TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING

Khwopa College Of Engineering
Libali, Bhaktapur
Department of Computer Engineering

A FINAL REPORT ON ON
3D HORROR GAME IN UNITY

Submitted in partial fulfillment of the requirements for the degree

BACHELOR OF COMPUTER ENGINEERING

Submitted by

Abhishek Neupane	KCE074BCT006
Amar Sunar	KCE074BCT008
Bibash Rajthala	KCE074BCT014
Rojash shahi	KCE074BCT034

Under the Supervision of
Er.Dinesh Man Gothe
Department Of Computer Engineering

Khwopa College Of Engineering
Libali, Bhaktapur
2020-21

Acknowledgement

We take this opportunity to express our deepest and sincere gratitude to our project supervisor Er. Dinesh Man Gothe, for his insightful advice, motivating suggestions, invaluable guidance, help and support in undertaking this project and also for his constant encouragement and advice throughout our Bachelor's program. We are privileged to express our sense of gratitude to our respected teacher Mrs. Bindu Bhandari whose moral fiber,encouragement and brainwave was an immense support in completing this project. We are deeply indebted to our Head of Department Er. Shiva Kumar Shrestha for providing the most important advice and giving realization of the practical scenario of the project. In Addition, we also want to thank our friends and contemporaries for their co-operation and complience.

Abhishek Neupane	KCE/074/BCT/006
Amar Sunar	KCE/074/BCT/008
Bibash Rajthala	KCE/074/BCT/014
Rojash Shahi	KCE/074/BCT/034

Abstract

Unlike most other video game genres, which are classified by their gameplay, horror games are nearly always based on narrative or visual presentation centered on horror fiction and typically designed to scare the player. Usually, the classification of video games into genres ignores the narrative themes, which would include science fiction or fantasy games. Horror games is the only narrative-based classification with the narrative genre label used broadly for games designed to scare players. This document shows various approaches taken to give a realism experience to the user in a horror world with the help of different story line of a haunted house. We are trying to create and unique and advanced UI system in order to make the whole system unique and make player more comfortable with the situations along with he horror experience.

Keywords: *Horror Fiction,Narrative Based Classification,Narrative Themes*

Contents

List of Tables

List of Figures

List of Symbols and Abbreviation

UI	User Interface
TOI	Time Of Impact
FSM	Finite State Machine
HSL	Hue Saturation and Luminance
FPS	Frame Per Second
HDD	Hard Disk Drive
RAM	Random Access Memory
GB	Gigabyte
3D	3-Dimensional
CPU	Central Processing Unit
IDE	Integrated Development Environment
PhysX	Realtime Physics Engine

Chapter 1

INTRODUCTION

1.1 Background Introduction

Computer games and video games have become very popular in children and adolescents' life and play a prominent role in the culture of young people . Games can now be played everywhere in technology-rich environments equipped with laptops, smart phones, game consoles (mobile and stationary), set-top, boxes and other digital devices. From this phenomenon, it is believed that the intrinsic motivation that young people shows towards games can be combined with educational content and objectives into what Prensky calls "digital game based learning". Besides of an abundant appearance of games in young students life, game development technology has matured and became more advanced than before . Based on various existing game development software, the whole duty of game development process can be divided into several domains and roles such as game programmers, 3D model creators, game designers, musicians, animators, and play writers. [4] With the advancement of the game development though all these years people have worked through the various platform., In today's world unity has been one of the most prominent method in the game development scene. It is highly appreciated because of its features and the efficiency. With the use of the C programming language to interact with the various object and models, Unity has been the top choice in the indie game development. Its and highly advanced game engine with the clean GUI .The coding part of the game development can be done in visual studio, VS code or mono. Unity is a cross-platform game engine developed by Unity Technologies, first announced and released in June 2005 at Apple Inc.'s Worldwide Developers Conference as a Mac OS X-exclusive game engine. As of 2018, the engine had been extended to support more than 25 platforms. The engine can be used to create three-dimensional, two-dimensional, virtual reality, and augmented reality games, as well as simulations and other experiences. The engine has been adopted by industries outside video gaming, such as film, automotive, architecture, engineering and construction. Our game is intended to give a realism experience to the user in a horror world with the help of different story line. The player has to research an haunted house and find out the clues related to the house past. With the different keys player can get access to the different part of the house and find out the mysterious darkness that lies within the house. We are planning to create and unique and advanced UI system in order to make the whole system unique and make player more comfortable with the situations.The

horror experience take place in the horror world of the JUJU valley where there lies an haunted house in the middle of the dense forest which is haunted by the spirit of the girl who has cursed the place and it is our mission to unveil the dark secret behind the house and lift the curse.

1.2 Problem Statement

With the vast majority of features available there is also some lacking in the Unity. First of all we cannot create complex models in unity so for the game design so we must use an another software called blender. For our project due to the time complexity and the lack of knowledge about the blender we preferred the ready made models . Another disadvantages of the unity is while creating the animations so, the animation part become too complex in unity. So in order to create a proper animation we used an website called mixamo.com which automatically generates different animation from our model with the help of the artificial intelligence and machine learning. Unity consumes too much ram and CPU and sometimes crash on the low end pc. Its is not suitable for making the AAA games in unity. While creating a project in unity sometimes the inbuilt Physx system comes with some performance issues.we also had problems while implementing out scripts due to the compiling bug that unity causes sometimes. We also had to face the problems of infinite looping of the programs and the bottleneck due to the vat operations.

1.3 Objective

The main aim of this project is:

- To create an playable and realistic looking 3D horror game in unity.

1.4 Limitations

- The limitations of our project that for publishing this project we have to take license for the use of the models,audio,materials in a commercial way and we also have to give royalty to the owners of those products.

Chapter 2

LITERATURE REVIEW

Since the dawn of gaming industries many games has been made till date. However, there seems to be one genre that's seemingly never changed. Sure, its mechanics have "evolved" and become more refined. But for this genre, the essence has always been fear. This genre we are talking about is horror genre. Horror game emphasizes the player's desire to stay alive throughout the game. Once the player completes a challenge, the player's pleasure (or relief) overrides fear or anxiety; this pleasure provokes the player to continue playing the game. The horror genre has featured in games since its creation. Fear of the unknown, the known, violence, bloodshed, psychological torment, childhood trauma, the macabre, the uncomfortably ordinary – the horror genre has pretty much run the gamut in trying to scare you. Horror video games target evolved defence mechanisms by confronting the player with fright-inducing stimuli such as darkness and hostile entities. Going back to the text based horror game Zork (1980), a section of the game had player stumbling around a dark cave scared of being eaten by a Grue, a situation no one had experienced before in a game. The earliest 3D survival horror video game is widely considered to be the original Alone in the Dark, released in 1992 for the PC. Alone in the Dark paved way for many horror game franchises like Silent Hill, Resident Evil, 7th Guest, Phantasmagoria, I Have No Mouth and I Must Scream. Our game project "Haunted House" is inspired from all these horror games and the striking similarity between these horror games and Haunted House is use of Jump scares, horror graphics and scary ambience meant to fear the user. [3] Our game is not so elaborate like the franchise developed games but it would help in providing a conceptual knowledge on how the base of these games were made. It lacks VFX and perfect jump scares unlike those franchise based games. Unlike those horror games, Haunted House has no levels and the plot of game unfolds as the player moves forward by reading notes placed along the gates, doors, cemeteries and corridors. As long as all the notes are not read by the player, the game won't finish. The player is also challenged to find keys that unlock the path to new areas. What are the effects of horror games; how do they work, and why do people play them? Teresa Lynch and Nicole Martins of Indiana University looked at college students' [2] experiences with horror video games and found that about half of their sample (53 percent) had tried playing such games and been frightened by them. They also found that: horror games produce these fright responses by targeting our evolved defence system (evolution has shaped us to be easily scared by the dangers that threatened our ancestors); that there are predictable individ-

ual differences in how likely people are to seek out and be scared by horror video games; and that interactivity is crucial to these effects. Moreover, the researchers found that horror video games can have strong spill-over effects, causing disrupted sleep and increased fearfulness after playing.The researchers had 269 undergrad students complete online forms on their experiences with frightening video games. They were asked to indicate which games had scared them, identify the game stimuli that scared them,and list the kinds of fright reactions they experienced during and after gameplay. Most of the respondents (97 per cent) were 18-24 years old. The researchers used a combination of forced-choice and open-ended questions.Thus, our game project "Haunted House" provides a scary ambience to feel the fear.The pickup keys finding serves as a puzzle for the player. As one English proverb says "Killing two birds with one stone" [1], Haunted House will feature both the scary part and puzzling part so user who prefers blend of two genres will absolutely love this game.

Chapter 3

REQUIREMENT STUDY

3.1 Functional Requirement

3.1.1 Audio

The game features music and sound effects. The sound effect will be 3 dimensional and 2 dimensional both which is intended to give a player a realistic horror environment in the game .It is aimed at giving some nervousness and scary feeling to the player

3.1.2 Key System

The key system means basically how a normal key system would work. Each door is specified with the different keys and when picking up different key we unlock the respective door.

3.1.3 Player Movement

The player can move freely with the help of the guided controls throughout the world and can navigate around the map.

3.1.4 Door System

The door is designed with the little bit of the details in our game , The door has 3 states i.e locked, unlocked, movable. The door has to be in either one of the three phased for a moment. It also has various animations for these different states.

3.1.5 Light

Various lights are used in our game from the player torch to the various lights inside or outside the house in the world or even in the jump scares . Lights give the realistic feeling of the environment and are interacted with the help of the various scripts.

3.2 Non-Functional Requirement

These requirements are not needed by the system but are essential for the better performance of sentiment engine. The points below focus on the non-functional requirement of the system.

3.2.1 Maintainability

Our game is properly maintained for bringing on new stuffs to the game and maintaining it if it undergoes some kind of the bug, The project is well managed and changes can be made according the requirements.

3.2.2 Performance

The game is intended to run on the various devices throughout the world nowadays. The graphics are not too fancy so it must be able to run smoothly on various low end devices also. It will not use unwanted memory and clocks and cores of the CPU.

3.2.3 Portability

The game for now is only intended to run on the windows only but with the certain modification to the code and download some sdk we can import the project to the Android, IOS,linux,etc platforms.

3.2.4 Ease of use

The game is made with the basic UI and the person who is playing video games for the first time can also play the game without having any difficulty while controlling and moving. The game has some help interface and control guidance also.

3.2.5 Optimization

The materials and the models used in the game were of high quality so the game graphics and the textures were reduced to obtained to get the higher frame per second which enables the smooth gameplay.

Chapter 4

FEASIBILITY STUDY

The following points describes the feasibility of the project.

4.1 Economic Feasibility

The total expenditure of the project is just computational power.The requirements for the game development is easily available. The project will cost some money only if it is introduced in the market. The asset were gained from the free source so we didn't have to pay any money to the owner as it is intended just for the educational purpose only.

4.2 Technical Feasibility

Since the project is completely coded in `C#` using the various extensions such as unity code snippets,debugger for unity,unity tool, mono behaviour snippets so the project is technically feasible

4.3 Operational Feasibility

The game will come to life just after the graphical user interfaces are constructed and when audio is introduced, the player or user can feel the horror of real nature in the game. The end users will enjoy the game since it contains all the requirements specified and it can be updated regularly through newer versions.

Chapter 5

TOOL USED

5.1 Unity 3D Game Engine

Unity 3D is a game engine and complete integrated development environment (IDE) with an integrated editor, asset workflow, scene builder, scripting, networking and more. There are five main views used in the Unity editor to get all the work done, the project view, scene view, game view, hierarchy view and inspector view, all of which are explained in more detail below.

5.1.1 The Scene View

The scene view is one of the most used views as this is where all the game objects are placed and scenes for the game are built.

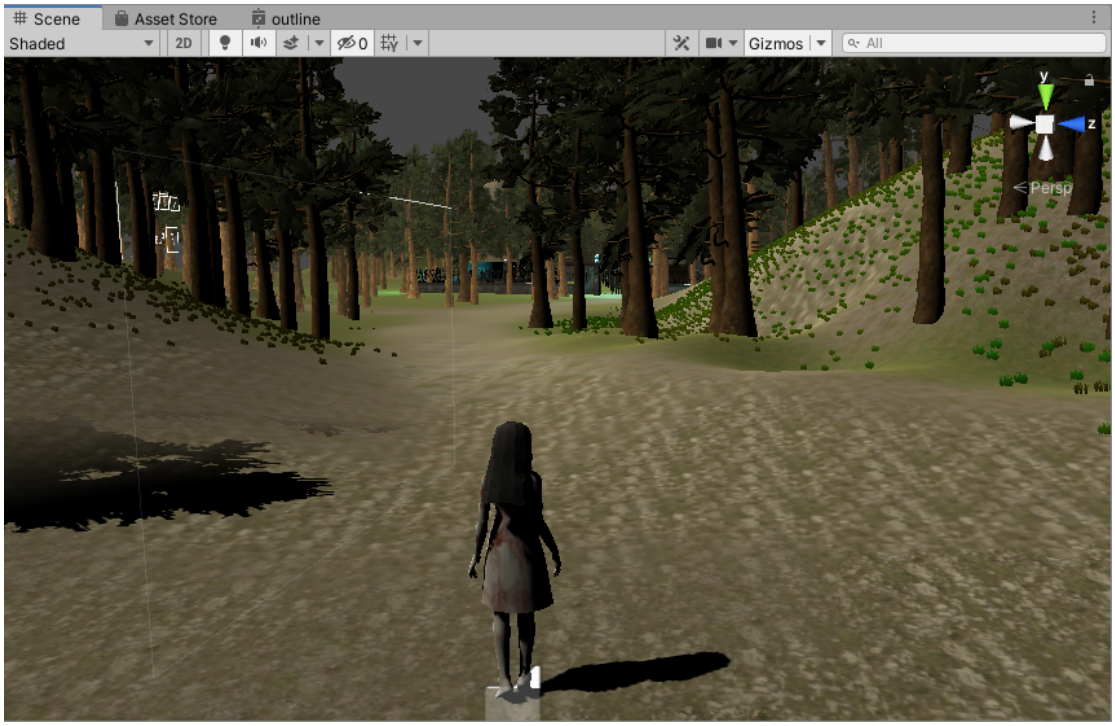

Figure 5.1: Scene view in Unity from Haunted House

5.1.2 The Game View

The game view is what user will see when the game is started. There are several options for this window. Across the top of the window there are several button/drop down menus which can change things from the perspective, full screen, and gizmos shown in the game view.

Figure 5.2: Game view in Unity from Haunted House

5.1.3 The Hierarchy View

The hierarchy view is where all the objects in the game can be created, accessed, grouped and manipulated to make the game. When the project is saved, the objects are saved in a scene file.

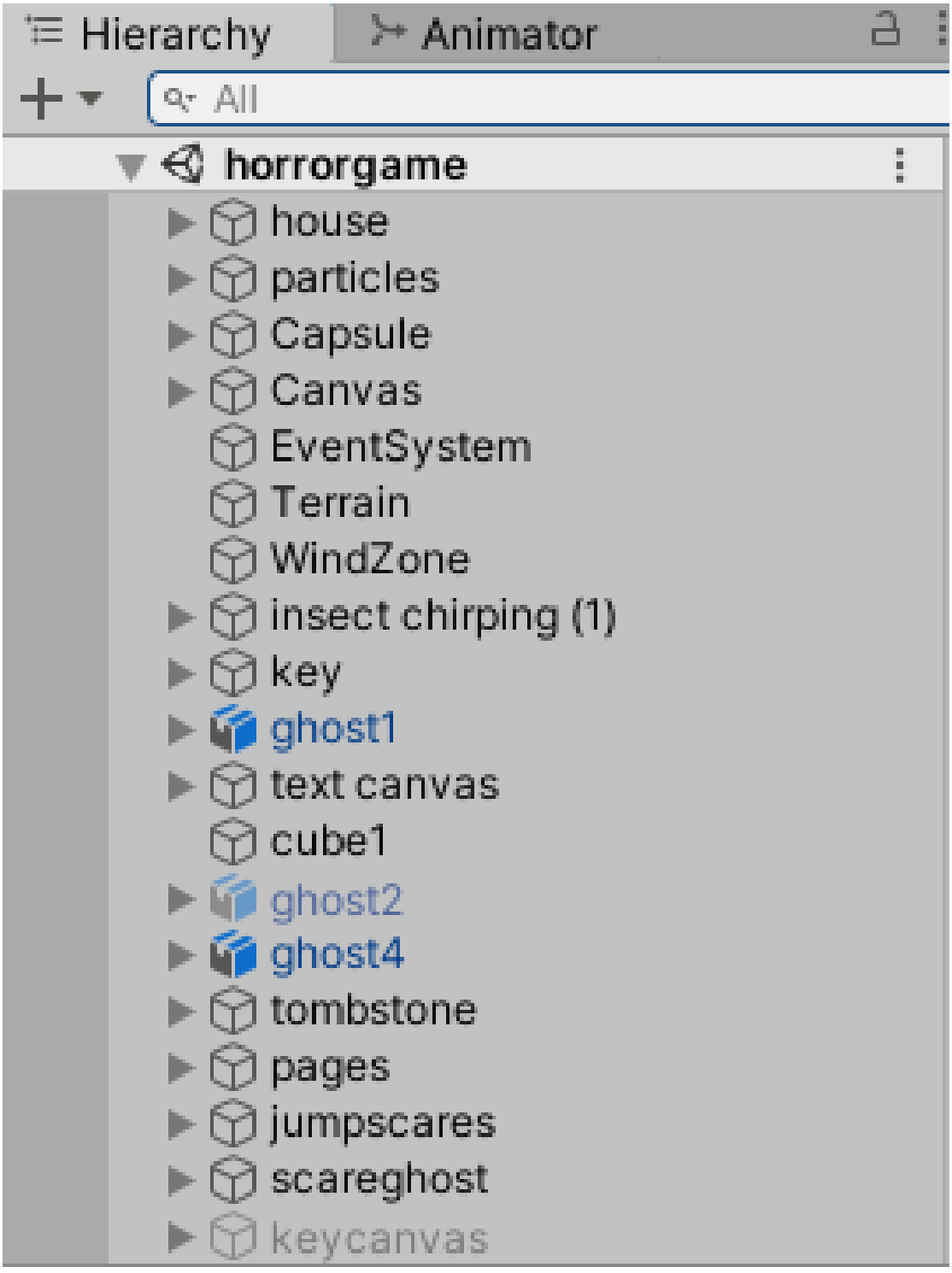

Figure 5.3: Hierarchy view in Unity from Haunted House

5.1.4 The Project View

The project view is where all the scripts and scenes are accessible from. This view is exactly like the file explorer on Windows or Mac and allows creating files and folders to help organize the projects assets.

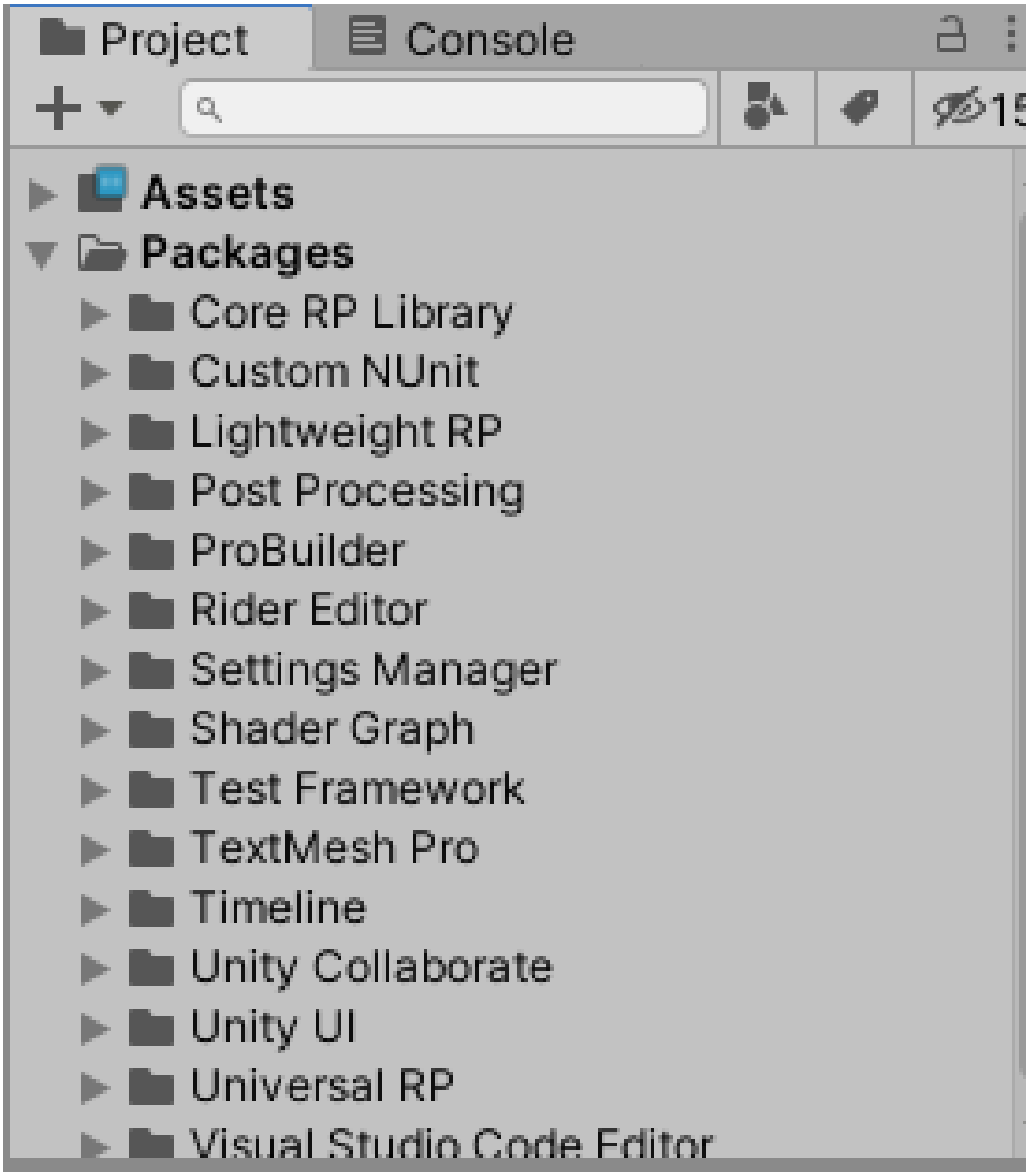

Figure 5.4: Project view in Unity from Haunted House

5.1.5 The Inspector View

The inspector view is where all the physics and properties of the objects are stored and accessed from. Every game object has a transform; this is what holds properties of the object such as rotation, position and scale. Other properties are the physics affecting the object, textures to load on the object and sound.

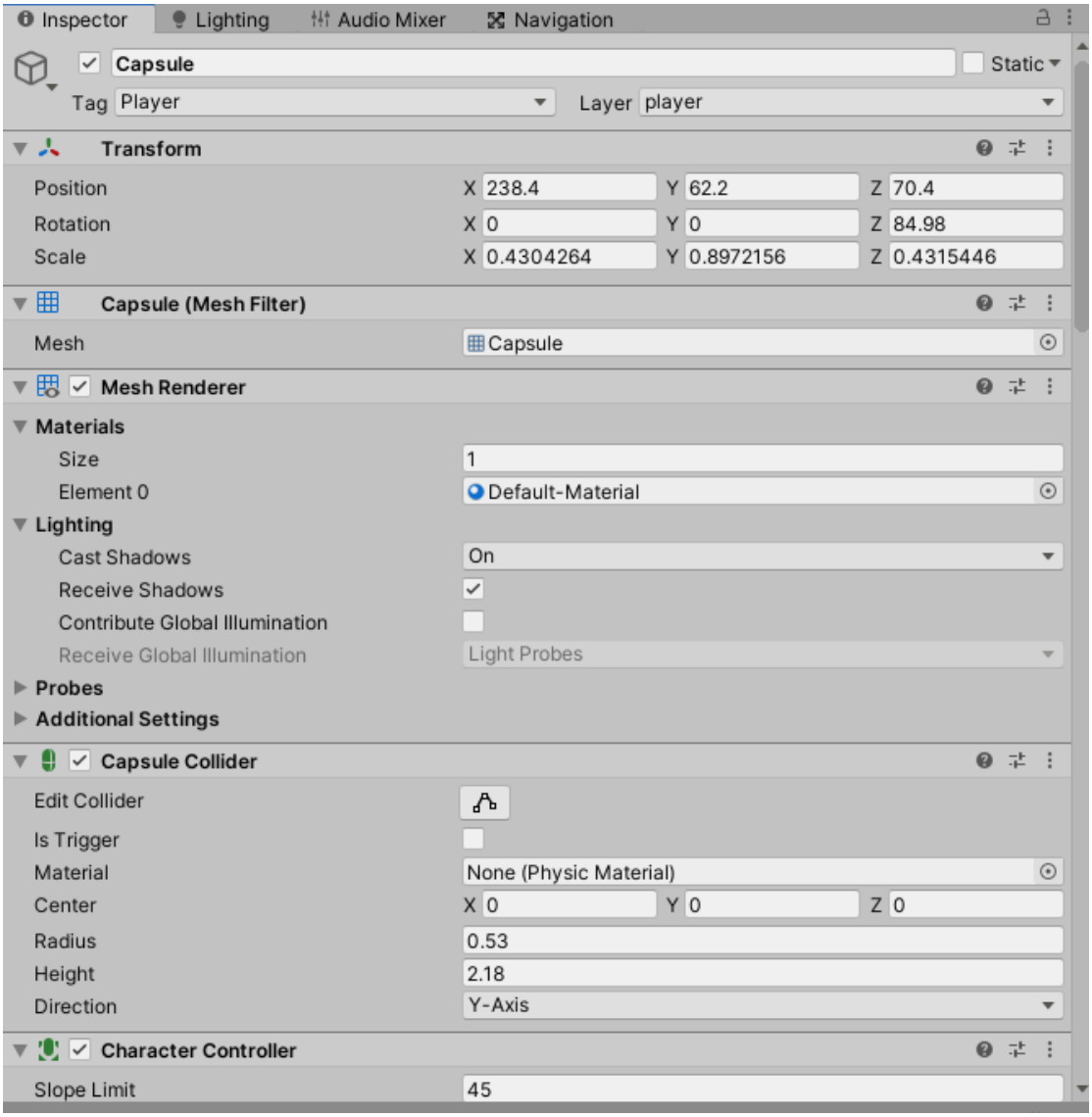

Figure 5.5: Inspector view in Unity from Haunted House

5.1.6 The Physic Engine

Unity contains the powerful NVIDIA® PhysX® Physics Engine. With this physic engine that is built into Unity many powerful things can be added to a game. Below is a list of several of the features usable in Unity.

- Hinges
- Character limbs
- Soft bodies
- Raycast
- Rigidbody
- Navmesh Agent

The physics engine is a great asset with all these features already built in. Without this a game programmer would need to create their own physics engine. Making a physics engine is an entire project in itself. Unity is truly an all-in-one game engine designed to create the next great game.

5.2 Adobe Photoshop CC 2019

Adobe Photoshop is an image processing tools used for the manipulating images and generating the textures.We used photoshop for the many purposed which are given below.

5.2.1 Grass and Textures

we used the photoshop in order to create the grass textures that were used in our game environment. With the help of the knowledge and work related to the depth of field and various channels of our picture we were able to create a realistic looking 3d grass overlay textures which would even work with the unity physics system and wind zone. we also created the regular grass and gravels textures that can be painted on the land and make it feel realistic with respect to the real life. We also created the alpha channel for the fire in the photoshop and used it later on the Particle system to give it a realistic fire effect in unity

5.2.2 Buttons

we also created all of our UI elements through the photoshop.We made those pictures with the help of the text and the rectangular tools which brings vibes about the traditional way of playing a game with the classic UI elements that guides the player within the game . We created E to interact textures which can be seen in the game in Unity.

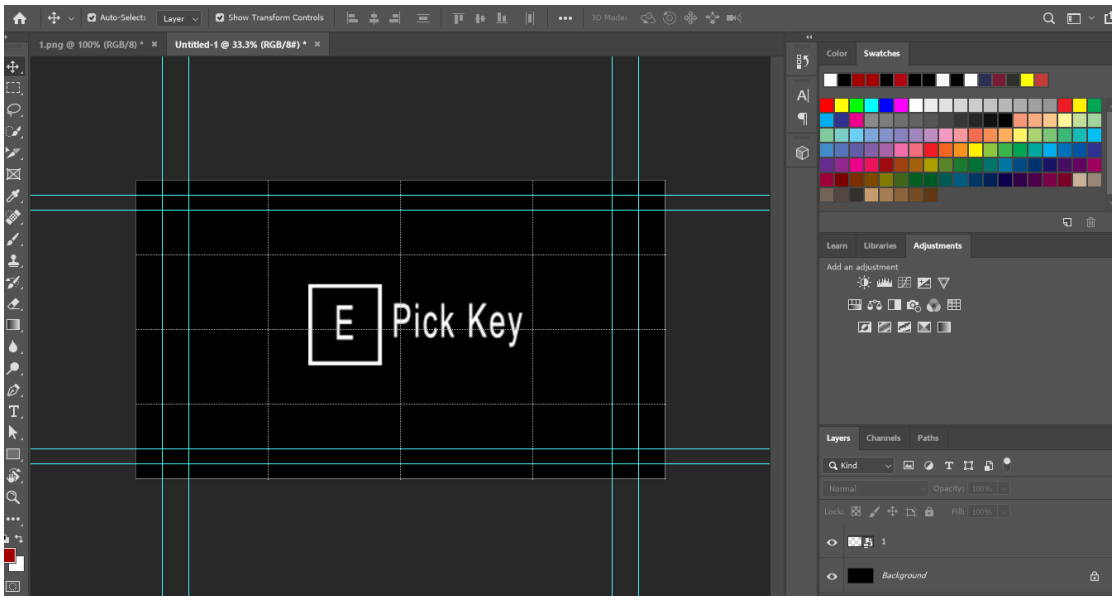

Figure 5.6: Buttons in Adobe Photoshop CC 2019

5.2.3 Notes

we also created all of our notes in the game from the scratch we also created the realistic looking paper texture in unity and wrote all the notes down manually through the text tool after the idea of the game was generated.

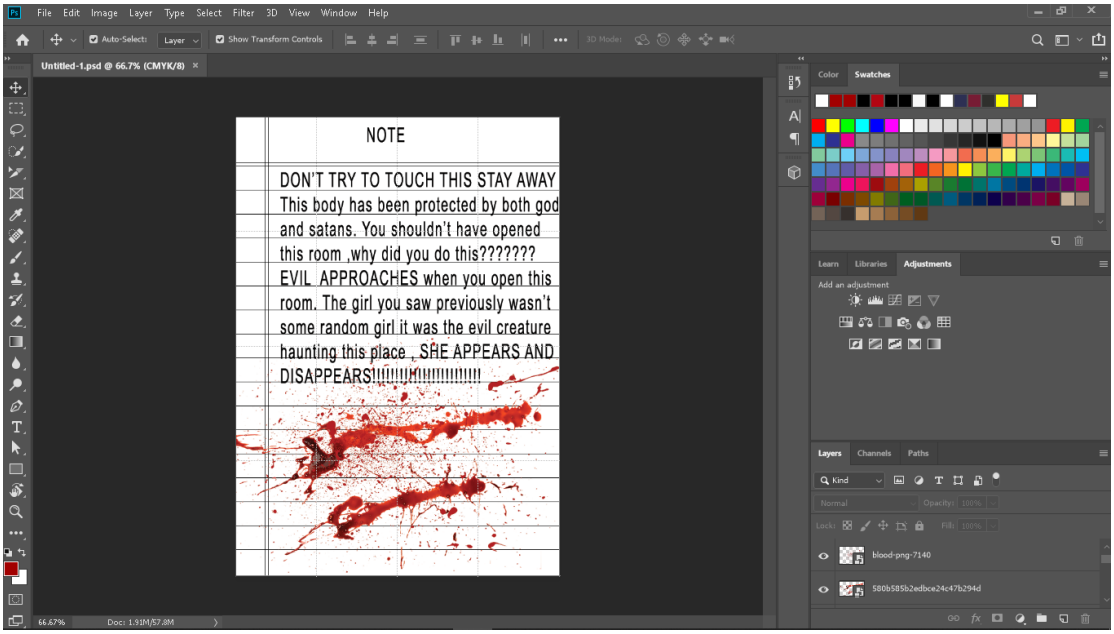

Figure 5.7: Notes creation in Adobe Photoshop CC 2019

5.3 Visual Studio Code

Visual Studio Code is a free source-code edit made by Microsoft for Windows, Linux and macOS. `C#` sharp extension must be installed from the Visual Studio marketplace to run C sharp programs in Visual Studio. Visual Studio Code uses the power of Roslyn and Omnisharp to offer an enhanced `C#` experience.All of our scripts are created in the visual studio code and it is linked with Unity with the help of the external editor function for the Unity.

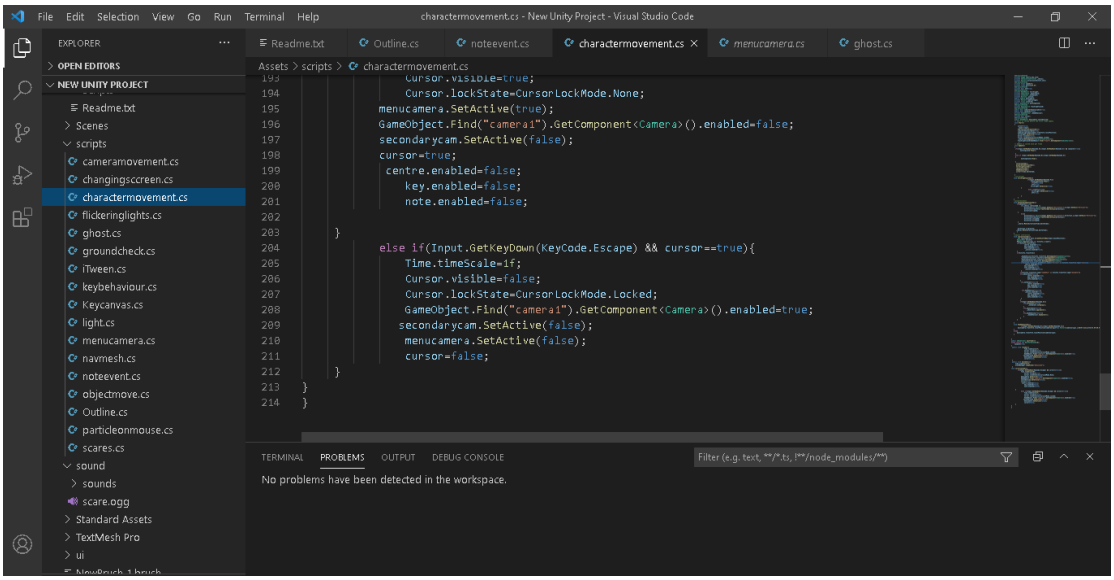

Figure 5.8: Scripts in Visual Studio Code

Chapter 6

SYSTEM DESIGN AND ARCHITECTURE

6.1 Class Diagram

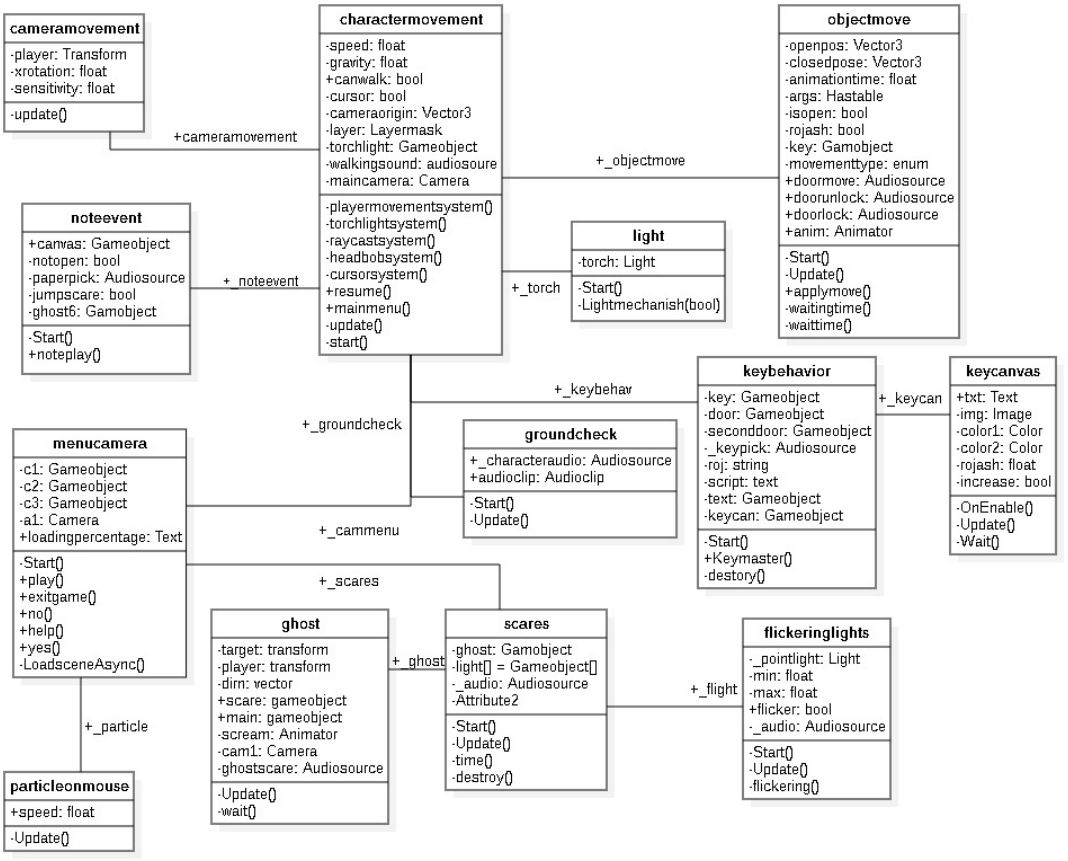

Figure 6.1: Class Diagram For 3D Horror Game

6.2 Sequence Diagram

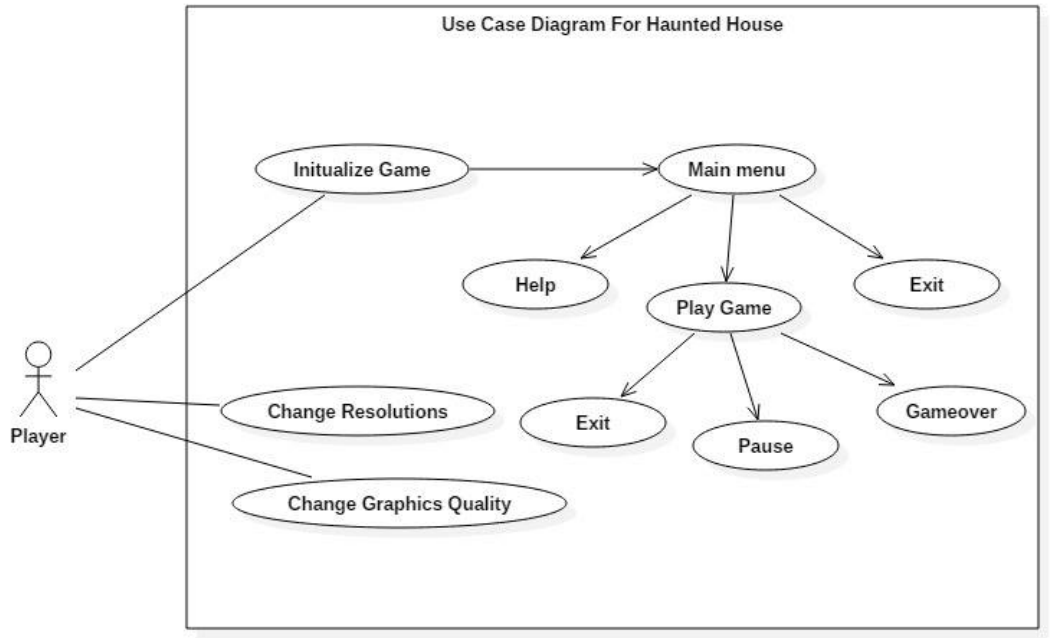

Figure 6.2: Use Case Diagram For 3D Horror Game

6.3 Activity Diagram

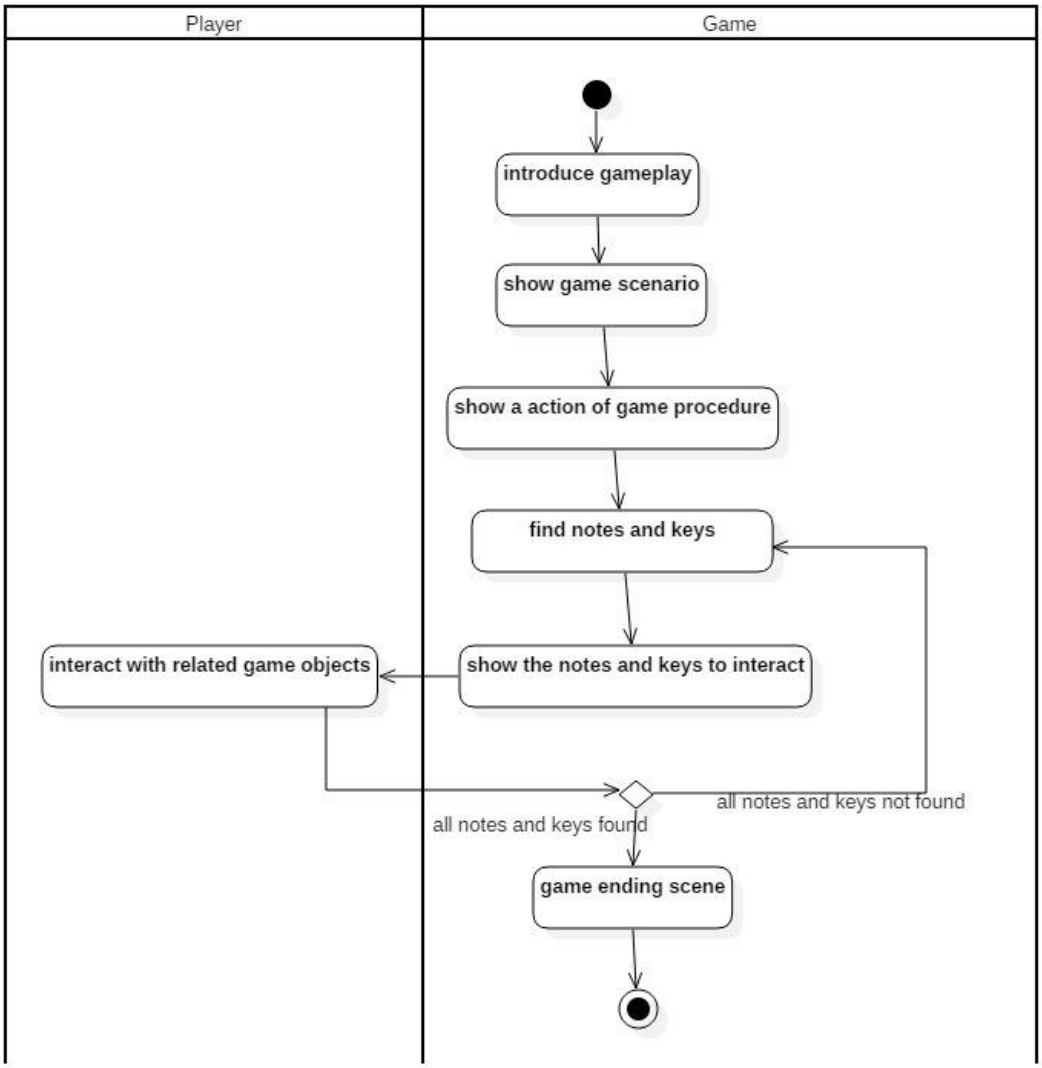

Figure 6.3: Activity Diagram For 3D Horror Game

6.4 Data Flow Diagram(DFD)

6.4.1 DFD0

Figure 6.4: DFD0 For 3D Horror Game

6.4.2 DFD1

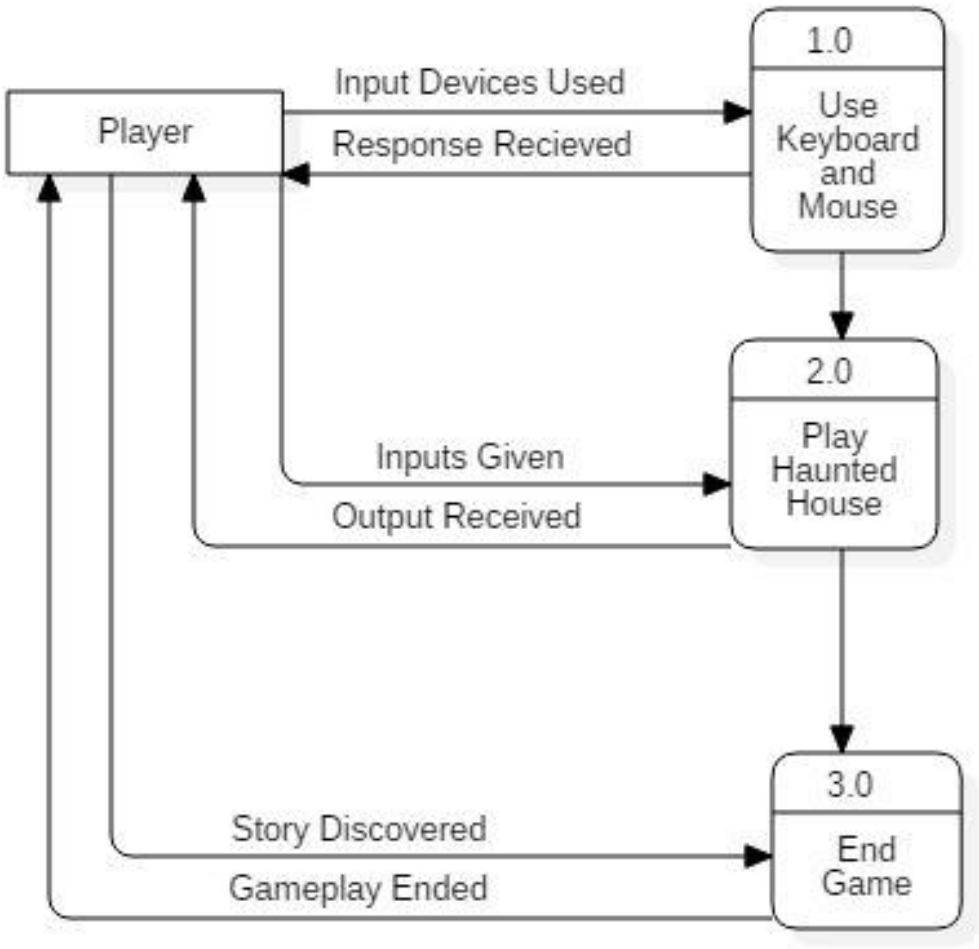

Figure 6.5: DFD1 For 3D Horror Game

6.5 Sequence Diagram

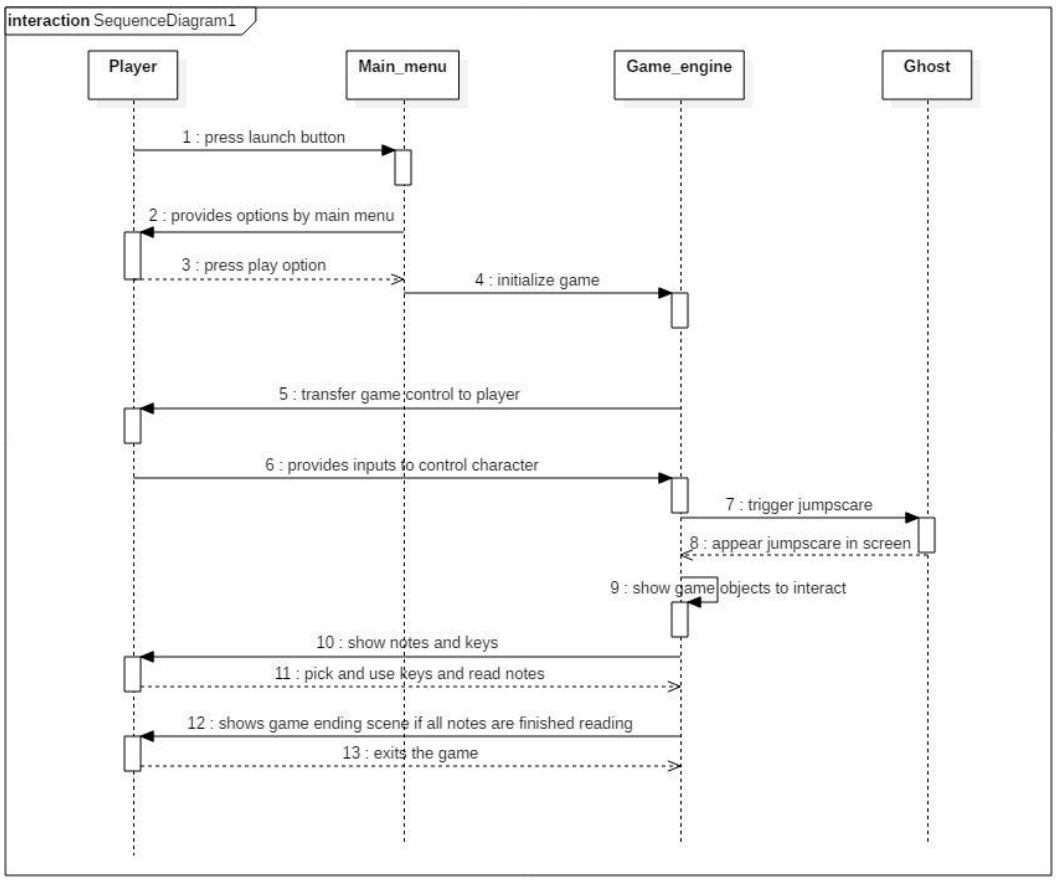

Figure 6.6: Sequence Diagram For 3D Horror Game

Chapter 7

METHODOLOGY

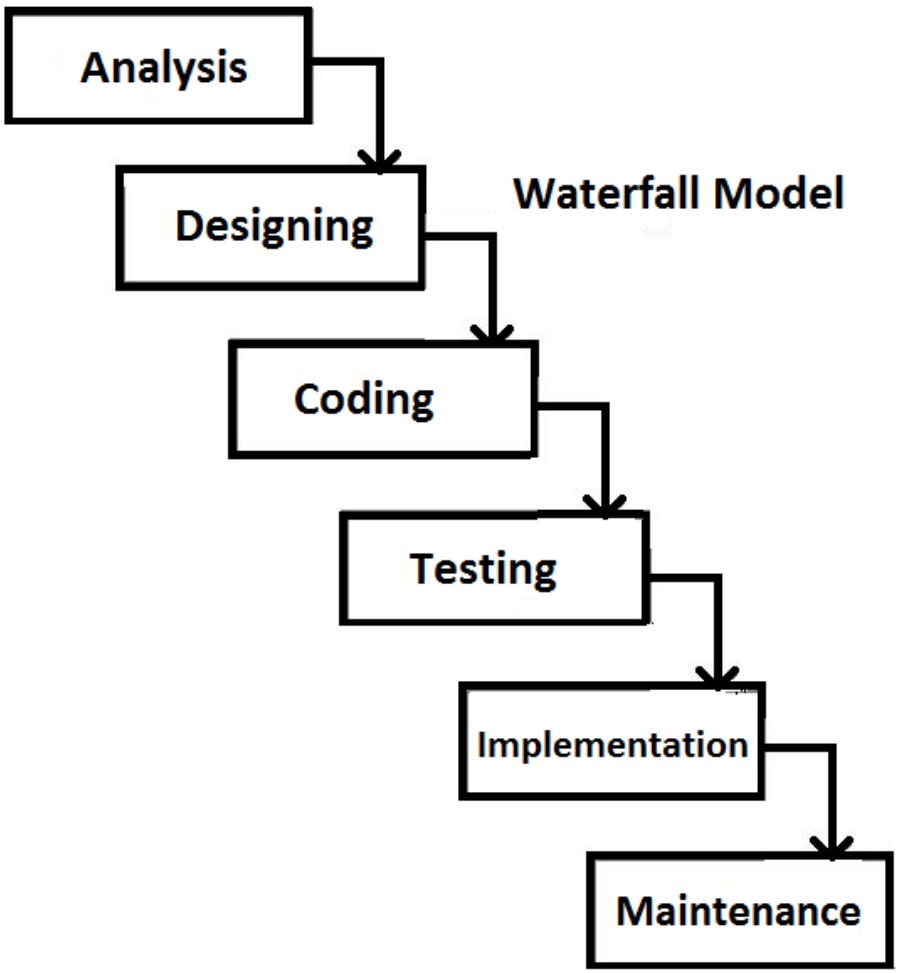

Figure 7.1: Waterfall Model For 3D Horror Game

For the development of our project we decided to move forward with the waterfall model as it was found to be the most suitable for the development of small scale games like ours which don't involve the marketing and risk analysis. The requirement also don't changes and all the materials and models are analyzed first only then the further work is done

7.1 Analysis

The overall project outline and demos were watched and we all calculated all the assets, models and texture we might have to use for our game. we also draw the scratch of horror game in paper. We also did the cost estimation whether our project need any investment or not. A lot of game demo on Youtube inspired us for our project.

7.2 Designing

For the designing of the environment we took the help of the various already made assets from the unity assets store and inbuilt unity features.

7.2.1 Terrain Design

For the design of the world and environment we used the Unity terrain System. The Unity Editor includes a built-in set of Terrain features that allowed us to add landscapes for our game. In the Editor, we created Terrain tiles, adjusted the height and appearance of our landscape, and added trees or grass. At runtime, Unity optimizes built-in Terrain rendering for efficiency.

7.2.2 House Design

The house was designed using the Unity horror pack 1 and 2 which you could easily find in the Unity asset store. The prefabs are there and we have to manually add the collider components to each and every part and we create a house by joining all those prefabs pieces by pieces.

7.2.3 Animations

The game objects used in our games like ghost model needed to be bring to real life using the animations so we needed to create the character rigging for the animations. So for the animations we took the help of the website name Mixamo.Mixamo also provides an online, automatic model rigging service known as the Auto-Rigger for creating animationslike walking,running,jumping etc.

7.2.4 UI Implementation

For the implementation of the UI we used the unity particle and terrain system, in order to create the fire in the main menu scene we used unity particle system and adobe photo shop cc to create the alpha map for the fire. we used several cameras swapping method in order to keep changing the position of the camera by clicking various buttons and for the pause menu we used the same approach. we also used terrain system and windzone to get the realistic environment effect.

7.2.5 Audio

Sound and audio are the most important part of the game for which we used the sounds from a open source websites named freesound.org which were executed by the audio source in the unity and we used the event for the sounds with the help of the codes.

7.2.6 Particle System

it is not possible to create the various particles in unity with the help of the codes alone. so unity particle system and shader graph are the extensions for the unity to create some stunning realistic looking objects and surroundings. In our game we created fire and white fog with the help of the unity particle system

7.3 Coding

For the code implementations we use visual studio code where we write our code and link them with the objects using the help of the hierarchy tab provided by the unity. we link the different scripts with the help of the handlers which are used to link 2 `C#` codes in unity. The various physics system were introduced like raycast , colliders , rigid body, light, inverse kinematics in order for our demo to make it work as the real life environment . Plugin named Itweens was used in the doors and drawers for their movements. Below are the list of scripts Used in our Project:

- **Cameramovement:**
  deals with movement of camera according to mouse movement in x an z axis.
- **Charactermovement:**
  all the player functionalities like movement, interact and use object with the help of keyboard.
- **Flickeringlight:**
  flicker light in some jump scares.
- **Ghost:**
  movement of ghost using AI or manually.
- **Groundcheck:**
  change the footstep sound according to nature of ground.
- **Keybehaviour:**
  unlock the respective door when key is picked.
- **keycanvas:**
  UI for the key pickups and door unlock.
- **Light:**
  flashlight system for player.
- **Noteevent:**
  read note and close it.

- **Objectmove:**
  movement of door and drawers.
- **Outlines:**
  shaders for the key.
- **Particleonmouse:**
  particle follows cursor for cool effect.
- **Scares:**
  triggering jump scares.

7.4 Testing

For the Testing of the codes we use the various test cases for this which are given below

Test Case ID	**Test Scenario**	**Test Steps**	**Expected Results**	**Actual Results**
TST01	Check Player movement with keyboard	Press W,A,S,D	Player should move in different directions when keys are pressed	As expected
TST02	Check POV movement along with movement of mouse	Slide mouse up,down, right,left	Camera should rotate as per the movement of mouse	As expected
TST03	Check the nature of ground and change footsteps sound	Move player on different surfaces	Footsteps Sounds should change when walking on different surfaces	As expected
TST04	Check whether the flashlight works or not	Press F to turn on and off the flashlight	Flashlight should activate and deactivate respectively	As expected
TST05	Check whether the object is interactable or not and raycast is working or not	Align cross hair with the objects within 2 metres	It should detect object and activate the respective UI box,image or text	As expected

TST06	Check Whether Button are responding according to the corresponding script	Press the different buttons on the game scene	Button should perform different function like changing scene, displaying text, resume game etc	As expected
TST07	Check Whether the note system is working or not	Press E on keyboard	When note is detected the Canvas for the note must be activated and player movement should be stopped and vice versa	As expected
TST08	Check the key system	Press E on keyboard	When note is detected by raycast Key should be picked and the respective door should be unlocked and the keypickup animated UI should be displayed	As expected
TST09	Check if the door is animating	Press E on keyboard	When door is detected by raycast and when the respective key is not picked the door should produce the locked sound and perform locked animation but if the key is picked it should produce unlock sound and then open	As expected
TST10	Check the player collision detection with the jump scare plane or object	Move player in the zone of the jump scare object	When the Player collides with the jump scares object it should activate the respective jump scares	As expected
TST11	Check whether the light are flickering when the certain jumps cares event are triggered	jumps care must be activated	The light must flicker when the paranormal object is nearby	As expected

TST12	Check the Ghost movement	Ghost object must be activated	ghost must perform the path finding and perform the respective animation and deactivate when it's object is completed	As expected
TST13	Check whether the particle like fire, volumetric fogs are working or not	Load the scene	Realistic fire particle and fog must be activated	As expected

Table 7.1: Test Cases

7.5 Implementation

Foe the implementation of our codes we have to link each script to it's respective game object and we also had to implement the various game dymanics for our game to follow physics

7.5.1 Common Game Mechanics

Every game that is played today is composed of some very common game mechanics: path finding, collision detection and input. These game mechanics have been around now for decades and have been improved on throughout the years. Here a few very common game mechanics are explained in more detail.

7.5.2 Raycast

A raycast is, essentially, a ray that gets sent out from a position in 3D or 2D space and moves in a specific direction. Unity 3D has built-in functions that can be used to implement a raycast in your game. In our game raycast is used to detect the the object whether it is key,door or note. Raycast origin is always at the centre of the player and applies on the range of the 2 metres.

7.5.3 Character Controller

A CharacterController allows you to easily do movement constrained by collisions without having to deal with a rigidbody.A CharacterController is not affected by forces and will only move when you call the Move function. It will then carry out the movement but be constrained by collisions. In our game we deals with the various vectors and inputs and then we move the character in the standard direction with the help of the implementation of the codes.

7.5.3.1 Collision Detection

The simplest definition of collision detection in relation to games is to determine if two rectangles in the same 2D or 3D space are overlapping.We are using the box, circle,polygon, capsule colliders etc for allowing our object to not pass through some layer or object and we also use those collider detection while creating a jump scare where the certain part of the object is intended to produce jump scare and scary sound effect when player collides with it. Collision Detection uses a Time Of Impact (TOI) algorithm to compute potential collisions for an object by sweeping its forward trajectory using its current velocity. If there are contacts along the object's moving direction, the algorithm computes the time of impact and moves the object until that time.
In Unity there is a method already created to help any game enthusiast create a game involving collision detection. The method is called OnCollisionEnter(Collision). In order for this method to work both objects in Unity need to have either a collider attached or a rigid body on the object. The method reports information such as points of contact, velocities, impact speeds and many other collision statistics.

7.5.3.2 Finite State Machine

A finite state machine at the simplest form is a model of how a system or a game will behave.Depending on the input from the player the state of the game can change. It is just the indication how the game will undergo change from one scene to another scene. In our game we used the scene management for the changing of the scene from menu scene to game scene when player choose the play option from the main menu and can also return from the game menu to main menu by pausing the game and selecting main menu. This is a resource consuming process.

7.5.3.3 Timers

There are many uses for timers in a game. It would be very surprising to find a game that does not use a timer or time in some fashion. There are countdown timers, count up timers, cool down timers, duration timers and timer based score.Co-routines are used to create the delay for the certain events and Time.timescale are used to create the timers and pause and resume of the game.

7.5.3.4 Path Finding

The general definition of path finding is plotting a path from a start point to an end point, done by a computer program or algorithm which is applied to a graph. In many cases the shortest path is the subject of interest to find. In the case of video games it is the same except it is done for a character or group of troops and it plots a path around obstacles on a map. In our game we add the navmesh agent to the ghost object and we set the destination for the ghost in the opening part of the game and the ghost follows the shortest path using the A* algorithm and we can also change the max slope ghost can climb,speed of the ghost etc.

A* Search Algorithm:

The A* algorithm is a spitting image of the Dijkstra's algorithm and works exactly the same except it adds an approximate distance to the end node as part of the weight system. This approximation is done by heuristic (a method where trading optimality, accuracy or completeness for speed when the tradition problem solving methods do not work). Using this method allows the algorithm to eliminate longer paths based off this approximation, in turn speeding up the resolution of the shortest path. Using this heuristic approach makes this algorithm faster than the Dijkstra's algorithm. The side effects of this approach are as the value of the heuristic increases A* examines fewer paths but does not guaranty an optimal path. For video games this compromise is not a problem and is why this is used over just the Dijkstra's algorithm. Whereas in routing protocols a guaranteed path is needed and the A* algorithm is not used.

7.6 Maintenance

7.6.1 Post Processing

Our game not only should work smoothly and efficiently but it must look good too so for this we used the post processing system provided by the unity which helps us to enchance our game, the features involves like ,ambient occlusion ,fog ,color grading ,vignetting ,HSL ,depth of field and motion blur

7.6.2 Optimization

The model we used in the unity were highly polished and details which is good but we also had to create a project for the low end pc so we decided to reduce the quality fo the same along with resolutions,shadows and the textures were reduced with a compression method equals to 10 times less size than the original size. At medium settings we were able to see an FPS boost from 25-45fps to 50-90fps.

Chapter 8

DISCUSSION

8.1 Evaluations

From our project we evaluated that Our project require more manpower to make a marketable product i.e only game developers are not needed along with them we needed script writers,model designer,animators, shader graph coders,sound designers etc. We needed to buy models or make them from scratch using blender for publishing purposes and also we must read all the license and use forum of the product before using it in our project. Since we have limited knowledge about other field we took the help of the various websites for the sound and models.We have also evaluated the specification at which our game might run in a smooth manner.

Hardware Requirements	**Software Requirements**
1.1GB HDD space 2.Nvidia 740MX 3.4GB RAM	Windows 7/8/10

Table 8.1: Specification Requirements

Chapter 9

CONCLUSION

We therefore developed a 3D horror game in unity where we included the basic characteristic for the horror game like player movement and object interactions. we also came to know with the various physics elements in our game and found that every game is bounded by the laws of the physics one way of the another.We can conclude that this project might act as a bridge for us in the further Game development in different platforms.

Bibliography

[1] Mathias Clasen. How do horror video games work, and why do people play them? 2015.

[2] Jason Dansie. Game development in unity: Game production, game mechanics and the effects of gaming. 2013.

[3] Ravi sinha. A comprehensive history of horror gaming. 2016.

[4] Bian Wu and Alf Inge Wang. A guideline for game development-based learning: a literature review. *International Journal of Computer Games Technology*, 2012, 2012.

Appendix

The output we got from our developed game are given below

Figure 9.1: Fog system in our Game

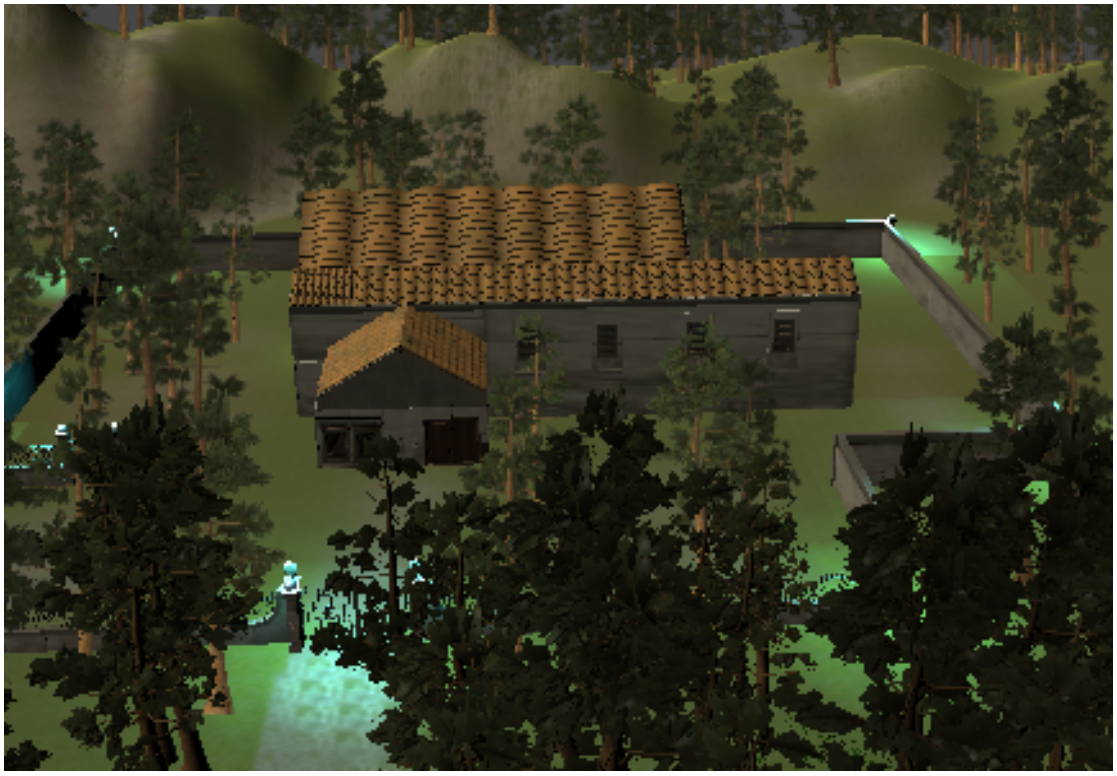

Figure 9.2: House and Environment in our Game

Figure 9.3: Ghost model in our Game

Figure 9.4: main menu in our Game

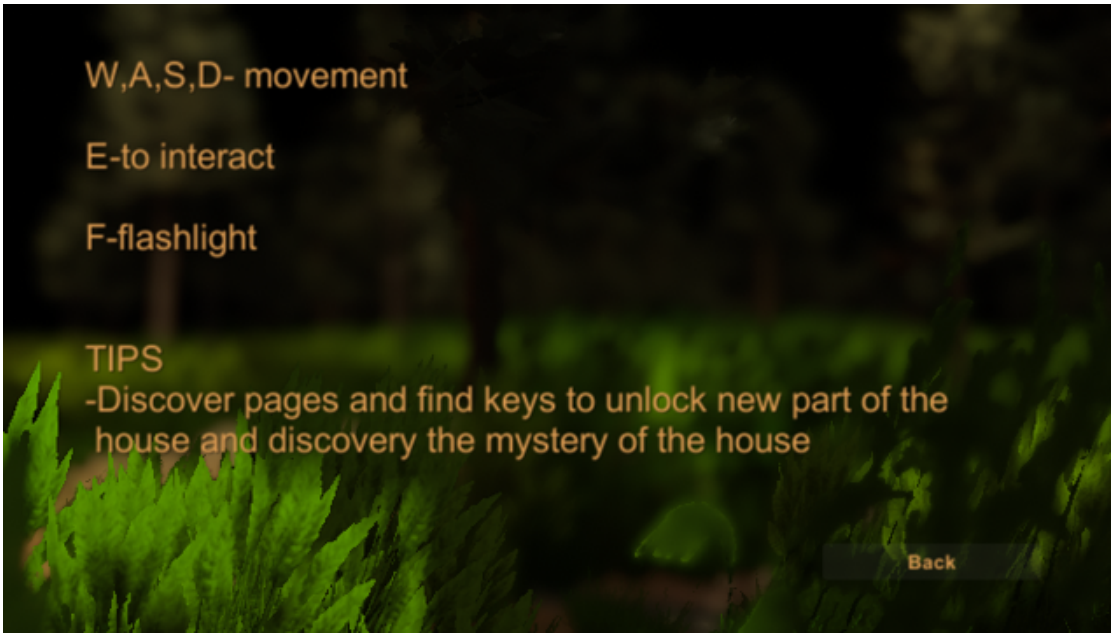

Figure 9.5: Help menu in our Game

Figure 9.6: Exit menu in our Game