Final Year Project Report

Full Unit – Interim Report

Building a Game

Md Mubin Morshed

A report submitted in part fulfilment of the degree of

BSc (Hons) in Computer Science

Supervisor: Angeliki Koutsoukou Argyraki

Department of Computer Science
Royal Holloway, University of London

April 09, 2024

Declaration

This report has been prepared based on my own work. Where other published and unpublished source materials have been used, these have been acknowledged.

Word Count: 10,628

Student Name: Md Mubin Morshed

Date of Submission: 08/12/2023

Signature: Md Mubin Morshed

Table of Contents

Abstract

Game development is now high in demands. It is popular among almost world-wide thank to the game engines and the modern powerful computers with powerful CPU (Central processing unit) and GPU (Graphical processing unit). It has become one of the most popular medium of entertainment. This project is all about building a game. To be more precise, the whole report will be made on building a tower defence game. A tower defence game is about building different towers or buildings and protecting it from the invaders. In tower defence game we can add many games design and, we ca manipulate the game objective if we want. The game will be made with unity. Unity is a game engine which is quite popular nowadays. Using unity, we will create the scene and the buildings that we need to run the game. Moving on we will use scripting to provide functionality to the components which are necessary. The UI design going to be achieved by unity engine itself. We can say that unity is just not a game engine but an editor itself. We are going to make the game from the scratch. For some component we can use the unity store to get the assets and use it in our scene. The functionalities will be hard coded with unity. The unity game engine has prebuilt libraries which will help us in the process of providing the functionalities to the game objects. The theme of the game will be there will be a starting point and an end point. The enemies will spawn at the starting point. And the task of the enemy will be travel through the path till the end point. We must stop the enemies to reach there to win the game. There will be non-stop spawning of enemies with a time lapse. As the time passes the game will be harder. There will be turrets to stop the enemies from reaching the end point. It will shoot bullets and upon touching the enemy object it will be destroyed. It will leave a hit effect after destroying the enemy. The game is set to the low polygon setting for memory management and better performance. There will be the concept of a shop and the game currency. The game will consist of multiple scenes. Each scene will serve unique purpose as Menu, Level Selector and the scenes which will serve as the levels. The first scene will serve as a start menu or the place where will be started the game and the second page will hold all the philosophy and the milestone of the tower defence game. There will be some range of variety of the towers along with the enemies, each having unique attributes. The game will give the player several chances which will be considered as lives. Upon losing all the chances the game will be over. The player can press retry and start a new or can go back to the main menu and quit. There will use of canvas to show the scores and the waves. The scores are the number of enemies that has been destroyed by the player. There will be some variety of defence building to stop the enemies to achieve it's goal and destroy it. The enemies will be spawned in time to time. The next waves of enemies will not appear until all the enemies are destroyed. The over all game will be in very low graphics because of device friendliness. Also, I have tried to make the game in a way that it consumes as less memory as possible with the appropriate game design. Using game designs might not bring the exclusive graphic but it will make the game much more smother comparing to instantiating the game object without having a boundary.

Chapter 1: **Introduction**

We are all aware of the term game. It is the most popular way to enjoy our free time. The computer game history has a root. It started from 1947.The first game was tic-tac-toe, and it was played on cathode ray tube in US Earlier the game was straight forward and plain2D game like Pacman.[1]

After that the era of game started to evolve gradually. In modern days the games are now highly developed and have massive graphics. They are visually attracting, and they have many features. To add more the game itself can contain a story like movie. Thanks to the inventions and hard work for the game developers the video game now has much more fields. From 2D to 3D there are many more aspects. Thanks to the field of software engineering and scripting language we can now increase the performance of the games. One more thing that took game development to another level is the game engines. There are a lot of game engines nowadays e.g., Unity3D, Unreal Engine, Godot etc. Some gaming entertainment company made their own game engine to develop their games like Ubisoft Anvil Next. With the help of the game engine, we can make the game much better, attractive, and efficient. The game performance depends on the memory usage and most important on scripting or coding. With help of the game engine, we can optimise the game performance because we can use it to give the graphical view with just drag and drop. And with the help of scripting, we can provide functionality as well as we can manipulate the view with certain range.

1.1 Aims and Goals

There are countless games we can see nowadays as well as with different genres (e.g. First person Shooting game or FPS game, Tower Defence, Arcade etc). One of the most popular genres is tower defence. It is popular because the game can be relaxing and there can be various concepts as well. There are a lot of game in this genre such as Warcraft, clash of clans, Plants vs zombies etc. It is also popular to improve the mathematical skills. The idea behind the tower defence game is to the player will be building towers or any other offensive or defensive buildings. The objectives of building these towers or turrets or any buildings by the player to stop the enemies or creeps to prevent them reaching a finishing point which can be a base of player or an end point or defend the headquarters or the whole base itself. The enemies will spawn like waves with an increasing number each time creating difficulties for the player. The more time passes the more difficult the game becomes. Players will upgrade their buildings or buy and build more defensive/offensive towers to make their base stronger and to prevent the enemies to reach to the targeted point. If the enemies manage to reach the destination the game will be over, and the player will lose.[2] There will be also some life limits such as how many times a player can loss or let an enemy pass or there will be a durability for the headquarter to get destroyed. There can be much more to add as like drag the buildings inside the game or some towers to slow down the enemy as well.

I am going to build a Tower defence game with the help of unity. I am going to follow the concept of the tower defence game and make a 3D game. The concept of my game is, there will be two portals. One will be a start point and another one will be an end point. The enemies will be spawning from the portal and start travelling towards the end portal. The player must build turrets, or any other defence building to kill the enemies and stopping them from reaching there. To describe the turrets variety there will be a standard turret which will through bullets, a missile turret which will damage some enemies in a declared range, lastly a laser turret which will slow down one enemy and damage over time. The damage over time means it will damage over increasing time as long as the enemy is the range of the turret. Of course, the enemy will keep spawning in waves. In each wave the number of the enemies will be increasing. In the bigger picture if any wave of the enemy reaches to the end point the game will be over for the player. There are several

chances. As an example, if more than 4 enemy have crossed the end portal then the game will be over. There will be game currency upon killing enemies. For each enemy there will be $10. The currency will be used to build more turrets. I can achieve the game theme with the help of Unity Game engine. I am going to use the game engine to develop the game scene and enemies with other buildings. To establish the functionalities, we are going to use C# scripting. As Unity supports C# as it's scripting language.[4]

My aim is to build the tower defence game with adding some game feature and designs. The idea is to demonstrate the software engineering methods to improvise the game and make it much efficient and performance friendly. Adding more, I want to make a game where will be no bug and visually attractive as well. I will keep the memory management in my mind for efficiency. I want to make a low poly game so that everyone can play the game on pretty much every device. Lastly, the game will be a cross platform game which can be played in any device. To make it cross platform I have used the URP 3D template in the unity. Unity comes with templates and with the renderer for every platform. SO, it makes easy for the game developer to choose the platform during the development of the game.

1.2 Objectives

When I was young one of my favourite games was Plants vs Zombies. The concept of the game is to I must kill the zombies with the plants to defend myself or my base. There are many kinds of maps each maps follows the same objective. There are some changes of events depending on levels or any occasion e.g., Christmas.[5]

The game logic stated above made me attracted to the genre of tower defence game. It is now a one of the most popular genres in the world. Game developers also have made this genre attractive with the help of design patterns and other standards (e.g. visual effects, more game events, more gifts, new levels along with the updates). When I got the project, I was determined to do a 3D low poly tower defence game as it has an interesting concept and can be fun to work with it. We can add so many design patterns as object pool, fly weight as well as we can use the concept of OOP to enhance the game performance and the security. As we are using Unity we do not have to worry about the graphics as it can be handled by the game engine. I decided to take the development to a level so that it can be an example of standard software development methodology. I want to see myself making a user interface which can be attractive to anybody. Furthermore, I want to keep it simple so that a child can play that as well.

We can say the objective is to make a cross platform game that can be played by anyone regarding their age and enjoy the leisure time. To keep up with my objective. I have decided to keep the game in a range of low polygons so that it does not look anything fancy. Because my objective is to make the game for all the people not just for the pro gamers or streamers who plays high graphical game. In the game there is a score system. This will show the score as the number of enemy that the player has destroyed and the number of the wave that the player has survived. The players objective will be to survive as much waves as possible while defeating the enemies as the adversary and it will create more interests. In the game I have increased the difficulty level with each wave. I have already added the functionality to increase the number of the enemy with each wave. I have added health bar so the player can see the end of the enemies that comes through the game. By increasing the difficulty, the player will not lose their interest as at some point the player will ty to achieve the impossible task to clear that wave which he was unable to clear. I wanted to keep an option to upgrade the weapons or inventory to increase their offense so that they can achieve the

strength to clear difficult waves. The upgrades should have been achieved through the game currency. My objective is the currency will be produced upon killing the enemy. In this way the player can use and gain the coins to buy the materials to strive to the level where he has enough strength to clear the waves. There will be 3 variety of the enemies with different attributes. And the next wave will not start until all the enemies will be defeated. The game will consist of two scenes. First scene will be the main menu which will lead the player to start or quit the game and the other scene will hold the terrain map along with all the game concept and the ideals.

The game will consist of multiple scenes. Each scene will have their own unique features. There will be a scene which will represent the main menu. Upon clicking the play button another scene will be load which will indicate the levels. Upon clicking any level another scene corresponding to the button will be loaded for the player to play. The objective of this part is to connect all the scene with the script to access each scene by the player in the game. All the scenes will be accessed by button which will be bound by methods. There will be multiple methods correspond the buttons which will indicate the buttons. There will be a chance to add multiple levels and connect it through methods. There will also be some game over UI which will pop up upon players defeat.

Chapter 2: **Tools**

A game can be built with many tools. And the same can be made with different tools or approach. Some game can be hard coding which can be made with some predefine library as JavaFX, pygame etc. However, with coding and GUI library the game can be made with only low-level gaming as well as the graphics will be low. To create a high-level game, we can use game engine such as Unreal engine 4, Unreal engine 5, Unity3D, Godot etc. We can also create our own game engine to make a game.

For the project I am going to use a game engine named Unity3D. With a game engine we need an IDE to script for the game objects. As the IDE we are going to use visual studio to edit the script and give functionality to the game. For the coding standards and check style purpose I am going to use visual studio code which have an extension called prettier to clean up the code for a standard.

2.1 Unity

Unity is one of the most popular game engines. Most of the game developer now uses it. And it is a very beginner friendly game engine. Unity is an open-source game engine. To add more it is a multi-platform or cross platform game engine. To elaborate more, the game created with unity 3D can be run on the phone, either iOS or an android phone, can be run on macOS, Windows or Linux operating system. As it is open source it does not cost anything. Anyone can download and access the unity engine and can build their game.[6]

Unity has a paid version as well. Which cost $1500 and it called unity pro version. It has more upper hand comparing to the free version. As an example, pro version has the real-time shadows and audio filter, custom splash screen, video playback/streaming etc. However, the free version is strong as well as it has rigid physics, terrain editor, tree editor and many more to explain.[7] I am going to use the free version of the unity game engine for the project version 3.6.1.

In unity every single component such as terrain, empty, cube, sphere, sun even the trees are considered as game objects. We can create and destroy these game objects. On the top of that, we can add any feature in this game objects. WE can change the size; shape and we can transform it with the help of unity game engine. More precisely, they are containers that we can transform or transfigure to an appropriate figure that we need in the game.[8]

Unity can provide various aspects as well. It is quite interesting because unity can be used as an IDE as well.[9] The reason of saying that with unity3D it is possible to build a game without programming or scripting. So, those who are not able code or unable to learn code we can use the unity blueprints to provide the functionality to the game object.[10]

To begin with, we must install unity hub to open the unity editor as there is no other way to use the unity editor. Then we must use a version of unity editor, in my case it is Unity 2023.1.9f1. Before doing that, we must add the free license to run our unity. For the more advance development or better optimisation we can buy the license but for now I will stick with the free license.[11]

After creating a unity project with a project template, we will see some windows. Project, hierarchy, space world, a sun, Inspector and many more. We have to create the game objects in the empty space where Z axis exist alongside the X and Y axis as we are building a 3D game.[12]

I have added a terrain which will be my map and edited it to carve a path for my enemy objects to follow. I have added a start point where the enemy objects will be spawning and at the end point the enemy will reach there and they will be destroyed. To follow the path, I have placed a several numbers of waypoints which are empty game objects. The enemy will follow the waypoints and

reach at the end points. The functionality of these will be provided by the scripts. The enemy object will move towards a waypoint. Before reaching there, it will target the next way point. All the way points will be the child of a game object where we will attach our script. The enemy object is a prefab which is imported from the unity asset store. It is a skeleton. The spawning of the enemy objects will be controlled by an empty game object which will be considered as a game master. There is a turret game object which is a prefab imported from the asset store. Turrets going to shoot bullets towards the enemy if the enemy is within the range. The enemy objects will be destroyed if the bullet hits. This functionality is achieved with the help of scripting. In the turret object only, the head will move.

2.2 Other Tools

The other tools that I am going to use:

- **Visual Studio the code editor:** All the C# scripting will be edited and modified with visual studio. Visual Studio is a Microsoft provided text editor which can act as an IDE as well. We can install plugins to make it more efficient. Inside the editor we can set the code template manually. I could not find any way to upload a xml file to auto configure it like eclipse. Apart from that it is a very powerful IDE. It supports the necessary C# scripts and the plugins that we need.

- **Version Control:** A version control system is a software development tools to organize, manage and co-ordinate the project. For the developers these tools can be used for collaboration and storing the code.[13] For the version control I am going to use GitLab. The repository will be provided by gitlab.cim.rhul.ac.uk. I publish my work there.

Chapter 3: **Design Patterns**

There are various reasons to use design patterns in a game. One of them are interaction within the game. It is crucial to establish a communication between the objects in the game as well as with the user. Game designs offers such architecture that we can achieve the motive that we want. Even if a developer failed to understand the concept, in the pattern there are potential solution can be found within the game. It also makes the development interesting. It is also can be used to communicate with the peers and professionals. As the design patterns describes the functionality or motive of the game mechanism it is easy to understand and read the program just by studying the design patterns.[14]

Design patterns allows us to organize our code as well as giving us some technical advantages. It increases code readability as well as it allows us to re-use the objects. So, we do not have to make the program messy which can be expensive to manage, we can keep it simple and can increase re-usability to have some free memory space. It also helps us to have a loosely coupled relation between the classes which allows the game not to crash. And there it goes with the singleton pattern which can keep one responsibility bounded within one class which can be helpful to understand the mechanism. And having the knowledge of gaming pattern we can do de-coupling easily as we just have to understand the functionality not the code then we can re-factor it with the implementation of a design pattern.[15]

3.1 Flyweight Design Pattern

When we see an animation with a view of the developers, we can understand that millions of polygons have been put in the memory somehow. Let us take an example for the trees in the forest. As we know that everything in unity engine is a game object, every time we place an object it will be instantiate as an object in the game. For that, each instance of trees will take different memory space for their own. That can cost a lot of memory and can make our game slow. This can be solved with the flyweight pattern as this design patterns allows to use the shared memory among the same type of object. By using this pattern, we can tell the GPU to render trees in different positions but this time with the shared memory. In flyweight design pattern divides the objects internal state into two states. One is called intrinsic data and extrinsic data. Intrinsic data is the methods or the components which relies in the class and without them the class cannot operate such as bark, branches, leaves; it will be stored into a class. And extrinsic data are those which will be stored outside of the class and without them the object can operate such as colour. We can re-use the extrinsic class in many aspects as the grass colour and the colour of the leaves can be the same. There will be a factory will be used most likely a factory class to create those objects instead of a constructor. This way it is easy to keep track for the data. And as the data is shared more precisely a big number of trees will be instantiate under the same class which will be a shared memory will reduce the pressure on the memory and can make the game quite efficient. [16]

I cannot say I have used this design pattern fully in my game, but I have used some of the approach. Such as shared memory. If we take instance our TurretFactory.cs and Wave.cs script I have classified the common criteria as turret game object and the cost and put it to one class with the access modifier to get the cost to extract from the player's money. With the Wave.cs class I have pointed out the enemy prefab the number of the enemy that going to spawn during the wave and the rate of the wave. It all shares the memory. We do not need to instantiate each time with a different variable. All the variable which will be created via the class will share the same memory space which going to be allocated by the script during the runtime. It gives the advantage of saving the memory space and run the game much more smoothly than having different variables for each game object which going to populate the memory unreasonably.

3.2 Object Pool

If we want to run our game smoothy then the best way is to allocate a large amount of memory space and keep it occupied until the task ends. However, the problem arises when we have to create new objects. Because of fragmentation it can be quite deadly. This problem can be solved by the object pool design pattern approach. The design pattern is as it sounds as object pool. It is a collection of similar objects. The objects are re-usable. Upon initializing the pool if the objects are free then it will be used and it will keep spawning. If the pool is empty which means if there is no object free to use, then it will create the similar object as per the requirement. This will reduce the workload of the garbage collector. As the objects will be re-used not destroyed the garbage collector does not need to work so the pressure on the memory will be reduced. Moreover, the instantiation will be very less so it will be faster for the computer to load the frames. Re-using the objects also improve the mechanism as there is less work to do comparing instantiate and create new objects. We have to keep on mind that:

- The object is similar type.
- This can only be applicable if there is a frequent need of destroying and creating objects.
- If the fragmentation problems occur.[17]

3.3 Singleton

A singleton pattern means we can create only one instance with one class. That is mean it will be instantiated with one instanced. There will be only one task assign to it. The advantages we will get:

- It will not initialize until it is needed. That allows to save memory.
- It will be initialized at the moment of runtime. Which allows to let the other important work done before the singleton allocate the memories.
- The singleton can be inherited. That is mean it can have sub classes.

There are several reasons not to use this pattern. Some of them are:

- It is a global variable which makes the debugging quite difficult. If someone else programmed it is hard to find the reasoning between the global variable and the method that we want to debug.
- It encourages coupling. In contrast the game will be beautiful if it is loosely coupled so each class will not be depending on each other. As an example, we do not want our shadow coupled with the wind or audio or physics.
- It is not compatible with concurrency. [17]

I tried to avoid this pattern since the demerits are heavier, but it is some kind of necessary evil that cannot be avoided. I have used it in quite some classes to give functionality and make it easier to use. As an example, I have used this design pattern in my Buildmanager.cs script to make it easier to access the method and reduce the initialising since it costs memory. It also helps us to avoid null reference pointer error when we instantiate an object, but we did to initialise it. To initialise it we

need to set a reference of the game object which will take some memory space otherwise it wil through null reference error.

Chapter 4: **Work Evaluation**

I am gradually building my Tower Defence game with the help of unity. The amount of work I have done:

- I have set up the unity and created a project with a universal pipeline which will allow me to run the game on any platform. Which means the game can be switch between any platform and have separate builds. For now, I am keeping the game for windows, mac and Linux build. But it can be changed anytime if it needed to be. Unity made it easier with the help of URP. The graphic is automatically set by the unity engine, so we barely have to worry about the graphic API such as Vulkan or WebGL.

- I have added a terrain to make my map area and carved a path with the terrain editor. And I have painted my terrain in green colour grass as well. The terrain is low poly so that it can be device friendly. The terrain will serve the purpose as the platform and the map in the game.

- I have added a start and an end point where the enemy will be spawn and destroyed upon reaching the end point. They are portals which are imported from the unity asset store.

- I have added the way points which are empty game objects. I have arranged in such way that all the way points follow the path starting from the starting point till the end points. I have made all the way points child object of an empty game object. We will add a C# script to the empty object to put all the child objects into an array. The way points can only be seen in the scene, and it is invisible in the game. They are diamond shaped objects which make them quite unique comparing to a simple traditional game object.

- The enemy object is imported from the unity asset store. It is a collection of skeletons. Then I have added the functionality with the script named Enemy. The enemy class will use the waypoints script array to guide the enemy to target the nearest waypoint. Upon reaching just before the waypoint the enemy object will target the next waypoint. The enemy object will be destroyed if it reaches to the last waypoint. In the game or in player's view the enemy will be following the path that has been carved through the terrain. The enemy will be the target of the player to destroy and stop the enemy to achieve its goal. There are three types of skeletons which will serve different purpose.

- There is an empty game object which will be considered as a game master or game manager which will control the waves with the help of the WaveSpawner.cs script. The enemies will be spawning in the waves with a time lapse. Each wave will spawn some fixed amount of enemies with different traits. The number of the enemies which will spawn can be changed inside the game engine which will allow the game to be harder as the time passes. The game master object will hold three more script. GameManager.cs, pauseMenu.cs, PlayerStats.cs. Each script will serve unique purpose of managing the game. The GameManager.cs script will handle the UI (User Interface) of game over UI. There will be canvas and panel consisting of buttons corresponding to which UI will be shown. The game over will be indicated if all the lives that the player has been used up. The Boolean End Game will be enabled. I have set up a canvas which is an UI element. The canvas will have two buttons "Retry" which will restart the game. The game will be started from the very beginning giving the same number of lives which was given to the player at the start of the game. And the "Continue" button will lead the game to load the scene which is a start menu. In the canvas there will be text of GAME OVER and displaying the waves that the player has survived and the number of the enemies that the player has killed or destroyed. The pause feature will be handled by the pauseMenu.cs script. If the 'P' key is pressed the game will be paused and another similar canvas UI with

panel, texts and buttons will appear. There will be three buttons that will pop up: Continue (Which will take the player to the point where the game was paused), 'Quit' will take place to exit the game, and 'Retry' will allow the player to beginning from the start. The PlayerStats.cs script will display the life and the money and the amount lives of the player. It can be edited through the script and the unity game engine.

- To damage or kill the enemy I have imported some turret models. I have a script named TurretBehavior.cs to give the rotation capabilities to the turret head. Moreover, I have scripted to give the ability to the turrets to fire bullets. For the bullets there is a script BulletChase.cs which will allow the bullets to hit the target and destroyed upon hitting. The enemy object will be destroyed as well. The bullet will be a sphere game object which is round and black which gives us a cannon like feeling. The motive of the turrets will be it will hit the enemy object as soon as it reaches the range of the turrets. But the bullet will hit the enemy once it launched even if the enemy is not in the turrets range. There will be three types of turret models. Each of the turrets will be given unique ability. The turrets are imported prefabs FBX files from the unity asset store which was free of costs. First comes the standard bullet turret which will shoot bullet towards an enemy and deals some damage. The bullet is a sphere game object and have a black material which was edited by me. Each time the turret shoots the bullet will be instantiated. There is a variant of missile turret which will occupy same as the standard turret. There are slight differences. The bullet will be a missile prefab from the unity asset store. The missile will explode and damage multiple enemies in the range. The script turret behaviour will control the movements of the turrets. When the bullet is instantiated the bullet chase script will give the functionality to hit the enemy. The missile will follow the enemy as the simple bullet does, but it will explode and damage the enemies in range. The laser beamer turret will not have any bullet prefab to shoot. It will not shoot any bullet, but it will shoot laser. The laser will be a line renderer. He functionality of the laser will be slowing down a single target and it will not have to use indirectly the bullet chase the functionality is in the TurretBehavior.cs class. Each turret will cost some money. The standard turret will be $10, the missile turret will be $15, and the laser beamer will cost $10. The money will be deducted from the player stats.

- I have added some functionality to the keyboard for camera. The camera will move up, down, left, and right by pressing the keys on the key board W, S, A and D. The mouse will be able to zoom in and out in a certain range. If we scroll up the mouse or the mouse pad, then the camera will be zooming out and if we scroll in it will zoom in. After zooming in player can use the W, A, S, D to move around the camera to see the map.

- There is a shop added in a Canvas UI along with the money and the lives that we will be given at the start of the game. At the beginning of the game the player will be provided with some amount of money. Upon killing each enemy, the player will be given $30 to replenish their treasury to buy more turrets to stop the waves of the enemies. The shop will have 3 buttons attached with 3 images of the turrets. Each image will represent each turret. Upon clicking on the button, the turret corresponding to the image will be initialised on the terrain along with the functionality that is provided for the turrets.

- There is a building system to instantiate and drag the turret on the map freely. The script that will be responsible for this functionality is Buildmanager.cs. In this script we will be referencing a grid. The grid will be created by the unity game engine as well. The script will be attached with the grid. The system that I have used to give the functionality is called grid building system. There has to be collider attached to the game objects that we are going to instantiate and drag or move around the terrain. The grid sells will be holding the information for the game objects that is placed on the grid as a snap co-ordinate and upon clicking our mouse the world mouse position for the unity will be recorded by the script. ObjectPlace.cs script will allow to move the objects using mouse Down. To be more precise on right click of the mouse or the mousepad the game object which have the object

place attached can be moved. The whole shop will be a canvas which will stretch, and it will be relevant to the world position so that it will not move along with the camera.

- There is a functionality of adding level. Initially the game consists of 4 scenes. The menu scene will be the first page to come up when the player open or start the game. The menu scene will have a turret as the symbol of tower defence game, and it will consist of two buttons. Play button which will take the layer to the new scene and the quit button will allow the player to exit the application. The new scene which will be loaded upon clicking the play button is the level button. For now, there are only two buttons which will indicate two level. Upon clicking any button, the scene corresponds to the level will be load. The functionality to load the level scene is achieved by MainMenu.cs script. In the script I have used the UnityEngine.SceneManagement library to load the scene. Application.Quit() is the unity method which will going to be used to quit the application window.

- To load the levels, I have used the LevelScript.cs script to load the scene. I have used the scene management library as told before to load the level scene in the same way. There will be a concept for level clearing as well. There will be UI pop up if the number of the wave is cleared which is 3 at the moment. In three wave three types of enemies will be instantiated and player will purchase the turrets to destroy those enemies. After clearing the wave, the number of the rounds which is the number of waves will appear with a message level passed in a UI canvas and will have two buttons. One of the buttons named Menu will direct the player to the main menu where the play and quit button exist. The other button will take the player in the level scene to access and load the second or the following level.

- The traits of the second level can be changed within the game engine. We do not have to access the attribute that we have fixed for the enemies, turrets, UI, the scene, or any other game object with the functionalities. The speed of the enemies or the amount of the enemies that needs to be spawn or the amount of the lives and money all can be set inside the unity engine and can be set the level. But if we want to have a map with a different design, we have to change the terrain manually, have to set the way points again. Because ethe way points are stored in a form of array and we have to maintain the serial of the waypoints. Because the enemy object will target the waypoints depending on the array index. If the way point at the third index is placed at the last point of the path the enemy will target that one and it will travel there instead of targeting the nearest way point. We can use different algorithms to target the nearest waypoint but for the sake of simplicity I have just used the waypoints and set them on the map in subsequent order.

- All the variables in unity have been encapsulated and the approach of OOP is followed. However, in unity if a variable is private then in the inspector it cannot be shown as well as the developers or the users have to edit the attributes inside the game. To solve this problem, I have used a tag which id in third bracket, [SerializedField]. This will allow the script to display the attribute in the game. And the access modifier can be used in the game. If we want to serialize or display multiple attributes which is similar, or the game object have multiple attributes we can put the attributes in one class. As example, TurretFactory.cs and Wave.cs class. TurretFactory class have the turret game object and the cost of the turret. The class is using the tag [System.Serializable]. In this class we have SerializedField fields which will be shown in the unity. If we use a public or a SerializedField private field on the script like we used in the Shop.cs script. In the script we used three serialized field of TurretFactory class to add the game object and set the price. Wave.cs have the same approach as the TurretFactory class having the attribute fields of the enemy object, number of enemies that we are going to spawn and the rate of spawning. And we used the array waves in the wave spawner class to have the set of enemies in the array and spawn it. Both of these classes have required necessary access modifier. I did not find any exceptional chance to use inheritance in my development.

Chapter 5: **The Problems I have Faced**

Till this journey I have faced some problems. Some are still unsolved as I could not figure out the solution. Some problems were solved as well with the proper resources and some platforms such as stack overflow, reddit, Facebook. Some of the problems are:

- When I was setting up unity the project was not opening showing a message invalid project folder which was certainly corrupted. When I tried to look into the solution, I had two ways. First one was to uninstall the unity hub and re install it which did not work. The second one was to revoke the license and again activate it which was a failure as well. The solution that actually worked which I got from stack exchange that install another version of unity editor and create a new unity project with it which was success.

- I was having problem with the camera functionality. It was not reading my key press from the keyboard. Then when checked out Unity documentation I applied a different approach which was a success.

- Another problem occurred when I was about to push the project in my GitLab repository. Some unity file was not letting me to push the files. So, I had to configure the .config file to grant the permission.

- When I first created the map and arranged the waypoints my enemy prefabs were not following the path. It was moving randomly. The reason behind was the way points were not added in the array in a sequence. For that reason, my enemy prefab could not follow the path and was moving randomly. To resolve that matter I had to re-arrange the way points in a sequence and it worked.

- There is a bug in my building system. The bug is I could not restrict the buildings and stop the collision between the game objects. That means I wanted to create a building system that the game will not allow the player to overlap the towers or turrets to collide with each other or any other object. I tried to use collider, tags and other unity tools and libraries with control flow conditions but failed to do so.

- In my shop functionality the turrets that I buy always spawn in same point and overlap each other. I have tried to use the random position on the terrain, but the result always comes out the same.

- I could not provide the functionality of upgrade and sell. The reason is I could not get the game object that I click. I tried to use collider and tags as well as using loop to check through all the game objects but failed to do so. I have looked into other resources but could not find any solution. This led for me to provide a faulty version where I still want to try and solve the issue.

- I have faced some problems with my time management. I tried to complete everything on time however, failed to do so. I worked almost regularly even though I could not achieve some tasks (e.g. Tests cases, adding music, adding more visual graphics, adding terrain generator) to complete. This mismanagement of time made me realise there was a load of works needed to be done but from as much as I have time left it is infeasible for me to complete all the tasks.

Conclusion

In conclusion I can say that I had a pleasant, new and an exciting journey. I have learned many things in this game development project. This is a very different than any other part of the Computer Science world. I got to learn how games actually works and how it can be made.

I got learn how to use new tools, game designs, professional work ethics, professionalism. I have also learned that how hard the developers work in order to build a beautiful game. Though it takes a lot of time to build a visually astounding game along with no bugs. For the graphics using the tools like blender can be complicated and time consuming. Blender has a very steep learning curve. To make game with super high polygons like Call of Duty: Warzone can take years after years. And the functionality is quite complex as well. As well as to create and play a high polygon game the user and a developer must have a computer or machine with high configuration.

Game design, OOP and other patterns and software engineering approach can enhance the game performance. Also, reducing the polygons allows the game to run on low specification devices as well. With the help of game engine we do not have to worry about making game in different language(As swift is a language for iOS development) or do not have to have another complicated configuration. Unity does that for us. At the beginning things can be looking hard and scary but gradually it becomes very fun and it is easy to implement game design with unity due to have a lot of resources. All the unity editors works same but the UI might be a little different.

Bibliography

[1] Li, F.W., 2008. Computer Games.

[2] Asfarian, A., Ramadhan, W., Putra, W.A., Raharjanto, G.M. and Frisky, R., 2019. Creating a circular tower defense game: development and game experience measurement of orbital defense X. *Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control*, pp.249-258.

[3] Xie, J., 2012, July. Research on key technologies base Unity3D game engine. In 2012 7th international conference on computer science & education (ICCSE) (pp. 695-699). IEEE.

[4] Unity (2023) Unity Manual: Creating and Using Scripts. Available at: https://docs.Unity3d.com/2023.3/Documentation/Manual/CreatingAndUsingScripts.html (Accessed: 23 September 2023)

[5] Games, P., 2009. Plants vs. zombies. *PopCap Games*.

[6] 1. Xie, J., 2012, July. Research on key technologies base Unity3D game engine. In 2012 7th international conference on computer science & education (ICCSE) (pp. 695-699). IEEE.

[7] Winkler, F. and Barrett, C., 2011. Introduction to Unity3D (vers. 3.4).

[8] Geig, M., 2013. Unity Game Development in 24 Hours, Sams Teach Yourself. Sams Publishing.

[9] Haas, J.K., 2014. A history of the Unity game engine. Diss. Worcester Polytechnic Institute, 483(2014), p.484.

[10] Bertolini, L., 2018. *Hands-On Game Development without Coding: Create 2D and 3D games with Visual Scripting in Unity*. Packt Publishing Ltd.

[11] Blackman, S. and Wang, J., 2014. *Unity for absolute beginners*. Apress.

[12] Goldstone, W., 2009. Unity game development essentials. Packt Publishing Ltd.

[13] Zolkifli, N.N., Ngah, A. and Deraman, A., 2018. Version control system: A review. *Procedia Computer Science*, *135*, pp.408-415.

[14] Holopainen, J. and Björk, S., 2003. Game design patterns. *Lecture Notes for GDC*.

[15] Rautakopra, A., 2018. Game Design Patterns: Utilizing Design Patterns in Game Programming.

[16] Harmes, R. and Diaz, D., 2008. The Flyweight Pattern. *Pro JavaScript Design Patterns*, pp.179-195.

[17] Nystrom, R., 2014. *Game programming patterns*. Genever Benning.

[18] Cote, A.C. and Harris, B.C., 2021. 'Weekends became something other people did': Understanding and intervening in the habitus of video game crunch. *Convergence*, *27*(1), pp.161-176.

[19] Anderson, S. and Orme, S., 2022. Mental Health, Illness, Crunch, and Burnout: Discourses in Video Games Culture.

[20] Edholm, H., Lidström, M., Steghöfer, J.P. and Burden, H., 2017, May. Crunch time: The reasons and effects of unpaid overtime in the games industry. In *2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering in Practice Track (ICSE-SEIP)* (pp. 43-52). IEEE.

[21] Edholm, H., Lidström, M., Steghöfer, J.P. and Burden, H., 2017, May. Crunch time: The reasons and effects of unpaid overtime in the games industry. In *2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering in Practice Track (ICSE-SEIP)* (pp. 43-52). IEEE.

Appendix

Professional Issues

The professional issues that I want to talk about and which is quite similar to my issue as well during the project is the Crunch Culture.

To elaborate if the workload is massive along with a tight time limit, we can call it a crunch culture. As I was proceeding with my project I have faced many errors, bugs in my program, bugs in the tool, complex code structure or tough implementation etc. It takes significant amount of time to complete a game along with a team. Still there are a lot of work to do to build a game. The game must have colliders and a building system, an inventory enemy functionality, player restriction and many more complex attributes. All of these can be achieved by code. However, the code structure and the implementations are hard. It takes a lot of time for developer to achieve as I have worked on a grid building system.

In the grid building system we have a grid placed on the terrain and we have to keep the calculations of the position of each grid cell as we are going to use these cells place the objects that we need to build. The implementation for this is to get the mouse position and instantiate the game object to the position where the mouse is clicked. Or in a random position. Then comes the inventory and shop which is a very complex functionality to add. The developers have to keep a lot in mind such as the graphics, smooth performance, game designs and structure, game logic, game objectives etc. All of this sometimes needs a lot of time. And within a tight time limit it is quite impossible to complete the game. The developers have to work overtime in order to complete the game. This also hamper the work life balance.

Crunch is a period of time where developers work overtime. To add more the overtime is unpaid. This is due to complete the lagging projects. It has been done for many years now it has become a common phenomenon in the development of game industry. [18]

Because of the overload of the work pressure burnout, mental illness as trauma can be the result. Because of the moving economy most of the companies relies and put pressure on the developers for unpaid overtime. This also results the personal and family time. Sometime the developers have to work during the weekends.

During the project I have faced this kind of issue. Even though I was given a good time limit it was really hard for me to complete the project while doing other course works and assignments. This led me to work over night and on weekends. Game development is a long road and there is lot to do. To make a complete game there is a lot of effort, time and resources are needed. Because sometimes there are some errors occurs and it takes a lot of time to resolve it otherwise it causes bugs that makes the game unpleasant. As I was new to Unity, and I was learning while building the game it increased the workload and day by day it was becoming harder to manage until it became overwhelming to handle and caused me to burnout. As an example, to create the grid building system in my game it took a lot of time of mine due to the difference of my game type the resources were hard to find. There is another example of my work is my inventory. An inventory or a shop plays a significant role to keep the game functional to move the game forward. It took me while to have a shop inventory for my game as it was a complex functionality to give due to being a beginner in game development and using Unity as my game development tool. I also learnt C#

while building the game. So, I was not aware of all the functionalities of C# and Unity C# functionalities and how to use them. Therefore, I had to rely on the documentation and some other sources as stack overflow, Unity forum, social media as Facebook, reddit etc. It took me a while to process all the information and get used to it.

In the game industry, this issue is more common due to the ongoing achievement of the beautiful games. The industry is far more popular to push their developers to work uncompensated overtime and weekends. During the production or development period the developers or employees have to work 12 hours per day for weeks or months to complete the project. It has become so severe that nowadays the employees tend to believe this is normal and unchangeable which leads them to accept the dire situation. Because, In this tight schedule, it is not feasible to complete the project as the game has to be full of unique features, functionality and most importantly bug free. Even though the research shows that crunch time can even delay the deliveries or allow developers deliver low quality products.[20]

Working extra hours frequently causes sleep deprivation and can reduce developers' ability significantly. This also effects personal relation and mental health. [21]

In conclusion we can say that the crunch culture in game industry brings detrimental results. Even if we think that working extra hours might allow us to complete the product, it might lead the developer to burn out and reduce the efficiency of their workflow. The time management is a very important skill to resolve his issue. However, if the deadline is very tight schedule and the the workload is over proportional then no matter how good the developer is it is not feasible to finish the product perfectly within time.

User Manual

In this section I will be explaining to build the game from unity and how to play. The game cannot be played until we build an executable format. We also, cannot deploy our game into any platform without having the executable. We can run it within the Unity game engine but then will be more like testing it rather than playing the actual game. To build the game:

- First, we have to Install the Unity Hub from the Unity official website.
- The game was built with the Unity editor version 2023.1.9f1. We have to download it with the Unity Editor. We can download it after opening the Unity Hub followed by going to the install window on the left-hand side panel. After clicking the button install editor we can choose the editor version. [note: we can install the version after we have the game file on our local disk Unity will guide us to there. So, if any case the desire version is not found we do not have to worry.

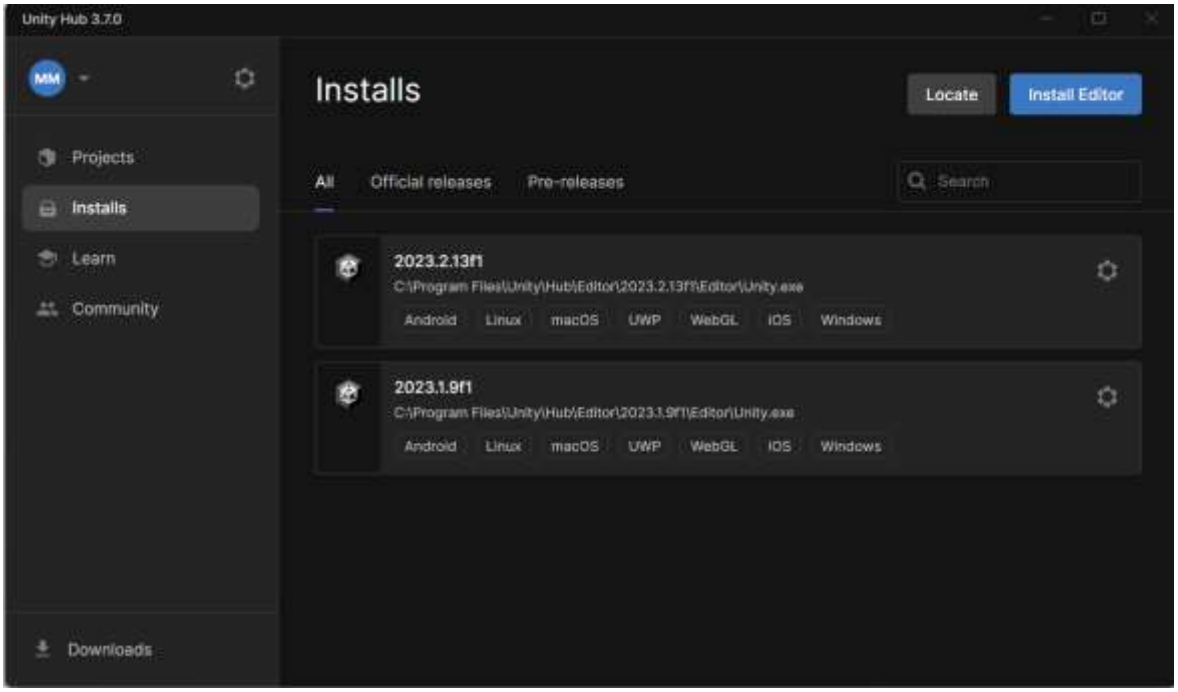

Figure 1. Unity Hub

- Then we can add the modules for our necessary builds. Such as Android SDK for android build, Windows, mac and Linux packages for the build for the cross platform builds. This might take some time as well.

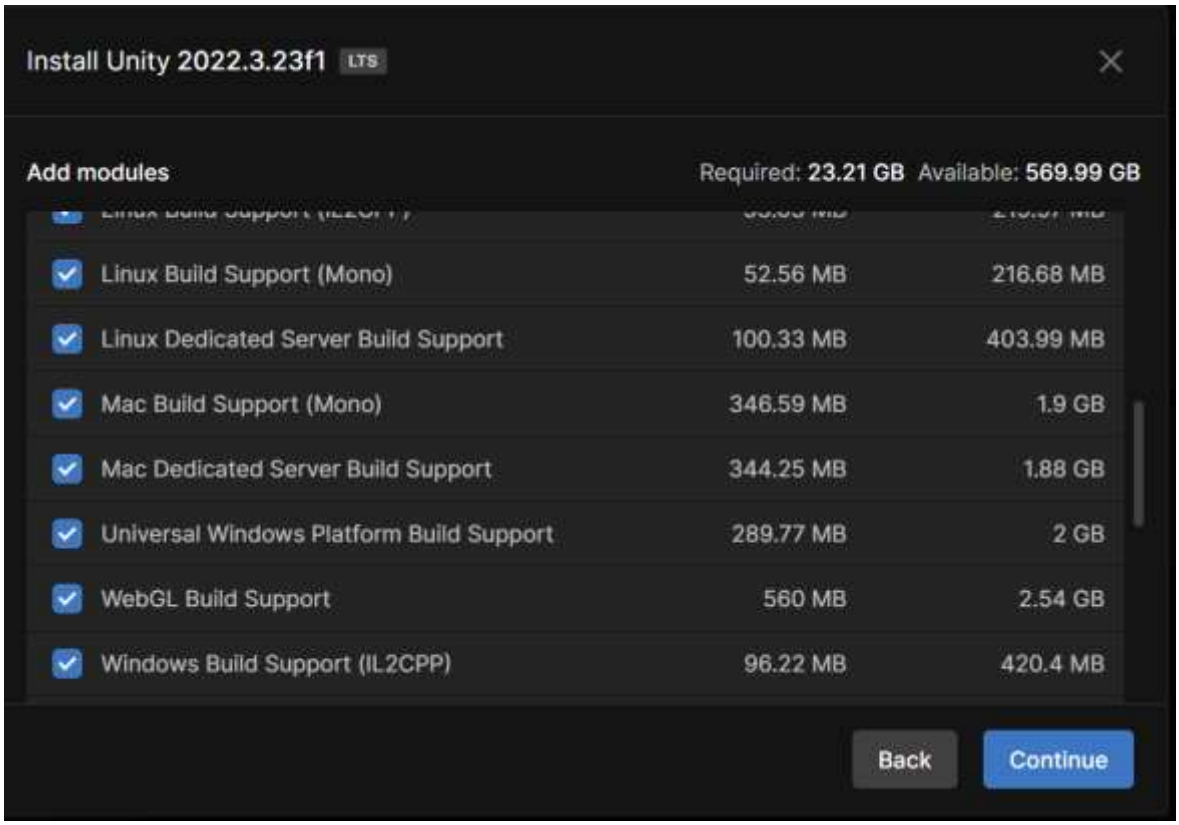

Figure 2. Modules

Then we can use git to have the repository in our local machine and have the game file. We can use

```
git clone https://gitlab.cim.rhul.ac.uk/wkis090/PROJECT.git
```

By doing this can have the project in our local machine and can open it with unity by clicking the ‘Add’ button on the top right corner of the project window [Shown in the Figure 3].

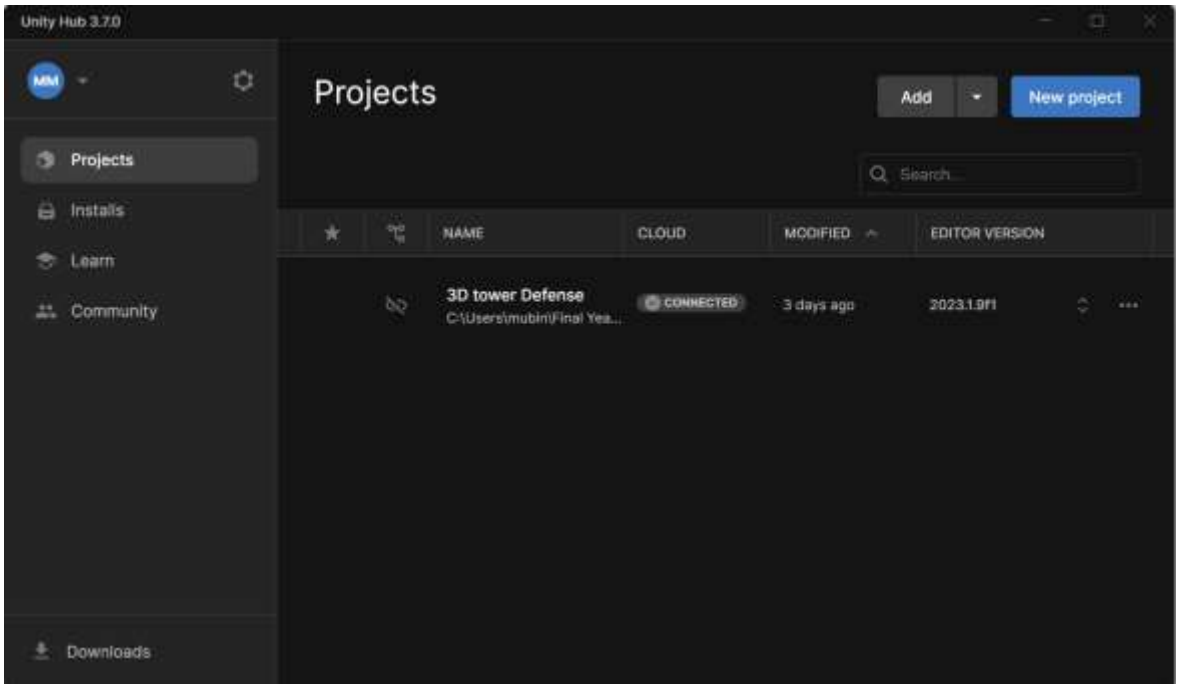

Figure 3. Adding the project

- We can open the file in Unity Editor by clicking it. After the file is open, we have to click on The file shown in the figure 4 then followed by build setting. Then We have to set the index of scene and press build. Which will allow us to build an executable for correspond to the platform we choose in the build setting (Figure. 5).

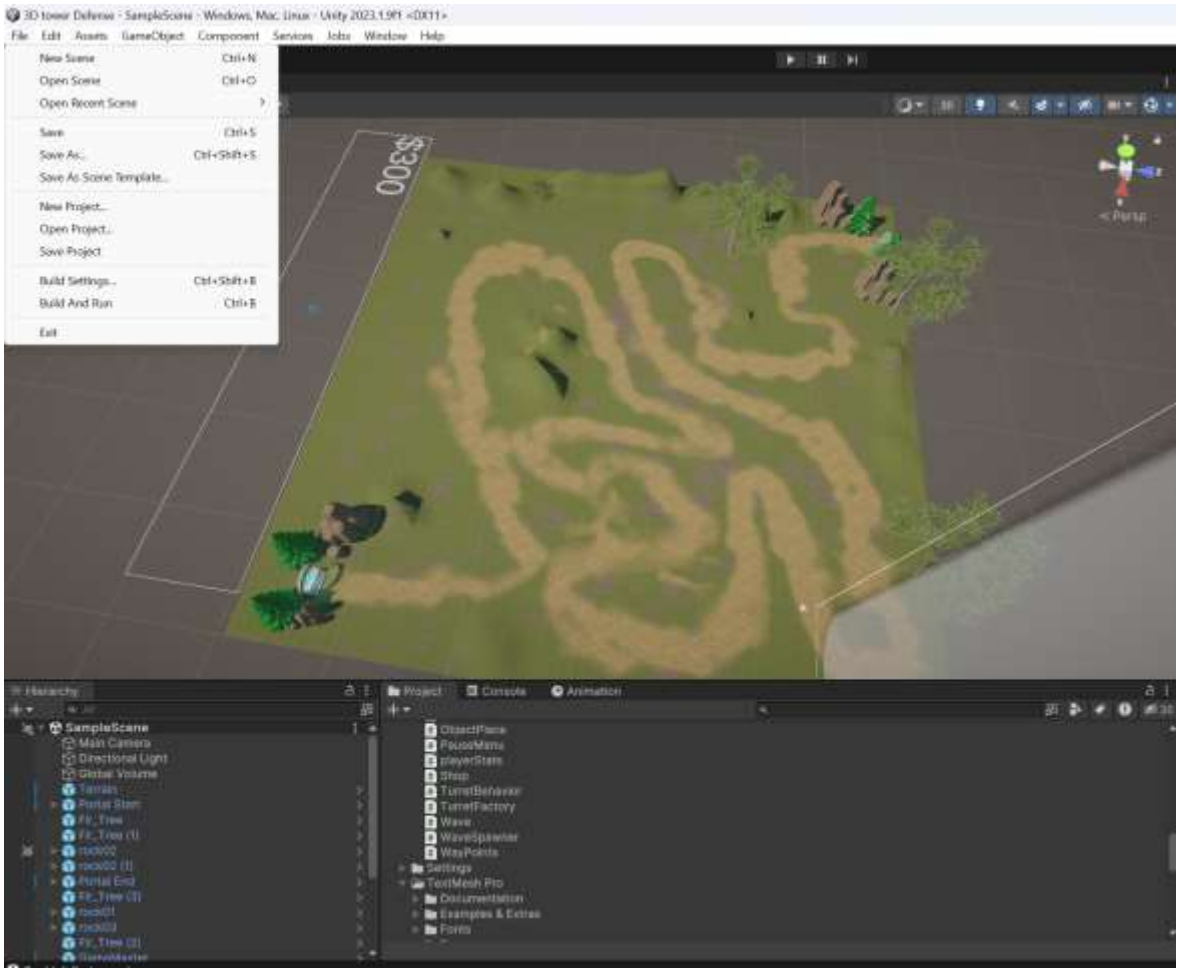

Figure 4. Build

To add the scene index sub consequently we can just the drag the scene and put it to the top if we want to load the scene first then the second scene at second and so on.

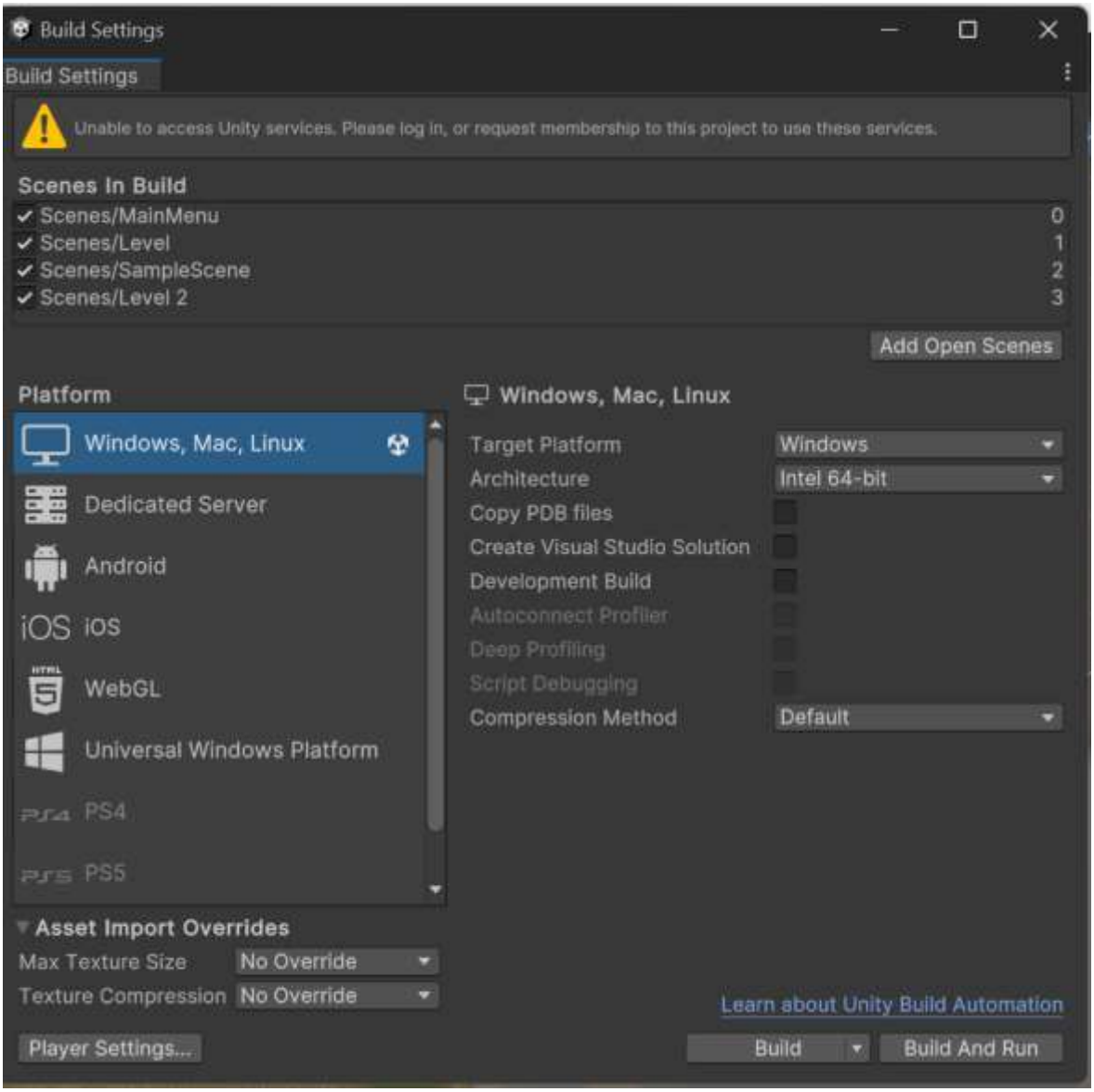

Figure 5.

- After setting the desired file location we can have our executable file we can now play our game.

How to play

Playing the game is pretty simple. There will be some amount of enemy which will spawn in waves. We have to stop the enemy and kill them with the turrets shown in the shop. We will be given some initial amount of the money, and we have to use that buy turrets. We will get money by killing the enemies. We can drag the turrets in the map with the double click on the mouse pad. We can move the camera forward, backward, left and right using W, S, A and D. We can zoom in and out by scrolling up and down on the mouse pad. There will be three variation of enemies. One of them will be a normal or balanced enemy. One of th other two will be fast but with less hit points

or HP and the last one which is golden will be the toughest but slowest enemy. Three types of turrets will be there as well. One will be the standard turrets shooting just a simple bullet, missile turrets will through missile which will damage multiple enemies in a range. Laser turret will slow down a single target with damaging over time. All of the three turrets will be displayed on the shop. We just have to click on the image in order to buy

Diary

27/09/2023

I met with my supervisor and showed my plan. Also, showed some of the work I did.

01/10/2023

Trying to play with the unity to get myself acquainted with the technology and be able to work with it. Also looked in the YouTube to get a visual and better learning about the unity.

03/10/2023

Joined some of the social media forums of unity developers in order to seek help if I get any kind of error.

05/10/2023

I have completed my project plan and submitted. Also, I have created the project file to start creating my game. Today I will add Universal pipeline to make my final game to run on every platform.

10/11/2023

I have switched from 2D to 3D game view. Since I cannot access my GitLab I cannot push this to my project. I am going to delete the 2D project and push the 3D project. I have made the terrain and designed with the path and some hills. I have added the texture from the unity asset store.

14/11/2023

Added the waypoints and added the script to give enemy a functionality.

15/11/2023

Added the enemy asset from the unity asset store. Established the portal and the start point for the enemy. I have given enemy AI to move forward to the end point. I have faced some error about the start node.

16/11/2023

Re arranged the way points to keep the enemy movements on the tracks and sorted the errors.

17//11/2023

I have added turrets to my game and gave the functionality through some script so that my turrets move the way the enemy is.

20/11/2023

I have created the bullets with some minimal effects. I have added the functionality to the turrets to instantiate the bullet and added the functionality to the bullet to hit the enemy object. I have faced a bit error, but I have resolved it as well.

21/11/2023

I have added the functionality for the camera.

04/01/2024

I have added the shop panel and added the buttons for the buildings. I will add a script and icons to make my shop more useful and functional.

18/01/2024

I have added the functionality to the shop in order to buy the turrets from the shop to build more turrets.

to make it work I had to work on the building syste.

22/01/2024

I have added the grid building system to add and drag the game objects while playing the game.

30/01/2024

I have solved the bug which was leading to create my turrets at the different place. Now it will be created at the point of my mouse.

01/02/2024

I have added the missile turret icon to the shop and I will be giving the functionality of the turret.

02/02/2023

I have added the hit effect for the missile. As well as used a function so that the missile will always face towards the enemy.

13/02/2024

I have added the money or the game currency in the game. Also solved some bugs as well. I have faced a bit difficulty during adding the game currency because in unity there are several variable types for a text to display. Besides I had a hard time to attach the script with the money canvas panel due to unity bugs.

23/02/2024

I have added the cost ui to the turret in the shop panel in order to see the turret cost in the game.

29/02/2024

I have added the laser beamer in the game. I have added it to the shop as well. I have used a different class turret factory and refactored the code for shop.cs to use the turret as the shared memory. So all the turrets that will e created via shop now will be sharing memory. This will reduce the memory cost and it will make my game much efficiet. We can say it is a little implement of the flyweight design pattern.

I have added the laser. The laser will be a line renderer which will follow the enemies when they are in range.

The laser will now damage the enemy overtime and slows it down.

02/03/2024

I have added the Gamer Over UI and the functionality to the Game. I have added a panel and some text to show the Game over message and the score as well. Then I have added a retry button with the functionality as well. In retry button the scene will be loading and re setting.

05/03/2024 - 10/03/2024

I have tried to add the upgrade and sell method with the UI. However, I failed. I could not make it work. In the editor the game is not working well with the game object.

14/03/2024

I have added the pause functionality with the UI.

15/03/2024 - 21/03/2024

I have added the health bar and the functionality to the game. i have used a green image and a image UI for the bar. I have given the functionality with the cs script and allowed unity to change the health bar on runtime if the enemy gets hit.

30/03/2024

I have added some additional functionality to the wave spawner. This will allow the game master to spawn three different variants enemies with different attributes.

01/04/2024

I have added another scene to serve as the purpose of level. This scene will be accessed by the Main Menu scene's button play. And from there the player can access the scene and play the game.

03/04/2024

I have added the functionality to clear the game levels. After clearing the level another level can be accessed through the Level scene.

05/04/2024

I have made documentation for the codes that I have written for better readability.