IMPLEMENTASI APLIKASI GAME TRIVIA CODING C# DENGAN METODE GDLC BERBASIS ANDROID

SKRIPSI

Oleh:

Winsen

190210005

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK DAN KOMPUTER
UNIVERSITAS PUTERA BATAM
2023

IMPLEMENTASI APLIKASI GAME TRIVIA CODING C# DENGAN METODE GDLC BERBASIS ANDROID

SKRIPSI

Untuk memenuhi salah satu syarat
memperoleh gelar sarjana

Oleh:

Winsen

190210005

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK DAN KOMPUTER
UNIVERSITAS PUTERA BATAM
2023

SURAT PERNYATAAN ORISINALITAS

Yang bertanda tangan di bawah ini saya:

Nama	: Winsen
NPM	: 190210005
Fakultas	: Teknik dan Komputer
Program Studi	: Teknik Informatika

Menyatakan bahwa **"Skripsi"** yang saya buat dengan judul:

Implementasi Aplikasi Game Trivia Coding C# Dengan Metode GDLC Berbasis Android

Adalah hasil karya sendiri dan bukan "duplikasi" dari karya orang lain. Sepengetahuan saya, di dalam naskah Skripsi ini tidak terdapat karya ilmiah atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis dikutip didalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila ternyata di dalam naskah Skripsi ini dapat dibuktikan terdapat unsur-unsur PLAGIASI, saya bersedia naskah Skripsi ini digugurkan dan gelar akademik yang saya peroleh dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku.

Demikian pernyataan ini saya buat dengan sebenarnya tanpa ada paksaan dari siapapun

Batam 27 Januari 2023

Winsen
190210005

IMPLEMENTASI APLIKASI GAME TRIVIA CODING C# DENGAN METODE GDLC BERBASIS ANDROID

SKRIPSI

Untuk memenuhi salah satu syarat
memperoleh gelar sarjana

Oleh:
Winsen
190210005

Telah disetujui oleh Pembimbing pada tanggal
seperti yang tertera di bawah ini

Batam, 27 Januari 2023

Ellbert Hutabri, S.Kom., M.Kom.
Pembimbing

ABSTRAK

C# dapat digunakan untuk mengembangkan banyak jenis aplikasi termasuk: Aplikasi desktop (untuk Windows, WPF), Aplikasi Web (ASP.Net), Aplikasi Seluler (Xamarin) sehingga terdapat banyak sekali peluang kerja jika belajar C#. Menurut survey oleh di tahun 2023 C# menduduki peringkat ke 4 untuk bahasa pemograman yang paling dicari untuk pekerjaan, bahkan lebih populer dari typescipt, C++, dan php. 34% dari game seluler gratis teratas dibuat menggunakan C#, bersama dengan aplikasi realitas virtual. Tetapi di *playstore* dan sumber lainnya minim sekali aplikasi dan media pembelajaran C# yang berbahasa indonesia akibatnya pengguna yang tidak berbahasa inggris susah sekali belajar bahasa pemograman ini. Maka untuk itu *game* ini dikembangkan karena kurangnya media pembelajaran C# berbasis game dan bahasa Indonesia. Tujuan pengembangan game ini adalah untuk menambah media pembelajaran coding C# berbasis bahasa Indonesia dan meningkatkan keterampilan pengguna dalam coding C#. Perancangan game ini menggunakan metode GDLC yang memiliki tujuh tahapan yaitu Initiation, Team Building, Pre Production, Main Production, Alpha Version, Beta Version dan Release Version. Penelitian ini juga menggunakan UML untuk menggambarkan cara kerja game ini yaitu Use Case Diagram, Activity Diagram, Sequence Diagram dan Class Diagram. Peneliti menggunakan Unity dan Visual Studio Code untuk mendesain game ini. Hasil dari penelitian ini adalah sebuah game trivia coding C# berbasis Android yang berbahasa indonesia dan mampu membantu kalangan pengguna yang tidak berbahas inggris dalam pembelajaran *coding C#.* Hasil pengujian game ini berfungsi sesuai dengan yang diinginkan.

Kata Kunci : *GDLC, Game Edukasi, UML, Unity, Visual Studio Code*

ABSTRACT

C# can be used to develop many types of applications including: Desktop applications (for Windows, WPF), Web Applications (ASP.Net), Mobile Applications (Xamarin) so there are tons of job opportunities when learning C#. According to a survey by 2023, C# is ranked 4th for the most sought-after programming language for work, even more popular than typecipt, C++, and PHP. 34% of the top free mobile games are built using C#, along with virtual reality applications. But in Playstore and other sources, there are very few applications and C# learning media that speak Indonesian, as a result, users who don't speak English find it very difficult to learn this programming language. So for that this game was developed due to the lack of game-based C# learning media and Indonesian language. The purpose of developing this game is to add Indonesian language-based C# coding learning media and improve user skills in C# coding. The design of this game uses the GDLC method which has seven stages, namely Initiation, Team Building, Pre Production, Main Production, Alpha Version, Beta Version and Release Version. This study also uses UML to describe how this game works, namely Use Case Diagrams, Activity Diagrams, Sequence Diagrams and Class Diagrams. Researchers use Unity and Visual Studio Code to design this game. The results of this study are an Android-based C# coding trivia game that is in Indonesian and is able to help users who don't speak English in learning C# coding. The results of testing this game function as desired.

*Keywords**:** Education Game, GDLC, UML, Unity, Visual Studio Code*.

KATA PENGANTAR

Puji syukur kepada Tuhan Yang Maha Esa yang telah melimpahkan segala rahmat dan karunianya, sehingga penulis dapat menyelesaikan laporan tugas akhir yang merupakan salah satu persyaratan untuk menyelesaikan program studi strata satu (S1) pada Program Studi Teknik Informatika Universitas Putera Batam.

Penulis menyadari bahwa skripsi ini masih jauh dari sempurna. Karena itu, kritik dan saran akan senantiasa penulis terima dengan senang hati. Dengan segala keterbatasan, penulis menyadari pula bahwa skripsi ini takkan terwujud tanpa bantuan, bimbingan, dan dorongan dari berbagai pihak. Untuk itu, dengan segala kerendahan hati, penulis menyampaikan ucapan terima kasih kepada:

1. Ibu Dr. Nur Elfi Husda, S.Kom., M.SI. selaku Rektor Universitas Putera Batam;
2. Bapak Welly Sugianto, S.T., M.M. selaku Dekan Fakultas Teknik dan Komputer;
3. Bapak Andi Maslan, S.T., M.SI. selaku Ketua Program Studi Teknik Informatika;
4. Bapak Ellbert Hutabri, S.Kom., M.Kom. selaku pembimbing skripsi pada Program Studi Teknik Informatika Universitas Putera Batam;
5. Dosen dan Staff Universitas Putera Batam;
6. Orang tua penulis yang memberikan dukungan dan doa agar penulis dapat menyelesaikan laporan ini;

7. Teman-teman Teknik Informatika 2019 yang memberikan semangat dan bantuan selama penyusunan laporan ini;

Semoga Tuhan Yang Maha Esa membalas kebaikan dan selalu mencurahkan hidayah serta taufik-Nya, Amin

Batam, 27 Januari 2023

Winsen

190210005

DAFTAR ISI

HALAMAN SAMPUL ..... **i**
HALAMAN JUDUL ..... **ii**
SURAT PERNYATAAN ORISINALITAS ..... Error! Bookmark not defined.
ABSTRAK ..... **v**
ABSTRACT ..... **vi**
KATA PENGANTAR ..... **vii**
DAFTAR ISI ..... **ix**
DAFTAR GAMBAR ..... **xi**
DAFTAR TABEL ..... **xiii**
BAB I PENDAHULUAN **1**
1.1 Latar Belakang 1
1.2 Identifikasi Masalah 3
1.3 Batasan Masalah 4
1.4 Rumusan Masalah 4
1.5 Tujuan Penelitian 5
1.6 Manfaat Penelitian 5
BAB II TINJAUAN PUSTAKA **7**
2.1 Teori Dasar 7
2.1.1 Game 7
2.1.2 *Game* Edukasi 13
2.1.3 *Android* 15
2.1.4 C# 18
2.2 Teori Khusus 21
2.2.1 UML (Unified Modeling Language) 21
2.2.2 *GDLC (Game Development Life Cycle)* 29
2.2.3 *Software* pendukung 32
2.2.3.1 *UNITY* 32
2.2.3.2 Visual Studio Code 33
2.3 Penelitian Terdahulu 35
2.4 Kerangka Pemikiran 42

BAB III METODOLOGI PENELITIAN 43
3.1 Metode Penelitian 43
3.2 Desain Penelitian 43
3.3 Proses Perancangan Sistem 43
3.4 Peralatan Yang Digunakan 46
3.5 *UML* (Desain Unified Modeling Language) 47
3.6 Perancangan Tampilan 62
3.7 Jadwal Penelitian 70
3.8 Metode Pengujian Sistem 70
3.9 Lokasi Penelitian 70
BAB IV HASIL DAN PEMBAHASAN 71
4.1 Hasil Penelitian 71
4.1.1 Halaman Login 71
4.1.2 Halaman Utama 72
4.1.3 Halaman Levels 73
4.1.4 Halaman GamePlay 74
4.1.5 Halaman Level Cleared 75
4.1.6 Halaman *Learn* 76
4.1.7 Halaman *About* 77
4.1.8 Halaman *Game Over* 78
4.2 Pembahasan 79
BAB V KESIMPULAN DAN SARAN 85
5.1 Kesimpulan 85
5.2 Saran 85
DAFTAR PUSTAKA 87
LAMPIRAN 89
Lampiran 1 : Surat Izin Penelitian 89
Lampiran 2 : Lampiran Jurnal 90
Lampiran 3 : Lampiran Hasil Turnitin Skripsi 91
Lampiran 4 : Lampiran Hasil Turnitin Jurnal 92
Lampiran 5 : Lampiran Pengujian 93
Lampiran 6 : Riwayat Hidup 94

DAFTAR GAMBAR

Gambar 2.1 Logo *Android*..... 16
Gambar 2.2 Logo *C#*..... 18
Gambar 2.3 *Game Development Life Cycle*..... 29
Gambar 2.4 Logo *Unity*..... 32
Gambar 2.5 Logo *Visual Studio Code*..... 34
Gambar 2.6 Kerangka Pemikiran 42
Gambar 3.1 Tahapan Penelitian 43
Gambar 3.2 *Use Case Diagram* 47
Gambar 3.3 *Activity Diagram Login*..... 48
Gambar 3.4 *Activity Diagram Menu Play* 50
Gambar 3.5 *Activity Diagram Menu Learn* 52
Gambar 3.6 *Activity Diagram Menu About* 53
Gambar 3.7 *Activity Diagram Menu Exit* 54
Gambar 3.8 *Sequence Diagram Login* 55
Gambar 3.9 *Sequence Diagram Register* 56
Gambar 3.10 *Sequence Diagram Play* 57
Gambar 3.11 *Sequence Diagram Learn* 58
Gambar 3.12 *Sequence Diagram About* 59
Gambar 3.13 *Sequence Diagram Exit* 60
Gambar 3.14 *Class Diagram* 61
Gambar 3.15 Rancangan Tampilan Login 62
Gambar 3.16 Rancangan Tampilan Utama 63
Gambar 3.17 Rancangan Tampilan Levels 64
Gambar 3.18 Rancangan Tampilan *Play* 65
Gambar 3.19 Rancangan Tampilan *Level Cleared* 66
Gambar 3.20 Rancangan Tampilan *Learn* 67
Gambar 3.21 Rancangan Tampilan *About* 68

Gambar 3.22 Rancangan Tampilan *Gameover* 69
Gambar 3.23 Lokasi Penelitian 70
Gambar 4.1 Tampilan Halaman Login 71
Gambar 4.2 Tampilan Halaman Utama 72
Gambar 4.3 Tampilan Halaman Levels 73
Gambar 4.4 Tampilan Halaman *Play* 74
Gambar 4.5 Tampilan Halaman Level Cleared 75
Gambar 4.6 Tampilan Halaman *Learn* 76
Gambar 4.7 Tampilan Halaman *About* 77
Gambar 4.8 Tampilan Halaman *Gameover* 78

DAFTAR TABEL

Tabel 1.1 Rating Bahasa Pemograman 1
Tabel 2.1 *Use Case Diagram* 22
Tabel 2.2 *Activity Diagram* 24
Tabel 2.3 *Sequence Diagram* 24
Tabel 2.4 *Class Diagram* 27
Tabel 3.1 Data Peneliti 46
Tabel 3.2 Jadwal Penelitian 70
Tabel 4.1 Pengujian Tampilan Login 79
Tabel 4.2 Pengujian Tampilan Utama 80
Tabel 4.3 Pengujian Tampilan Levels 81
Tabel 4.4 Pengujian Tampilan *Gameplay* 82
Tabel 4.5 Pengujian Tampilan *Level Cleared* 83
Tabel 4.6 Pengujian Tampilan *Learn* 83
Tabel 4.7 Pengujian Tampilan *About* 84
Tabel 4.8 Pengujian Tampilan *Gameover* 84

BAB I

PENDAHULUAN

1.1 Latar Belakang

C *Sharp* (C#) adalah salah satu bahasa pemrograman yang paling populer, bersama dengan *Ruby on Rails* dan *Java*. Dua dekade setelah pembuatannya, C# terus menjadi salah satu bahasa pemrograman yang paling banyak digunakan (Krajewski, 2021). C# berada pada urutan kelima setelah *Python*, C, *Java*, *C++* yang paling banyak digunakan dalam merancang aplikasi, dapat dilihat pada tabel berikut.

Tabel 1.1 Rating Bahasa Pemograman

Jun 2022	Jun 2021	Change	Programming language	Ratings	Change
1	2	↑	Python	12.20%	+0.35%
2	1	↓	C	11.91%	-0.64%
3	3		Java	10.47%	-1.07%
4	4		C++	9.63%	+2.26%
5	5		C#	6.12%	+1.79%
6	6		Visual Basic	5.42%	+1.40%
7	7		JavaScript	2.09%	-0.24%
8	10	↑	SQL	1.94%	+0.06%
9	9		Assembly language	1.85%	-0.21%
10	16	↑↑	Swift	1.55%	+0.44%
11	11		Classic Visual Basic	1.33%	-0.40%
12	18	↑↑	Delphi/Object Pascal	1.32%	+0.26%
13	8	↓↓	PHP	1.25%	-0.97%
14	23	↑↑	Objective-C	1.02%	+0.33%
15	20	↑↑	Go	1.02%	+0.07%
16	14	↓	R	0.98%	-0.22%
17	15	↓	Perl	0.76%	-0.41%
18	38	↑↑	Lua	0.76%	+0.43%
19	13	↓↓	Ruby	0.75%	-0.48%
20	26	↑↑	Prolog	0.74%	+0.18%

Sumber : (TIOBE, 2022)

C# dapat digunakan untuk mengembangkan banyak jenis aplikasi termasuk: Aplikasi desktop (untuk Windows, WPF), Aplikasi Web (ASP.Net), Aplikasi Seluler (Xamarin) sehingga terdapat banyak sekali peluang kerja jika belajar C#.

Menurut survey oleh (Figuière, 2023) di tahun 2023 C# menduduki peringkat ke 4 untuk bahasa pemograman yang paling dicari untuk pekerjaan, bahkan lebih populer dari typescipt, C++, dan php. C# sering digunakan untuk aplikasi perusahaan besar seperti memproses transaksi bank. Secara khusus, jika organisasi Anda bekerja dalam industri game (atau berencana) maka C# adalah investasi yang berharga. 34% dari game seluler gratis teratas dibuat menggunakan C#, bersama dengan aplikasi realitas virtual. Ini sering digunakan untuk mengembangkan game menggunakan *Unity*, *termasuk Temple Run Trilogy* dan *Assassin's Creed: Identity*. Pengembang Anda juga dapat menemukan kesalahan dengan lebih mudah di C# karena kode tersebut diperiksa oleh kerangka kerja sebelum menjadi aplikasi. Ini juga memberikan landasan yang layak dalam bahasa C lain seperti C ++ membuatnya lebih mudah dipelajari dalam jangka panjang. Menawarkan jalur pembelajaran dan pengembangan (L&D) yang jelas. Namun, dengan semua itu, C# memang membutuhkan lebih banyak waktu dan upaya untuk belajar daripada Python.

Berdasarkan fakta diatas maka programmer perlu belajar dan menguasai bahasa pemograman ini dengan banyaknya permintaan keahlian perancangan aplikasi menggunakan C#. Tetapi C# bukan bahasa pemroraman yang mudah untuk dipelajari secara mandiri, karena bukan bahasa pemrograman yang sering diajarkan pada sekolah ataupun universitas. Selain itu C# menggunakan pustaka *.NET* menambah lapisan

kerumitan lainnya. Perpustakaan di *.NET* sering diperbarui dan ada ribuan sumber daya yang perlu di pelajari sebelum mendapatkan pekerjaan dalam pemrograman C#, selain itu kesulitan mempelajari c# adalah sumber belajar masih kurang terutama yang berbahasa Indonesia. Beberapa media belajar pemrograman C# salah sarunya *w3school,* tetapi tidak berbasis *game* dan tidak berbahasa Indonesia, dan aplikasi di *playstore* berbahasa indonesia yang mengajarkan tentang *coding* C# masih sangat sedikit.

Berdasarkan masalah di atas peneliti tertarik untuk membuat sebuah aplikasi *game* trivia *coding* C# dengan metode *GDLC* berbasis *android* agar dpat digunakan untuk belajar bahasa pemrograman C# secara mandiri dan menyenangkan sambil bermain, sehingga pelajar tidak cepat bosan dan lebih mudah mengerti tentang C#.

1.2 Identifikasi Masalah

Berdasarkan dari latar belakang diatas, dapat kita kemukakan beberapa hal yang bisa menjadi akar dari pemasalahan yang ada, yaitu :

1. Sudah ada media pembelajaran C# tetapi belum berbasis *game*.
2. Sumber belajar bahasa pemograman C# masih kurang, terutama yang berbahasa Indonesia.
3. Aplikasi di *play*store berbahasa Indonesia yang mengajarkan tentang *coding* C# masih sangat minim.

4. C# menduduki peringkat ke 4 untuk bahasa pemograman yang paling dicari untuk pekerjaan

1.3 Batasan Masalah

Berdasarkan dari latar belakang yang sudah ada dan juga indentifikasi beberapa permasalahan, maka batasan masalah dari penelitian ini yaitu :

1. Materi pada *game* ini hanya meliputi pemograman C# dasar.
2. Aplikasi yang dirancang hanya bisa dimainkan oleh sistem *android* versi 8 keatas.
3. Perancangan *game* ini mengunakan *Unity* versi 2021.3.5f1.
4. *Game* yang dirancang menggunakan jenis *game* trivia yang hanya bisa dimainkan oleh *single player*.
5. Perancangan game ini menarget orang yang ingin belajar C# dan menjadi programmer C# tetapi tidak begitu memahami bahasa *inggris.*
6. Penelitian ini dilakukan di tempat tinggal sang peneliti.

1.4 Rumusan Masalah

Berdasarkan dari latar belakang yang sudah dijelaskan, maka di simpulkan rumusan masalah penelitian ini adalah :

1. Bagaimana cara merancang *game* trivia *coding* C# dengan *unity* dan *visual studio code*?
2. Bagaimana game ini bermanfaat dalam pembelajaran C# bagi yang tidak berbahasa inggris?

1.5 Tujuan Penelitian

Adapun beberapa tujuan dari penelitian yang sudah dilakukan oleh peneliti, yang diantaranya yaitu :

1. Merancang *game* trivia *coding* c# dengan *unity* dan *visual studio code*.
2. Mengimplementasikan *game* trivia *coding* c# dengan metode *GDLC* berbasis *android*.
3. *Game* ini dibuat untuk mempermudah pembelajaran *coding* C.

1.6 Manfaat Penelitian

Adapun juga beberapa manfaat penelitian yang diharapkan oleh peneliti yaitu :

1. Manfaat Teoritis
    a. Penelitian ini diharapkan dapat membuat para pembaca dapat memahami dan mengerti tentang *coding* C#
    b. Hasil dari penelitian ini, diharapkan bisa digunakan sebagai referensi serta acuan dalam pengembangan *game* trivia *coding* C# berbasis *android*.
    c. Penelitian ini diharapkan dapat meningkatkan kemampuan program C#
2. Manfaat Praktis
    1. Penulis/Peneliti
        a. Meningkatkan pengetahuan merancang *game* trivia *coding* c# dengan metode *GDLC* berbasis *android*
        b. Meningkatkan kemampuan menggunakan *UNITY* dalam merancang *game* trivia *coding* c# dengan metode *GDLC* berbasis *android*

2. Pengguna
    a. Untuk menjadikan pembelajaran *coding* C# dasar sesuatu yang menyenangkan.
    b. Untuk menambah pengetahuan pengguna tentang dasar bahasa pemrograman C#.
    c. Untuk meningkatkan ketertarikan pengguna untuk menjadi seorang pengembang perangkat lunak berbasis bahasa pemrograman C#.

BAB II

TINJAUAN PUSTAKA

2.1 Teori Dasar

2.1.1 Game

Menurut (Tandoğan, 2021) *Game* merupakan Suatu kegiatan yang dirancang untuk suatu tujuan, *game* ini melibatkan pengalaman bermain dengan menciptakan dunianya sendiri. (Sharon & Crawford, 2021) berpendapat *game* adalah sebuah aktivitas, bentuk permainan, atau olahraga yang memiliki aturan main, kemampuan terukur, dan tujuan yang jelas. Pendapat lain mengatakan (González, García, & González, 2020) merupakan proposal pemandangan fiksi yang mencakup tujuan, seperangkat aturan dan struktur pengembangan serta j*UML*ah pemain per game (satu orang atau tim). Sebuah game dapat dimainkan untuk bersenang-senang atau dengan tujuan pendidikan karena simulasi kehidupan nyata di dalamnya. Itu juga dapat mewakili situasi yang tidak nyata. Berdasarkan pendapat di atas, dapat disimpulkan bahwa *game* adalah suatu kegiatan atau sistem yang dirancang untuk suatu tujuan yang berbentuk suatu permainan atau olahraga dan memiliki aturan main dan digunakan untuk bersenang-senang dengan tujuan pendidikan

Jenis – jenis *game* menurut (Pavlovic, 2020) adalah sebagai berkut :

a. *Sandbox*

Istilah "*Sandbox*" mungkin lebih dikenali dari penggunaannya di bidang teknologi atau bahkan sebagai mode terbuka yang tersedia dalam *game* tertentu. Ini sering dikaitkan dengan pilihan pemain, lingkungan terbuka, dan *gameplay* non-linear. Genre kotak pasir telah berkembang dari ceruk kecil menjadi mencakup berbagai macam judul

Dalam *game* ini, pemain sering kali memiliki tujuan dan jalur naratif yang kurang konkret untuk dikejar. Alih-alih mengalahkan bos dan menyelamatkan sang putri, pemain mungkin menghadapi berbagai tugas yang dapat pemain selesaikan dengan berbagai cara. Hal ini menarik pemain ke dalam pengalaman yang lebih imersif, mendorong eksperimen dengan mekanika yang mungkin asing.

Contoh *Minecraft, Grand Theft Auto, Sim.*

b. *Real time strategy (RTS)*

Awalnya diciptakan sebagai istilah pemasaran untuk *Dune II Westwood Studios*, *game* strategi *real-time* sudah ada selama bertahun-tahun sebelum sebagian besar pemain mengetahui genre itu. Berkat popularitasnya yang bertahan lama dan pertumbuhan sub-genre baru, *game* RTS tetap menjadi bagian yang mencolok dari lanskap video *game*. Dalam judul RTS pola dasar, pemain manusia dan *AI Dune II* mengontrol faksi yang berbeda dan bersaing satu sama lain secara

bersamaan dalam "waktu nyata", oleh karena itu disebut "strategi waktu nyata", sebagai lawan dari strategi berbasis giliran. Gim ini biasanya mencakup manajemen sumber daya dan peta, dan sering menampilkan tampilan dari atas ke bawah. *Warcraft, Age of Empires, dan Command & Conquer* adalah beberapa judul RTS paling popular.

c. Shooter (FPS dan TPS)

Penembak adalah genre lama lainnya yang mengembangkan beberapa cabang awal dan bercabang menjadi dua sub-genre utama: penembak orang pertama (FPS) dan penembak orang ketiga (TPS). Perbedaan utamanya adalah perspektif. FPS mensimulasikan *sudut* pandang manusia pada umumnya, menunjukkan pada dasarnya apa yang dilihat karakter dalam *game* pemain di waralaba seperti *Half-Life, Call of Duty,* dan *DOOM*. TPS menarik perspektif ke belakang dan menampilkan seluruh karakter pemain dan lingkungan sekitarnya, seperti dalam seri *Gears of War dan Tom Clancy's The Division*.

d. *Multiplayer online battle arena* (MOBA)

Subgenre yang semakin populer dengan koneksi ke berbagai gaya lain, *game* arena pertempuran online multipemain berbagi banyak fitur dengan *game* strategi waktu nyata. Ada perspektif top-down yang menekankan manajemen peta dan sumber daya, ditambah persaingan waktu nyata antar pemain. Perbedaan utama antara *game* MOBA dan RTS adalah karakter dan peran pemain. Dalam MOBA, pemain mungkin memiliki penyelarasan faksi dan banyak dasar RTS yang

dimainkan, tetapi biasanya pemain hanya mengontrol satu karakter. Itu sangat kontras dengan kebanyakan *game* RTS, tempat pemain membangun komunitas dan memerintahkan banyak unit. Contoh *Dota 2, League of Legends*.

e. *Role-playing games* (RPG, ARPG dan banyak lagi)

Premis dasar dari permainan role-*play*ing sederhana dan ada di mana-mana di banyak permainan: pemain membuat atau mengendalikan karakter yang kemudian dapat pemain tingkatkan melalui poin pengalaman. RPG adalah landasan *game*, tetapi tidak ada satu *game* pun yang dapat mewakili genre tersebut karena RPG tumbuh dan berkembang menjadi banyak sub-genre. Contoh *RPG Skyrim, The Witcher 3 (ARPG)*.

f. Simulasi dan olahraga

Genre ini telah banyak berkembang selama bertahun-tahun dan *user* mungkin benar-benar melihatnya dalam sudut pandang yang sama. Tetapi hanya dengan kemajuan teknologi grafis mereka mulai menawarkan pengalaman imersif yang unik. Iterasi terbaru memberikan tingkat detail yang mengesankan dan menunjukkan seberapa banyak yang bisa dilakukan dengan *game*. Permainan olahraga telah berkembang dalam variasi, menawarkan kemitraan penuh dengan organisasi olahraga besar, mulai dari trek balap hingga lapangan atau lapangan. *NBA 2K dan Madden NFL* adalah dua contoh terkenal yang menampilkan rekreasi mendetail dari bola basket dan sepak bola

profesional, sedangkan Forza adalah *game* balap mobil bergaya simulasi. Contoh *Forza Motorsport, NBA2K*.

g. Teka-teki dan permainan pesta

Teka-teki dan permainan pesta juga memiliki tumpang tindih yang signifikan, dengan keduanya menekankan mekanisme permainan. Pemain dapat mengharapkan untuk memainkan permainan berdasarkan tema atau permainan meja tradisional dengan aturan tertentu. *Game* pesta mengambil premis itu sedikit lebih jauh dan sering kali menyertakan elemen multipemain. Mereka juga melipatgandakan *gameplay*. Seri *Mario Party* sangat populer, dan menghasilkan lebih dari 10 angsuran dan spin-off. Contoh *The Talos, Jackbox Party Pack*.

h. Aksi petualangan

Di antara genre hybrid paling awal yang dapat dikenali, *game* aksi-petualangan memiliki fokus mendalam pada plot dan pertarungan melalui keterlibatan cerita dan mekanisme *gameplay* yang ketat. Alhasil, banyak *game* yang masuk ke dalam kategori ini, termasuk franchise klasik *Legend of Zelda* yang membuka jalan bagi banyak franchise. Sebagian besar pengguna menarik garis antara aksi dan petualangan dalam cara *game* menyeimbangkan cerita dan fitur seperti pertarungan simulasi. Seri *Assassin's Creed Ubisoft* telah menjadi andalan sejak 2007 berkat rilis baru hampir setiap tahun yang menampilkan pencelupan tingkat tinggi. Mereka juga membawa para

*game*r ke berbagai lokasi dan lanskap bersejarah mulai dari London selama Revolusi Industri (Sindikat) hingga Yunani kuno (Odyssey) Contoh Sekiro: *Shadows Die Twice, Assasin's Creed*.

i. *Survival dan horror*

Game survival dan horor memiliki banyak kesamaan, bahkan mereka menciptakan sub-genre mereka sendiri (survival horror). Secara khusus, *game* horor sering kali memiliki beberapa fitur dasar yang sama dengan *game* bertahan hidup, meskipun kebalikannya jarang terjadi. Karena pengembang menambahkan lebih banyak fitur FPS dan penembak konvensional ke judul-judul horor dan bertahan hidup tertentu, bahkan ada perdebatan yang sedang berlangsung tentang cara mendefinisikan *game-game* ini. Mekanika inti dari permainan bertahan hidup berpusat pada manajemen sumber daya, sering kali menggabungkan sistem kerajinan atau penyelamatan yang dapat pemain gunakan untuk membantu menjaga karakter pemain tetap hidup. Minecraft adalah menonjol populer, seperti *Don't Starve*. Dan kemudian ada *game* seperti *The Long Dark*, yang berfokus sepenuhnya pada elemen bertahan hidup dengan mode khusus yang meningkatkan kesulitan. contoh *The Long Dark. Don't Starve*.

j. *Platformer*

Mencari genre yang konsepnya tidak banyak berubah selama bertahun-tahun? Dalam hal nostalgia dan pengabdian pada kerajinan, genre platformer mencakup berbagai macam *game* yang masih dengan bangga

memamerkan akarnya di penggulir samping 2D paling awal. Platformer melibatkan berlari, memanjat, dan melompat saat pemain mengeksplorasi dan bekerja melalui level yang menantang. *Game* platform menampilkan tampilan samping dan kontrol sederhana, dengan *Donkey Kong* sering dianggap sebagai contoh nyata pertama. *Game* itu meneruskan obor ke *Super Mario Bros*. dan kemudian *Sonic the Hedgehog*. Bergerak beberapa tahun ke depan, judul *Crash Bandicoot* pertama *Naughty Dog* menemukan penontonnya dengan *sudut* kamera yang berbeda (head-on) dan banyak pesona.Contoh *Cuphead, Crash Bandicoot*.

2.1.2 *Game* Edukasi

Game edukasi adalah *game* yang dirancang untuk mengajar orang tentang subjek tertentu atau keterampilan tertentu. Mereka dibuat terutama untuk anak-anak dan siswa dari segala usia, dan dapat digunakan baik di dalam maupun di luar kelas. *Game* edukasi adalah bagian dari *game* serius, salah satu perbedaan penting adalah antara *game* edukasi dan pembelajaran berbasis *game.* Pembelajaran berbasis permainan adalah tindakan mempelajari informasi atau keterampilan melalui permainan. Itu bisa terjadi di hampir semua *game*, baik serius maupun tidak. Contoh dari ini mungkin logika pengajaran catur dan keterampilan strategi. *Game* edukasi telah dikembangkan dengan mempertimbangkan hasil pembelajaran tertentu. Pembelajaran berbasis *game* terjadi dalam *game* edukasi, menjadikan *game* edukasi sebagai istilah umum untuk tindakan pembelajaran dan metode pendidikan (King, 2021).

Jenis – jenis *game* edukasi menurut (schoolsoftware, 2022) adalah sebagai berikut :

a. *Quizalize* (Kuis atau Trivia)

   Permainan kuis yang menyenangkan dan menarik ini memungkinkan pemain menguji pengetahuan seseorang.

   Trivia game (Murphy, 2022) adalah salah satu jenis *Game* edukasi, dimana merupakan jenis permainan pemain (yang dapat bermain secara individu atau dalam tim) ditanyai tentang topik yang berbeda dan mereka harus mendapatkan jawaban yang benar sebanyak mungkin.

b. *Pictionary*

   Klasik lama tetapi juga cara yang bagus bagi pemain untuk mem*visual*isasikan pemahaman mereka dalam permainan tim yang menyenangkan.

c. *Draw swords*

   Permainan cepat ini menguji keterampilan motorik halus pemain dan mendorong pemikiran cepat, serta menghasilkan kompetisi yang sehat.

d. *Puzzles*

   Permainan kelompok kreatif ini mendorong pemain untuk bekerja sama dan mem*visual*isasikan konsep akademik secara abstrak.

e. *Bingo*

   Sebuah permainan cepat dan sederhana yang tidak pernah gagal untuk memotivasi pemain dalam pembelajaran mereka, papan tulis dan pulpen

atau kertas dan pulpen/pensil, ditambah daftar istilah atau konsep khusus mata pelajaran misalnya, angka, fonik, kosakata kunci, rumus ilmiah, atau tokoh sejarah.

f. Scatter-gories

Permainan yang menyenangkan ini akan mendorong pemain untuk berpikir 'di luar kotak' dan memanfaatkan berbagai pengetahuan mata pelajaran.

g. Hangman

Gim tradisional namun interaktif yang meningkatkan pengejaan dan pengetahuan pemain, tetapi juga menyenangkan.

h. Charades

Permainan sederhana namun klasik ini adalah cara yang bagus untuk mendorong pemain agar bangkit dari tempat duduknya dan berpartisipasi.

2.1.3 *Android*

Android adalah Sistem Operasi *open source* berbasis *Linux* untuk perangkat mobile seperti *smartphone* dan komputer tablet. *Android* menawarkan pendekatan terpadu untuk pengembangan aplikasi untuk perangkat seluler yang berarti pengembang hanya perlu mengembangkan untuk *android*, dan aplikasi mereka harus dapat berjalan di perangkat berbeda yang diberdayakan oleh *Android*. Versi beta

pertama dari *Android Software Development Kit* (SDK) dirilis oleh Google pada tahun 2007 dimana versi komersial pertama, *android* 1.0, dirilis pada bulan September 2008.

Gambar 2.1 Logo *Android*
Sumber : developer.*android*.com

Kode sumber untuk *Android* tersedia di bawah lisensi perangkat lunak bebas dan sumber terbuka. Google menerbitkan sebagian besar kode di bawah Lisensi Apache versi 2.0 dan sisanya, perubahan kernel Linux, di bawah Lisensi Publik Umum GNU versi 2. Aplikasi *Android* biasanya dikembangkan dalam bahasa *Java* menggunakan Kit Pengembangan Perangkat Lunak *Android*. Setelah dikembangkan, aplikasi *Android* dapat dikemas dengan mudah dan terjual habis baik melalui toko seperti *Google Play*, *SlideME, Opera Mobile Store, Mobango, F-droid* dan *Amazon Appstore*. Kode nama *android* berkisar dari A sampai N saat ini, seperti *Aestro, Blender, Cupcake, Donut, Eclair, Froyo, Gingerbread, Honeycomb, Ice Cream Sandwitch, Jelly Bean, KitKat, Lollipop dan Marshmallow*. Mari kita pahami sejarah *android* secara berurutan. (Overview, 2022)

Android 8.0 Oreo adalah pembaruan besar kedelapan untuk sistem operasi Android, menghadirkan fitur dan peningkatan baru untuk pengembang aplikasi. Ini dirilis pada 21 Agustus 2017 dan digunakan pada perangkat Android dalam berbagai bentuk untuk kasus penggunaan bisnis dan pribadi. Disusul dengan Android 9.0 Pie yang pertama kali diungkap ke publik pada 6 Agustus 2018. Salah satu fitur terbaru Android Oreo adalah mode picture-in-picture. Ini memungkinkan pengguna untuk menonton video yang diskalakan ke bagian layar yang lebih kecil saat menggunakan aplikasi lain. Masa pakai baterai juga diperpanjang dengan meminimalkan aktivitas latar belakang pada aplikasi yang jarang digunakan. Pengumuman juga telah diperbarui. Pengembang dapat membuat saluran yang dapat dikonfigurasi pengguna untuk pemberitahuan jenis tampilan. Pengguna juga dapat menunda notifikasi. Aplikasi di layar beranda menampilkan titik notifikasi, yang merupakan indikasi lain dari notifikasi. Selain itu, aplikasi ini memiliki instruksi penyimpanan data yang lebih baik dengan kuota ruang yang ditentukan untuk data yang di-cache. Grup navigasi keyboard tersedia untuk developer yang ingin mengoptimalkan aplikasinya untuk perangkat Chrome OS. Aplikasi Android dapat didistribusikan ke Chromebook dan perangkat Chrome OS lainnya melalui Google Play Store. Versi Oreo - API level 26 - juga menyertakan beberapa alat yang dapat diintegrasikan pengembang ke dalam aplikasi mereka. Salah satu alatnya adalah kerangka kerja IsiOtomatis yang memungkinkan pengguna untuk secara otomatis mengisi informasi yang mereka simpan saat mengirimkan informasi ke formulir aplikasi. Platform ini juga memungkinkan penyematan pintasan dan widget di aplikasi, dukungan multilayar, pemilihan teks cerdas, dan ikon peluncur adaptif.

2.1.4 C#

Variabel pada penelitian ini adalah C#. C# atau yang dibacakan C *sharp* adalah bahasa pemrograman sederhana yang digunakan untuk umum, dalam artian bahasa pemrograman ini dapat digunakan untuk berbagai fungsi misalnya untuk pemrograman server-side pada situs web, membangun aplikasi desktop ataupun mobile, pemrograman *game* dan sebagainya. Selain itu C# juga bahasa pemrograman yang berorientasi objek, jadi C# juga mengusung konsep objek seperti inheritance, *Class*, polymorphism dan encapsulation.

Gambar 2.2 Logo C#
Sumber : www.seeklogo.com

Dalam prakteknya C# sangat tepat dengan framework yang disebut .NET *Framework*, *framework* yang nanti digunakan untuk mengkompilasi dan menjalankan kode C#. Tujuan dibangunnya C# adalah sebagai bahasa pemrograman utama dalam lingkungan .NET *Framework* (lihat C#). Banyak pihak juga menganggap bahwa *Java* dengan C# saling bersaing, bahkan ada juga yang menyatakan jika belajar *Java* maka belajar C# akan sangat mudah dan begitu juga sebaliknya. Meskipun hal tersebut sebenarnya tidak salah karena perlu diketahui sebelum adanya C# *Microsoft* mengembangkan J++ dengan mencoba membuat *Java* agar berjalan pada platform

Windows, karena adanya masalah dari pihak luar maka Microsoft mengembangkan proyek J++ dan beralih untuk mengembangkan bahasa baru yaitu C#.

Manfaat Menggunakan C#. Keserbagunaan mungkin merupakan fitur C# yang paling menonjol, tetapi ada banyak keuntungan lain bagi siapa saja yang bekerja dengannya. Beberapa yang paling penting termasuk:

a. Waktu pengembangan lebih cepat

   C# memiliki beberapa fitur yang memungkinkan pengembang untuk membuat kode lebih cepat dibandingkan dengan bahasa lain. Beberapa fitur tersebut termasuk bahasa yang diketik secara statis dan mudah dibaca, sintaks yang terasa seperti versi *Java* yang diperluas, dan perpustakaan besar yang diisi dengan fungsionalitas tingkat tinggi.

b. Skalabilitas tinggi

   Sifat pengkodean statis dari C# mengubah semua programnya menjadi produk andal yang dapat dengan mudah disesuaikan dan diubah. Itu berarti para insinyur dapat dengan cepat melakukan penyesuaian dan membangun di atas program C# apa pun untuk memperluas fungsionalitasnya dan mendukung lebih banyak pengguna.

c. Berorientasi pada objek

   C# telah merangkul pemrograman berorientasi objek sedemikian rupa sehingga mungkin bahasa yang lebih baik memanfaatkannya. Faktanya, berorientasi objek memungkinkan C# menjadi sangat efisien dan sangat fleksibel, yang semuanya membuat pengembangan lebih mudah dan tidak terlalu intensif sumber daya.

d. Kurva belajar yang lembut

Sebagai bahasa tingkat tinggi, C# sangat mudah dipelajari dan dipahami. Dan itu tanpa mempertimbangkan banyak fitur bawaan yang sangat mudah digunakan. Terlebih lagi, setiap insinyur yang sudah mengetahui C++ atau *Java* akan merasa betah saat pertama kali menggunakan C#, karena bahasa ini memiliki banyak fitur yang sama dan pendekatan keseluruhan untuk pemrograman.

e. Komunitas besar

C# adalah salah satu bahasa yang paling banyak digunakan di dunia, yang berarti ada banyak pengembang C# yang siap membantu pegguna. Bukan itu saja. Menjadi produk Microsoft, C # mendapat dukungan dari raksasa teknologi, yang diterjemahkan menjadi bantuan ahli, sumber daya tambahan, dan pembaruan yang sering.

Menurut (bairesdev, 2022) C# memiliki serangkaian kelemahan yang perlu pengguna pertimbangkan sebelum menerapkannya untuk proyek digital pengguna. Yang paling menonjol termasuk:

a. Berbasis *Windows*

Karena C# adalah bagian dari ekosistem .NET, aplikasinya hampir secara eksklusif untuk sistem berbasis *Windows*. Jika pengguna memilih untuk bekerja dengan OS yang berbeda, Pengguna mungkin menemukan bahwa beberapa fitur C# tidak berfungsi atau tidak tersedia.

b. Ketergantungan .NET

Meskipun C# serbaguna dan dapat melayani cukup banyak proyek, kemampuan itu dilengkapi dengan peringatan: pengguna memerlukan kerangka kerja .NET agar semuanya berjalan dengan lancar.

c. Ketidakmungkinan untuk mengkodekan solusi tingkat rendah

C# adalah bahasa tingkat tinggi, yang tidak hanya berarti bahwa pendekatan sintaksis dan pengkodean lebih abstrak tetapi juga tidak mungkin menghubungkan produk C# dengan perangkat keras.

2.2 Teori Khusus

2.2.1 UML (Unified Modeling Language)

Menurut (Editorial, 2021) *UML* (*Unified Modeling Language*) adalah bahasa *visual* yang menyediakan cara bagi insinyur dan pengembang perangkat lunak untuk membangun, mendokumentasikan, dan memvisualisasikan sistem perangkat lunak. Meskipun *UML* bukan bahasa pemrograman, ia dapat memberikan representasi *visual* yang membantu pengembang perangkat lunak lebih memahami potensi hasil atau kesalahan dalam program. Ini dapat menghemat waktu dan meningkatkan kolaborasi di antara anggota tim. Manfaat utama penggunaan *UML* dalam analisa sistem yaitu:

1. Komunikasi yang ditingkatkan: *UML* memungkinkan pengembang untuk berkomunikasi tentang topik yang mencakup sistem dan persyaratannya. Dengan cara ini, ini dapat membantu para profesional berkomunikasi dengan rekan kerja dan di antara berbagai tim.

2. Sistem model yang jelas: Sistem model *UML* menggunakan representasi subjek dan kombinasi sekumpulan ide untuk membantu komunikasi *visual*.
3. Standardisasi: *UML* menyatukan teknologi dan sistem informasi dan menerapkan teknik khusus untuk mengembangkan sistem yang sukses dan standar.

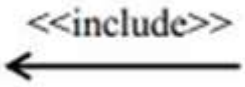
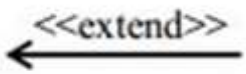
Diagram *UML* yang sering digunakan dalam analisa perangkat lunak menurut (Editorial, 2021) yaitu:

1. *Use case* Diagram

Diagram *use case* menunjukkan bagian-bagian dari sistem atau fungsionalitas dari suatu sistem dan bagaimana mereka berhubungan satu sama lain. Pengembang dapat mengetahui cara sesuatu berfungsi tanpa harus mengetahui cara atau detail dari implementasi.

Tabel 2.1 *Use Case Diagram*

Simbol	Nama	Keterangan
	Aktor	Aktor : Mewakilli Peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan *use case*

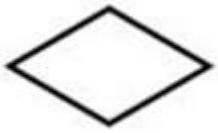
	*Use case*	*Use case* : Abstraksi dan interaksi antara sistem dan actor
	*Association*	Association: Abstraksi dari penghubung antara aktor dengan *use case*
	Generalisasi	Generalisasi : menunjukan spesialisasi aktor untuk dapat berpartisipasi dengan *use case*
	*Include*	Menunjukkan bahwa suatu *use case* seluruhnya merupakan fungsionalitas dari *use case* lainnnya
	*Extend*	Menunjukkan bahwa suatu *use case* merupakan tambahan fungsional dari *use case* lainnya jika suatu kondisi terpenuhi

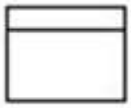
Sumber : www.visual-paradigm.com

2. *Activity Diagram*

Diagram ini menggambarkan aliran kontrol dalam suatu sistem dan mungkin berguna sebagai referensi untuk diikuti saat mengeksekusi diagram *use case*. Diagram aktivitas juga dapat menggambarkan penyebab suatu peristiwa tertentu.

Tabel 2.2 *Activity Diagram*

Simbol	Nama	Keterangan
	Status awal	Sebuah diagram aktivitas memiliki sebuah status awal
	Aktivitas	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
	Percabangan/*Decision*	Percabangan dimana ada pilihan aktivitas yang lebih dari satu.
	Penggabungan/*Join*	Penggabungan dimana yang mana lebih dari satu aktivitas lalu digabungkan jadi satu.

	Status akhir	Status akhir yang dilakukan sistem sebuah diagram aktivitas memiliki sebuah status akhir
	*Swimlane*	*Swimlane* memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

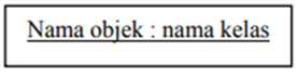
Sumber : www.visual-paradigm.com

3. Sequence Diagram

Diagram ini menunjukkan bagaimana objek berkomunikasi satu sama lain secara berurutan. Ini biasanya digunakan oleh pengembang perangkat lunak untuk menyimpan dan memahami persyaratan yang diperlukan untuk membuat sistem baru atau mendapatkan lebih banyak pengetahuan tentang sistem yang ada.

Tabel 2.3 *Sequence Diagram*

Simbol	Nama	Deskripsi
	Aktor	Lambang ini adalah aktor/pengguna digunakan untuk

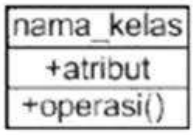
		berinteraksi dengan sistem yang dibuat dari luar sistem itu sendiri.
	Garis Hidup/ *Lifeline*	Untuk garis hidup yang ada pada objek
	Objek	Menyatakan objek yang komunikasi dengan menggunakan pesan
	Pesan tipe call	Melakukan pemanggilan pada suatu objek lain atau sendirinya
	Pesan tipe send	Melakukan Pemanggilan pada suatu objek lain atau sendiri dengan mengirim pesan.
	Pesan tipe *return*	Perancangan pemanggilan yang menghasilkan pada suatu hal tertentu

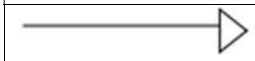
Sumber : www.visual-paradigm.com

4. Class Diagram

Class Diagram atau diagram kelas merupakan prinsip dasar dari setiap sistem perangkat lunak berorientasi objek, dan menggambarkan kelas dan rangkaian hubungan antar kelas. Kelas-kelas dalam suatu sistem atau operasi menggambarkan strukturnya, yang membantu perekayasa perangkat lunak dan pengembang mengidentifikasi hubungan antara setiap objek.

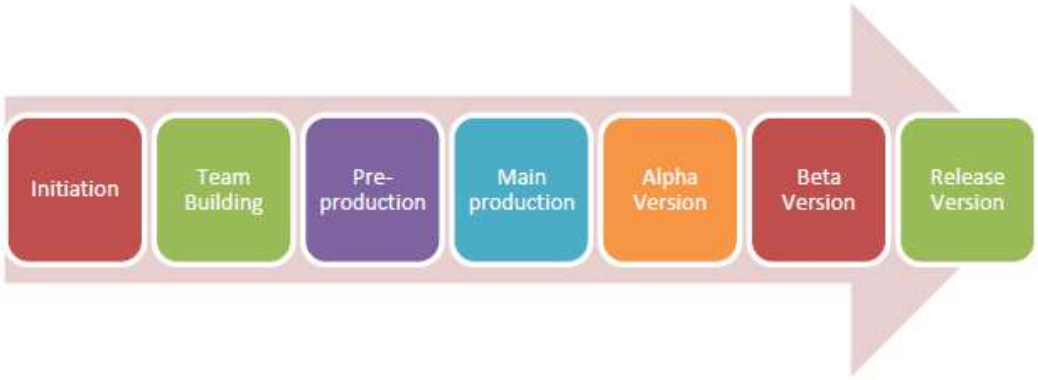
Tabel 2.4 *Class Diagram*

Simbol	Nama	Deskripsi
	*Class*/Kelas	Lambang yang memberikan informasi dari pelaku.
	Asosiasi Berarah	Interaksi yang dapat membentuk berarah pada kelas lain dan beragam
	Generalisasi	Lambang yang berbentuk terikat

	*Interface*/Antar muka	Simbol ini sama dengan kelas akan tetapi Langkah dideklarasi tidak sesuai konten.
	*Depedency*/Kebergantungan	Relasi antar kelas yang mempunyai makna umum - khusus

Sumber : www.visual-paradigm.com

2.2.2 *GDLC (Game Development Life Cycle)*



Gambar 2.3 *Game Development Life Cycle*
Sumber : personanonymous.wordpress.com

Banyak pengembang menggambarkan siklus hidup pengembangan *game* (*GDLC*) sebagai siklus hidup pengembangan perangkat lunak dengan tujuan untuk menghibur pengguna. Dibandingkan dengan siklus hidup pengembangan perangkat lunak, pengembang menghadapi banyak tantangan saat mengembangkan *game* dan karenanya mereka mengikuti pendekatan berbeda untuk mengembangkan *game* yang dikenal sebagai *GDLC*.

Menurut (Singh, 2022) model *GDLC* mengikuti beberapa langkah mendasar dari tujuh tahap yaitu sebagai berikut:

1. Inisiasi: Inisiasi adalah langkah pertama dari *game* di mana pengembang memutuskan jenis *game* apa yang akan mereka hasilkan seperti target audiens atau pemain, jenis *game*, pahlawan/protagonis *game*, tema *game*, dll. .
2. Membangun tim: Menentukan kategori *game*, pengembang membuat tim terbaik untuk mengembangkan *game* tertentu. Ini terdiri dari beberapa langkah sebagai berikut:

a. Mempekerjakan
b. Mengembangkan struktur tim
c. Mendefinisikan pekerjaan untuk anggota

3. Praproduksi: Praproduksi adalah salah satu tahapan terpenting sebelum terjun ke pengkodean *game* di mana desainer, artis, dan pemrogram membuat prototipe *game*. Proses pra-produksi mengikuti beberapa langkah:
   a. Mengembangkan cerita/ naskah: Pada tahap ini, desainer dan pendongeng membuat cerita *game* yang terdefinisi dengan baik dengan tema inti untuk audiens tertentu. Kisah ini membantu memastikan alur yang mulus di antara level. Menulis cerita membutuhkan imajinasi dan kreativitas yang nyata.
   b. Desain *gameplay*: Memimpin desainer membangun desain *game* sesuai dengan alur cerita dan membuat storyboard. Setelah desain siap, desainer membuat level permainan. Segera setelah desainnya siap, para seniman mengerjakan konsep seni *game* tersebut. Papan cerita lengkap ini memberikan detail mekanisme permainan. Yang penting, desain terdiri dari desain karakter, desain latar belakang, dan desain set dan properti.
   c. Mesin *game*: Setelah desain siap, pemrogram memutuskan mesin pengembangan *game* seluler yang tepat untuk *game* tersebut.
   d. Prototipe: Dengan garis besar *game*, konsep seni, dan level *game*, desainer membangun prototipe *game* yang menggabungkan semua faktor.

e. Dokumentasi: Setelah prototipe siap, dokumentasi untuk seni, tim teknis perlu mulai mengerjakan langkah selanjutnya.

4. Produksi: Desainer, artis, dan pemrogram menggunakan prototipe yang dibuat dalam proses praproduksi untuk mengembangkan *game*. Dalam proses ini, cetak biru desain *game* diubah menjadi pengkodean. Seniman membuat tekstur model dan animasi karakter, lingkungan. Di sisi lain, pemrogram memulai dengan pengkodean. Langkah-langkah pembuatannya adalah sebagai berikut:
    a. Pengkodean
    b. Desain tata letak
    c. Pemodelan
    d. Tekstur
    e. Animasi
    f. Evaluasi
5. Versi alfa: Dalam versi alfa, *game* dapat dimainkan, tetapi tidak lengkap. Dan itu terdiri dari langkah-langkah seperti pengujian alfa dan perbaikan bug. *Gameplay* sepenuhnya diperbaiki di sini pada langkah ini, dan langkah ini pada dasarnya digunakan untuk memeriksa kesalahan
6. Versi beta: Dalam versi beta, *game* sudah selesai dan final. Langkah-langkahnya adalah pengujian beta, perbaikan bug, dan penyeimbangan.
7. Versi rilis: *Game* siap diluncurkan dalam versi rilis. Setelah pengembangan dan pengujian, *game* ini akhirnya siap dirilis di pasar.

2.2.3 *Software* pendukung

2.2.3.1 *UNITY*

Unity adalah mesin *game* 3D/2D dan IDE lintas platform yang kuat untuk pengembang. Sebagai mesin *game*, *Unity* mampu menyediakan banyak fitur bawaan terpenting yang membuat *game* berfungsi. Itu berarti hal-hal seperti fisika, rendering 3D, dan deteksi tabrakan. Dari *sudut* pandang pengembang, ini berarti tidak perlu menemukan kembali roda. Daripada memulai proyek baru dengan membuat mesin fisika baru dari awal–menghitung setiap gerakan terakhir dari setiap material, atau cara cahaya memantul dari permukaan yang berbeda. Apa yang membuat *Unity* lebih kuat, adalah bahwa itu juga mencakup "Toko Aset" yang berkembang pesat. Ini pada dasarnya adalah tempat di mana pengembang dapat mengunggah kreasi mereka dan membuatnya tersedia untuk komunitas. Semua ini berarti bahwa pengembang *game* bebas untuk fokus pada hal-hal yang penting: merancang pengalaman yang unik dan menyenangkan, sementara hanya mengodekan fitur-fitur yang unik untuk visi tersebut.

Gambar 2.4 Logo Unity
Sumber : 1000logos.net

Tentu saja, ada mesin *game* besar lainnya yang tersedia untuk dikembangkan. Mesin *game Unity* menghadapi persaingan ketat dari orang-orang seperti Unreal Engine dan Cryengine. Meskipun perangkat lunak ini

sebelumnya dikenal sebagai "*Unity* 3D", perangkat lunak ini telah berkembang menjadi alat pengembangan 2D yang setara. Tidak hanya itu, tetapi cara penanganan grafik membuatnya sangat mudah untuk mentransfer pengalaman ke perangkat keras yang lebih rendah. Karena alasan inilah *Unity* memperkuat sebagian besar judul di *Google Play Store*. Karena *Unity* adalah lintas platform, ini berarti membuat *game* untuk *IOS*, PC, atau bahkan konsol *game* sama mudahnya. *Unity* juga menawarkan dukungan *VR* yang luar biasa bagi para pengembang yang tertarik untuk mengembangkan *Oculus Rift* atau *HTC Vive*. (Sinicki, 2021)

2.2.3.2 Visual Studio Code

Visual Studio Code adalah editor kode sumber gratis, ringan namun kuat yang berjalan di desktop dan di web dan tersedia untuk *Windows, macOS, Linux*, dan *Raspberry Pi OS*. Muncul dengan dukungan bawaan untuk JavaScript, TypeScript, dan Node.js dan memiliki ekosistem ekstensi yang kaya untuk bahasa pemrograman lain (seperti *C++, C#, Java, Python, PHP,* dan *Go*), *runtime* (seperti *.NET* dan *Unity*), lingkungan (seperti *Docker* dan *Kubernetes*), dan cloud (seperti *Amazon Web Services, Microsoft Azure,* dan *Google Cloud Platform*).

Gambar 2.5 Logo Visual Studio Code
Sumber : *Visual Studio*

Selain seluruh gagasan untuk menjadi ringan dan memulai dengan cepat, *Visual Studio Code* memiliki penyelesaian kode IntelliSense untuk variabel, metode, dan modul yang diimpor; debugging grafis; linting, pengeditan multi-kursor, petunjuk parameter, dan fitur pengeditan canggih lainnya; navigasi dan refactoring kode yang keren; dan kontrol kode sumber bawaan termasuk dukungan Git. Banyak dari ini diadaptasi dari teknologi *Visual Studio*.

Visual Studio code yang tepat dibangun menggunakan *shell Electron, Node.js, TypeScript*, dan Protokol Server Bahasa, dan diperbarui setiap bulan. Banyak ekstensi diperbarui sesering yang diperlukan. Kode dalam repositori *Visual Studio Code* adalah open source di bawah Lisensi MIT. Produk *Visual Studio Code* sendiri dikirim di bawah lisensi produk Microsoft standar, karena memiliki persentase kecil dari kustomisasi khusus Microsoft. Ini gratis meskipun memiliki lisensi komersial.

2.3 Penelitian Terdahulu

Berikut ini adalah beberapa penelitian terdahulu yang bisa digunakan sebagai referensi untuk peneliti, diantaranya adalah :

1. (Hakim, 2020, *GAME* EDUKASI PENGENALAN BAHASA KOMERING UNTUK MASYARAKAT MARTAPURA MENGGUNAKAN ALGORITMA FUZZY SUGENO, ISSN 2723-3367) Bahasa daerah Indonesia mengalami penurunan yan g serius bahkan terancam punah. Masyarakat bahasa daerah dapat dikenalkan kepada anak-anak sedini mungkin dan menggunakan bahasa tersebut dalam kehidupan sehari-hari, namun kenyataannya saat ini masyarakat sudah jarang menggunakan bahasa daerah. Tujuan dari penelitian ini adalah untuk memberikan pendidikan dan pembelajaran kepada masyarakat tentang pentingnya melestarikan bahasa Korea ini. Pada penelitian ini digunakan metode pengembangan *GDLC* (*Game Development Life Cycle*) yang pengujiannya menggunakan standar ISO 25010, pengujian yang dilakukan menyangkut kesesuaian fungsi, kegunaan dan kompatibilitas. Hasil penelitian menunjukkan bahwa metode developer *GDLC* (*Game Development Life Cycle*) dapat digunakan dalam proses pengembangan *game* yang menggabungkan unsur edukasi dan hiburan. Pengujian yang dilakukan pada *game* edukasi pengantar bahasa Korea memiliki 3 aspek diantaranya penerapan fungsional yang memiliki proporsi 100%. Dari aspek usability 82, aspek compatibility, instalasi aplikasi pada perangkat

(smartphone) yang menjalankan *OS Lollipop, Oreo, Pie* dan *Q* memenuhi aspek compatibility.

2. (Prasetyo et al., 2021, Rancang Dan Bangun *Game* Edukasi Anak-Anak Berbasis *Android* Dengan Unity Menggunakan Metode *Game* Development Life Cycle, e-ISSN: 2746-1343) Teknologi berkembang seiring dengan perkembangan zaman, terus dan terus berkembang menciptakan terobosan-terobosan baru dalam segala aspek kehidupan. Laporan *Studi* Literasi Membaca IEA menemukan bahwa keterampilan siswa sekolah dasar di Indonesia sangat rendah. Indonesia menempati urutan ke-30 dari 31 negara yang disurvei. Dengan latar belakang tersebut, peneliti dalam penelitian ini merumuskan masalah bagaimana merancang dan membangun sebuah *game* edukasi berbasis *Android* dengan *Unity* menggunakan teknologi *Game Development Life Cycle* (*GDLC*). Penelitian ini dilakukan pada bulan Agustus 2021 di Desa Ulu 9/10 Kecamatan Plaju Palembang dengan sampel 10 anak. Action Research digunakan dalam metode penelitian dan *GDLC* dalam metode pengembangan aplikasi. Hasil pengujian black box sistem *game* dapat digunakan tanpa kendala, dan hasil uji kelayakan *game* 84.6% pada kategori yang sesuai.Hasil pengembangan dengan *Unity* dan *GDLC* berhasil membuat sumber daya pendidikan berdasarkan *game Android* dengan cukup baik.
3. (Windawati & Koeswanti, 2021, Pengembangan *Game* Edukasi Berbasis *Android* untuk Meningkatkan Hasil Belajar Siswa di Sekolah Dasar, p-ISSN 2580-3735 e-ISSN 2580-1147) Media pembelajaran merupakan alat

guru untuk menyampaikan informasi dalam kegiatan pembelajaran. Namun saat ini masih banyak sekolah dasar yang melakukan kegiatan pembelajaran hanya melalui media audio *visual*, alat peraga dan benda-benda yang ada di lingkungannya, sehingga siswa mudah bosan ketika mengikuti kegiatan pembelajaran. Dengan adanya teknologi dapat dijadikan sebagai media pembelajaran yang menarik berupa *game* edukasi berbasis *android*. Tujuan penelitian ini adalah untuk mengetahui tingkat validitas berdasarkan pendapat ahli terhadap produk pengembangan *game* edukasi berbasis *android* untuk meningkatkan hasil belajar siswa kelas IV. Jenis penelitian ini adalah pengembangan atau research and development (R&D). Penelitian dilakukan dengan menggunakan model perencanaan pengembangan ASSURE yaitu. Analisis Pembelajar, Tujuan Umum, Metode Terpilih, Media dan Bahan, Penggunaan Media dan Bahan Ajar (Media dan Bahan Bermanfaat), Libatkan Siswa dalam Kegiatan Pembelajaran (Posisikan Pembelajar) dan Evaluasi. dan revisi (Evaluasi & Revisi). Hasil uji validasi ahli diperoleh sebagai berikut: (1) proporsi yang diperoleh dari uji validasi ahli materi sebesar 73%, skor yang diperoleh tinggi; (2) Persentase yang dicapai dalam uji validasi media adalah 97% dan penilaian penilaian yang dicapai sangat tinggi. Berdasarkan uji validasi ahli media dan ahli materi, dapat diartikan bahwa media pembelajaran berupa *game* pembelajaran berbasis *android* dapat dikatakan bermanfaat untuk meningkatkan hasil belajar siswa kelas 7 IV. Kata kunci: *Game* Edukasi, *Android*, Hasil Belajar

4. (Kie & Simanjuntak, 2022, PERANCANGAN *GAME* EDUKASI MENYUSUN HURUF NAMA HEWAN BERBASIS *ANDROID*, ISSN (Online) 2715-6265) Penelitian ini bertujuan untuk mengetahui hasil pengembangan *game* edukasi sebagai alat ajar untuk meningkatkan hasil belajar siswa kelas I Sekolah Dasar, serta reaksi siswa terhadap penggunaan *game* sebagai alat ajar melalui representasi nama-nama siswa SD. pendidikan. . binatang. Metode penelitian bercak adalah pengembangan atau dikenal dengan metode penelitian dan pengembangan. Klasifikasi ssdan pengujian. Tes tersebut berupa konfirmasi oleh ahli atau ahli media dan sangat informatif. Kemudian siswa menguji media tersebut. Setelah divalidasi, alat tersebut memenuhi standar interoperabilitas yang sesuai dan pengujian atau penelitian yang diperlukan untuk menentukan kompetensi industri. Penelitian dilakukan di SD Negeri 001, Batam dan diikuti oleh 38 siswa. Pada penelitian ini, hasil permainan edukasi dalam kategori ini sangat bermanfaat. Mudah digunakan untuk setiap siswa. Oleh karena itu, dapat diartikan bahwa lingkungan belajar berupa permainan urutan pembelajaran abjad nama hewan untuk kelas 1 SD merupakan alat bantu yang bermanfaat dalam pembelajaran.
5. (AlAli & Al Hosni, 2022, *Game* Edukasi di Pendidikan Dasar: Membuka Kunci Potensi, ISSN 2690-3644 (Print) ISSN 2690-3652 (Online)) Perkembangan teknologi telah membuat langkah besar dalam perkembangan kehidupan manusia di berbagai bidang. Integrasi teknologi ke dalam kelas telah memperkenalkan metode pengajaran lain yang dapat

meningkatkan dan meniru metode pengajaran tradisional. Penggunaan *game* edukatif merupakan produk sampingan dari pengintegrasian teknologi ke dalam kelas untuk meningkatkan metode pengajaran dan kinerja siswa. Oleh karena itu, tujuan dari penelitian ini adalah untuk mengevaluasi pengaruh penggunaan permainan edukasi dalam pelajaran matematika untuk siswa kelas dua di sekolah Palestina dengan menggunakan pendekatan eksperimen semu. Tiga puluh siswa laki-laki dan perempuan kelas dua dari Sekolah Terpadu Al Aqsa, Kuala Lumpur menjadi sampel sasaran. Sampel dibagi menjadi kelompok eksperimen dan kelompok kontrol. Pelajaran matematika dijelaskan melalui permainan pembelajaran yang dipilih, yaitu. 99 atau 999 ditambahkan ke grup uji. Konten yang sama diajarkan kepada kelompok kontrol dengan menggunakan metode siswa tradisional. Hasil penelitian menunjukkan bahwa pengajaran matematika menggunakan permainan edukatif secara signifikan meningkatkan kinerja siswa dalam kelompok eksperimen dibandingkan dengan teman sebayanya yang mengajar menggunakan metode tradisional. Hasil ini menunjukkan bahwa permainan edukatif dapat meningkatkan keterampilan siswa matematika kelas dua secara signifikan dibandingkan dengan metode pengajaran tradisional.

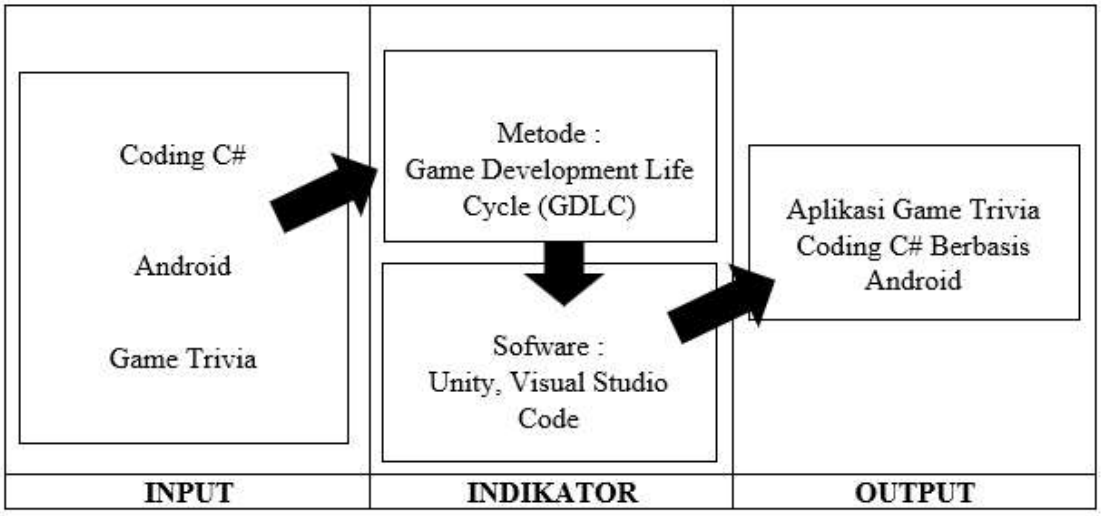
6. (Ahmed et al., 2022, Perburuan Elemen (*Game* Edukasi), ISSN (O) 2278-1021, ISSN (P) 2319-5940) Artikel ini berdasarkan pada analisis mendalam tentang sistem permainan dan bagaimana generasi muda berguna dalam menyelesaikan berbagai masalah dan memperkuat etos India. Sistem

tersebut merupakan aplikasi *game* yang menggunakan konsep tabel periodik kimia. Mempertimbangkan minat anak-anak, kami mengembangkan sistem ini berdasarkan konsep tabel periodik kimia. Setiap tingkat mewakili elemen tabel periodik dan sifat-sifat elemen yang digunakan untuk menemukan solusi untuk masalah tertentu. *Game* ini memiliki empat fase: (i) Bercerita dengan menonjolkan suasana India, (ii) bermain dan menyelesaikan level tertentu, (iii) mencoba dan melewati MCQ. Tujuan dari pekerjaan penelitian kami adalah memberikan kesempatan kepada kaum muda untuk memperoleh pengetahuan di bidang tertentu melalui permainan edukatif ini.

7. (Sungkaew et al., 2022, Rekayasa perangkat lunak pengembangan *game*: pendidikan digital permainan yang mempromosikan pemikiran algoritmik, ISSN: 2088-8708) Tujuan dari penelitian ini adalah untuk membuat sebuah *game* edukasi digital yangmempromosikan pemikiran algoritmik untuk siswa sekolah dasar. Namun,proses pengembangan *game* berbeda dari pengembangan perangkat lunak tradisionalyang tidak dapat menjamin efektivitasnya dalam hal manusia-mesinantarmuka. Pada artikel ini, kami mengusulkan perangkat lunak pengembangan *game* baruengineering (GDSE) sebagai model untuk pengembangan *game.* Model baru inibertujuan untuk melengkapi dan mengurangi kekurangan perangkat lunak tradisionalperkembangan. Prinsip-prinsip interaksi manusia-komputer sekarangdimasukkan ke dalam model baru. GDSE meliputi desain, pengembangan,inspeksi kegunaan, evaluasi pengalaman *game*, nilai

pendidikanevaluasi dan pelepasan. Itu digunakan sebagai metode penelitian untuk mengembangkan *game*yang mempromosikan pemikiran algoritmik untuk anak-anak. Hasil penelitian ini adalahbukan hanya *game* edukasi digital yang mempromosikan pemikiran algoritmik untukanak-anak tetapi juga siklus hidup pengembangan *game* baru yang menjaminkinerja *game* digital dalam hal peningkatan kegunaan, *game*pengalaman dan nilai pendidikan.

2.4 Kerangka Pemikiran



Gambar 2.6 Kerangka Pemikiran
Sumber : Data peneliti

Berdasarkan gambar diatas, penelitian diawali dengan peneliti mengunakan input yaitu *coding C#, android* dan *Game Trivia* setelah itu penulis mengembangkan *game* trivia C# menggunakan metode *GDLC* dalam Analisa dan perancangan *game* digunakan unity, dan visual studio code. Hasil dari penelitian ini adalah aplikasi *game* trivia coding c# berbasis *android*.

BAB III

METODOLOGI PENELITIAN

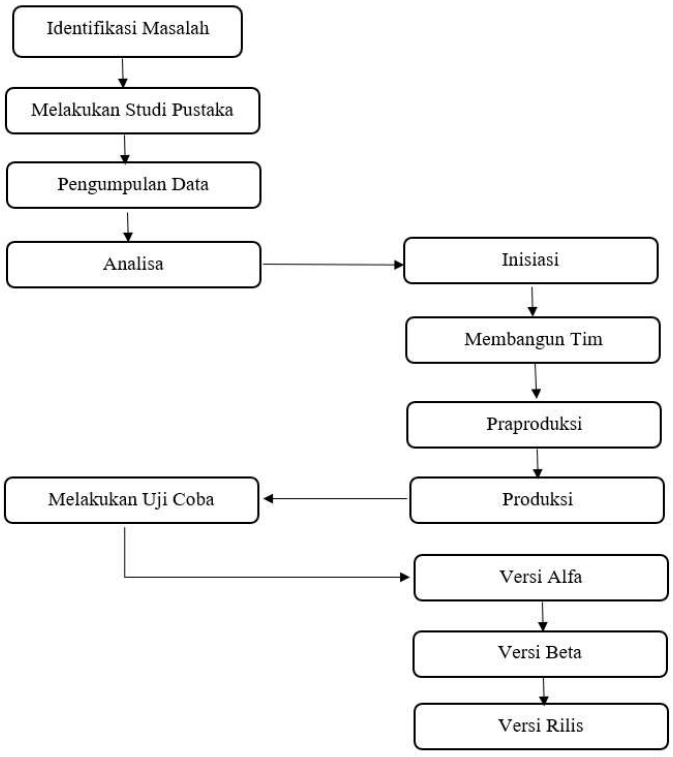
3.1 Metode Penelitian

3.2 Desain Penelitian

Desain penelitian pada penelitian ini digunakan untuk mempermudah dan menstrukturkan tahapan-tahapan perancangan *game* yang akan dibuat.

3.3 Proses Perancangan Sistem

Dalam tahap penelitian yang dilakukan peneliti, mencakup beberapa langkah-langkah penelitian dari awal sampai selesai. Masing-masing langkah dari penelitian diuraikan secara spesifik dibawah ini, antara lain :



Gambar 3.1 Tahapan Penelitian
Sumber : Data peneliti

Berdasarkan gambar 3.1 tahapan penelitian ini adalah :

1. Identifikasi Masalah

   Dalam tahapan ini, peneliti mengidentifikasi sebuah permasalahan yang ada yaitu minimnya sumber belajar bahasa pemograman C# dan minimnya aplikasi *game* yang mengajarkan tentang coding C# terutama yang berbahasa Indonesia.

2. Melakukan Studi Pustaka

   Dalam tahapan ini, peneliti melakukan penelitian tentang studi pustaka, untuk mendapatkan teori-teori pendukung yang berkaitan dengan materi C#, metode, dan teori pendukung lainnya yang digunakan sebagai acuan dalam melaksanakan penelitian.

3. Pengumpulan Data

   Dilakukan pengamatan terhadap *junior programmer* PT Inotek Mitra Indonesia tentang bagaimana mereka menggunakan C# dalam menyelesaikan pekerjaan sehari – hari.

4. Analisa

   Setelah berhasil melakukan pengumpulan data, peneliti harus melakukan analisis terhadap data yang didapatkan guna merancang *game* edukasi *trivia* C# dengan metode *GDLC* berbasis *android,* alat bantu analisa *game* digunakan *UML*.

5. Inisiasi

   Dilangkah ini peneliti memutuskan jenis *game* apa yang akan diciptakan seperti materi – materi yang akan digunakan. Adapun langkah-langkah inisiasi yang peneliti lakukan adalah :

a. Membangun Tim

Ditahapan ini dikarenakan peneliti mandiri maka tidak dibutuhkan untuk mengembangkan struktur tim dan pembagian tugas.

b. Praproduksi

Ditahapan ini peneliti memutuskan *gameplay* permainan, sistem *level*, sistem skor, dan mesin *game* yang akan digunakan. Ditahapan ini juga peneliti membangun prototipe.

c. Produksi

Pada tahap ini peneliti menggunakan prototipe yang dibuat pada saat praproduksi dan mengembangkan *game.* Disni peneliti mengubah rancangan menjadi pengkodean menggunakan aplikasi *unity* dan *visual studio code* dan akhirnya menghasilkan *game* yang dapat dimainkan tetapi belum lengkap sepenuhnya.

6. Melakukan Uji Coba

Ditahapan ini peneliti melakukan uji coba terhadap *game* yang dibuat pada saat produksi

a. Versi *Alfa*

Disini peneliti melakukan pengujian *alfa* dan memperbaiki *bug* yang ditemukan pada *game* versi *alfa*

b. Versi *Beta*

Disini peneliti melakukan pengujian *beta* terhadap *game* yang sudah selesai dan final. Disini peneliti memperbaiki *bug* yang ditemukan pada *game* versi *beta*

c. Versi Rilis

Ini merupakan tahap pengujian terakhir, dimana *game* hasil pengujian pada versi *beta* disempurnakan. Pengujian pada tahap ini menggunakan metode *black box.*

3.4 Peralatan Yang Digunakan

Dari rancangan yang akan dibangun, peneliti membedakan beberapa kategori dalam peralatan yang digunakan, dimana kriterianya adalah sebagai berikut :

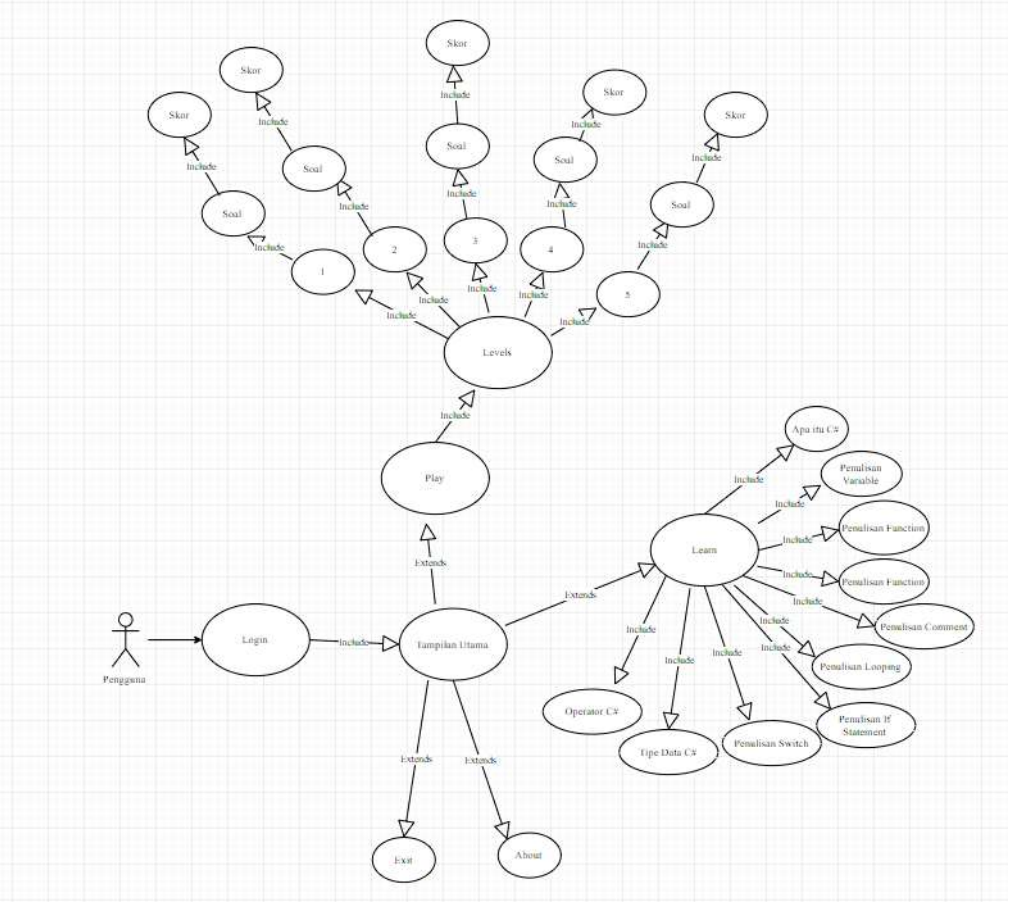
Tabel 3.1 Data Peneliti

Jenis Alat & Bahan	Alat & Bahan
Hardware	Laptop *Asus*
	Handphone Redmi Note 8 Pro
Software	*Unity*
	Visual Studio Code

Sumber : Data peneliti

3.5 *UML* (Desain Unified Modeling Language)

1. Use Case Diagram



Gambar 3.2 *Use Case Diagram*
Sumber : Data peneliti

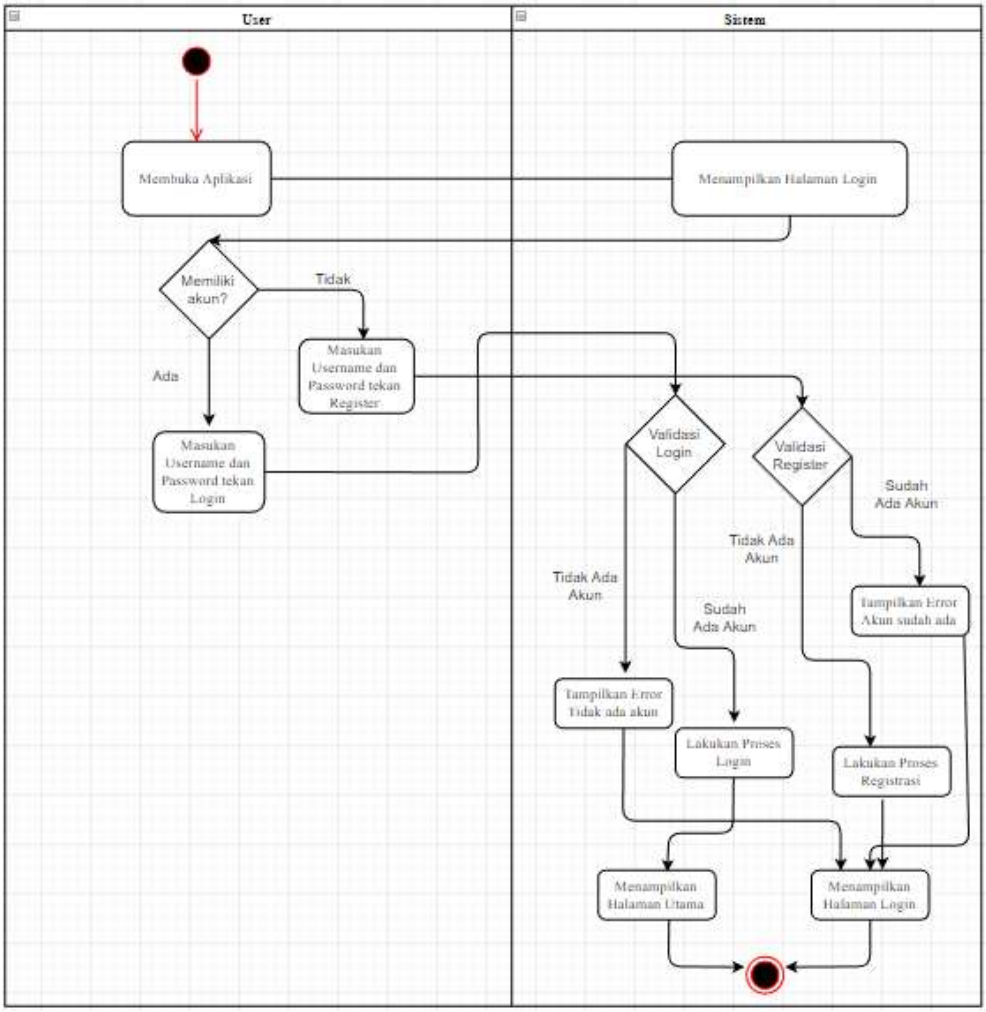
Aktor pada *game* ini adalah orang-orang yang sedang belajar atau akan belajar pemograman C#. Setelah aplikasi *game* dijalankan pengguna akan di melihat tampilan *login* yang memiliki dua *input text field* yaitu *username* dan *password* setelah itu *user* dapat melakukan *login* atau *register*. Setelah itu *user* jika *login* sukses maka *user* akan diarahkan ke tampilan utama yang memiliki empat menu yaitu *play*, *learn*, *about* dan *exit*. Pada menu *play* pengguna dapat memilih level setelah itu permainan akan dimulai, menu *learn* pengguna dapat

belajar materi C#, sedangkan pada menu *about* pengguna dapat melihat informasi tentang aplikasi *game* trivia c#, dan pada menu *exit* pengguna dapat keluar dari aplikasi *game*.

2. Activity Diagram

Activity diagram digunakan untuk menjelaskan aktivitas dari sebuah sistem menu yang ada pada perangkat lunak. Activity diagram pada penelitian ini sebagai berikut:

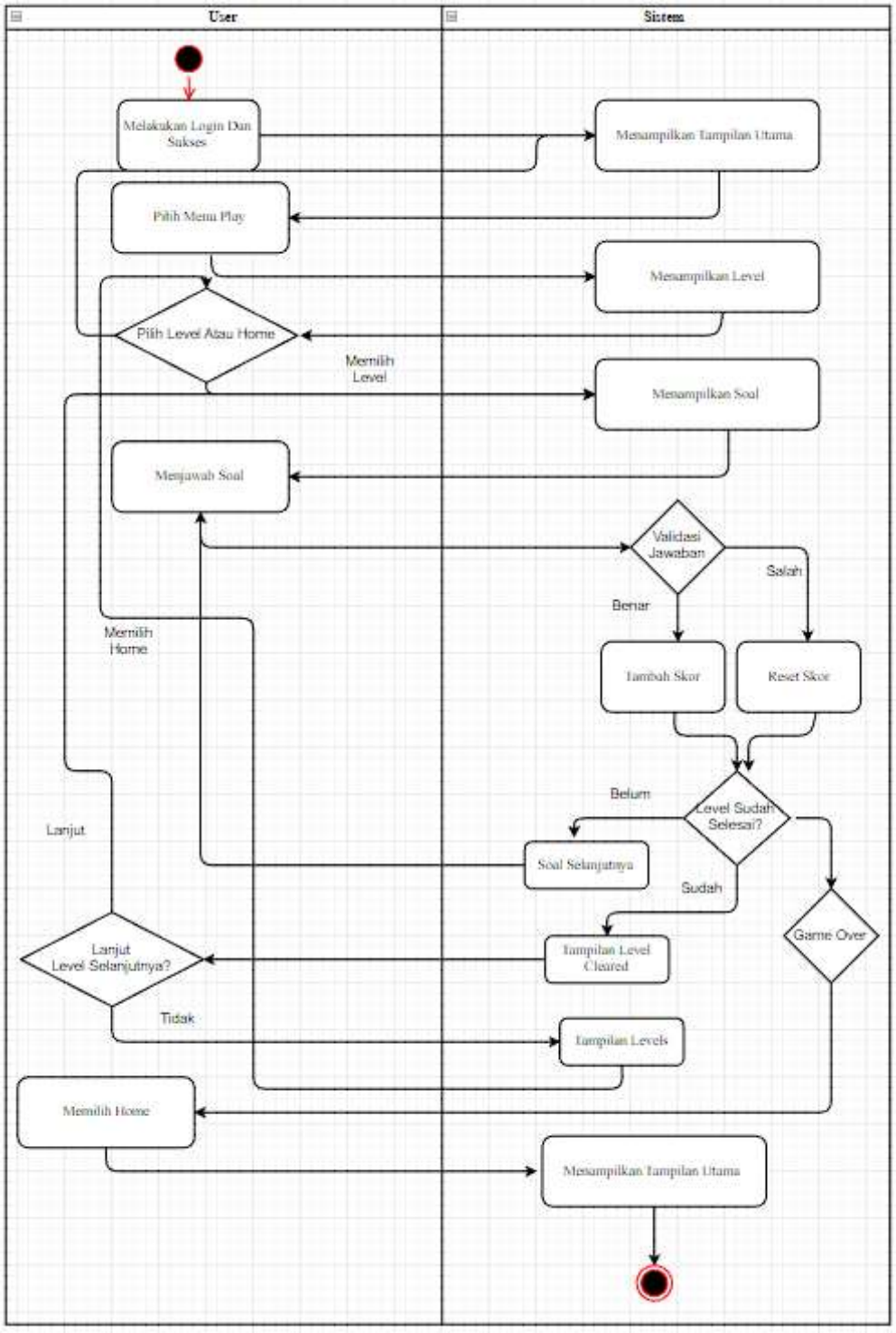
a. Activity Diagram Login



Gambar 3.3 *Activity Diagram Login*
Sumber : Data peneliti

Berdasarkan gambar diatas, *user* menjalankan aplikasi dan kemudian aplikasi akan menampilkan halaman *login*. Jika *user* memiliki akun maka *user* memasukan *username* dan *password* serta menekan tombol *login* dan sistem akan melakukan validasi. Jika hasil validasi adalah akun sudah ada maka sistem akan menampilkan halaman utama. Jika hasil validasi adalah akun tidak ada maka sistem akan menampikan *error* dan kembali ke halaman *login*. Jika *user* tidak memiliki akun maka *user* memasukan *username* dan *password* serta menekan tombol *register* dan sistem akan melakukan validasi. Jika hasil validasi adalah akun belum ada maka sistem akan melakukan registrasi akun dan menampilkan halaman *login*. Jika *user* sudah ada maka sistem akan menampilkan *error* dan kembali ke halaman *login*.

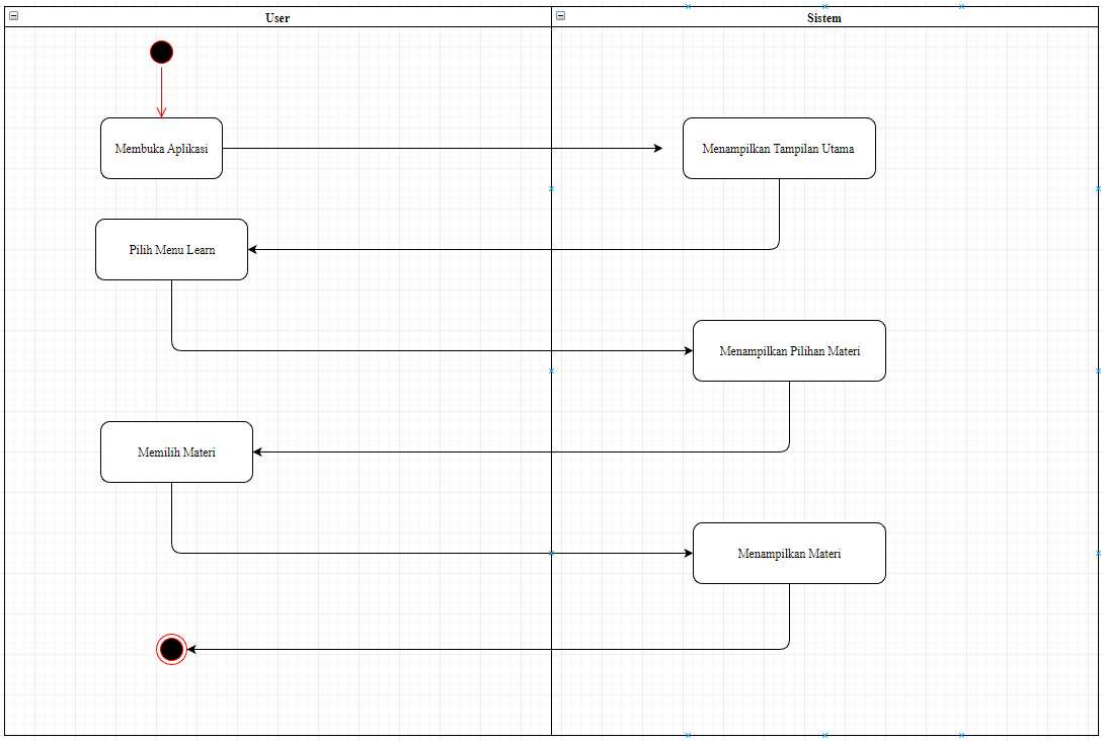
b. Activity Diagram Menu Play



Gambar 3.4 *Activity Diagram Menu Play*
Sumber : Data peneliti

Berdasarkan gambar diatas, setelah *user* sukses melakukan login, sistem akan menampilkan halaman utama. Setelah itu *user* memilih menu *play* dan sistem menampilkan pilihan levels. *User* memilih salah satu *level* dan sistem menampilkan soal setelah itu *user* menjawab soal dan sistem melakukan validasi jawaban jika jawaban benar maka skor akan bertambah dan sistem akan menampilkan soal selanjutnya jika jawaban salah maka skor akan *reset* dan sistem akan menampilkan soal pertama *level* tersebut. Jika *level* sudah diselesaikan maka sistem akan menampilkan halaman *level cleared*. Setelah itu jika *user* memilih untuk lanjut maka sistem akan menampilkan soal dan melakukan perulangan yang sama seperti diatas. Jika semua *level* sudah berhasil diselesaikan maka sistem akan menampilkan tampilan *gameover*.

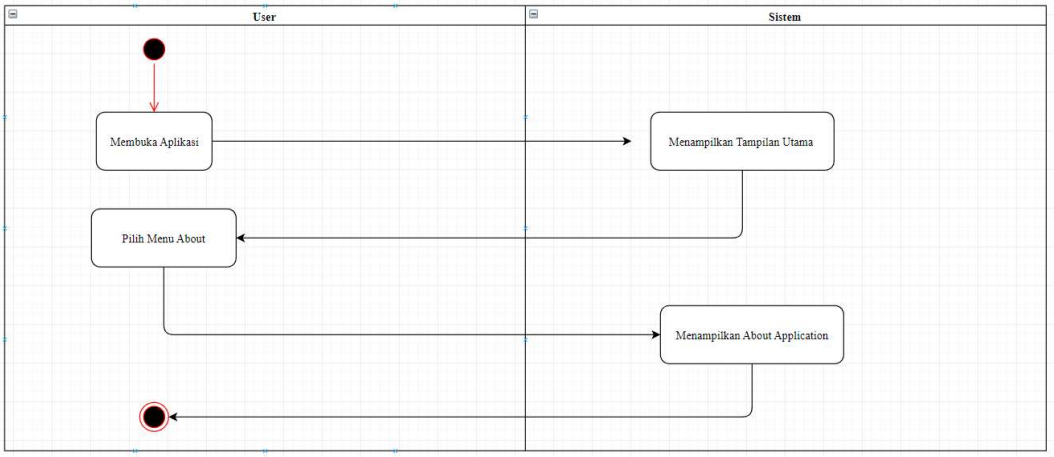
c. Activity Diagram Menu Learn



Gambar 3.5 *Activity Diagram Menu Learn*
Sumber : Data peneliti

Berdasarkan gambar diatas, *user* menjalankan aplikasi dan kemudian aplikasi akan menampilkan halaman utama. Setelah itu *user* sudah berada di dalam *menu* utama, user mengklik *menu learn* dan aplikasi akan menampilkan halaman *learn* yang dimana *user* akan memilih materi tentang pemograman *C#.* Setelah itu *user* memilih salah satu materi di halaman *learn* dan aplikasi akan menampilkan materi yang dipilih *user*. Setelah *user* menyelesaikan materi tersebut *user* dapat menekan tombol *back* dan aplikasi akan kembali ke halaman *learn* setelah itu *user* dapat memilih materi selanjutnya atau kembali ke halaman utama.

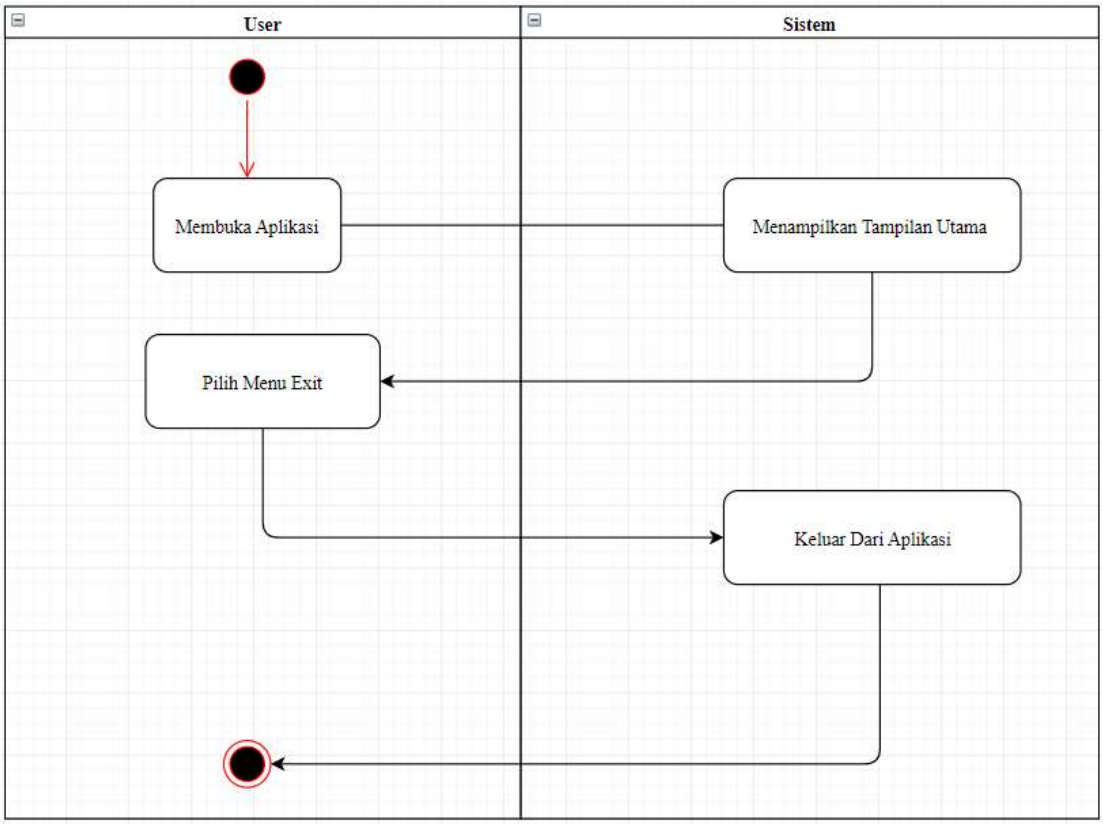
d. Activity Diagram Menu About



Gambar 3.6 *Activity Diagram Menu About*
Sumber : Data peneliti

Berdasarkan gambar diatas, *user* menjalankan aplikasi dan kemudian aplikasi akan menampilkan halaman utama. Setelah itu *user* sudah berada di dalam *menu* utama, user mengklik *menu about* dan aplikasi akan menampilkan halaman *about* yang dimana *user* dapat melihat sedikit informasi tentang aplikasi seperti versi dan nama alias pengembang. Setelah itu *user* untuk kembali ke halaman *user* dapat menekan tombol *back* dan aplikasi akan kembali ke halaman utama.

e. Activity Diagram Menu Exit



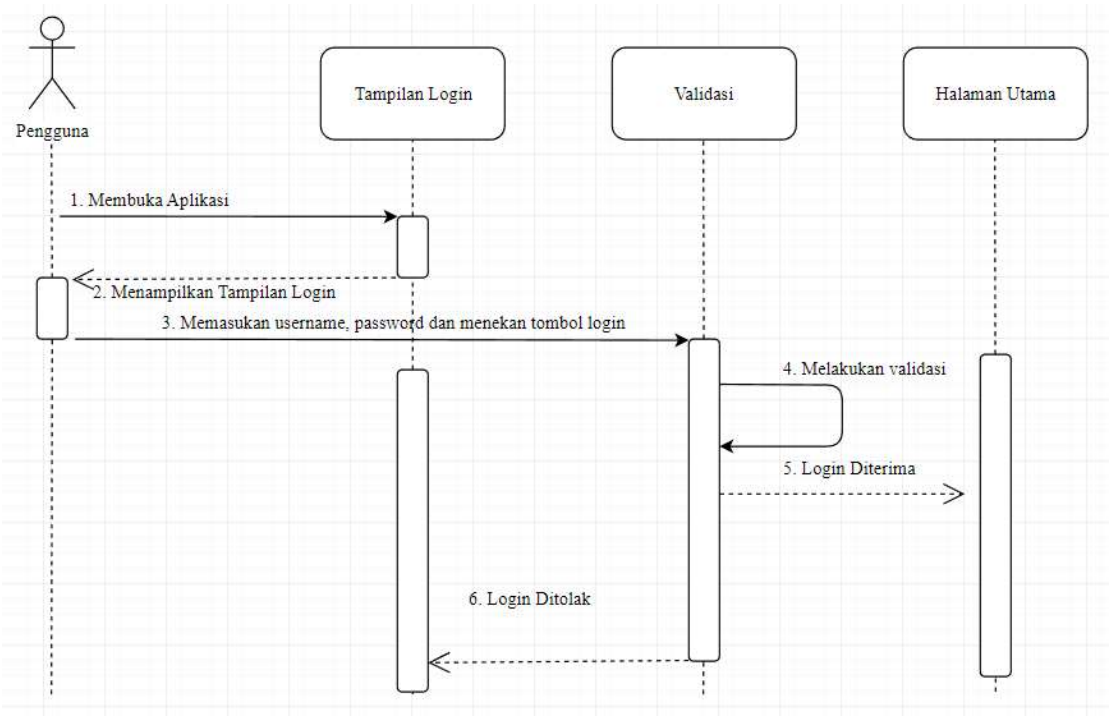
Gamber 3.7 *Activity Diagram Menu Exit*
Sumber : Data peneliti

Berdasarkan gambar diatas, *user* menjalankan aplikasi dan kemudian aplikasi akan menampilkan halaman utama. Setelah itu *user* sudah berada di dalam *menu* utama, user mengklik *menu exit* dan aplikasi akan keluar.

3. Sequence Diagram

Seqence Diagram berguna untuk menjelaskan waktu hidup objek dan pesan yang dikirim maupun yang akan diterima.

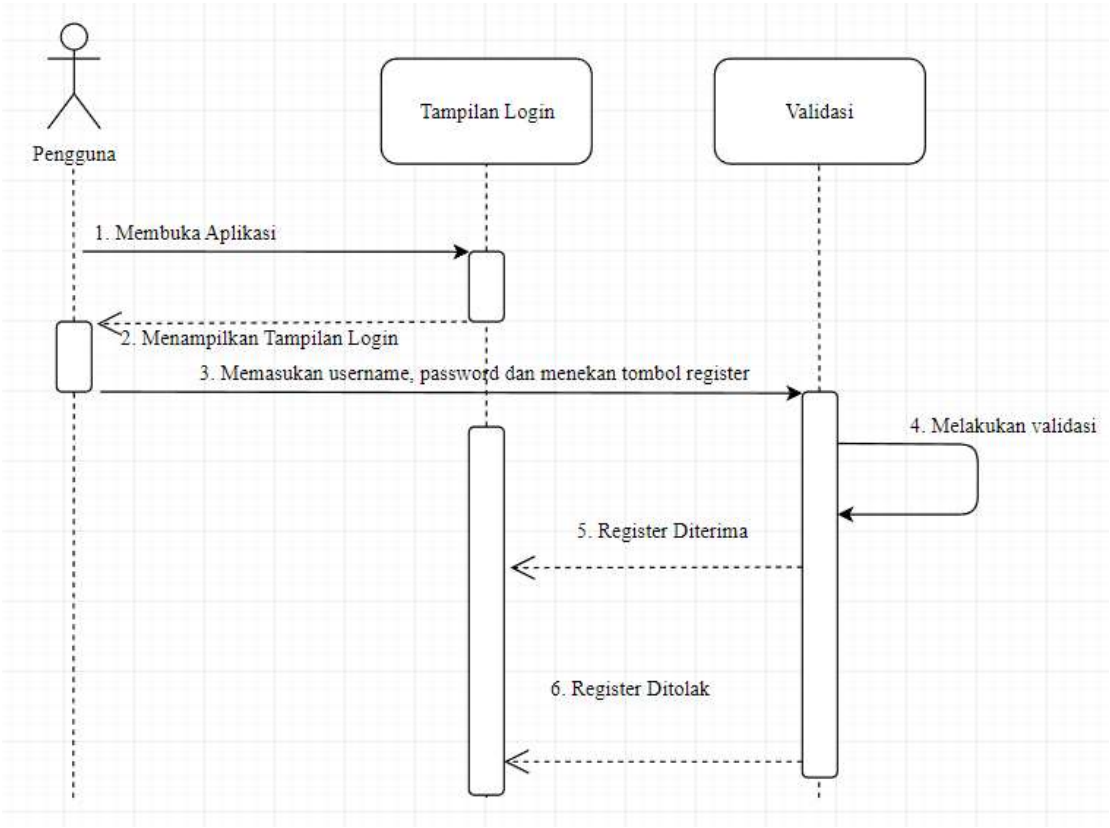
a. Sequence Diagram Login



Gambar 3.8 *Sequence Diagram Login*
Sumber : Data peneliti

Berdasarkan gambar diatas, setelah *user* menjalankan aplikasi sistem akan menampilkan halaman *login*, setelah itu *user* memasukan *username* dan *password* serta menekan tombol *login*, setelah itu sistem melakukan validasi apakah *username* dan *password* tersebut benar. Jika benar maka sistem akan menampilkan halaman utama. Jika salah maka sistem akan menampilkan kembali halaman *login*

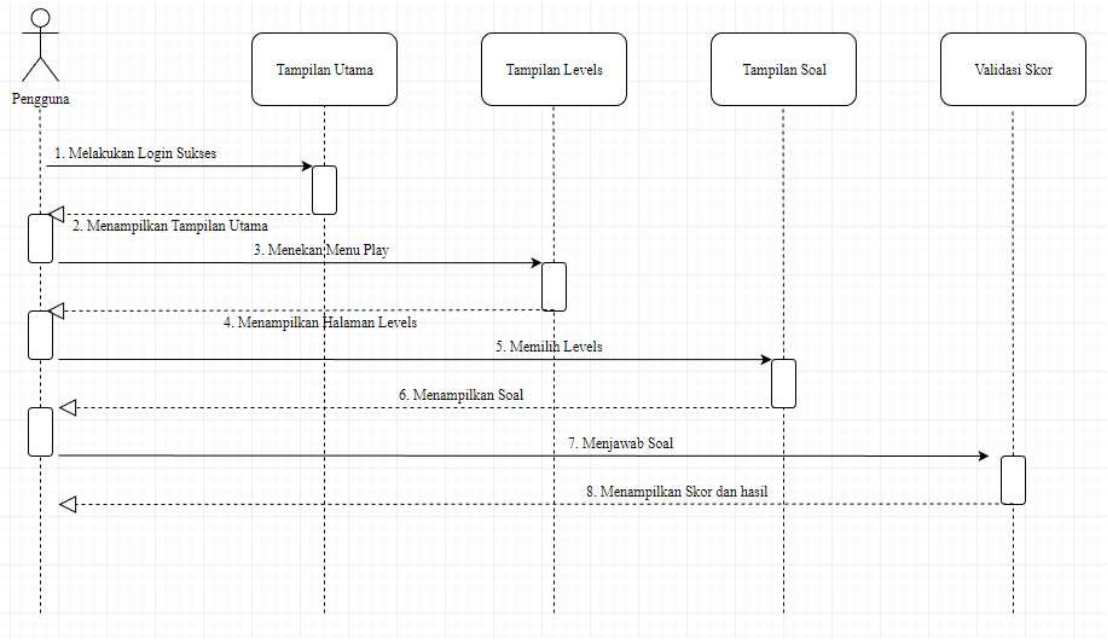
b. Sequence Diagram Register



Gambar 3.9 *Sequence Diagram Register*
Sumber : Data peneliti

Berdasarkan gambar diatas, setelah *user* menjalankan aplikasi sistem akan menampilkan halaman *login*, setelah itu *user* memasukan *username* dan *password* serta menekan tombol *register*, setelah itu sistem melakukan validasi apakah *username* dan *password* tersebut sudah terdapat di sistem. Jika di sistem sudah ada maka sistem akan menampilkan halaman *login* dan pesan *error*. Jika di sistem belum ada maka sistem akan menampilkan kembali halaman *login* dan pesan *success*.

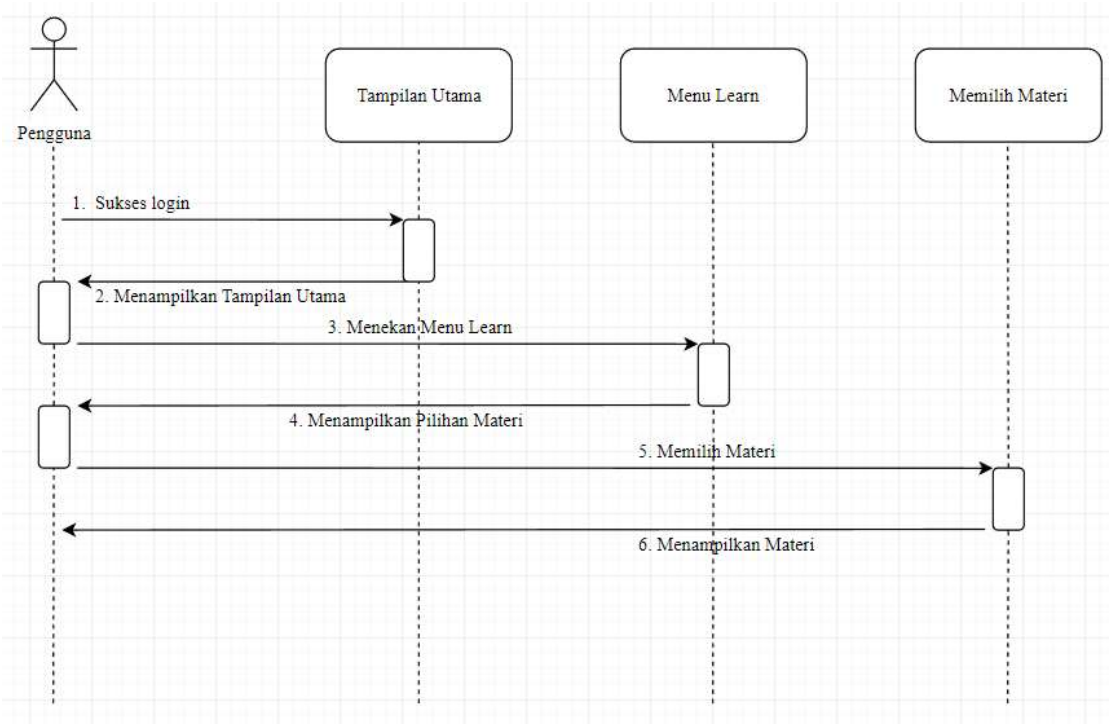
c. Sequence Diagram Play



Gambar 3.10 *Sequence Diagram Play*
Sumber : Data peneliti

Dapat dilihat dari gambar diatas, setelah *user* melakukan *login* dengan sukses maka sistem menampilkan halaman utama. Setelah itu *user* memilih menu *play* dan sistem menampilkan halaman *levels* yang berisi lima *level* dan tiap *level* memiliki lima butir soal objektif. Selanjutnya *user* menjawab soal dan aplikasi melakukan pengecekan apakah jawaban yang dipilih *user* benar, jika benar maka skor akan *diupdate* dan soal selanjutnya akan ditampilkan, jika salah maka skor dan soal akan *direset*.

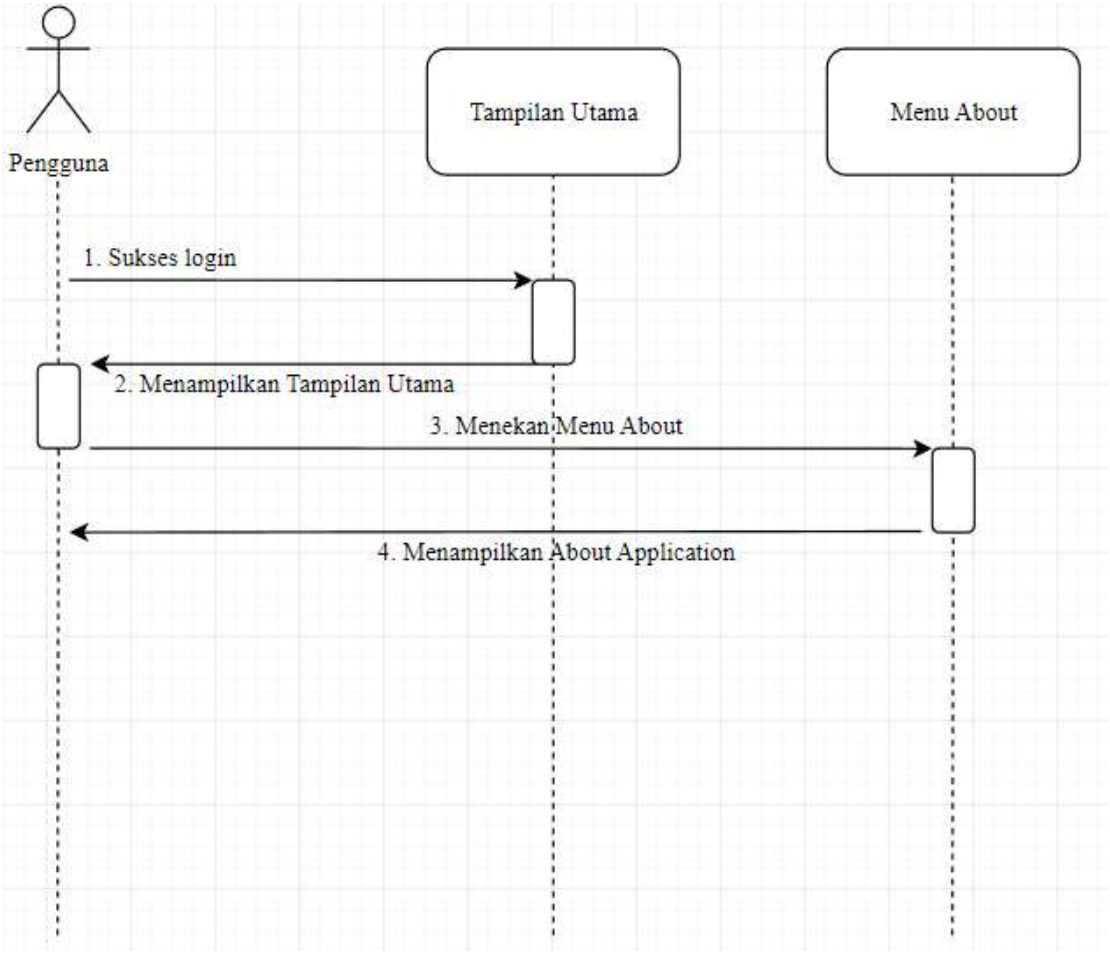
d. Sequence Diagram Learn



Gambar 3.11 *Sequence Diagram Learn*
Sumber : Data peneliti

Berdasarkan gambar diatas, *user* menjalankan aplikasi dan kemudian aplikasi akan menampilkan halaman utama. Selanjutnya *user* menekan *menu learn*, aplikasi menampilkan halaman *learn* yang berisi materi yang berkaitan dengan pemograman *C#.* Selanjutnya *user* memilih materi yang diminati dan aplikasi akan menampilkan materi tersebut.

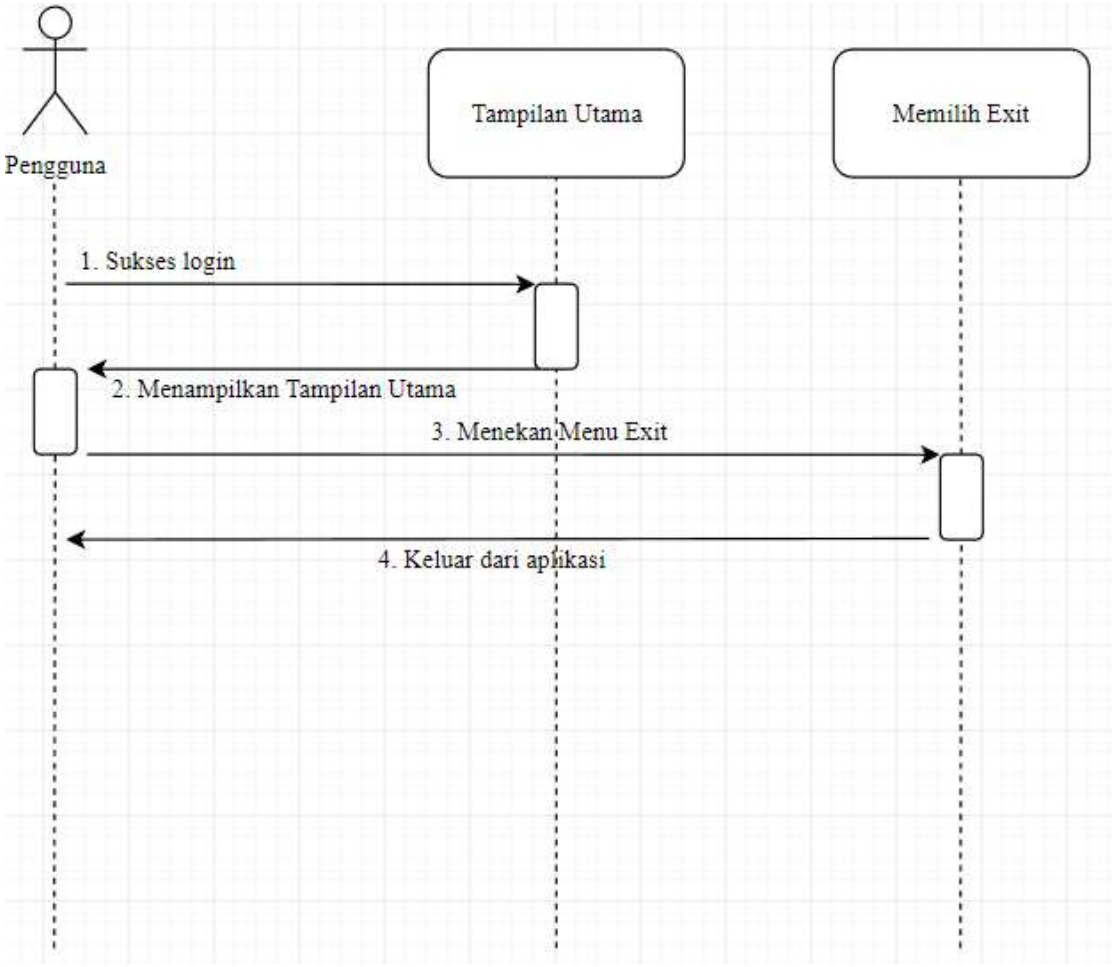
e. Sequence Diagram About



Gamber 3.12 *Sequence Diagram About*
Sumber : Data peneliti

Berdasarkan gambar diatas, *user* menjalankan aplikasi dan kemudian aplikasi akan menampilkan halaman utama. Selanjutnya *user* menekan *menu about*, aplikasi menampilkan halaman *about* yang berisi informasi yang berkaitan dengan aplikasi, didalam menu about terdapat versi dan nama pengembang. Selanjutnya *user* dapat menekan tombol *back* untuk kembali ke halaman utama

f. Sequence Diagram Exit

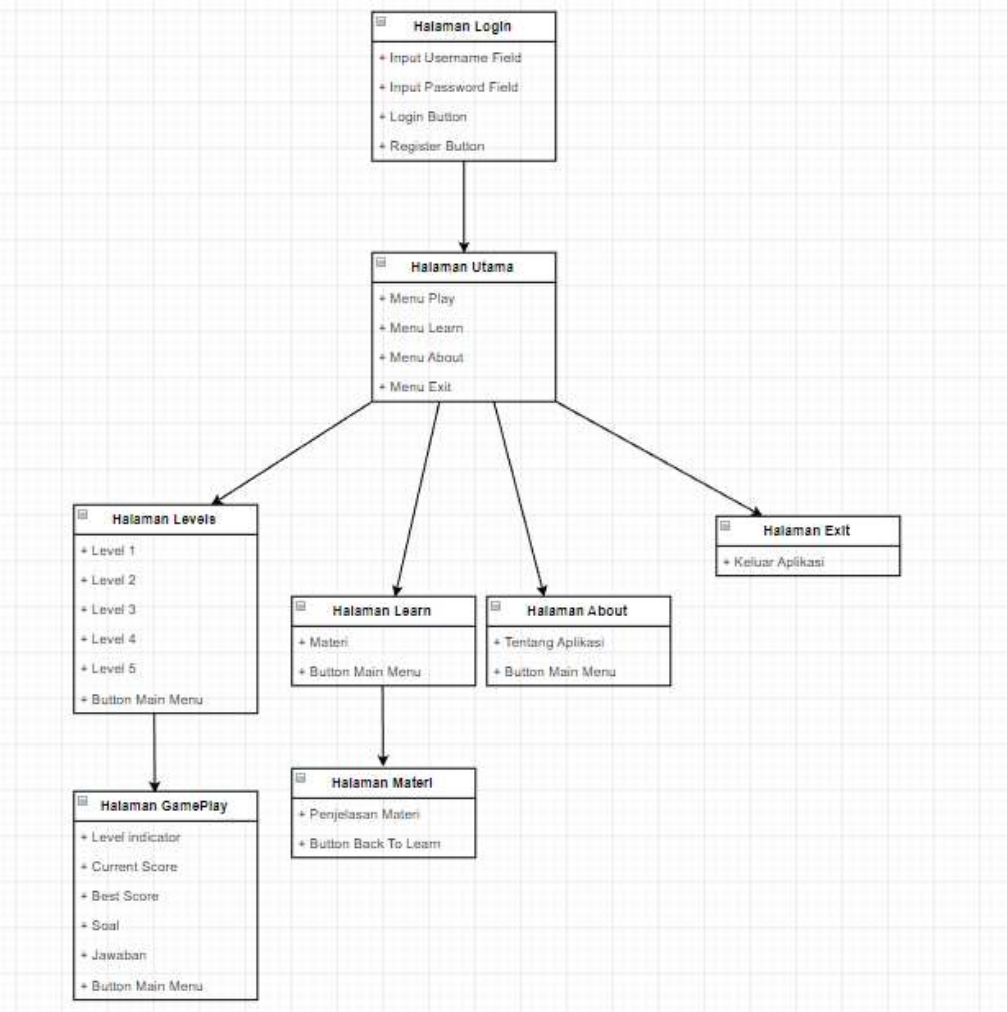


Gambar 3.13 *Sequence Diagram Exit*
Sumber : Data peneliti

Berdasarkan gambar diatas, *user* menjalankan aplikasi dan kemudian aplikasi akan menampilkan halaman utama. Selanjutnya *user* menekan *menu exit*, setelah itu aplikasi akan keluar.

4. *Class Diagram*

Class Diagram digunakan untuk menggambarkan kelas yang berorientasi objek dalam sistem yang saling berhubungan satu sama lain. Diagram *class* pada penelitian ini sebagai berikut:



Gamber 3.14 *Class Diagram*
Sumber : Data peneliti

3.6 Perancangan Tampilan

Perancangan tampilan ini digunakan untuk menjelaskan gambaran atau desain rancangan aplikasi *game* edukasi *coding* c# berbasis *android* yang akan dibuat.

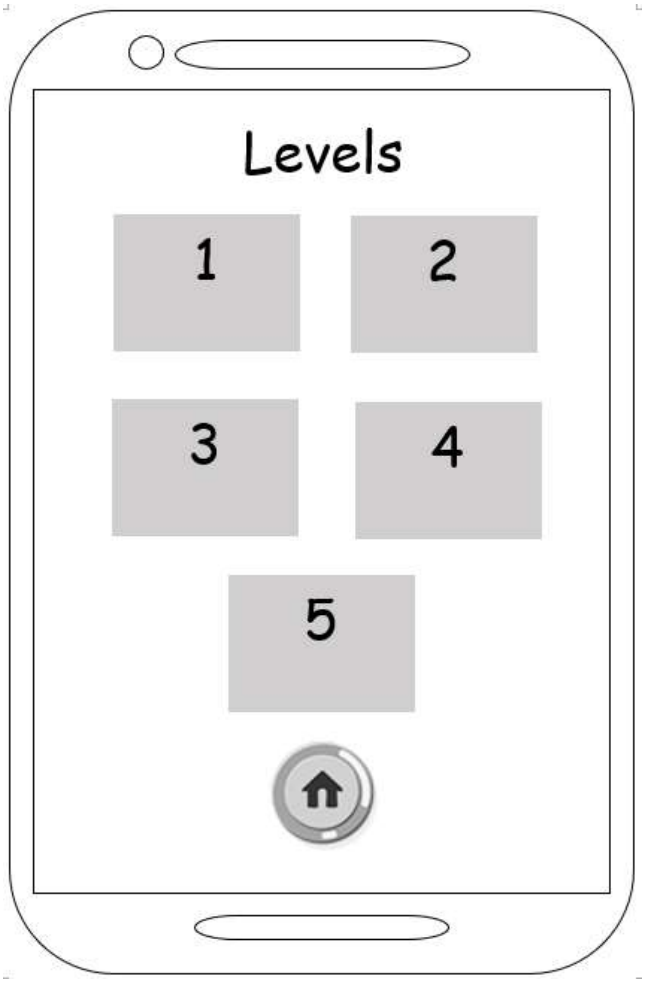
1. Rancangan Tampilan Login

Gambar 3.15 Rancangan Tampilan Login
Sumber : Data peneliti

2. Rancangan Tampilan Utama

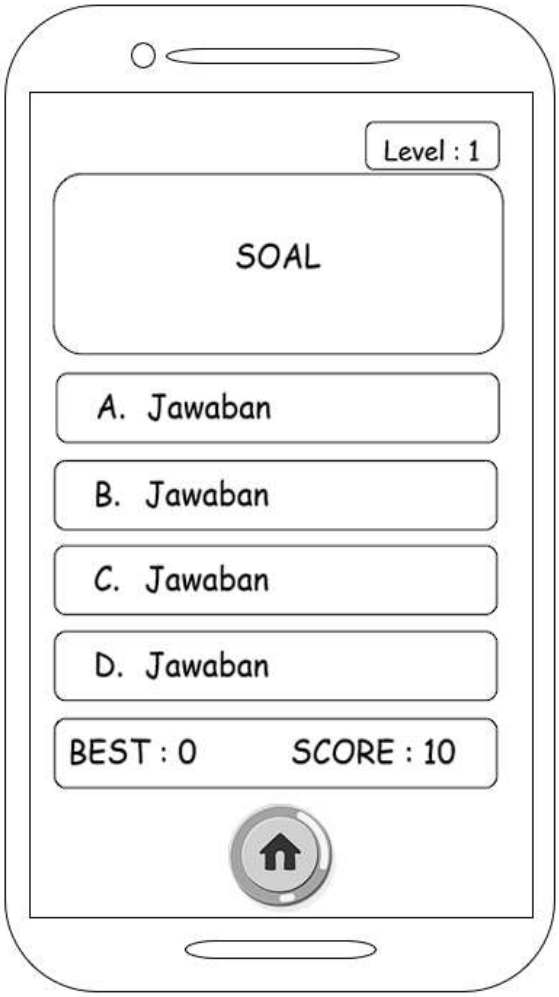
Gambar 3.16 Rancangan Tampilan Utama
Sumber : Data peneliti

3. Rancangan Tampilan *Levels*



Gambar 3.17 Rancangan Tampilan Levels
Sumber : Data peneliti

4. Rancangan Tampilan *GamePlay*

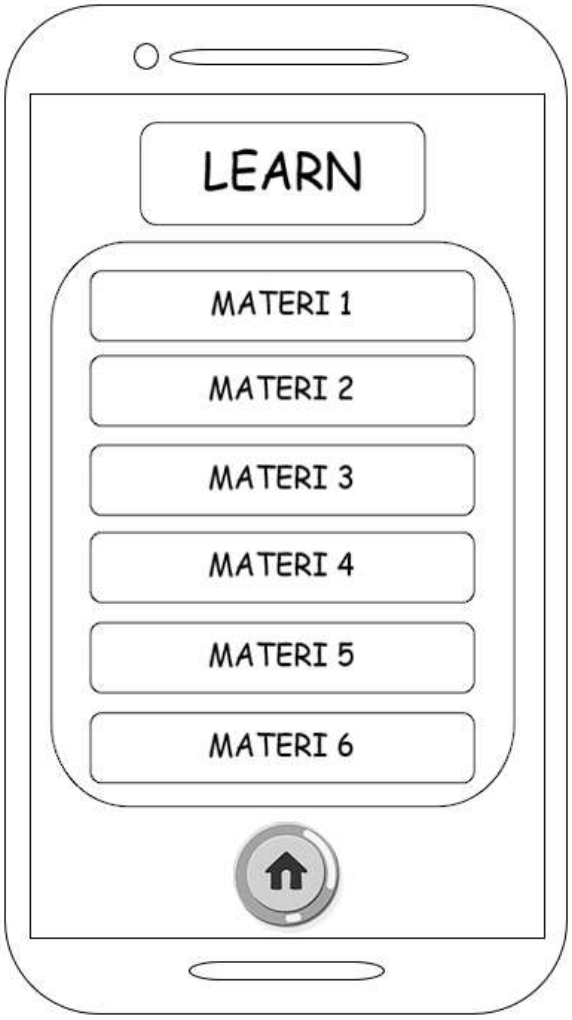


Gambar 3.18 Rancangan Tampilan *GamePlay*
Sumber : Data peneliti

5. Rancangan Tampilan *Level Cleared*

Gambar 3.19 Rancangan Tampilan *Learn*
Sumber : Data peneliti

6. Rancangan Tampilan *Learn*



Gambar 3.20 Rancangan Tampilan *Learn*
Sumber : Data peneliti

7. Rancangan Tampilan Jawaban *About*

Gambar 3.21 Rancangan Tampilan *About*
Sumber : Data peneliti

8. Rancangan Tampilan *Gameover*

Gambar 3.22 Rancangan Tampilan *Gameover*
Sumber : Data peneliti

3.7 Jadwal Penelitian

Penelitian ini di laksanakan oleh peneliti selama 5 bulan. Berikut adalah tabel jadwal penelitian yang dilakukan oleh peneliti:

Tabel 3.2 Jadwal Penelitan

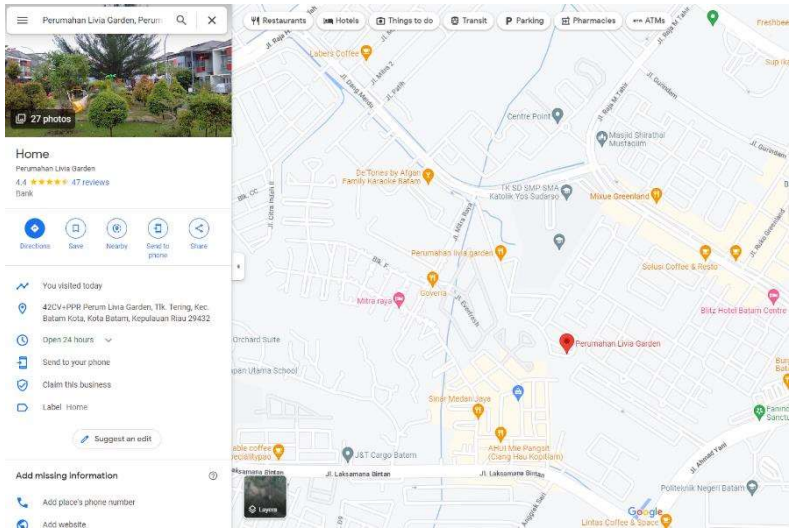
Kegiatan	Tahun 2022																Tahun 2023											
	September				Oktober				November				Desember				Januari				Februari				Maret			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Pengajuan Judul																												
Penyusunan Bab I																												
Penyusunan Bab II																												
Penyusunan Bab III																												
Penyusunan Bab IV																												
Penyusunan Bab V																												
Pengumpulan Skripsi																												

Sumber : Data peneliti

3.8 Metode Pengujian Sistem

Aplikasi *game* ini menggunakan metode *blackbox* untuk melakukan pengujian terhadap tampilan dan fungsi nya. Agar dapat mengetahui apakah program sudah berfungsi sesuai dengan yang diinginkan.

3.9 Lokasi Penelitian



Gambar 3.23 Lokasi Penelitian
Sumber : Data Peneliti