FIRST- PERSON SHOOTER GAME:

GALACTIC MACHINES

PROJECT REPORT

Submitted By

VISMAYA VIJAY

(Reg. No: MCT20MCA-2038)

To

APJ Abdul Kalam Technological University

In partial fulfilment of the requirements for the award of Degree in

MASTER OF COMPUTER APPLICATIONS

DEPARTMENT OF COMPUTER APPLICATIONS

MOHANDAS COLLEGE OF ENGINEERING & TECHNOLOGY

Anad, Nedumangad

Thiruvananthapuram

695544

2022

MASTER OF COMPUTER APPLICATIONS

MOHANDAS COLLEGE OF ENGINEERING AND TECHNOLOGY

ANAD, NEDUMANGAD, TRIVANDRUM-695544

CERTIFICATE

This is to certify that the main project report entitled **"FIRST PERSON SHOOTER GAME: GALACTIC MACHINES"** submitted by **VISMAYA VIJAY** (Registration number: **MCT20MCA- 2038**) to the **APJ Abdul Kalam Technological University**, in partial fulfilment of the requirements for the award of the Degree of **Master of Computer Applications,** is a bona fide record of the project work carried out by her under my/our guidance and supervision. This report, in any form, has not been submitted to any other University or Institute for any purpose.

Internal Supervisor(s) **Project Coordinator**

Head of the Department **External Examiner**

DECLARATION

I undersigned hereby declare that the main project report for the First-Person Shooter game "GALACTIC MACHINES" submitted for partial fulfilment of the requirements for the award of degree of Master of Computer Applications from APJ Abdul Kalam Technological University, Kerala, is a bona fide work done by me under the supervision of Professor Subha Ramachandran. This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources.

I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the Institute and or the University and can also evoke penal action from the sources which have thus not been properly cited, or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other university.

Place: Trivandrum

Date:

Submitted by

VISMAYA VIJAY

ACKNOWLEDGEMENT

I am overwhelmed in all humbleness and gratefulness to acknowledge in depth to all those who have helped me to put these ideas, well above the level of simplicity into something concrete. I would like to express my special thanks and gratitude to our principal **Dr. S. Sheela** and our Director **Dr. Ashalata Thampuran** for providing all necessary facilities.

I express my sincere thanks and deep sense of gratitude to **Prof. Sreeja K**, Head of the Department, Department of Computer Applications and my guide **Prof. Subha Ramachandran**, for all the kind words of inspiration and for bestowing valuable guidance and suggestions throughout the entire work.

I thank her for giving me the golden opportunity to do this project **"GALACTIC MACHINES",** which I am deeply interested in and invested in, which drove me to do a lot of research that helped me become familiar with many novel things.

Last but certainly not the least, I would also like to thank all the staff of the Department of Computer Applications for their help and cooperation.

Any attempt at any level cannot be satisfactorily completed without the support and guidance of my parents and friends. I would like to thank them too, them, who in spite of their busy schedules, gave me diverse and versatile ideas to make this project unique in its own right.

BY

VISMAYA VIJAY

TABLE OF CONTENTS

ABSTRACT

Game designing is a really interactive and creative part of modern IT culture and this has motivated various developers to create interesting games. Thus, the main aim is to develop an entertaining and fun 3D game based on childhood favourites, that is a First-Person Shooter game in URP (Universal Render Pipeline). The platform of the game is developed using Unity 3D game engine which is a cross-platform game engine developed by Unity Technologies.

The objective is to create a single player, multi-level game that consists of various props and weapons with enemies thirsty for blood and power boosts to give the user a more addictive experience. It is a space-themed game with androids and robots acting as enemies. The player has to complete two objectives to win the game and advance to the next level:-

(1)Defeat all the enemies on that particular level

(2)Reach a certain illuminated goal point.

Its main coding is in C sharp, hence full game development will cover implementation of real time graphics physics engine network support as well as sound effects.Other technologies like SNAPS and Progrids, Nav-mesh are also used in its development.

Thus, a game that is extremely beguiling, engaging and amusing along with great graphics which makes it visually bewitching is created.

INTRODUCTION

1.1 About the project

Video game development is a diverse field. It is a hybrid of game production and game design and has requisite skills from both fields forming the core of a video game developer's knowledge. A video game developer is usually a big-picture position in the creation of a video game, guiding the project through multiple phases. A video game developer is a cross between a producer and a programmer, they are a coordinating administrator with an artistic vision who also possesses the technical skill to oversee and contribute to software engineering, image rendering, editing and other aspects of game design.

The first video games were non-commercial, and were developed in the 1960s. They required mainframe computers to run and were not available to the general public. Commercial game development began in the 1970s with the advent of first-generation video game consoles and early home computers like the Apple I. Approaching the 21st century, ever-increasing computer processing power and heightened consumer expectations made it difficult for a single person to produce a mainstream console or PC game. The average cost of producing a triple-A video game slowly rose from US$1–4 million in 2000 to over $5 million in 2006, then to over $20 million by 2010.

Mainstream PC and console games are generally developed in phases. First, in pre-production, pitches, prototypes, and game design documents are written. If the idea is approved and the developer receives funding, a full-scale development begins. This usually involves a team of 20–100 individuals with various responsibilities, including designers, artists, programmers, and testers.

Games are produced through the software development process. Games are developed as a creative outlet and to generate profit. Development is normally funded by a publisher.

However, it is important to estimate a game's financial requirements, such as development costs of individual features. Failing to provide clear implications of the game's expectations may result in exceeding the allocated budget.

In fact, the majority of commercial games do not produce profit. Most developers cannot afford changing the development schedule and require estimating their capabilities with available resources before production. The game industry requires innovations, as publishers cannot profit from constant release of repetitive sequels and imitations. Similarly, many developers close down because they cannot find a publishing contract or their production is not profitable. Nevertheless, growth of the casual and mobile game market has allowed developers with smaller teams to enter the market.

1.2. Proposed Development Methodology

AGILE DEVELOPMENT METHODOLOGY

- Agile software development refers to a group of software development methodologies based on iterative development.
- Promotes a disciplined project management process that encourages frequent inspection and adaptation, encourages self-organisation and accountability.
- A set of engineering best practices intended to allow for rapid delivery of high-quality software.
- One of the most popular approaches to project management due to its flexibility, adaptability to change, and high level of customer input.
- Agile project management is not a singular framework — rather, it can be used as an umbrella term to include many different frameworks.
- Agile project management can refer to terms including Scrum, Kanban, Extreme Programming (XP), and Adaptive Project Framework (APF).

AGILE DEVELOPMENT METHODOLOGY IN GAME DEVELOPMENT

In a nutshell, Agile in game development means dividing the process of game creation into short iterations. Thus, instead of working on the whole project from the very beginning to the release, the development team works on small projects which are normally called features.

The development is mostly based on **Scrum** but it becomes more effective during post-development.Every sprint has a goal and if a particular goal is not achieved in 2 week sprints, it may be prolonged to three or four weeks but no longer.

The Scrum team usually includes three roles:

(1) PRODUCT OWNER

- Controlling ROI, or profitability of the project
- Prioritising tasks
- Updating features
- Approving work results

(2) SCRUM MASTER

- Assigning tasks
- Shielding the team from any external interference
- Planning meetings

(3) DEVELOPMENT TEAM(6-8 MEMBERS)

- Choosing backlogs
- Changing workflows
- Searching for the best solutions

WORKING OF SCRUM

Sprints:

The managers divide the team's tasks into short spans known as sprints. Sprints are a time frame under which the team has to clear the backlog. Now, the backlog accommodates tasks required to construct the product as set out by the Product Owner.

Usually, the time frame is two weeks or fewer. But according to the task, it can extend anywhere up to four weeks.

Daily Scrum:

Teams can follow daily Scrum for added efficiency. It's a meeting held daily where the team reports on their backlog status. There is a discussion and a brief review of the completed tasks. Also, there's a discussion about how to prepare for the next sprint.

It enables every team to remain on the same page. However, a reliable Scrum board is required. It'll convert the entire procedure into a cakewalk. The board will ensure smooth assigning of backlogs and preparing the team for the next sprints.

2. Kanban:

Kanban is a visual implementation of the Agile system. In this process, tasks from backlogs are picked and are completed as required. It's also known as the Just-in-time approach. It works with the help of a Kanban board.

A Kanban board can be a real time whiteboard with three columns: –

1. To-Do.
2. Doing.
3. Done.

Extreme Programming:

XP or extreme programming is an Agile methodology meant for improving the software development process. It helps developers embrace changes according to customer requirements.

The developer's craft Unit tests before programming begins. It helps in keeping the issues minimal and the entire app bug-free. As the tests are automated, it saves time and capital spent. This process also involves a unique system of programming in pairs.

1.3. Development process

- Pre-production
- High concept
- Pitch
- Sprint and Product log presentation
- Game environment design
- Script writing
- AI Pathfinding (Navmesh)
- Audio and art production
- Designing extra features
- First playable
- Release
- Maintenance

2. SYSTEM SPECIFICATION

2.1 Software Requirements

- Windows 7 or higher
- DirectX 11 or above

2.2 Hardware Requirements

- CPU :Intel i5 3rd Generation/AMD Ryzen 3 or above
- RAM :4GB, 8GB(for best performance)
- GPU :Intel HD 4000/Nvidia GT 710/AMD HD 3450 or above
- Disk Space: At least 5 GB

2.3 Game Engine and Language Description

2.3.1 C#

C# (C Sharp) is a general object-oriented programming (OOP) language for networking and Web development. C# is specified as a common language infrastructure (CLI) language. In January 1999, Dutch software engineer Anders Hejlsberg formed a team to develop C# as a complement to Microsoft's NET framework. Initially, C# was developed as C-Like Object Oriented Language (Cool). The actual name was changed to avert potential trademark issues. In January 2000, NET was released as C#. Its NET framework promotes multiple Web technologies.

2.3.2 Game Engine- Unity 3D

Unity is a cross platform game engine developed by Unity Technologies and is used to develop video games for PC, consoles, mobile devices and websites. Unity

gives users the ability to create games in both 2D and 3D, and the engine offers a primary scripting API in C#, for both the Unity editor in the form of plugins, and games themselves, as well as drag and drop functionality.

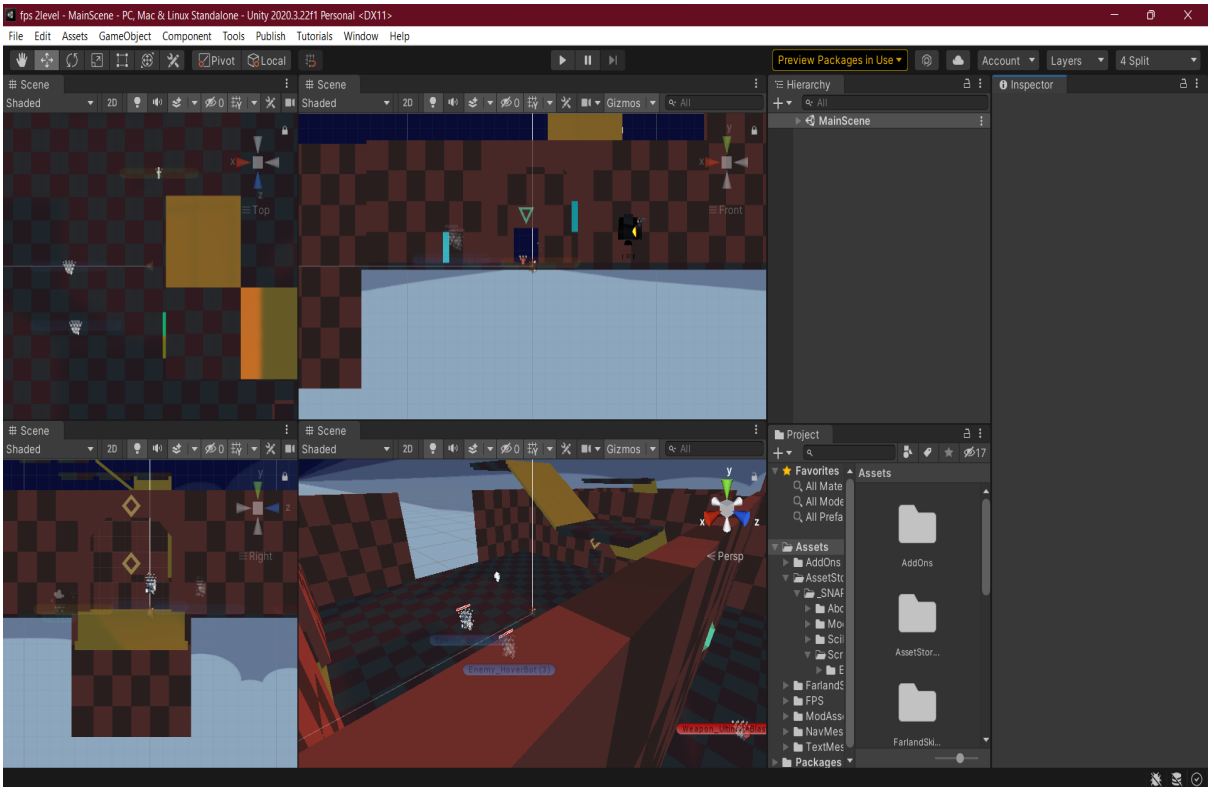
Within 2D games, Unity allows importation of sprites and an advanced 2D world renderer. For 3D games, Unity allows specification of texture compression and resolution settings for each platform that the game engine supports and provides support for bump mapping, reflection mapping, parallax mapping, screen space ambient occlusion (SSAO), dynamic shadows using shadow maps, render-to texture and full-screen post-processing effects. It also offers HDRP and URP rendering.

The High-Definition Render Pipeline (HDRP) is a high-fidelity Scriptable Render Pipeline built by Unity to target modern (Compute Shader compatible) platforms. HDRP utilises Physically-Based Lighting techniques, linear lighting, HDR lighting, and a configurable hybrid Tile/Cluster deferred/Forward lighting architecture. It gives you the tools you need to create applications such as games, technical demos, and animations to a high graphical standard.

URP or Universal Render Pipeline is a prebuilt scriptable render pipeline that is quick to customise and helps create optimised graphics across a wide range of platforms.

Unity also offers services to developers, these are: Unity Ads, Unity Analytics, Unity Certification, Unity Cloud Build, Unity Everyplay, Unity IAP, Unity Multiplayer, Unity Performance Reporting, Unity Collaborate and Unity Hub.

Unity supports the creation of custom vertexes, fragments (or pixels), tessellation, compute shaders and Unity's own surface shaders using Cg, a modified version of Microsoft's High-Level Shading Language developed by NVIDIA.



2.3.3. Microsoft Visual Studio

Microsoft Visual Studio is an Integrated Development Environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps. It can produce both native code and managed code. Visual Studio includes a code editor supporting IntelliSense as well as code refactoring. The integrated debugging works both as a source level debugger and a machine level debugger. Visual Studio supports 36 different programming languages and allows the code editor and debugger to support, to varying degrees, nearly any programming language, provided a language- specific service exists. Built-in languages include C, C++, C++ / CLI, Visual Basic .NET, C #, F #, JavaScript, Typescript, XML, XSLT, HTML and CSS.

3. SYSTEM ANALYSIS

3.1 Introduction

System/Requirement analysis phase is considered to be one of the most important phases in the system development life cycle. It is immensely important that the software developer make a thorough study of the existing systems. It refers to the process of examining situations with the intent of improving it through better procedures and methods. System Analysis is the process of planning a new system to either replace or complement an existing system. But before any planning is done, the old system must be thoroughly understood and the requirements must be understood. System Analysis is therefore the process of gathering and interpreting facts, diagnosing problems and using the information to make improvements in the system. System analysis is done with the following objectives in mind:

- Identification of public need
- Evaluation of the system concept for feasibility
- Economic and Technical analysis

3.2 Identification of need

In today's market of games, we noticed a distinct lack of local co-op games, even though this feature was cherished by many while we were young. This is why I intended to make a fun First-Person Shooter game that will keep you at the edge of your seat.

We recognized an untapped market in the category of arcade gaming and chose to act on it. Games nowadays rely too much on microtransactions and fail to focus on the gameplay.

3.3 Existing System

A number of first person as well as third person shooter games are already present in the market. However, most of them are way too violent and there is a distinct lack of games in recent years in the Arcade Shooting games market of the videogame industry. A distinct feature of arcade shooting games has been its lenient physics allowing for a more "fun" approach to shooting down enemies, their infinite

respawns, and where players don't have to worry as often about their own health declining.

Games like 'The House of the Dead 2' is a light gun arcade game with a horror theme and the second game in the '**The House of the Dead' series** of video games, developed by Sega for video arcades in 1998. **Galaga** is a 1981 fixed shooter arcade game developed and published by Namco. Controlling a starship, the player is tasked with destroying the Galaga forces in each stage while avoiding enemies and projectiles. They feature thrilling racing action but are sorely lacking in visual quality by today's standards.

3.3.1 HOUSE OF THE DEAD by SEGA

The House of the Dead is a rail shooter light gun game developed by Sega for video arcades in 1998. Players use a light gun (or mouse, in the PC version) to aim and shoot at approaching zombies. When a player sustains damage or shoots an innocent, one point of health is removed. If the player runs out of continues, the game is over. First-aid packs are available throughout the game which restore one point of health. Throughout the course of the game, players are faced with numerous situations in which their action (or inaction) will have an effect on the direction of gameplay.

3.3.2 S.T.A.L.K.E.R: SHADOW OF CHERNOBYL

S.T.A.L.K.E.R.: Shadow of Chernobyl is a first-person shooter,single-player survival horror video game developed by GSC Game World and published by THQ in 2007 following a long development. The game is set in an alternative reality, where a second disaster of mysterious origin occurred at the Chernobyl Exclusion Zone, causing strange changes in the area around it. The game features a non-linear storyline and includes role-playing gameplay elements such as trading and two-way communication with non-player characters.

In the game, the player assumes the identity of the Marked One, an amnesiac man trying to find and kill the mysterious Strelok within the Zone, a forbidden territory surrounding the Chernobyl Nuclear Power Plant. It is set after a fictitious second Chernobyl disaster, which further contaminated the surrounding area with radiation, and caused strange otherworldly changes in local fauna, flora, and the laws of physics. The background and some terminology of the game are borrowed from the Russian novella Roadside Picnic and its film adaptation Stalker.

3.4. Proposed System

We aim to make a classic arcade shooting game with photorealistic graphics which one can play for unadulterated fun.

We will be using the Unity 2020.3.22f1 to build a clean, simple and balanced UX to create a game application interface.

The proposed game will be a single player,multi-level, First Person Shooter game.

4. FEASIBILITY STUDY

Feasibility study is a set of system proposals according to its workability, impact on the organisation, ability to meet user needs and effective use of resources. The objective of feasibility study is not to solve the problem, but to acquire a sense of its scope. Three key considerations are involved in the feasibility analysis. These are:

- Economic Feasibility
- Behavioural Feasibility
- Technical Feasibility

4.1 Economic Feasibility

Economic analysis is the most frequently used method for evaluating the effectiveness of the candidate system. This is more commonly known as cost/benefit analysis. The procedure is to determine the benefits that are obtained from the candidate system and compare that to the cost of building the system

Developing a game/software product from scratch is a significant challenge for any organisation. It requires considerable investments in terms of effort and cost and also confirms client involvement, knowledge about client markets like Steam, Google Play, etc.

We target a demographic of people left behind in the modern shooting game market. Arcade-style shooting games put fun and a fast-paced experience above all else. A key feature of arcade-style shooters that specifically distinguishes them from simulation shooters is their far more liberal physics. Arcade shooting games used to be popular in the late 90s and early 2000s but have been left behind in the modern game market with Video Game companies pushing for more cinematic games over straightforward arcade shooting action.

The video game industry is growing at an incredible pace, projected to be worth an astronomical 120 billion USD by 2019.

The global gaming market was valued at USD 173.70 billion in 2020, and it is expected to reach a value of USD 314.40 billion by 2026, registering a **CAGR of 9.64% over 2021-2026**.As such, we can get at least a guaranteed return on investment with this kind of growing market.

4.2 Behavioural Feasibility

It includes how strong the reaction of staff will be towards the development of a new system that involves computer's use in their daily work. In this case, we see how the player reacts to a new game in the genre.

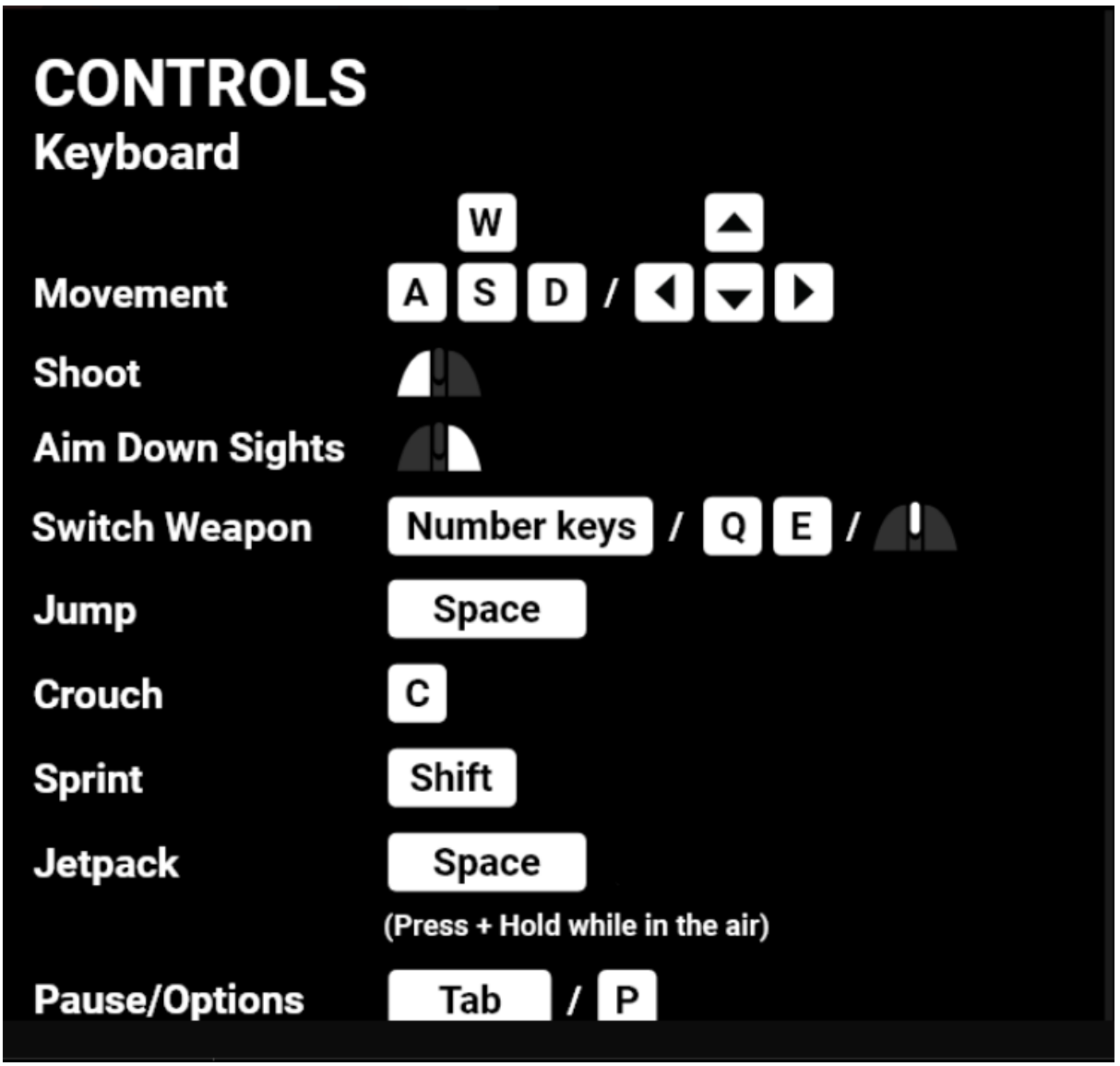
4.3 Technical Feasibility

It checks whether the existing computer systems support the candidate system or not or up to what extent it supports. In this case, we have fairly low system requirements in order to run the game at an acceptable frame rate. This is done to support a large number of systems and hence helps in gathering a wide audience.

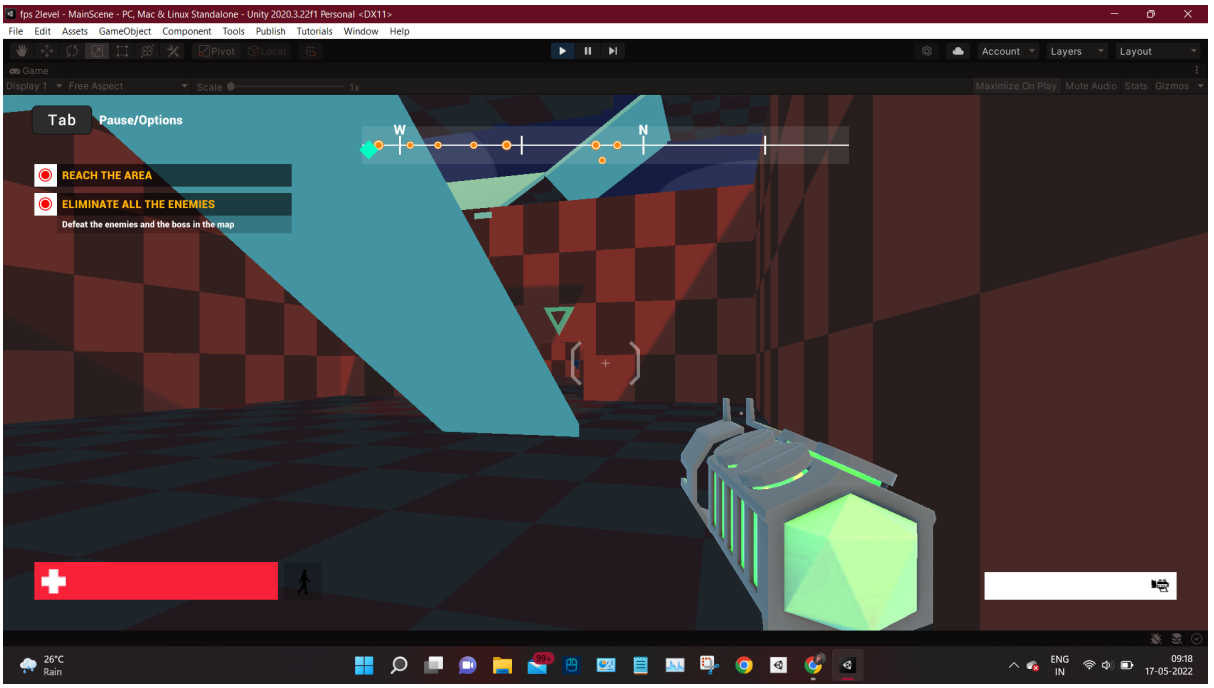
However, if the user is in possession of a high-end system, she/he will be able to run the game with a very high visual quality and high frame rates, making for a more visually appealing game.

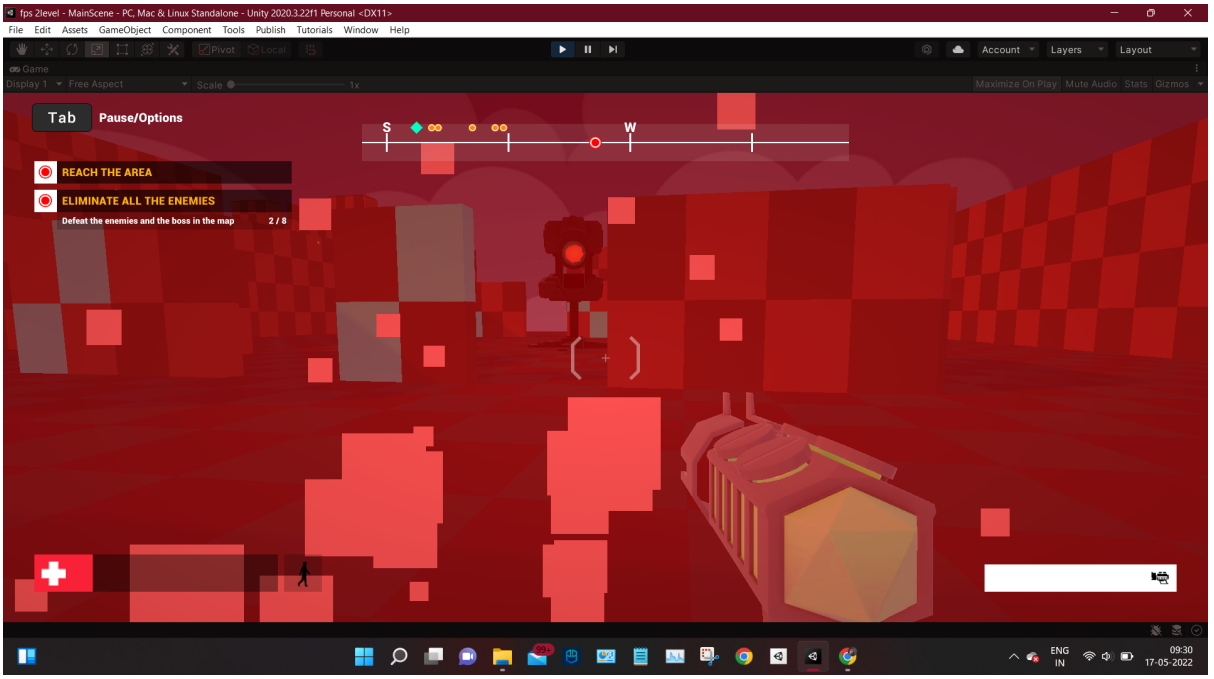
5. SYSTEM DESIGN

5.1 Input Design



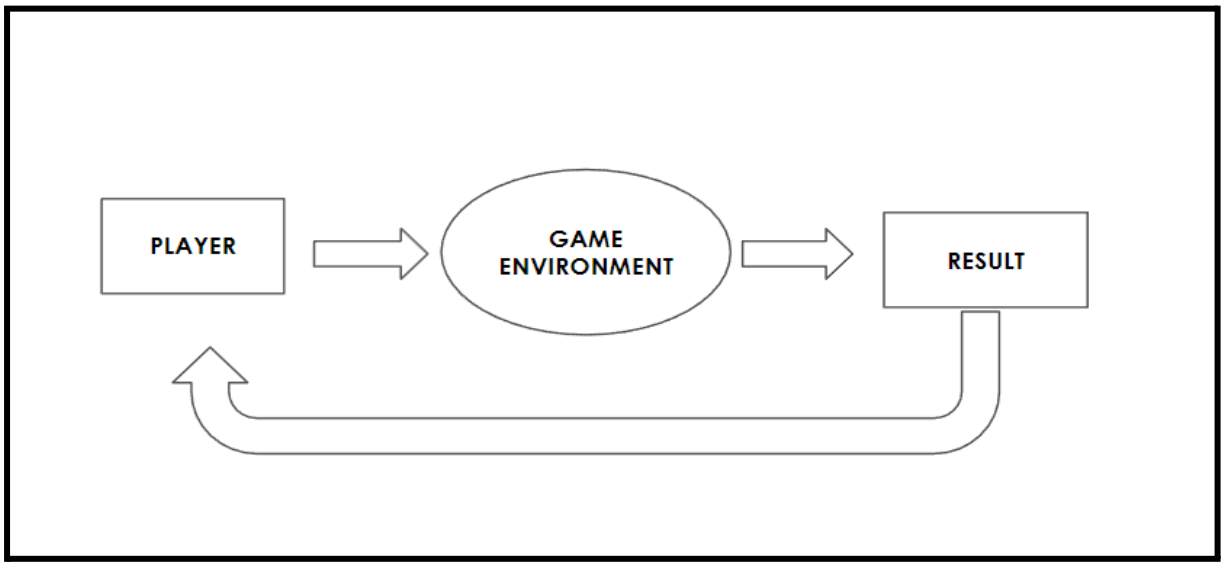
5.2 Output Design



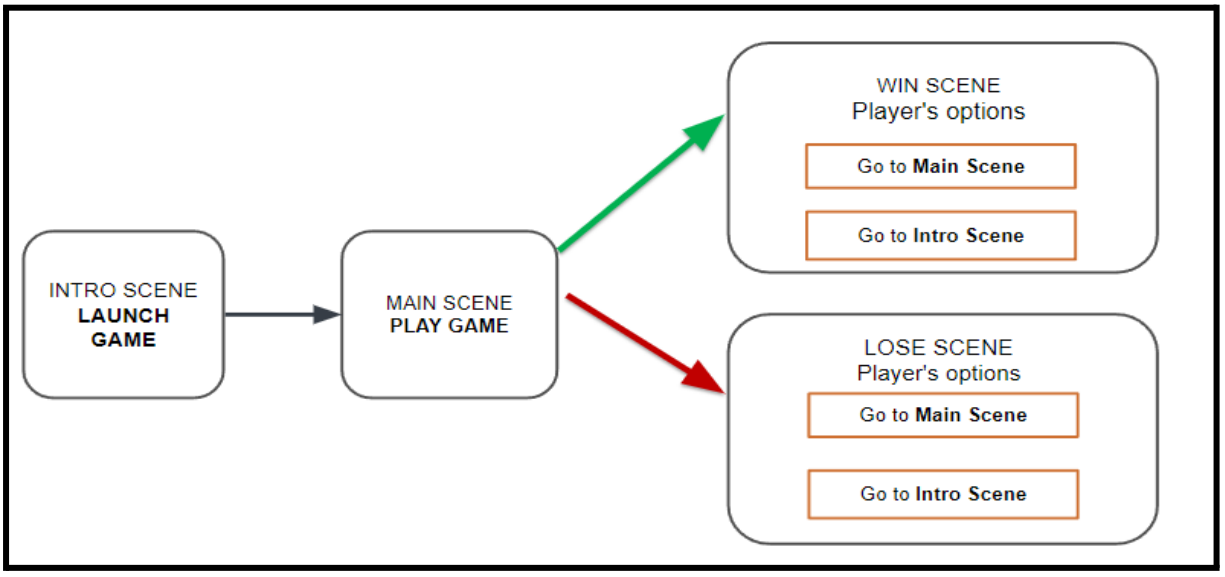


5.3. Data -Flow Diagram

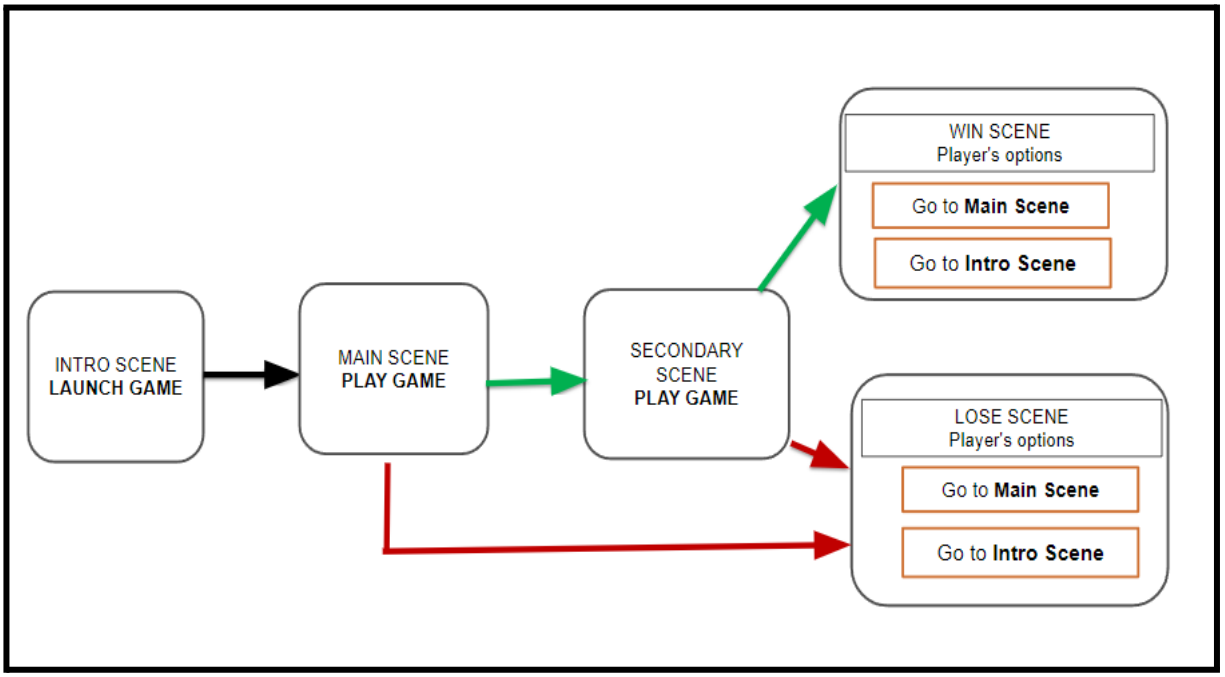
LEVEL 0



LEVEL 1



LEVEL 2


INTRO SCENE
LAUNCH GAME
MAIN SCENE
PLAY GAME
SECONDARY
SCENE
PLAY GAME
WIN SCENE
Player's options
Go to Main Scene
Go to Intro Scene
LOSE SCENE
Player's options
Go to Main Scene
Go to Intro Scene

6. TESTING

6.1 Unit Testing

It is a level of software testing where individual units/ components of software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output. In procedural programming, a unit may be an individual program, function, procedure, etc. In object-oriented programming, the smallest unit is a method, which may belong to a base/ super class, abstract class or derived/ child class. (Some treat a module of an application as a unit. This is to be discouraged as there will probably be many individual units within that module.) Unit testing frameworks, drivers, stubs, and mock/ fake objects are used to assist in unit testing.

Unit testing increases confidence in changing/ maintaining code. If good unit tests are written and if they are run every time any code is changed, we will be able to promptly catch any defects introduced due to the change. Also, if codes are already made less interdependent to make unit testing possible, the unintended impact of changes to any code is less.

Codes are more reusable. In order to make unit testing possible, codes need to be modular. This means that codes are easier to reuse. Writing tests takes time but the time is compensated by the less amount of time it takes to run the tests; You need not fire up the GUI and provide all those inputs. The effort required to find and fix defects found during unit testing is very less in comparison to the effort required to fix defects found during system testing or acceptance testing. The cost of fixing a defect detected during unit testing is lesser in comparison to that of defects detected at higher levels. Debugging is easy. When a test fails, only the latest changes need to be debugged.

6.2 Integration Testing

Integration testing is the phase in software testing in which individual software modules are combined and tested as a group. Integration testing is conducted to evaluate the compliance of a system or component with specified functional requirements. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

6.3 Acceptance Testing

Acceptance testing is also known as user acceptance testing (UAT), end-user testing, Operational Acceptance Testing (OAT) or field (acceptance) testing. Acceptance criteria are the criteria that a system or component must satisfy in order to be accepted by a user, customer, or other authorised entity. User acceptance testing consists of a process of verifying that a solution works for the user. It is not system testing (ensuring software does not crash and meets documented requirements), but rather ensures that the solution will work for the user (i.e., tests that the user accepts the solution); software vendors often refer to this as "Beta testing".

7. PRODUCT BACKLOG

SL. NO	USER STORIES	PRIORITY	COMMENT FROM SCRUM MASTER	COMMENT FROM PRODUCT OWNER
1	Creating game environment: Downloading Assets and its Preparation	Very High		Very important
2	The camera should stay oriented when it collides with enemies and other obstructions.	Very high		A first person view with the camera is used to resolve this.
3	It should be possible to see graphics and art design that complies with the game's universe.	High		Darker environment with lights and high contrast
4	The reactions and fighting should be synchronised	Very High		Based on types of enemies
5	A position indicator displayed on a monitor and making CCTV live feed	High		To give approximate positions of the enemies on a particular level
6	It should be possible to see enemies get tired and killed.	High		Once killed, enemies do not get respawned.
7	AI enemies should be able to immediately detect the player once it enters a room	Very high		Detection of the player means the AI will fight it
8	Player health	Very high		Very important
9	UI game controls for the given project should not be too complicated	Very high		Unity UI toolkit can be used to achieve this.
10	Adding required sound effects	High		Important for a good gaming experience
11	Testing	Very High		Very important to test the working of the game
12	Documentation	Very High		Very important

8. SPRINT BACKLOG

Sl.No.	USER STORY	NOT STARTED	IN PROGRESS	COMPLETED
1	Creating game environment			Completed
2	Downloading the required Assets			Completed
3	Adjusting camera orientation			Completed
4	A position indicator displayed on top of the game screen for navigation inside the level by the player			Completed
5	Sound effects			Completed
6	Programming/Script writing			Completed
7	Adjust fight navigations and AI pathfinding for mobile enemies			Completed
8	Player health			Completed
9	Adjusting enemy AI detection capacity in regards to the player and CCTV live feed			Completed
10	Boss fighting, damage, power boosts, weapons etc.			Completed
11	A smooth playing experience without bugs			Completed
12	Dual weapon wielding for the player			
13	Designing the second level using SNAPS technology			
14	Testing Game			Completed
15	Projectiles per each shot			Completed
16	Documentation			Completed

9. SYSTEM IMPLEMENTATION

System implementation uses the structure created during architectural design and the results of system analysis to construct system elements that meet the stakeholder's requirements and system requirements developed in the early lifecycle phases. These system elements are then integrated to form intermediate aggregates and finally the complete system-of-interest (SOI).

Implementation is the process that actually yields the lost level system elements in the system hierarchy (System Breakdown Structure). System elements made are reused. Production involves the hardware fabrication process of forming, removing, joining, and finishing, the software realisation process of coding and testing, or the operational procedures development process for operator's roles. If implementation involves a production process, a manufacturing system which uses the established technical and management process may be required.

The purpose of the implementation process is to design and create (or fabricate) a system element confirming to that element's design properties and requirements. This process bridges the system definition process and the integration process.

The following major activities and tasks are performed during this process:

• Define the implementation strategy - Implementation process activities begin with detailed design and include developing an implementation strategy that defines fabrication and coding procedures, tools and equipment to be used, implementation tolerances, and the means and criteria for auditing configuration of resulting elements to the detailed design documentation. In the case of repeated system element implementations (such as for mass manufacturing or replacement elements), the implementation strategy is defined and refined to achieve consistent and repeatable element production; it is retained in the project decision database for future use.

The implementation strategy contains the arrangements for packing, storing, and supplying the implemented element.

• Realise the system element - Realise or adapt and produce the concerned system element using the implementation strategy items as defined above. Realisation or adaptation is conducted with regard to standards that govern applicable safety, security, privacy, and environmental guidelines or legislation and the practices of the relevant implementation technology. This requires the fabrication of hardware elements, development of software elements, definition of training capabilities, drafting of training documentation, and the training of initial operators and maintainers.

• Provide evidence of compliance - Record evidence that the system element meets its requirements and the associated verification and validation criteria as well as the legislation policy. This requires the conduction of peer reviews and unit testing, as well as inspection of operation and maintenance manuals. Acquire measured properties that characterise the implemented element (weight, capacities, effectiveness, level of performance, reliability, availability, etc.).

• Package, store, and supply the implemented element - This should be defined in the implementation strategy.

10. SYSTEM MAINTENANCE

The results obtained from the evaluation process help the organisation to determine whether its information systems are effective and efficient or otherwise. The process of monitoring, evaluating, and modifying existing information systems to make required or desirable improvements may be termed as System Maintenance.

System maintenance is an ongoing activity, which covers a wide variety of activities, including removing program and design errors, updating documentation and test data and updating user support. For the purpose of convenience, maintenance may be categorised into three classes, namely:

i) Corrective Maintenance

Corrective Maintenance is a maintenance activity undertaken to overcome the failure or damage found during the preventive maintenance period. In general, corrective maintenance is not a scheduled maintenance activity, because it is done after a component is damaged and aims to restore the reliability of a component or system to its original state. Corrective maintenance also is known as breakdown or run to failure maintenance. Maintenance is only done after equipment or machinery is damaged. If this maintenance strategy is used as the main strategy will result in a high impact of unplanned maintenance activities and replacement parts inventory.

ii) Adaptive Maintenance

Adaptive maintenance is the implementation of changes in a part of the system, which has been affected by a change that occurred in some other part of the system. Modification of a software product performed after delivery to keep a software product usable in a changed or changing environment. Adaptive maintenance consists of adapting software to changes in the environment such as the hardware or the operating system.

The term environment in this context refers to the conditions and the influences which act (from outside) on the system. For example, business rules, work patterns, and government policies have a significant impact on the software system.

iii) Perfective Maintenance

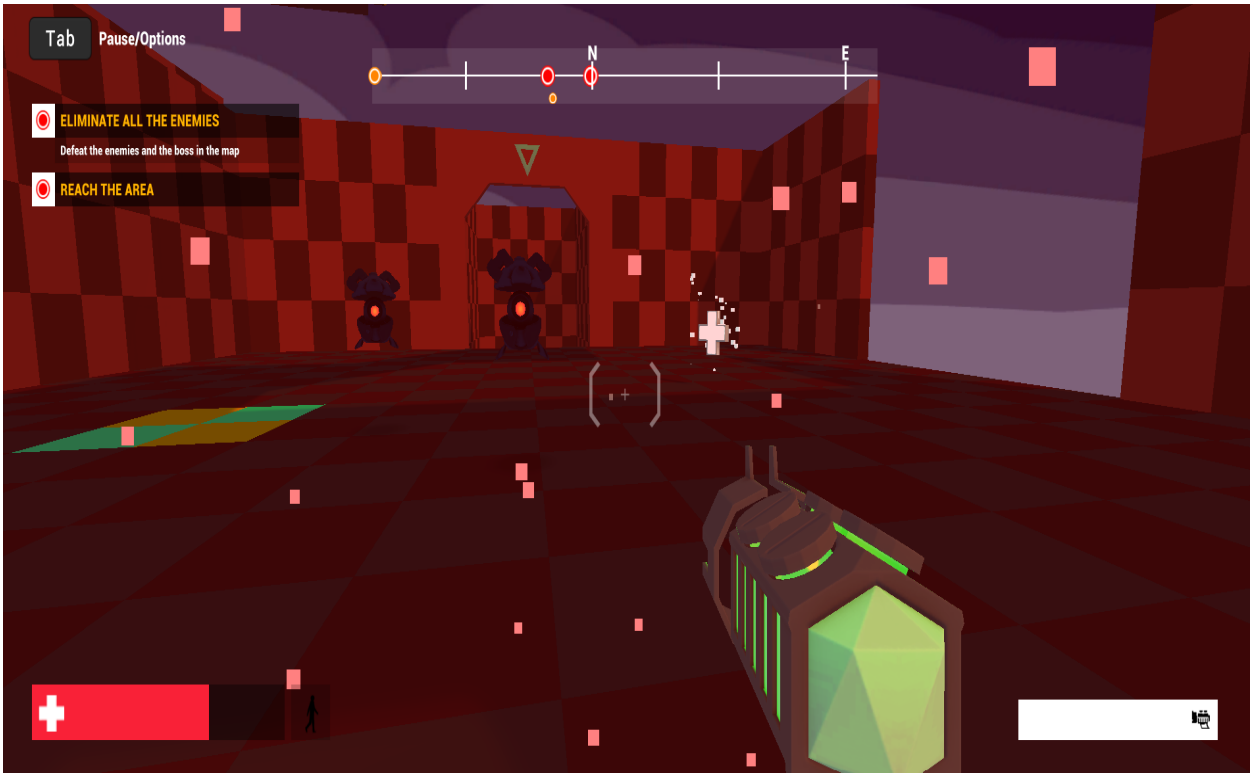
Perfective maintenance means adding new programs or modifying the existing programs to enhance the performance of the information system. This type of maintenance is undertaken to respond to user's additional needs which may be due to the changes within or outside of the organisation. Perfective maintenance mainly deals with implementing new or changed user requirements. Perfective maintenance involves making functional enhancements to the system in addition to the activities to increase the system's performance even when the changes have not been suggested by faults. This includes enhancing both the function and efficiency of the code and changing the functionalities of the system as per the users' changing needs.

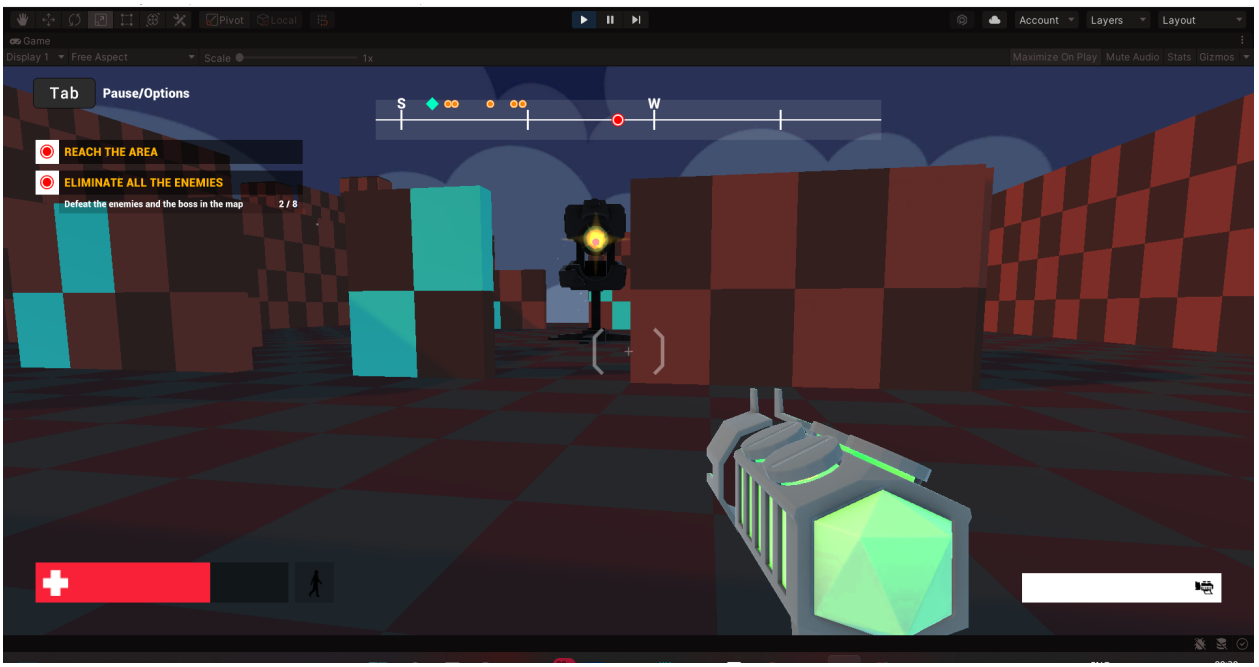
iv) Preventive Maintenance

Preventive maintenance is performed on a schedule, which detects and lessens the degradation of components and systems. Preventative maintenance has the aim of increasing reliability, saving costs from major failure or down time, reducing the risk of failure and extending the life of machines or facilities. Preventive maintenance cannot assure escape from catastrophic failures and sometimes may seem to be unnecessary. Preventative maintenance includes test operations as per design, diagnosis, replacing worn out parts, measurements, lubricating, tightening and adjustments

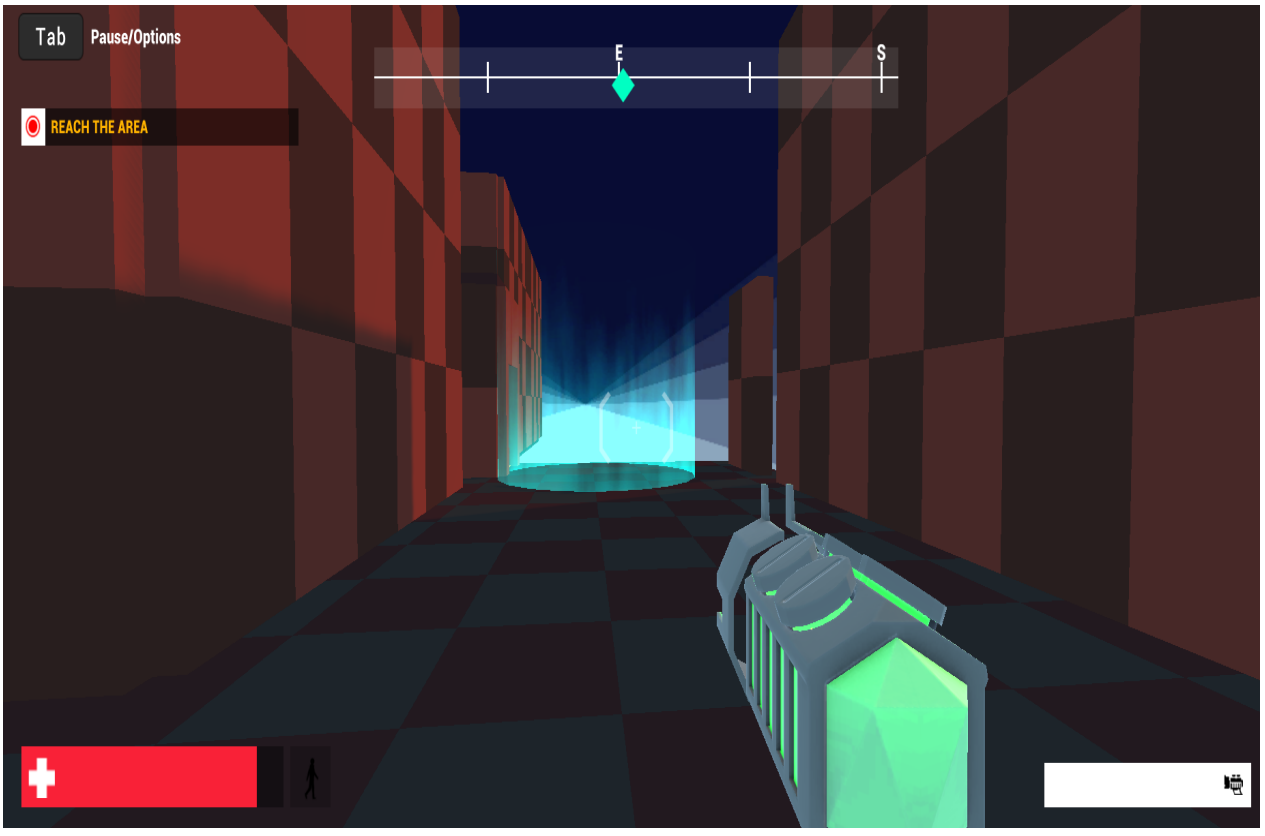
11.SCREENSHOTS

LEVEL 1



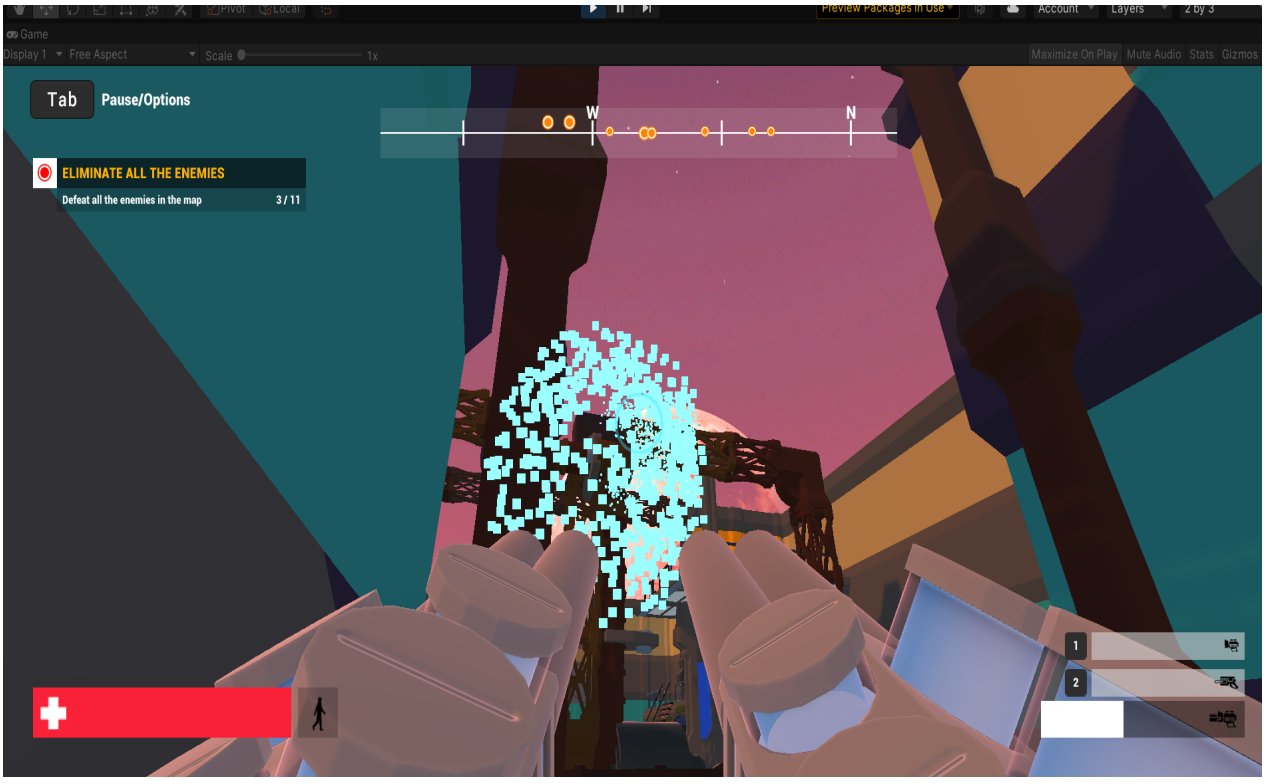



Tab
Pause/Options
N
E
REACH THE AREA
ELIMINATE ALL THE ENEMIES
Defeat the enemies and the boss in the map
2 / 8


Tab
Pause/Options
E
S
REACH THE AREA

LEVEL 2


Tab
Pause/Options
ELIMINATE ALL THE ENEMIES
Defeat all the enemies in the map
1 / 11
W
N
1
3


Tab
Pause/Options
ELIMINATE ALL THE ENEMIES
Defeat all the enemies in the map
3 / 11
W
N
1
2


Game
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
Stats
Gizmos
Tab
Pause/Options
S
W
ELIMINATE ALL THE ENEMIES
Defeat all the enemies in the map
7 / 11
1
3

12. FUTURE ENHANCEMENTS

Multi-player can be implemented into the system either locally through split screen or online. In this way, players will be able to login to the system and fight against each other in multiplayer mode and AI enemies to complete the objectives and advance to the next level.Third person perspective can also be included in time. Future updates can bring more maps, levels and other game features like time restraints.The game can also be modified so that the human players can fight against each other to see who is the last man standing and that player can be declared as the winner. As the system is highly modular, it is relatively easy to add and modify components as needed

Other technology like Visual scripting can be included to make the game development easier. Also including concepts of swarm-intelligence in AI enemy development can lead to a more interesting and harder strategic gaming experience for the player.

Since the gaming industry is an ever-evolving world with stunning enhancements getting introduced routinely, innovations in all aspects will always be welcomed with open arms. Thus, worrying about future scope is a moot point.

13. CONCLUSION

At the end of the project, a video game is successfully created that is both visually and mechanically appealing to a wider audience. It can easily be ported to other PC systems in order to widen the market reach and also be put up for sale on videogame markets like Steam and the Google Play Store.

Using WebGl, we can put the game up online for others to play and enjoy.

Since the gaming industry is an ever-evolving world with stunning enhancements getting introduced routinely, innovations in all aspects will always be welcomed with open arms. Thus, worrying about future scope is a moot point.

14.REFERENCES

- https://docs.unity3d.com/Manual/index.html
- Unity 3D Game Development by Example Beginner's Guide - Ryan Henson Creighton
- https://www.gamedesigning.org/learn/unity/
- FPS microgame development: UNITY
- UNITY LEARN
- Learn UNITY: Beginner’s game tutorial
- UNITY FPS microgame Tutorials