasi bahwa koin yang dimiliki pemain tidak cukup untuk membeli ikan.

Pada baris 283 hingga 286 merupakan prosedur yang berfungsi untuk menutup panel toko. Pada baris 288 hingga 292 merupakan prosedur yang berfungsi untuk menutup panel notifikasi ketika gagal membeli ikan. Pada baris 294 hingga 301 merupakan prosedur yang berfungsi untuk menentukan secara acak sirkuit yang digunakan untuk balapan. Pada baris 303 hingga 306 merupakan prosedur yang berfungsi untuk membuka halaman akuarium. Pada baris 308 hingga 311 merupakan prosedur yang berfungsi untuk membuka halaman cara bermain. Pada baris 313 hingga 316 merupakan prosedur yang berfungsi untuk menampilkan panel konfirmasi keluar. Pada baris 318 hingga 321 merupakan prosedur yang berfungsi untuk keluar dari *game*. Pada baris 323 hingga 326 merupakan prosedur yang berfungsi untuk menutup panel konfirmasi keluar Pada baris 328 hingga 338 merupakan prosedur yang berfungsi untuk menyalakan dan mematikan fitur *accelerometer* pada *game*, dan meyimpan pilihan pemain. *Script MenuScene* dapat dilihat pada Gambar 4.15.

```
1  using System.Collections;
2  using System.Collections.Generic;
```

```
3  using UnityEngine;
4  using UnityEngine.SceneManagement;
5  using UnityEngine.UI;
6
7  public class MenuScene : MonoBehaviour {
8
9   private Canvas Canvas;
10  private Canvas PanelKonfirmasiCanvas;
11  private Canvas ShopPop;
12  private Canvas Alert;
13  private Button Main;
14  private Button Akuarium;
15  private Button CaraMain;
16  private Button Keluar;
17  private Button AlTutup;
18  private Button Kembali;
19  private Button Shop;
20  private Button Ensiklopedia;
21  private Button CloseShop;
22  private Button Beli;
23
24  public RectTransform menuContainer;
25  public Transform PanelIsi;
26  public Vector3 desiredMenuPosition;
27
28  public Text BuyFishText;
29  public Text goldText;
30
31   private int[] fishCost = new int[] {0, 500, 1000, 1500, 2000, 2500, 3000, 3500,
     4000, 4500};
32  private int selectedfishIndex;
33  private int activeFishIndex;
34
35  public Button tiltControlButton;
36  public Color tiltControlEnabled;
37  public Color tiltControlDisabled;
38
39  // Use this for initialization
40  private void Start ()
41  {
42      // Check if we have accelerommeter
43      if (SystemInfo.supportsAccelerometer) {
44          // Is it currently enabled?
45          tiltControlButton.GetComponent<Image>            ().color            =
        (SaveManager.Instance.state.usingAccelerometer)   ?   tiltControlEnabled   :
        tiltControlDisabled;
46      } else
47      {
48          tiltControlButton.gameObject.SetActive (false);
49      }
50
51      // Show how much gold should be displayed
52      UpdateGoldText();
53
54      // Get Button and Canvas
55      Canvas = GameObject.Find ("Canvas").GetComponent<Canvas>();
56      Main = GameObject.Find ("Main").GetComponent<Button>();
57      Akuarium = GameObject.Find ("Akuarium").GetComponent<Button>();
58      CaraMain = GameObject.Find ("Cara Bermain").GetComponent<Button>();
59      Keluar = GameObject.Find ("Keluar").GetComponent<Button>();
60      PanelKonfirmasiCanvas                   =                   GameObject.Find
        ("PanelKonfirmasiCanvas").GetComponent<Canvas>();
61      PanelKonfirmasiCanvas.gameObject.SetActive (false);
62
63
64      //Canvas.gameObject.SetActive (true);
65      Kembali = GameObject.Find ("Kembali").GetComponent<Button>();
```

```
66      Shop = GameObject.Find ("Shop").GetComponent<Button>();
67      Ensiklopedia = GameObject.Find ("Ensiklopedia").GetComponent<Button>();
68
69      CloseShop = GameObject.Find ("CloseShop").GetComponent<Button>();
70
71      ShopPop = GameObject.Find ("ShopPop").GetComponent<Canvas>();
72      ShopPop.gameObject.SetActive (false);
73
74      Alert = GameObject.Find ("Alert").GetComponent<Canvas>();
75      AlTutup = GameObject.Find ("AlTutup").GetComponent<Button>();
76      Alert.gameObject.SetActive (false);
77
78      //Add button on-click events to shop button
79      InitShop();
80
81      //Set player's preferences (fish)
82      OnFishSelect(SaveManager.Instance.state.activeFish);
83      SetFish (SaveManager.Instance.state.activeFish);
84      //characterList [SaveManager.Instance.state.activeFish].SetActive (true);
85
86      //Make the buttons bigger for the selected items
87                          PanelIsi.GetChild(SaveManager.Instance.state.activeFish).
        GetComponent<RectTransform>() .localScale = Vector3.one * 1.125f;
88  }
89
90  // Update is called once per frame
91  private void Update ()
92  {
93      //Menu Navigation
94      menuContainer.anchoredPosition3D                                                =
        Vector3.Lerp(menuContainer.anchoredPosition3D, desiredMenuPosition, 0.1f);
95
96      if (PanelKonfirmasiCanvas.isActiveAndEnabled)
97      {
98      Main.interactable = false;
99      Akuarium.interactable = false;
100     CaraMain.interactable = false;
101     Keluar.interactable = false;
102     }
103     else
104     {
105         Main.interactable = true;
106         Akuarium.interactable = true;
107         CaraMain.interactable = true;
108         Keluar.interactable = true;
109     }
110
111
112     if (ShopPop.isActiveAndEnabled)
113     {
114         Kembali.interactable = false;
115         Shop.interactable = false;
116         Ensiklopedia.interactable = false;
117     }
118     else
119     {
120         Kembali.interactable = true;
121         Shop.interactable = true;
122         Ensiklopedia.interactable = true;
123     }
124
125     if (Alert.isActiveAndEnabled)
126     {
127         Kembali.interactable = false;
128         Shop.interactable = false;
129         CloseShop.interactable = false;
130         Ensiklopedia.interactable = false;
```

```
131     }
132     else
133     {
134         Kembali.interactable = true;
135         Shop.interactable = true;
136         CloseShop.interactable = true;
137         Ensiklopedia.interactable = true;
138     }
139 }
140 
141 private void InitShop()
142 {
143     // Just make sure we've assigned the reference
144     if (PanelIsi == null)
145         Debug.Log ("Panel masih kosong");
146 
147     //for every children transform under our panelisi, find the button and add
        onclick
148     int i = 0;
149     foreach (Transform t in PanelIsi)
150     {
151         int currentIndex = i;
152 
153         Button b = t.GetComponent<Button> ();
154         b.onClick.AddListener (() => OnFishSelect (currentIndex));
155 
156         // Set the fish of the image based on if owned or not
157         Image img = t.GetComponent<Image>();
158         img.color = SaveManager.Instance.IsFishOwned (i) ? Color.white : new
        Color (0.7f, 0.7f, 0.7f);
159 
160         i++;
161     }
162     i = 0;
163 }
164 
165 private void NavigateTo(int menuIndex){
166     switch (menuIndex) {
167     // 0 && default case == Mine Menu
168     default:
169     case 0:
170         desiredMenuPosition = Vector3.zero;
171         break;
172     //1 = Play Menu
173     case 1:
174         desiredMenuPosition = Vector3.left * 1280;
175         break;
176     }
177 }
178 
179 private void SetFish(int index)
180 {
181     // Set the active index offcolor
182     activeFishIndex = index;
183     SaveManager.Instance.state.activeFish = index;
184 
185     // Change beli button text
186     BuyFishText.text = "Aktif";
187     //Beli.interactable = false;
188 
189     // Remember preference
190     SaveManager.Instance.Save ();
191 }
192 
193 private void UpdateGoldText()
194 {
195     goldText.text = SaveManager.Instance.state.gold.ToString ();
```

```
196 }
197
198 //Button Section
199 private void OnFishSelect(int currentIndex)
200 {
201     Debug.Log("Select Fish : " + currentIndex);
202
203     // if the button clicked is already selected, exit
204     if(selectedfishIndex == currentIndex)
205         return;
206
207     // make the icon slightly bigger
208     PanelIsi.GetChild(currentIndex).GetComponent<RectTransform>().localScale =
       Vector3.one * 1.125f;
209
210     //put the previous one on normal scale
211  PanelIsi.GetChild(selectedfishIndex).GetComponent<RectTransform>().localScale
     = Vector3.one;
212
213     //owned fish
214     selectedfishIndex = currentIndex;
215
216     // Change the content of the buy button depending on the state of the fish
217     if (SaveManager.Instance.IsFishOwned (currentIndex))
218     {
219         //fish is owned
220         // is already our current character?
221         if (activeFishIndex == currentIndex)
222         {
223             BuyFishText.text = "Aktif";
224         }
225         else
226         {
227             BuyFishText.text = "Pilih";
228         }
229     }
230     else
231     {
232         // fish isn't owned
233         BuyFishText.text = "Beli : " + fishCost[currentIndex].ToString();
234     }
235 }
236
237 public void OnKembaliClick()
238 {
239     NavigateTo (0);
240 }
241
242 public void OnShopClick()
243 {
244     ShopPop.gameObject.SetActive (true);
245 }
246
247 public void OnDaftarIkanClick()
248 {
249     SceneManager.LoadScene ("DaftarIkan");
250 }
251
252 public void OnFishBuySet()
253 {
254     //Debug.Log("Beli Ikan");
255
256     // is the fish owned?
257     if (SaveManager.Instance.IsFishOwned (selectedfishIndex)) {
258         // Select as active
259         SetFish(selectedfishIndex);
260
```

```
261         SaveManager.Instance.Save ();
262         //BuyFishText.text = "Aktif";
263     } else {
264         // attempt to buy the fish
265         if          (SaveManager.Instance.BuyFish          (selectedfishIndex,
        fishCost[selectedfishIndex])) {
266             // success!!
267             SetFish(selectedfishIndex);
268 
269             //change the color of the button
270 
                PanelIsi.GetChild(selectedfishIndex).GetComponent<Image>().color  =
                Color.white;
271 
272             UpdateGoldText ();
273 
274             //BuyFishText.text = "Berhasil";
275 
276         } else {
277             Alert.gameObject.SetActive (true);
278             ShopPop.gameObject.SetActive (false);
279         }
280     }
281 }
282 
283 public void OnCloseShopClick()
284 {
285     ShopPop.gameObject.SetActive (false);
286 }
287 
288 public void OnAlTutup()
289 {
290     Alert.gameObject.SetActive (false);
291     ShopPop.gameObject.SetActive (true);
292 }
293 
294 public void OnMainClick()
295 {
296 
297     string[] zones = new string[5] {"Arena1", "Arena2", "Arena3", "Arena4",
        "Arena5"};
298     int random = Random.Range (0, 5);
299     SceneManager.LoadScene(zones[random]);
300 
301 }
302 
303 public void OnAkuariumClick()
304 {
305     NavigateTo (1);
306 }
307 
308 public void OnCaraBermainClick()
309 {
310     SceneManager.LoadScene("CaraBermain");
311 }
312 
313 public void OnKeluarClick()
314 {
315     PanelKonfirmasiCanvas.gameObject.SetActive (true);
316     }
317 
318 public void Ya()
319 {
320     Application.Quit();
321 }
322 
323 public void Tidak()
```

```
324 {
325     PanelKonfirmasiCanvas.gameObject.SetActive (false);
326 }
327
328 public void OnTiltControl()
329 {
330     // Toggle the accelerometer
331     SaveManager.Instance.state.usingAccelerometer                              =
       !SaveManager.Instance.state.usingAccelerometer;
332
333     // Make sure save player's preferences
334     SaveManager.Instance.Save();
335
336     // Change the display image of the tilt control button
337     tiltControlButton.GetComponent<Image>              ().color              =
       (SaveManager.Instance.state.usingAccelerometer)   ?   tiltControlEnabled   :
       tiltControlDisabled;
338 }
339}
```

Gambar 4.15 *Script* MenuScene

4.3.3 *Script* SpawnOwnedFish

Script SpawnOwnedFish ini digunakan untuk meletakkan ikan yang telah dibeli oleh pemain ke dalam akuarium. Pada baris 15 merupakan kode yang berfungsi untuk inisiasi *array* objek baru dengan nama *OwnedFish*. Pada baris 18 hingga 19 merupakan kode yang berfungi untuk memasukkan objek-objek yang telah disiapkan pada *array OwnedFish*. Pada baris 22 hingga baris 23 merupakan kode yang berfungsi untuk menonaktifkan objek-objek yang ada pada array *OwnedFish.* Pada baris 25 hingga 27 merupakan kode yang berfungsi untuk menentukan jarak acak untuk sumbu x, y, dan z yang disimpan pada variabel *randomX*, *randomY*, dan *randomZ*.

Pada baris 29 hingga 37 merupakan kode yang berfungsi untuk menampilkan ikan-ikan yang telah dimiliki oleh pemain pada posisi yang ditentukan oleh objek dengan variabel pos dengan parameter *randomX*, *randomY*, dan *randomZ*. Pada baris 41 hingga 57 merupakan kode yang berfungsi untuk menampilkan ikan yang dimiliki oleh pemain dan mengubah arah ikan berenang secara acak. *Script SpawnOwnedFish* dapat dilihat pada Gambar 4.16.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class SpawnOwnedFish : MonoBehaviour {
6
7   public static GameObject[] OwnedFish;
8   public GameObject goalPrefab;
9   public static int tanksize = 5;
10  public static Vector3 goalPos = new Vector3(1904, 325, 153);
11
12  // Use this for initialization
13  void Start () {
```

```
14
15      OwnedFish = new GameObject[transform.childCount];
16
17      //fill the array with our model
18      for (int i = 0; i < transform.childCount; i++)
19          OwnedFish[i] = transform.GetChild(i).gameObject;
20
21      // we toggle of their renderer
22      foreach(GameObject go in OwnedFish)
23          go.SetActive(false);
24
25      float randomX = Random.Range(1600,2100);
26      float randomY = Random.Range(276,480);
27      float randomZ = Random.Range(50,250);
28
29      int index = 0;
30      foreach (GameObject go in OwnedFish)
31          if (SaveManager.Instance.IsFishOwned (index)) {
32              Vector3 pos = new Vector3 (randomX, randomY, randomZ);
33              OwnedFish [index].SetActive (true);
34              OwnedFish[index].transform.position = pos;
35              index++;
36          }
37  }
38
39  // Update is called once per frame
40  void Update () {
41      // we toggle on the first index
42      int index = 0;
43      foreach (GameObject go in OwnedFish)
44          if (SaveManager.Instance.IsFishOwned (index)) {
45              OwnedFish [index].SetActive (true);
46              index++;
47          }
48
49      float randomX = Random.Range(200,600);
50      float randomY = Random.Range(276,480);
51      float randomZ = Random.Range(50,250);
52
53      if (Random.Range (0, 6000) < 50)
54      {
55      goalPos = new Vector3 (randomX, randomY, randomZ);
56          goalPrefab.transform.position = goalPos;
57      }
58  }
```

Gambar 4.16 *Script* SpawnOwnedFish

4.3.4 *Script* Flock

Pada *script* ini, di dalamnya berisi fungsi untuk membuat ikan berenang di dalam akuarium, mencegah ikan keluar dari area yang ditentukan, dan mencegah ikan berenang hanya menuju satu arah. Pada baris 20 merupakan kode yang berfungsi untuk menentukan kecepatan ikan berenang. Kecepatan ikan ditentukan secara acak dengan jarak acak yang telah ditentukan dan disimpan pada variabel *minSpeed* untuk nilai minimal dan *maxSpeed* untuk nilai maksimal. Pada baris 23 hingga 32 merupakan prosedur yang berfungsi untuk mengubah arah ikan berenang jika ikan menyentuh objek lain, baik itu dinding akuarium, ataupun ikan lainnya. Pada prosedur *update*, pada baris 42 hingga 46 merupakan kode yang yang berfungsi untuk

membuat ikan berbelok ketika ikan mulai mendekati dinding akuarium. Pada baris 48 hingga 61 merupakan kode yang berfungsi untuk menentukan arah ikan berenang setelah ikan berbelok ketika mendekati dinding, kecepatan ikan berbelok, dan kecepatan ikan berenang. Jika ikan tidak mendekati dinding dan masih berenang lurus dalam waktu tertentu, akan dipanggil prosedur *ApplyRules().* Pada baris 63 merupakan kode yang berfungsi untuk menentukan kecepatan ikan berenang. Pada baris 67 hingga 113 merupakan prosedur *AppyRules()* yang berfungsi untuk mengubah arah ikan berenang. *Script Flock* dapat dilihat pada Gambar 4.17.

```
1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4
5  public class Flock : MonoBehaviour {
6
7   public float baseSpeed = 10.0f;
8
9   float rotationSpeed = 0.5f;
10  float minSpeed = 10f;
11  float maxSpeed = 20f;
12  Vector3 averageHeading;
13  Vector3 averagePosition;
14  float neighbourDistance = 30.0f;
15
16  bool turning = false;
17
18  // Use this for initialization
19  void Start () {
20      baseSpeed = Random.Range (minSpeed, maxSpeed);
21  }
22
23  void OnTriggerEnter(Collider other)
24  {
25      if (other.gameObject)
26      {
27          if (!turning) {
28              SpawnOwnedFish.goalPos = this.transform.position -
     other.gameObject.transform.position;
29              turning = true;
30          }
31      }
32  }
33
34  void OnTriggerExit(Collider other)
35  {
36      turning = false;
37  }
38
39  // Update is called once per frame
40  void Update () {
41
42      if (Vector3.Distance (transform.position, Vector3.zero) >=
       SpawnOwnedFish.tanksize) {
43          turning = true;
44      }
45      else
46          turning = false;
```

```
47
48      if (turning) {
49          //Vector3 direction = new Vector3 (1904, 325, 153) -
      transform.position;
50          Vector3 direction = SpawnOwnedFish.goalPos - transform.position;
51          transform.rotation = Quaternion.Slerp (transform.rotation,
52              Quaternion.LookRotation (direction),
53              rotationSpeed * Time.deltaTime);
54
55          baseSpeed = Random.Range (minSpeed, minSpeed);
56      }
57      else
58      {
59          if (Random.Range (0, 5) < 1)
60              ApplyRules ();
61      }
62
63      transform.Translate (0, 0, Time.deltaTime * baseSpeed);
64
65  }
66
67  void ApplyRules()
68  {
69      GameObject[] fis;
70      fis = SpawnOwnedFish.OwnedFish;
71
72      Vector3 vcentre = SpawnOwnedFish.goalPos;
73      Vector3 vavoid = SpawnOwnedFish.goalPos;
74      float gSpeed = 2f;
75
76      Vector3 goalPos = SpawnOwnedFish.goalPos;
77
78      float dist;
79
80      int groupSize = 0;
81      foreach (GameObject go in fis)
82      {
83          if (go != this.gameObject)
84          {
85              dist = Vector3.Distance (go.transform.position,
      this.transform.position);
86              if (dist <= neighbourDistance)
87              {
88                  vcentre += go.transform.position;
89                  groupSize++;
90
91                  if (dist <= 5f)
92                  {
93                      vavoid = vavoid + (this.transform.position -
      go.transform.position);
94                      groupSize++;
95                  }
96                  Flock anotherFlock = go.GetComponent<Flock> ();
97                  gSpeed = gSpeed + anotherFlock.baseSpeed;
98              }
99          }
100     }
101
102     if (groupSize > 0)
103     {
104         vcentre = vcentre / groupSize + (goalPos - this.transform.position);
105         baseSpeed = gSpeed / groupSize;
106
107         Vector3 direction = (vcentre + vavoid) - transform.position;
108         if (direction != Vector3.zero)
109             transform.rotation = Quaternion.Slerp (transform.rotation,
110                 Quaternion.LookRotation (direction),
```

```
111                 rotationSpeed * Time.deltaTime);
112     }
113 }
```

Gambar 4.17 *Script* Flock

4.3.5 *Script* Ensiklopedia

Script ini digunakan untuk mengangani fungsi tombol keluar yang ada di halaman ensiklopedia. Pada baris kedelapan merupakan kode yang berfungsi untuk mendeklarasikan variabel Kembali dengan tipe data *Button*. Pada baris 13 merupakan kode yang digunakan untuk mengidentifikasi objek tombol dengan nama Kembali pada scene yang menyimpan objek pada variabel Kembali. Pada baris 16 hingga 19 merupakan prosedur yang berfungsi untuk kembali ke halaman *home*. *Script* Ensiklopedia dapat dilihat pada Gambar 4.18.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.UI;
5 using UnityEngine.SceneManagement;
6
7 public class Ensiklopedia : MonoBehaviour {
8   private Button Kembali;
9
10  // Use this for initialization
11  private void Start ()
12  {
13      Kembali = GameObject.Find ("Kembali").GetComponent<Button>();
14  }
15
16  public void OnKembaliClick()
17  {
18      SceneManager.LoadScene("Menu");
19  }
20}
```

Gambar 4.18 *Script* Eniklopedia

4.3.6 *Script* Arena

Script arena digunakan untuk menangani fungsi tombol-tombol yang ada di setiap arena balap agar tombol “Pause”, “Keluar”, dan “Resume” bisa berfungsi dengan baik. Pada baris 8 hingga 14 merupakan kode-kode yang digunakan untuk deklarasi variabel yang digunakan dalam *script* ini. Pada baris 17 hingga Pada baris 27 merupakan kode-kode yang berfungsi untuk mengindentifikasi objek yang ada pada scene dan menyimpan attribut serta nilainya pada veriabel yang telah dideklarasikan. Pada baris 30 hingga 35 merupakan kode yang digunakan untuk mengubah skala waktu menjadi 0 jika InfoPanel sedang aktif. Hal ini akan membuat *game* dihentikan sementara. Jika InfoPanel telah ditutup, maka *game* akan dilanjutkan. Pada baris 37 hingga 44 merupakan kode yang digunakan untuk mengubah skala waktu menjadi 0

jika *PanelPause* sedang aktif. Hal ini akan membuat *game* dihentikan sementara dan menghilangkan kapabilitas interaksi pada tombol *pause*. Jika *PanelPause* telah ditutup, maka *game* akan dilanjutkan dan tombol *pause* dapat kembali diakses.

Pada baris 50 hingga 53 merupakan prosedur yang berfungsi untuk menutup *PanelPause*. Pada baris 55 hingga 57 merupakan prosedur yang berfungsi untuk membawa pemain kembali ke halaman *home*. Pada baris 59 hingga 61 merupakan prosedur yang berfungsi untuk menutup InfoPanel. Script Arena dapat dilihat pada Gambar 4.19.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.SceneManagement;
5 using UnityEngine.UI;
6
7 public class Arena1 : MonoBehaviour{
8   private Canvas Canvas;
9   private Canvas PausePanel;
10  private Canvas InfoPanel;
11  private Button Pause;
12  private Button Resume;
13  private Button Exit;
14  private Button Tutup;
15
16  private void Start(){
17      Canvas = GameObject.Find ("Canvas").GetComponent<Canvas>();
18      Pause = GameObject.Find ("Pause").GetComponent<Button>();
19      Resume = GameObject.Find ("Resume").GetComponent<Button>();
20      Exit = GameObject.Find ("Exit").GetComponent<Button>();
21      Tutup = GameObject.Find ("Tutup").GetComponent<Button>();
22
23      PausePanel = GameObject.Find ("PausePanel").GetComponent<Canvas>();
24      PausePanel.gameObject.SetActive (true);
25      InfoPanel = GameObject.Find ("InfoPanel").GetComponent<Canvas>();
26      InfoPanel.gameObject.SetActive (true);
27  }
28
29  private void Update(){
30      if (InfoPanel.isActiveAndEnabled){
31          Time.timeScale = 0;
32      }
33      else{
34          Time.timeScale = 1;
35      }
36
37      if (PausePanel.isActiveAndEnabled){
38          Pause.interactable = false;
39          Time.timeScale = 0;
40      }
41      else{
42          Pause.interactable = true;
43          Time.timeScale = 1;
44      }
45  }
46
47  public void OnPause(){
48      PausePanel.gameObject.SetActive (true);
49  }
50
```

```
51  public void OnResume(){
52      PausePanel.gameObject.SetActive (false);
53  }
54
55  public void OnExit(){
56      SceneManager.LoadScene("Menu");
57  }
58
59  public void OnTutup(){
60      InfoPanel.gameObject.SetActive (false);
61       PausePanel.gameObject.SetActive (false); }
```

Gambar 4.19 *Scipt* Arena

4.3.7 *Script* NavMover

Script NavMover ini digunakan untuk membuat karakter musuh atau bisa disebut dengan AI bisa bergerak dengan sendirinya sesuai dengan pola yang telah ditentukan. Pada baris 6 merupakan kode untuk deklarasi array Transform. Pada baris 7 merupakan kode unuk deklarasi variabel *destPoint*. Pada baris 8 merupakan kode untuk deklarasi variabel *agent*. Pada baris 11 merupakan kode untuk menyimpan komponen *NavMeshAgent* pada variabel *agent*. Pada baris 15 merupakan kode untuk mencegah objek mengurangi kecepatan secara otomatis ketika mendekati titik tujuan. Pada baris 18 merupakan kode untuk memanggil prosedur *GotoNextPoint().*

Pada baris 21 hingga 43 merupakan prosedur yang berfungsi untuk menentukan titik yang dituju ketika objek bergerak. Pada baris 23 hingga 24 merupakan kode yang berfungsi untuk membuat objek kembali ke posisi awal jika jumlah titik tujuan sama dengan 0. Pada baris 25 hingga 26 merupakan kode yang berfungsi untuk membuat objek kembali ke posisi awal jika titik yang dituju merupakan titik terakhir. Pada baris 29 merupakan kode yang berfungsi untuk menggerakkan objek menuju satu titik yang telah dipilih. Pada baris 34 merupakan kode yang berfungsi untuk memilih tujuan selanjutnya dari *array points*. Pada baris 41 hingga 42 merupakan kode yang berfungsi untuk memanggil prosedur *GotoNextPoint()* jika jarak sisa menuju titik tujuan objek kurang dari 0.5 dan titik tujuan masih tersedia. *Script NavMover* dapat dilihat pada Gambar 4.20.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class NavMover : MonoBehaviour {
6   public Transform[] points;
7   private int destPoint = 0;
8   private UnityEngine.AI.NavMeshAgent agent;
9
10  void Start (){
11      agent = GetComponent<UnityEngine.AI.NavMeshAgent> ();
```

```
12
13      //Disabling auto-braking allows for continuous movement
14      //between points (ie, the agent doesn't allow slow down as it approaches a
    destination point).
15      agent.autoBraking = false;
16      agent.updateUpAxis = false;
17
18      GotoNextPoint ();
19  }
20
21  void GotoNextPoint() {
22            // Returns if no points have been set up
23            if (points.Length == 0)
24                return;
25      if (destPoint == points.Length)
26          return;
27
28            // Set the agent to go to the currently selected destination.
29            agent.destination = points[destPoint].position;
30
31            // Choose the next point in the array as the destination,
32            // cycling to the start if necessary.
33            //destPoint = (destPoint + 1) % points.Length;
34          destPoint = (destPoint + 1);
35        }
36
37        void Update () {
38            // Choose the next destination point when the agent gets
39            // close to the current one.
40
41            if (!agent.pathPending && agent.remainingDistance < 0.5f)
42                GotoNextPoint();
43        }
```

Gambar 4.20 *Script* NavMover

4.3.8 *Script* RandomCharacter

Script RandomCharacter ini digunakan untuk memilih secara acak AI yang akan menjadi musuh pemain dalam balapan. Pada baris 6 merupakan kode yang digunakan untuk mendeklarasikan variabel AI dengan tipe data berupa *array GameObject*. Pada baris 10 merupakan kode-kode yang berfungsi untuk menghitung objek-objek yang merupakan children dari GameObject AI dan meyimpannya ke dalam variabel AI. Pada baris 13 hingga 14 merupakan kode-kode yang berfungsi untuk menghitung objek-objek yang merupakan children dari GameObject AI dan masukkan objek-objek tersebut ke dalam *array* AI. Pada baris 17 hingga 18 merupakan kode-kode yang berfungsi untuk menonaktifkan setiap objek yang ada di dalam *array* AI. Pada baris 21 merupakan kode-kode yang berfungsi untuk mengaktifkan salah satu objek di dalam *array* AI secara acak. *Script* RandomCaharacter dapat dilihat pada Gambar 4.21.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
```

```
4
5 public class RandomCharacter : MonoBehaviour {
6   public GameObject[] AI;
7
8   // Use this for initialization
9   void Start () {
10      AI = new GameObject[transform.childCount];
11
12      //fill the array with our model
13      for (int i = 0; i < transform.childCount; i++)
14          AI[i] = transform.GetChild(i).gameObject;
15
16      // we toggle of their renderer
17      foreach(GameObject go in AI)
18          go.SetActive(false);
19
20      // we toggle on the first index
21      AI[Random.Range (0, AI.Length)].SetActive(true);
22  }
23}
```

Gambar 4.21 *Script* RandomCharacter

4.3.9 *Script* PlayerMotor

Script PlayerMotor digunakan untuk menggerakkan karakter pemain agar karakter tersebut mampu bergerak maju secara otomatis serta menentukan seberapa cepat karakter pemain bergerak maju dan berbelok. Pada baris 6 hingga 10 merupakan kode-kode yang digunakan untuk mendeklarasikan variabel yang akan digunakan pada *script* ini. Pada baris 13 merupakan kode-kode yang digunakan untuk mengakses komponen *CharacterController* yang ada pada objek yang berada di *scene*. Pada baris 18 merupakan kode-kode yang berfungsi mengerakkan objek untuk maju ke depan. Pada baris 21 merupakan kode-kode yang digunakan untuk memanggil prosedur GetPlayerInput() yang terletak pada *Script* Manager. Pada baris 24 hinggah baris 27 merupakan kode-kode yang berfungsi untuk mengarahkan objek ketika objek bergerak. *Script* PlayerMotor dapat dilihat pada Gambar 4.22.

```
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
4
5   public class PlayerMotor : MonoBehaviour {
6   private CharacterController controller;
7   private float baseSpeed = 19.5f;
8   private float rotSpeedX = 20.0f;
9   private float boostSpeed = 5.0f;
10  private float boostCooldown = 10.0f;
11
12  private void Start(){
13  controller = GetComponent<CharacterController> ();
14  }
15
16  private void Update(){
17  // Gove the player some forward velocity
18  Vector3 moveVector = transform.forward * baseSpeed;
```

```
19
20  // Gather Player's input
21  Vector3 inputs = Manager.Instance.GetPlayerInput();
22
23  // Get the delta direction
24  Vector3 yaw = inputs.x * transform.right * rotSpeedX * Time.deltaTime;
25  moveVector += yaw;
26  transform.rotation = Quaternion.LookRotation (moveVector);
27  controller.Move (moveVector * Time.deltaTime);
28  }
29  }
```

Gambar 4.22 *Script* PlayerMotor

4.3.10 *Script* SprintFinish

Pada *script* ini, terdapat fungsi untuk mengetahui siapa yang terlebih dahulu menyentuh garis akhir di sirkuit. Pada baris 7 merupakan prosedur yang berfungsi untuk menampilkan halaman kalah atau menang berdasarkan *tag* pada objek yang pertama kali menyentuh garis akhir. Pada kode-kode di baris 8 hingga 10, jika objek yang terlebih dahulu menyentuh garis memiliki *tag* "AI", maka sistem akan menampilkan halaman kalah. Pada kode-kode di baris 12 hingga 14, jika objek yang terlebih dahulu menyentuh garis memiliki *tag* "Player", maka sistem akan menampilkan halaman menang. *Script* SprintFinish dapat dilihat pada Gambar 4.23.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.SceneManagement;
5
6 public class SprintFinish : MonoBehaviour {
7   void OnTriggerEnter (Collider col){
8       if (col.gameObject.tag == "AI") {
9           SceneManager.LoadScene("Lose");
10      }
11
12      if (col.gameObject.tag == "Player") {
13          SceneManager.LoadScene("Win");
14      }
15  }
16}
```

Gambar 4.23 *Script* SprintFinish

4.3.11 *Script* Win

Script Win ini digunakan untuk menangani fungsi tombol yang ada pada halaman menang. Pada baris 8 merupakan kode-kode yang berfungsi untuk mendeklarasikan variabel Menu dengan tipe data Button. Pada baris 12 merupakan kode-kode yang berfungsi untuk memanggil prosedur WinRace() yang terletak pada *script* SaveManager. Pada baris 13

merupakan kode-kode yang berfungsi untuk membawa pemain ke halaman *home*. *Script* Win dapat dilihat pada Gambar 4.24.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.SceneManagement;
5 using UnityEngine.UI;
6
7 public class Win : MonoBehaviour {
8   private Button Menu;
9
10  public void BacktoMenu()
11  {
12      SaveManager.Instance.WinRace ();
13      SceneManager.LoadScene("Menu");
14  }
15}
```

Gambar 4.24 *Script* Win

4.3.12 *Script* Lose

Pada *script* Lose ini, terdapat fungsi untuk menangani tombol yang ada pada halaman kalah. Pada baris 8 merupakan kode-kode yang berfungsi untuk mendeklarasikan variabel Menu dengan tipe data Button. Pada baris 12 merupakan kode-kode yang berfungsi untuk membawa pemain ke halaman *home*. *Script* Lose dapat dilihat pada Gambar 4.25.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.SceneManagement;
5 using UnityEngine.UI;
6
7 public class Lose : MonoBehaviour {
8   private Button Menu;
9
10  public void BacktoMenu()
11  {
12      SceneManager.LoadScene("Menu");
13  }
14 }
```

Gambar 4.25 *Script* Lose

4.3.13 *Script* CaraBermain

Script CaraBermain ini merupakan *script* yang digunakan untuk menangani fungsi tombol “Kembali” pada halaman cara bermain. Pada baris 8, terdapat kode-kode yang berfungsi untuk membawa pemain ke halaman *home*. *Script* CaraBermain dapat dilihat pada Gambar 4.26.

```
1   using System.Collections;
```

```
2   using System.Collections.Generic;
3   using UnityEngine;
4   using UnityEngine.SceneManagement;
5
6   public class CaraBermain : MonoBehaviour {
7       public void OnKembaliClick(){
8           SceneManager.LoadScene("Menu");
9       }
10  }
```

Gambar 4.26 *Script* CaraBermain

4.3.14 *Script* Manager

Pada *script* Manager ini, di dalamnya terdapat fungsi untuk menerima *input* dari pemain ketika pemain menggunakan tipe kontrol *touch input*. Saat pemain memulai balapan, *script* PlayerMotor akan memanggil fungsi GetPlayerInput() yang ada pada script ini untuk menerima *input* dari pemain dan membelokkan ikan ke kiri atau ke kanan sesuai dengan *input* yang diterima. Pada baris 17 hingga baris 39, terdapat kode-kode yang berfungsi untuk menerima *input* dari pemain. Pada baris 17, kode-kode tersebut berfungsi untuk intansiasi variabel r. Pada baris 34, terdapat kode-kode untuk membatasi *swipe* yang dilakukan oleh pemain. Sehingga, ketika pemain melakukan *swipe* ke kanan atau ke kiri, maksimal pixel yang disentuh ketika melakukan *swipe* adalah 300. *Script* Manager dapat dilihat pada Gambar 4.27.

```
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
4
5   public class Manager : MonoBehaviour
6   {
7       public static Manager Instance { set; get; }
8       private Dictionary<int, Vector2> activateTouches = new Dictionary<int,
    Vector2>();
9
10      private void Awake(){
11          Instance = this;
12      }
13
14      public Vector3 GetPlayerInput(){
15          if (SaveManager.Instance.state.usingAccelerometer) {
16              Vector3 a = Input.acceleration;
17              a.y = a.z;
18              return a;
19          }
20          Vector3 r = Vector3.zero;
21          foreach (Touch touch in Input.touches) {
22              // If we just started pressing on the screen
23              if (touch.phase == TouchPhase.Began) {
24                  activateTouches.Add (touch.fingerId, touch.position);
25              }
26              // if we remove our finger off the screen
27              else if (touch.phase == TouchPhase.Ended) {
28                  if (activateTouches.ContainsKey (touch.fingerId))
29                      activateTouches.Remove (touch.fingerId);
30              }
```

```
31            // our finger is either moving, or stationary, in both cases, let's
     use delta
32            else {
33                float mag = 0;
34                r = (touch.position - activateTouches [touch.fingerId]);
35                mag = r.magnitude / 300; //
36                r = r.normalized * mag;
37            }
38        }
39        return r; }
40  }
```

Gambar 4.27 *Script* Manager

4.3.15 *Script* Helper

Script ini digunakan untuk memasukkan data yang ada pada *script* SaveState ke dalam sebuah string yang nantinya akan disimpan, dan sebaliknya. Pada baris 7 hingga baris 12, terdapat fungsi *serialize()* yang berfungsi untuk menyimpan data-data pemain ke dalam sebuah *string*. Fungsi ini nantinya akan dipanggil dalam prosedur *Save()*. Pada baris 14 hingga baris 18, terdapat fungsi *deserialize()* yang berfungsi untuk mengubah string yang berisi data pemain ke bentuk variabel-variabel seperti sebelumnya. Fungsi ini akan dipanggil dalam prosedur *Load*() pada *script* SaveManager. *Script* Helper dapat dilihat pada Gambar 4.28.

```
1 using System.IO;
2 using System.Xml.Serialization;
3
4 public static class Helper
5 {
6   // Serialize
7   public static string Serialize<T>(this T toSerialize){
8       XmlSerializer xml = new XmlSerializer (typeof(T));
9       StringWriter writer = new StringWriter ();
10      xml.Serialize (writer, toSerialize);
11      return writer.ToString ();
12  }
13  // Deserialize
14  public static T Deserialize<T>(this string toDeserialize){
15      XmlSerializer xml = new XmlSerializer (typeof(T));
16      StringReader reader = new StringReader (toDeserialize);
17      return (T)xml.Deserialize(reader);
18  }
19}
```

Gambar 4.28 *Script* Helper

4.3.16 *Script* SaveManager

Script SaveManager ini digunakan untuk menangani fungsi menyimpan atau memuat data, membeli ikan, dan menambahkan koin yang didapat dari balapan ke dalam data pemain yang nantinya akan disimpan. Pada baris 12, terdapat kode-kode yang berfungsi untuk mencegah perangkat beralih ke *sleep* ketika sedang memainkan *game*. Pada baris 15, terdapat

kode untuk memanggil prosedur *Load()*. Pada baris 18 hingga baris 22, terdapat kode-kode yang berfungsi untuk mengidentifikasi apakah perangkat pemain memiliki sensor *accelerometer*. Jika tidak, variabel *usingAccelerometer* akan bernilai *false* dan disimpan agar ketika di masa depan pemain menjalankan *game*, sistem tidak perlu melakukan pengecekan lagi. Pada baris 29, terdapat kode-kode untuk melakukan serialisasi data pemain ke dalam *string* "save" dan menyimpannya. Pada baris 34 hingga baris 46, terdapat kode-kode untuk melakukan pengecekan apakah terdapat penyimpanan berupa *string* "save", Jika iya, sistem akan melakukan deserialisasi terhadap string tersebut. Jika tidak, sistem akan membuat data baru dan membuka ikan pada indeks pertama pada toko agar bisa digunakan oleh pemain.

Pada baris 49, terdapat fungsi untuk melakukan pengecekan apakah ikan yang dipilih sudah dimiliki oleh pemain atau tidak. Pada baris 58, terdapat fungsi Buyfish. Pada fungsi itu, terdapat kode-kode yang berfungsi untuk melakukan pengecekan apakah total koin yang dimiliki pemain cukup untuk membeli ikan yang dipilih. Jika cukup, sistem akan mengurangkan total koin pemain berdasarkan harga ikan yang dibeli, lalu sistem akan memanggil prosedur *Unlockfish()* untuk membuka ikan yang dibeli agar bisa digunakan oleh pemain dan menyimpan data tersebut. Jika koin pemain tidak cukup, maka sistem mengembalikan nilai *false* yang berarti pemain gagal membeli ikan. Pada baris 74, terdapat prosedur *UnlockFish()*. Prosedur ini berfungsi untuk membuka ikan yang sebelumnya dikunci ketika pemain membeli ikan. Pada baris 79, terdapat prosedur *WinRace()*. Prosedur ini berfungsi untuk memberikan 500 koin kepada pemain jika ketika pemain berhasil memenangkan balapan. *Script* SaveManager dapat dilihat pada Gambar 4.29.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class SaveManager : MonoBehaviour {
6
7   public static SaveManager Instance { set; get;}
8   public SaveState state;
9
10  private void Awake()
11  {
12      Screen.sleepTimeout = (int)SleepTimeout.NeverSleep;
13      DontDestroyOnLoad (gameObject);
14      Instance = this;
15      Load ();
16
17      //Are we using accelerometer And can we use it?
18      if (state.usingAccelerometer && !SystemInfo.supportsAccelerometer) {
19          // if we can't, make sure we're not trying next time
20          state.usingAccelerometer = false;
21          Save ();
22      }
```

```
23  }
24
25  //save the whole of this saveState script to the player pref
26
27  public void Save()
28  {
29      PlayerPrefs.SetString ("save",Helper.Serialize<SaveState>(state));
30  }
31
32  //load the previuos saved state from the player prefs
33  public void Load()
34  {
35      //Do we already have a save?
36      if (PlayerPrefs.HasKey ("save")) {
37
38          state = Helper.Deserialize<SaveState> (PlayerPrefs.GetString ("save"));
39      }
40      else {
41          state = new SaveState ();
42          Save ();
43          Debug.Log ("No save file found, creating a new one!");
44          UnlockFish (0);
45      }
46  }
47
48  //Check if the fish is owned
49  public bool IsFishOwned(int index)
50  {
51      // Check if the bit is set, if so, the fish is owned
52      return (state.fishOwned & (1 << index)) !=0;
53  }
54
55  //attemp buying fish
56  public bool BuyFish(int index, int cost)
57  {
58      if (state.gold >= cost) {
59          // Enough money, remove from the current gold stack
60          state.gold -= cost;
61          UnlockFish(index);
62
63          // Save progress
64          Save ();
65
66          return true;
67      } else {
68          // Not enough money
69          return false;
70      }
71  }
72
73  // Unlock a fish in the fish owned
74  public void UnlockFish(int index){
75      // Toggle on the bit at index
76      state.fishOwned |= 1 << index;
77  }
78
79  public void WinRace(){
80      state.gold += 500;
81      Save ();
82  }
83
84  //RemoteSettings the whole save file
85}
```

Gambar 4.29 *Script* SaveManager

4.3.17 *Script* SaveState

Script SaveState ini merupakan *script* dimana data pemain seperti jumlah koin, karakter yang digunakan, jumlah ikan yang dimiliki, dan pilihan kontrol disimpan. Pada baris ketiga, terdapat kode-kode untuk menyimpan jumlah koin yang dimiliki pemain ke dalam variabel *gold*. Pada baris keempat, terdapat kode-kode yang berfungsi untuk menyimpan ikan yang dimiliki pemain. Data tersebut disimpan ke dalam variabel fishOwned. Pada baris keenam, ikan yang digunakan pemain disimpan pada variabel activeFish. Pada baris kedelapan, terdapat variabel usingAccelerometer dimana data mengenai apakah pemain menggunakan accelerometer atau tidak disimpan. *Script* SaveState dapat dilihat pada Gambar 4.30.

```
1   public class SaveState
2   {
3       public int gold = 0;
4       public int fishOwned = 11; // 0000 0000 0000 0000 0000 0000 0000 0000
5
6       public int activeFish = 0;
7
8       public bool usingAccelerometer = true;
9   }
```

Gambar 4.30 Script SaveState

4.3.18 *Script* Ensi

Script Ensi merupakan *script* yang berfungsi untuk menangani objek gambar-gambar informasi kasifikasi ilmiah ikan dan beberapa tombol pada halaman ensiklopedia. Pada baris ke-8 hingga baris ke-13, terdapat kode-kode yang berfungsi untuk mendeklarasi variabel yang digunakan dalam *script* ini. Pada baris ke-17 dan ke-18, terdapat kode-kode yang berfungsi untuk mengakses komponen dari tombol Prev lalu meyimpannya pada variabel Prev dan tombol Next lalu menyimpannya pada variabel Next. Pada baris ke-19 dan ke-20, terdapat kode-kode yang berfungsi untuk menghitung *children* dari objek Ikan dan objek List. Pada baris ke-23 hingga baris ke-28, terdapat kode-kode yang berfungsi untuk memasukkan *children* dari objek Ikan ke dalam *array* Ikan dan menonaktifkan seluruh objek di dalam *array* Ikan. Pada baris ke-31 hingga baris ke-36, terdapat kode-kode yang berfungsi untuk memasukkan *children* dari objek List ke dalam *array* List dan menonaktifkan seluruh objek di dalam *array* List. Kode-kode pada baris ke-38 dan 40 berfungsi untuk mengaktifkan objek pada indeks pertama yang terdapat di dalam *array* Ikan. Kode-kode pada baris ke-43 dan 44 berfungsi untuk mengaktifkan objek pada indeks pertama yang terdapat di dalam *array* List. Pada baris ke-47 terdapat prosedur untuk menonaktifkan objek pada indeks yang sedang dipilih, lalu berpindah

ke indeks sebelahnya dengan melakukan pengurangan pada indeks dan mengaktifkan objek pada indeks tersebut. Pada baris ke-63 terdapat prosedur untuk menonaktifkan objek pada indeks yang sedang dipilih, lalu berpindah ke indeks sebelahnya dengan melakukan penambahan pada indeks dan mengaktifkan objek pada indeks tersebut. Pada baris ke-79, terdapat prosedur untuk menghilangkan kapabilitas interaksi pada tombol Next jika indeks yang sedang aktif merupakan indeks terakhir dari array tersebut. Jika indeks yang sedang aktif adalah indeks pertama, maka sistem akan menghilangkan kapabilitas interaksi pada tombol Prev. *Script* Ensi dapat dilihat pada Gambar 4.31.

```
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
4   using UnityEngine.UI;
5
6   public class Ensi : MonoBehaviour {
7
8       public GameObject[] Ikan;
9       public GameObject[] List;
10      private int index;
11
12      private Button Prev;
13      private Button Next;
14
15      private void Start ()
16      {
17          Prev = GameObject.Find ("Prev").GetComponent<Button>();
18          Next = GameObject.Find ("Next").GetComponent<Button>();
19          Ikan = new GameObject[transform.childCount];
20          List = new GameObject[transform.childCount];
21
22          //fill the array with our model
23          for (int i = 0; i < transform.childCount; i++)
24              Ikan[i] = transform.GetChild(i).gameObject;
25
26          // we toggle of their renderer
27          foreach(GameObject go in Ikan)
28              go.SetActive(false);
29
30          //fill the array with our model
31          for (int i = 0; i < transform.childCount; i++)
32              List[i] = transform.GetChild(i).gameObject;
33
34          // we toggle of their renderer
35          foreach(GameObject go in List)
36              go.SetActive(false);
37
38          // we toggle on the first index
39          if(Ikan[0])
40              Ikan[0].SetActive(true);
41
42          // we toggle on the first index
43          if(List[0])
44              List[0].SetActive(true);
45      }
46
47      public void OnPrevClick(){
48          Ikan [index].SetActive (false);
```

```
49          List [index].SetActive (false);
50
51          index--;
52          if (index < 0)
53              index = Ikan.Length - 1;
54
55          Ikan [index].SetActive (true);
56
57          if (index < 0)
58              index = List.Length - 1;
59
60          List [index].SetActive (true);
61      }
62
63      public void OnNextClick(){
64          Ikan [index].SetActive (false);
65          List [index].SetActive (false);
66
67          index++;
68          if (index == Ikan.Length)
69              index = 0;
70
71          Ikan [index].SetActive (true);
72
73          if (index == List.Length)
74              index = 0;
75
76          List [index].SetActive (true);
77      }
78
79      private void Update () {
80          if (index == Ikan.Length - 1)
81              Next.interactable = false;
82          else
83              Next.interactable = true;
84
85          if (index == 0)
86              Prev.interactable = false;
87          else
88              Prev.interactable = true;
89      }
90  }
```

Gambar 4.31 *Script* Ensi

4.3.19 *Script InfoSelection*

Script ini berfungsi untuk menampilkan informasi klasifikasi ilmiah ikan yang digunakan pemain sesaat sebelum balapan dimulai. Pada baris ke-6, terdapat kode-kode yang berfungsi untuk mendeklarasikan variabel *array gameobject* dengan nama List. Pada baris ke-11, terdapat kode-kode yang berfungsi untuk menghitung *children* dari objek List. Pada baris ke-14 hingga baris ke-23, terdapat kode-kode yang berfungsi untuk memasukkan *children* dari objek List ke dalam *array* List dan menonaktifkan seluruh objek di dalam *array* List, lalu mengaktifkan objek pada indeks yang sama dengan indeks yang tersimpan pada variabel *activeFish*. Kode-kode pada baris ke-28 dan 29 berfungsi untuk menonaktifkan seluruh objek yang terdapat di dalam *array* List. Kode-kode pada baris ke-32 dan 33 berfungsi untuk

mengaktifkan objek pada indeks yang sama dengan indeks yang tersimpan pada variabel *activeFish*. *Script* InfoSelection dapat dilihat pada Gambar 4.32.

```
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
4
5   public class InfoSelection : MonoBehaviour {
6       public GameObject[] List;
7
8       // Use this for initialization
9       void Start ()
10      {
11          List = new GameObject[transform.childCount];
12
13          //fill the array with our model
14          for (int i = 0; i < transform.childCount; i++)
15              List[i] = transform.GetChild(i).gameObject;
16
17          // we toggle of their renderer
18          foreach(GameObject go in List)
19              go.SetActive(false);
20
21          // we toggle on the first index
22          if(List[SaveManager.Instance.state.activeFish])
23              List[SaveManager.Instance.state.activeFish].SetActive(true);
24      }
25
26      void Update(){
27          // we toggle of their renderer
28          foreach(GameObject go in List)
29              go.SetActive(false);
30
31          // we toggle on the first index
32          if(List[SaveManager.Instance.state.activeFish])
33              List[SaveManager.Instance.state.activeFish].SetActive(true);
34      }
35  }
```

Gambar 4.32 *Script* InfoSelection

4.3.20 *Script MinimapCamera*

Script MinimapCamera merupakan *script* yang menangani kamera yang betugas untuk menampilkan *minimap*. Pada baris ke-6, terdapat kode-kode yang berfungsi untuk mendeklarasikan variabel lookAt. Kode-kode pada baris ke-11 berfungsi untuk melakukan instansiasi *newPosition* yang bertipe *Vector3*. Kode-kode pada baris ke-12 berfungsi untuk menentukan posisi kamera pada sumbu y. Kode-kode pada baris ke-13 berfungsi untuk menentukan posisi kamera. Pada baris ke-16, terdapat kode-kode untuk memberikan fungsi rotasi pada kamera. *Script* MinimapCamera dapat dililhat pada Gambar 4.33.

```
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
```

```
4
5   public class MinimapCamera : MonoBehaviour {
6       public Transform lookAt;
7
8       private void Update()
9       {
10          //Update Posisiton
11          Vector3 newPosition = lookAt.position;
12          newPosition.y = transform.position.y;
13          transform.position = newPosition;
14
15          //Update the rotation
16          transform.rotation = Quaternion.Euler(90f, lookAt.eulerAngles.y, 0f);
17      }
18  }
```

Gambar 4.33 *Script* MinimapCamera

4.3.21 *Script* CharacterSelection

Script CharacterSelection merupakan *script* yang berfungsi untuk menangani proses pemilihan karakter. Pada baris ke-6, terdapat kode-kode yang berfungsi untuk mendeklarasikan variabel *array gameobject* dengan nama Player. Pada baris ke-10, terdapat kode-kode yang berfungsi untuk menghitung *children* dari objek Player. Pada baris ke-13 hingga baris ke-22, terdapat kode-kode yang berfungsi untuk memasukkan *children* dari objek Player ke dalam *array* Player dan menonaktifkan seluruh objek di dalam *array* Player, lalu mengaktifkan objek pada indeks yang sama dengan indeks yang tersimpan pada variabel *activeFish*. Kode-kode pada baris ke-28 dan 29 berfungsi untuk menonaktifkan seluruh objek yang terdapat di dalam *array* Player. Kode-kode pada baris ke-32 dan 33 berfungsi untuk mengaktifkan objek pada indeks yang sama dengan indeks yang tersimpan pada variabel *activeFish*. *Script* CharacterSelection dapat dilihat pada Gambar 4.34.

```
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
4
5   public class CharacterSelection : MonoBehaviour {
6       public GameObject[] Player;
7
8       // Use this for initialization
9       void Start () {
10          Player = new GameObject[transform.childCount];
11
12          //fill the array with our model
13          for (int i = 0; i < transform.childCount; i++)
14              Player[i] = transform.GetChild(i).gameObject;
15
16          // we toggle of their renderer
17          foreach(GameObject go in Player)
18              go.SetActive(false);
19
20          // we toggle on the first index
21          if(Player[SaveManager.Instance.state.activeFish])
```

```
22              Player[SaveManager.Instance.state.activeFish].SetActive(true);
23      }
24
25      void Update(){
26
27          // we toggle of their renderer
28          foreach(GameObject go in Player)
29              go.SetActive(false);
30
31          // we toggle on the first index
32          if(Player[SaveManager.Instance.state.activeFish])
33              Player[SaveManager.Instance.state.activeFish].SetActive(true);
34      }
35  }
```

Gambar 4.34 *Script* CharacterSelection

4.4 Hasil Pengujian

Hasil pengujian merupakan hasil dari pengujian yang telah dilakukan pada *game* yang telah dibangun. Pengujian ini dilakukan untuk mengetahui kelebihan dan kekurangan dari *game* Pengenalan Hewan Laut. Pengujian yang dilakukan ada dua jenis. Pengujian pertama adalah pengujian menggunakan kuesioner, dan pengujian kedua ada pengujian perangkat.

4.4.1 Pengujian Menggunakan Kuesioner

Saat melakukan pengujian menggunakan kuesioner, penulis meminta responden memainkan *game* yang telah dibangun. Setelah pemain selesai memainkan game tersebut, penulis meminta responden untuk mengisi kuesioner yang telah disiapkan. Kuesioner yang penulis siapkan terdiri dari dua jenis, yaitu cetak, dan *online* menggunakan Google Form. Responden pengujian ini terdiri dari 23 responden, dan tidak terfokus di suatu daerah atau usia tertentu. Daftar responden dapat dilihat pada Tabel 4.1.

Tabel 4.1 Daftar responden

No.	Nama	Usia
1.	Jarot Satrio	27
2.	Hermawan	-
3.	Gustama Cahya N.	24
4.	Adrian Mulyawan	20
5.	Juliandro Maulid Gantara	21
6.	Safariyana	19
7.	Yuyun	34
8.	Ridho	23
9.	Dwi Septiyono	25
10.	Nur fatimah H.M	23
11.	Rizky Fandra	25
12.	Datinasari	22
13.	Sahadi	37

14.	Santo Destriano	16
15.	Gita Thea Samudra	16
16.	Devi Mulyana	27
17.	Sylvia	35
18.	Zulfia Desnovita	34
19.	Dinda Evana Farica	10
20.	Yurais Jiwantoro	20
21.	Harry Pranata R.	23
22.	Wimas	22
23.	Nur Ramadhani	19

Data yang telah diperoleh dari kuesioner akan diolah untuk mendapatkan kesimpulan dari responden atas *game* yang telah dibangun. Untuk memudahkan menghitung hasil dari kuesioner, maka setiap jawaban diberi nilai sebagai berikut :

a. Nilai 5 untuk jawaban sangat baik (SB).
b. Nilai 4 untuk jawaban baik (B).
c. Nilai 3 untuk jawaban cukup (C).
d. Nilai 2 untuk jawaban kurang (K).
e. Nilai 1 untuk jawaban sangat kurang (SK).

Nilai-nilai tersebut digunakan untuk menghitung nilai akhir dari jawaban responden terhadap game yang telah dibangun. Rumus untuk menghitung nilai akhir tersebut dapat dilihat pada persamaan (4.1).

TN = Total Nilai

R = Jumlah responden.

NA = Nilai akhir

NT = Nilai tertinggi dari pilihan jawaban.

$$\mathrm{NA} = \frac{\left(\frac{TN}{R}\right)}{\mathrm{NT}} . 100\% \qquad (4.1)$$

1. Angka 0% - 20% = Sangat Kurang.
2. Angka 21% - 400% = Kurang.
3. Angka 41% - 60% = Cukup.
4. Angka 61% - 80% = Baik.
5. Angka 81% - 100% = Sangat Baik

Hasil dari jawaban kuesioner yang diberikan kepada responden terhadap *game* ini dapat dilihat pada Tabel 4.2.

Tabel 4.2 Hasil kusioner responden

No.	Pernyataan	SK	K	C	B	SB	%
		1	2	3	4	5	
1.	Topik pengenalan hewan laut yang diangkat penulis cukup menarik.	-	1	2	9	11	86%
2.	Informasi mengenai klasifikasi ilmiah ikan yang diberikan sudah akurat.	-	-	2	10	11	87.8%
3.	Topik pengenalan laut yang diangkat sesuai untuk pengguna dengan kategori semua umur.	-	1	3	7	12	86%
4.	Aplikasi Game Pengenalan Hewan Laut merupakan media pembelajaran alternatif yang efektif.	-	-	3	8	12	87.8%
5.	Media pembelajaran alternatif ini mampu untuk meningkatkan motivasi belajar.	-	-	4	6	13	87.8%
6.	Kemudahan penggunaan aplikasi.	-	-	1	6	16	93%
7.	Antarmuka aplikasi.	-	2	4	6	11	82.6%
8.	Kualitas objek yang digunakan pada aplikasi ini, seperti ikan, sirkuit, dan batu.	-	2	6	5	10	80%
9.	Gameplay aplikasi Game Pengenalan Hewan Laut cukup menarik.	-	2	2	10	9	82.6%
10.	Tingkat kesulitan gameplay.	1	1	5	6	10	80%

Berdasarkan hasil perhitungan dan persentase dari haril kuesioner diatas, dilakukan analisis terhadap kinerja Game Pengenalan Hewan Laut. Analisis dapat dilihat pada Tabel 4.3.

Tabel 4.3 Analisis hasil kuesioner

No.	Pernyataan	Nilai	Keterangan
1.	Topik pengenalan hewan laut yang diangkat penulis cukup menarik.	86%	Pada aspek ini dinyatakan sangat baik dengan nilai akhir sebesar 86% karena topik yang diangkat menurut responden sudah menarik.
2.	Informasi mengenai klasifikasi ilmiah ikan yang diberikan sudah akurat.	87.8%	Pada aspek ini dinyatakan sangat baik dengan nilai akhir sebesar 87.8% karena informasi yang didiberikan sudah akurat.
3.	Topik pengenalan laut yang diangkat sesuai untuk pengguna dengan kategori semua umur.	86%	Pada aspek ini dinyatakan sangat baik dengan nilai akhir sebesar 86% karena topik yang diangkat sudah sesuai untuk pengguna semua umur.
4.	Aplikasi Game Pengenalan Hewan Laut merupakan media pembelajaran alternatif yang efektif.	87.8%	Pada aspek ini dinyatakan sangat baik dengan nilai akhir sebesar 87.8% karena aplikasi ini merupakan media pemberlajaran alternatif yang efektif.

5.	Media pembelajaran alternatif ini mampu untuk meningkatkan motivasi belajar.	87.8%	Pada aspek ini dinyatakan sangat baik dengan nilai akhir sebesar 87.8% karena media pemberlajaran alternatif ini mampu meningkatkan motivasi untuk belajar.
6.	Kemudahan penggunaan aplikasi.	93%	Pada aspek ini dinyatakan sangat baik dengan nilai akhir sebesar 93% karena aplikasi ini sangat mudah untuk digunakan.
7.	Antarmuka aplikasi.	82.6%	Pada aspek antarmuka ini dinyatakan sangat baik dengan nilai akhir sebesar 82.6%. Akan tetapi, banyak responden yang menyarankan agar antarmuka lebih ditingkatkan.
8.	Kualitas objek yang digunakan pada aplikasi ini, seperti ikan, sirkuit, dan batu.	80%	Pada aspek ini dinyatakan baik dengan nilai akhir sebesar 80% karena objek yang digunakan menurut responden sudah cukup baik. Terdapat saran yang meminta jumlah biota laut ditingkatkan.
9.	Gameplay aplikasi Game Pengenalan Hewan Laut cukup menarik.	82.6%	Pada aspek ini dinyatakan sangat baik dengan nilai akhir sebesar 82.6% karena *gameplay* menurut responden aplikasi ini sudah menarik. Responden menyarankan untuk meningkatkan animasi ikan dan menambahkan hiasan dan penghalang di sirkuit.
10.	Tingkat kesulitan gameplay.	80%	Pada aspek ini dinyatakan baik dengan nilai akhir sebesar 80% karena menurut responden tingkat kesulitan dari gameplay aplikasi ini sudah baik.

4.4.2 Pengujian Perangkat

Pada pengujian perangkat ini, penulis menjalankan *game* yang telah dibangun pada beberapa perangkat untuk menguji apakah game ini dapat berjalan di perangkat lainnya. Penulis tidak akan memberikan spesifikasi penuh dari perangkat yang digunakan, melainkan hanya spesifikasi yang berpengaruh dalam menjalankan *game* ini.

a. Infinix Hot Note X551.

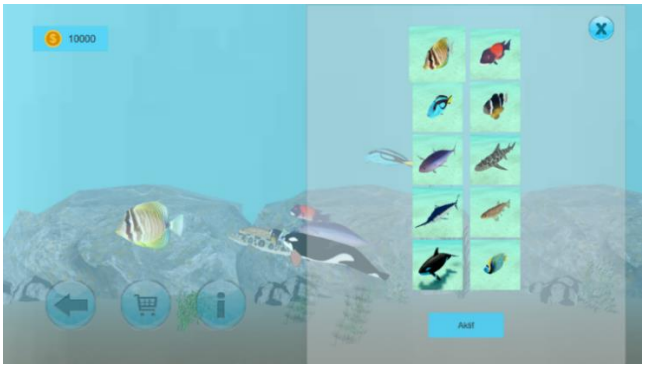
Tabel 4.4 Spesifikasi Infinix Hot Note X551

Display	Display type	TFT capacitive touchscreen, 16M colors
	Display size	5.5 inches
	Resolution	720 x 1280 pixels (~267 ppi pixel density)
Memory	Internal memory	16 GB, 2 GB RAM
Platform	OS	Android OS v5.1 Lollipop
	Chipset	MediaTek MT6592

	CPU	Octa core 1.4 GHz Cortex-A7
	GPU	Mali-450
Features	Sensors	Accelerometer, Proximity, Light Sensor, G-Sensor

Berdasarkan spesifikasi perangkat yang dapat dilihat pada Tabel 4.4, pengujian menggunakan perangkat ini semuanya berjalan lancar sesuai dengan apa yang dibangun penulis. Hasil pengujian pada perangkat ini dapat dilihat pada Gambar 4.35 untuk halaman akuarium dan Gambar 4.36 untuk panel toko.

Gambar 4.35 Halaman Akuarium pada Infinix Hot Note X551



Gambar 4.36 Panel Toko pada Infinix Hot Note X551

b. Oppo Neo 7.

Tabel 4.5 Spesifikasi Oppo Neo 7

Display	Display type	IPS LCD capacitive touchscreen, 16M colors
	Display size	5.0 inches
	Resolution	540 x 960 pixels, 16:9 ratio (~220 ppi density)
Memory	Internal memory	16 GB, 1 GB RAM
Platform	OS	Android OS v5.1 Lollipop
	Chipset	Qualcomm MSM8916 Snapdragon 410
	CPU	Quad-core 1.2 GHz Cortex-A53

	GPU	Mali-400MP2
Features	Sensors	Accelerometer, proximity, compass

Berdasarkan spesifikasi perangkat yang dapat dilihat pada Tabel 4.5, pada pengujian menggunakan perangkat ini terdapat beberapa masalah peletakan objek. Hal ini dikarenakan adanya perbedaan resolusi pada perangkat dengan acuan resolusi yang digunakan ketika *game* dibangun. Hasil pengujian pada perangkat ini dapat dilihat pada Gambar 4.37 untuk halaman akuarium, dan Gambar 4.38 untuk panel toko.

Gambar 4.37 Halaman Akuarium pada Oppo Neo 7

Gambar 4.38 Panel Toko pada Oppo Neo 7

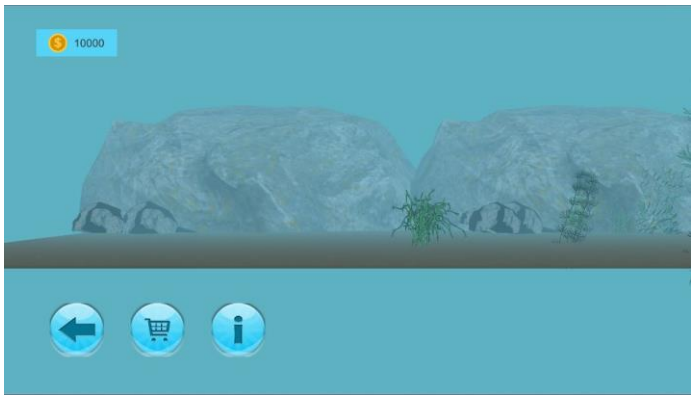
c. Lenovo Vibe K5 Plus

Tabel 4.6 Spesifikasi Lenovo Vibe K5 Plus

Display	Display type	IPS LCD capacitive touchscreen, 16M colors
	Display size	5.0 inches
	Resolution	1080 x 1920 pixels, 16:9 ratio (~441 ppi density)
Memory	Internal memory	16 GB, 1 GB RAM

Platform	OS	Android OS v5.1 Lollipop
	Chipset	Qualcomm MSM8939v2 Snapdragon 616
	CPU	Octa-core (4x1.5 GHz Cortex-A53 & 4x1.2 GHz Cortex-A53)
	GPU	Adreno 405
Features	Sensors	Accelerometer, proximity

Berdasarkan spesifikasi perangkat yang dapat dilihat pada Tabel 4.6, pada pengujian menggunakan perangkat ini juga terdapat beberapa masalah pada peletakan objek. Hal ini dikarenakan adanya perbedaan resolusi pada perangkat dengan acuan resolusi yang digunakan ketika *game* dibangun. Hasil pengujian pada perangkat ini dapat dilihat pada Gambar 4.39 untuk halaman akuarium, dan Gambar 4.40 untuk panel toko.



Gambar 4.39 Halaman Akuarium pada Lenovo Vibe K5 Plus

Gambar 4.40 Panel Toko pada Lenovo Vibe K5 Plus

4.5 Kelebihan dan Kekurangan Aplikasi

Dalam membangun sebuah aplikasi, tentunya terdapat beberapa kesalahan dan kekurangan. Hal ini tidak terlepas dari keterbatasan yang dimiliki oleh penulis. Adapun rincinan kekurangan pada *game* Pengenalan Hewan Laut ini adalah sebagai berikut:

a. Kelebihan Aplikasi:
    1. Aplikasi atau *game* ini memberikan pengetahuan mengenai klasifikasi ilmiah hewan laut.
    2. Aplikasi ini cukup ringan untuk dijalankan di berbagai jenis perangkat berbasis android yang hanya memiliki RAM 1 GB.
    3. Aplikasi ini cukup mudah untuk digunakan oleh pengguna kategori semua umur.

b. Kekurangan Aplikasi:
    1. Tampilan antarmuka masih terlalu sederhana dan animasi untuk karakter pemain masih terlihat kaku.
    2. Tidak adanya tingkatan level kesulitan seperti mudah, normal, dan sulit.
    3. Tidak adanya penerapan salah jalan ketika sedang balapan, dan kurangnya hambatan di sirkuit.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil penelitian, analisis, perancangan, implementasi, hingga *Game* Pengenalan Hewan Laut ini dibangun dan diuji, maka dapat diambil kesimpulan antara lain :

a. Aplikasi ini telah berhasil dibangun dan dapat dijalankan pada perangkat berbasis Android meskipun masih memiliki beberapa kekurangan.
b. Aplikasi ini dapat digunakan sebagai media belajar alternatif dalam mengenal hewan laut.
c. Aplikasi ini dapat digunakan untuk kategori pengguna semua umur.

5.2 Saran

Berdasarkan kekurangan dan keterbatasan yang ada pada aplikasi ini, terdapat beberapa saran yang ingin penulis sampaikan :

a. Untuk pengembangan ke depan perlu ada perbaikan pada antarmuka dan animasi karakter agar terlihat lebih menarik.
b. Untuk pengembangan ke depan perlu ditambahkan penerapan tingkat kesulitan pada *gameplay*, seperti tingkat kecepatan, hambatan pada sirkuit, fitur kemampuan khusus yang bisa digunakan saat balapan.
c. Untuk pengembangan ke depan perlu ditambahkan jenis-jenis hewan laut lainnya agar lebih bervariasi dan terlihat menarik.

DAFTAR PUSTAKA

Bagaimanamultimedia. (2013). *Pengertian Multimedia dan Semua Tentang Multimedia*. Diambil 16 Juli 2017, dari https://bagaimanamultimedia.wordpress.com/2011/07/23/pengertian-multimedia-dan-semua-tentang-multimedia/

Blender.org. (2017). *Home of the Blender Project - Free 3D Creation Software*. Diambil 15 Juli 2017, dari https://www.blender.org/

IGRS.ID. (2016). *Pertanyaan Umum | IGRS - Indonesian Game Rating System*. Diambil 16 Juli 2016, dari https://igrs.id/faq

Oktriaviani, D. (2012). *Accelerometer & Gyroscope*. Diambil 16 Juli 2017, dari http://oktriaviani.blogspot.co.id/2012/06/accelerometer-gyroscope_16.html

Salen, K., & Zimmerman, E. (2003). *Rules of Play: Game Design Fundamentals*. *MIT Press*, *2004*, 688. https://doi.org/10.1093/intimm/dxs150

Wikipedia. (2016). *FL Studio*. Diambil 16 Juli 2017, dari https://en.wikipedia.org/wiki/FL_Studio

Wikipedia. (2017a). *Adobe Photohsop*. Diambil 16 Juli 2017, dari https://en.wikipedia.org/wiki/Adobe_Photoshop#File_format

Wikipedia. (2017b). *Android (operating system)*. Diambil 15 Juli 2017, dari https://en.wikipedia.org/wiki/Android_(operating_system)

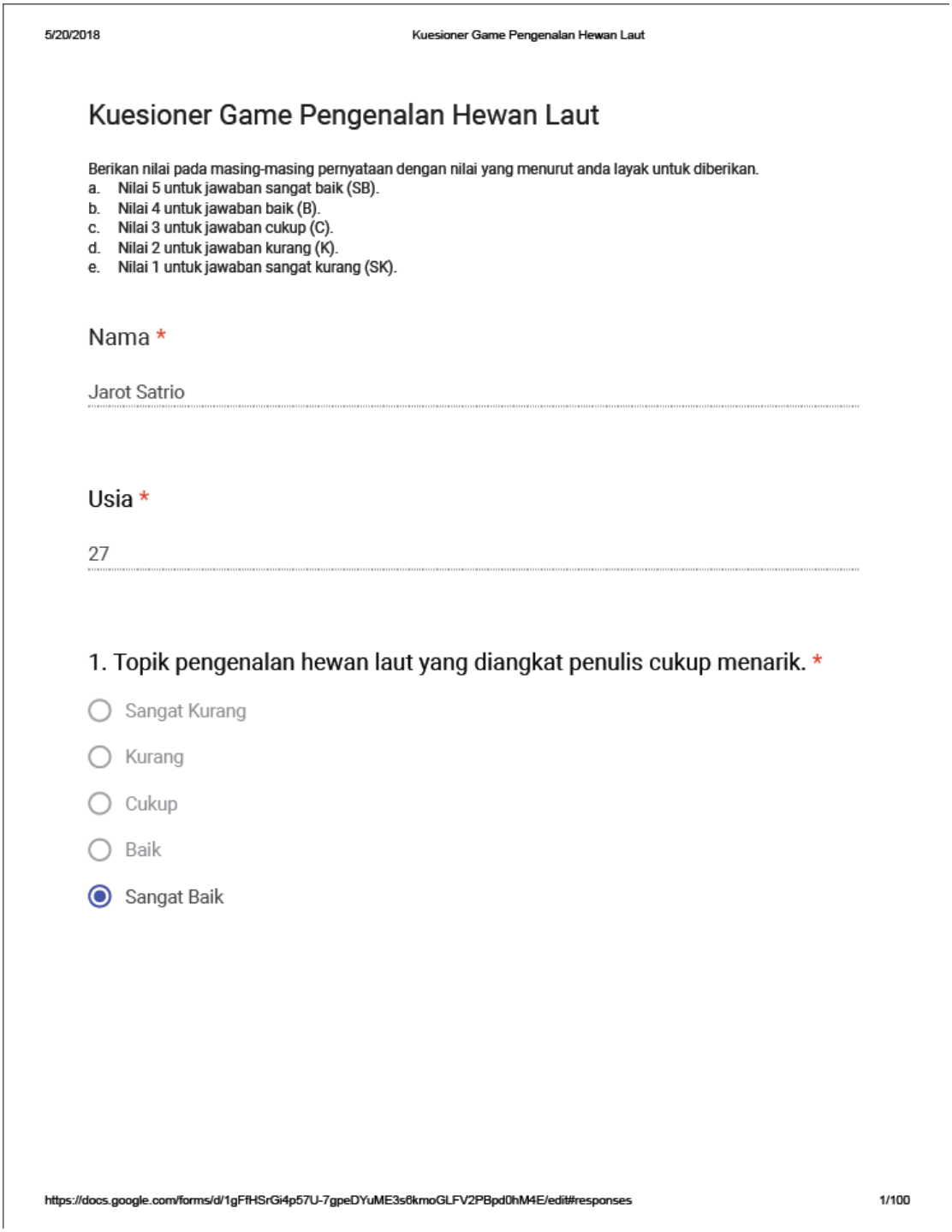
Wikipedia. (2017c). *C Sharp (programming language)*. Diambil 16 Juli 2017, dari https://en.wikipedia.org/wiki/C_Sharp_(programming_language)#Versions

Wikipedia. (2017d). *Unity (game engine)*. Diambil 16 Juli 2017, dari https://en.wikipedia.org/wiki/Unity_(game_engine)

LAMPIRAN

Lampiran A

Kuesioner (Respon dari masing-masing responden dengan total 23 responden dapat dilihat pada *file* yang telah penulis lampirkan di dalam cd).



5/20/2018 Kuesioner Game Pengenalan Hewan Laut

Kuesioner Game Pengenalan Hewan Laut

Berikan nilai pada masing-masing pernyataan dengan nilai yang menurut anda layak untuk diberikan.
a. Nilai 5 untuk jawaban sangat baik (SB).
b. Nilai 4 untuk jawaban baik (B).
c. Nilai 3 untuk jawaban cukup (C).
d. Nilai 2 untuk jawaban kurang (K).
e. Nilai 1 untuk jawaban sangat kurang (SK).

Nama *

Jarot Satrio

Usia *

27

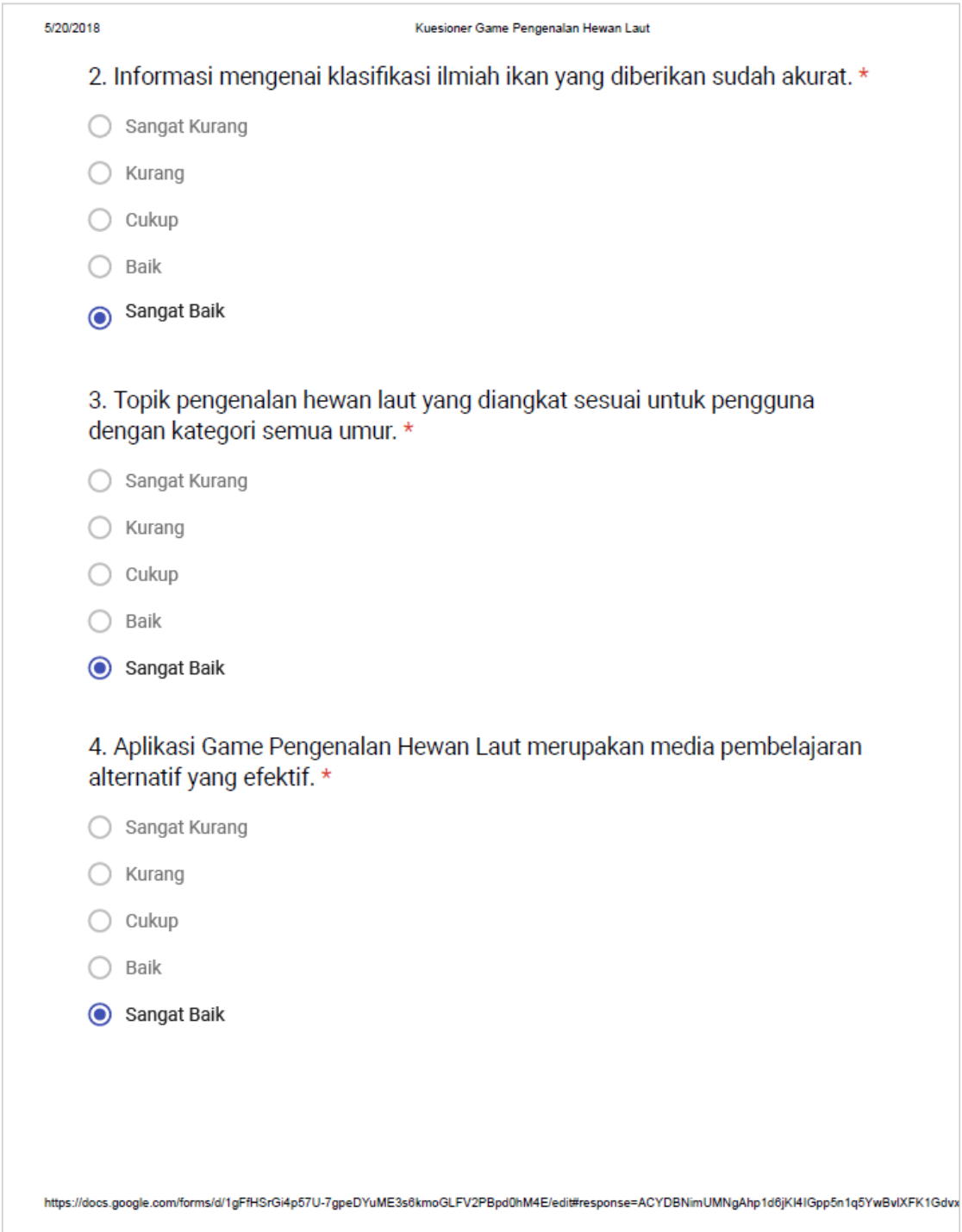
1. Topik pengenalan hewan laut yang diangkat penulis cukup menarik. *

- ○ Sangat Kurang
- ○ Kurang
- ○ Cukup
- ○ Baik
- ◉ Sangat Baik

https://docs.google.com/forms/d/1gFfHSrGi4p57U-7gpeDYuME3s8kmoGLFV2PBpd0hM4E/edit#responses 1/100

Lampiran A

Kuesioner



5/20/2018 Kuesioner Game Pengenalan Hewan Laut

2. Informasi mengenai klasifikasi ilmiah ikan yang diberikan sudah akurat. *

- Sangat Kurang
- Kurang
- Cukup
- Baik
- **Sangat Baik**

3. Topik pengenalan hewan laut yang diangkat sesuai untuk pengguna dengan kategori semua umur. *

- Sangat Kurang
- Kurang
- Cukup
- Baik
- **Sangat Baik**

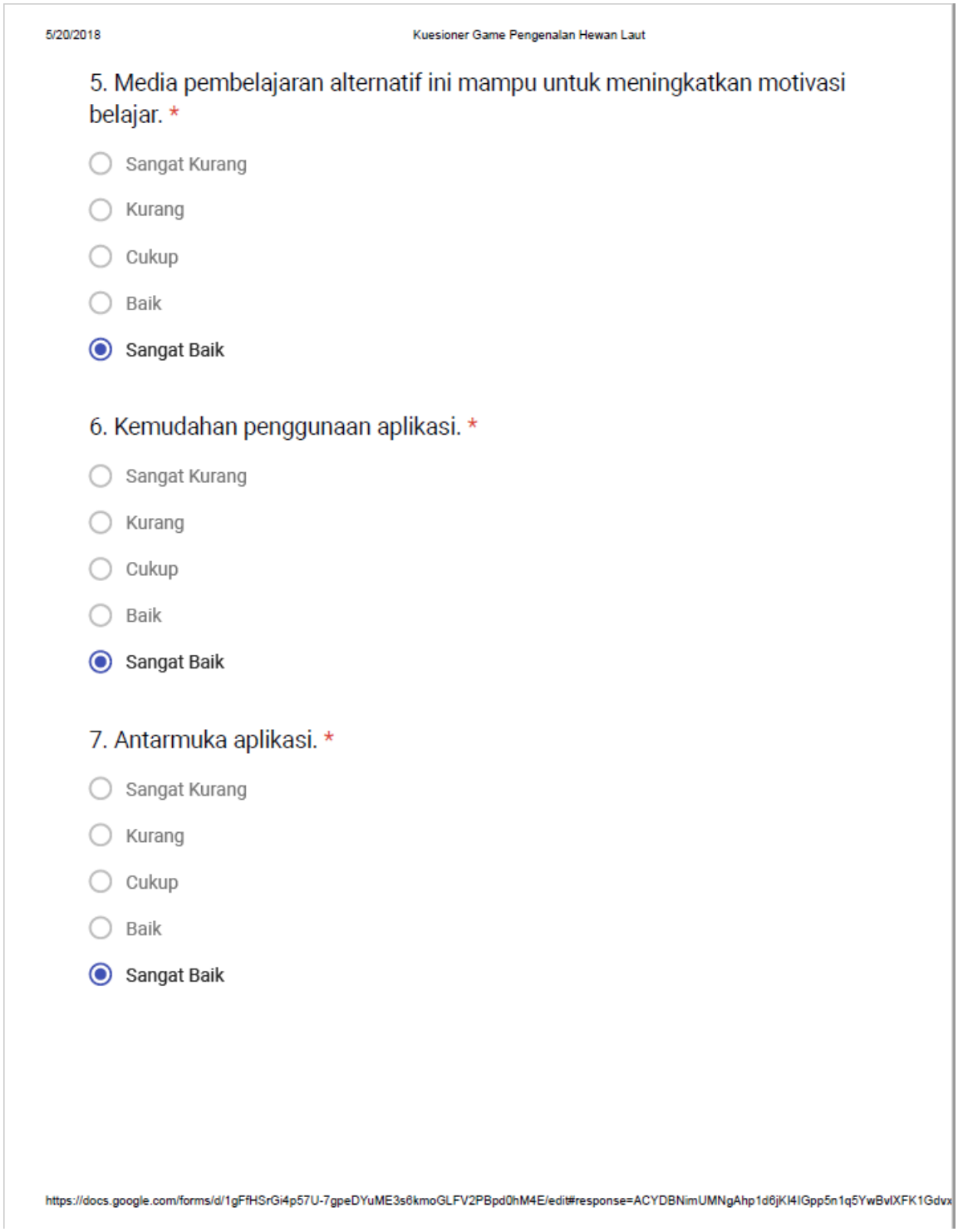
4. Aplikasi Game Pengenalan Hewan Laut merupakan media pembelajaran alternatif yang efektif. *

- Sangat Kurang
- Kurang
- Cukup
- Baik
- **Sangat Baik**

https://docs.google.com/forms/d/1gFfHSrGi4p57U-7gpeDYuME3s6kmoGLFV2PBpd0hM4E/edit#response=ACYDBNimUMNgAhp1d6jKI4IGpp5n1q5YwBvlXFK1Gdvx

Lampiran A

Kuesioner



5/20/2018 Kuesioner Game Pengenalan Hewan Laut

5. Media pembelajaran alternatif ini mampu untuk meningkatkan motivasi belajar. *

- ○ Sangat Kurang
- ○ Kurang
- ○ Cukup
- ○ Baik
- ◉ Sangat Baik

6. Kemudahan penggunaan aplikasi. *

- ○ Sangat Kurang
- ○ Kurang
- ○ Cukup
- ○ Baik
- ◉ Sangat Baik

7. Antarmuka aplikasi. *

- ○ Sangat Kurang
- ○ Kurang
- ○ Cukup
- ○ Baik
- ◉ Sangat Baik

https://docs.google.com/forms/d/1gFfHSrGi4p57U-7gpeDYuME3s8kmoGLFV2PBpd0hM4E/edit#response=ACYDBNimUMNgAhp1d6jKI4IGpp5n1q5YwBvlXFK1Gdvx

Lampiran A

Kuesioner

8. Kualitas objek yang digunakan pada aplikasi ini, seperti ikan, sirkuit, dan batu. *

- ○ Sangat Kurang
- ○ Kurang
- ○ Cukup
- ○ Baik
- ◉ Sangat Baik

9. Gameplay aplikasi Game Pengenalan Hewan Laut cukup menarik. *

- ○ Sangat Kurang
- ○ Kurang
- ○ Cukup
- ○ Baik
- ◉ Sangat Baik

10. Tingkat kesulitan gameplay. *

- ○ Sangat Kurang
- ○ Kurang
- ○ Cukup
- ○ Baik
- ◉ Sangat Baik

Saran *

UI / UX lebih ditingkatkan lagi.