

PREMIUM

VIRTUAL REALITY AND IT'S APPLICATIONS  
FINAL PROJECT

# THE MAZE MASTER







**UNDER THE GUIDANCE OF**  
**Dr. Sharad Sharma**

**Team Members**

1. Sravani Reddy Meda (11593747)
2. Pramodh Athota (11611995)
3. Naga Sirisha Ponnaganti (11563226)



# CONTENTS

- ABSTRACT
- MODELLING
- PROGRAMMING
- FUNCTIONALITY
- WHY VR ?
- FUTURE WORK
- CONCLUSION





# ABSTRACT

- A VR experience that is both immersive and interactive was developed using Unity for the Maze Master project. The goal of this project is to give players a fun and interesting approach to go into a maze-like environment and solve puzzles.
- The target audience for Maze Master game are primarily individuals who enjoy puzzle-solving and are interested in exploring new and innovative virtual environments.
- The application is useful because it provides a fun and interactive way for individuals to improve their problem-solving skills while also engaging with cutting-edge technology.



Time - 00:19

## MODELLING ( Easy Level )

### DESERT THEME

- WALLS & GROUND – SAND STONE TEXTURE
- 4K TEXTURES FROM BLENDER & POLY HAVEN
- PRO-BUILDER TOOL (To construct Maze)





Time - 00:19

## MODELLING ( Medium Level )

### FOREST THEME

- WALLS & GROUND –GREEN FOREST TEXTURE
- VOLUMETRIC FOG – USING PARTICLE GENERATOR PLUGIN
- PRO-BUILDER TOOL (To construct Maze)



**Time - 00:23**



**MODELLING  
( Difficult Level )**

**CITY THEME**

- **WHITE CITY PACKAGE FROM UNITY ASSET STORE**
- **6000 BUILDINGS**
- **PRO-BUILDER TOOL (To construct Maze)**



```
5 public class CharacterSelection : MonoBehaviour
6 {
7     public GameObject[] characters;
8     public int selectedCharacter = 0;
9     private int activeCharacter = 0;
10
11     public void onClick_Back() {
12         characters[selectedCharacter].SetActive(false);
13         selectedCharacter = activeCharacter;
14         characters[selectedCharacter].SetActive(true);
15     }
16
17     public void NextCharacter()
18     {
19         characters[selectedCharacter].SetActive(false);
20         selectedCharacter = (selectedCharacter + 1) % characters.Length;
21         characters[selectedCharacter].SetActive(true);
22     }
23
24     public void PreviousCharacter()
25     {
26         characters[selectedCharacter].SetActive(false);
27         selectedCharacter--;
28         if (selectedCharacter < 0)
29         {
30             selectedCharacter += characters.Length;
31         }
32     }
33 }
```

```
SetActive(true);
};
er", activeCharacter);
};
};
```

```
39 }
40
41 void Start() {
42     foreach (GameObject character in characters)
43     {
```

# PROGRAMMING

( Character Selection and Loading )



CharacterSelection.cs

LoadCharacter.cs

LevelComplete.cs X

MazeEndPortal.cs

Users > pramodhroy > Downloads > My project (6) > Assets > Scripts > LevelComplete.cs

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using TMPro;
5
6 public class LevelComplete : MonoBehaviour //Unused Script
7 {
8     public TMP_Text contextLable;
9     public TMP_Text scoresLable;
10
11     void StartNot()
12     {
13         string completionTime = "0" + PlayerPrefs.GetString("completedLevelTime") + ":00";
14         string completionScore = PlayerPrefs.GetString("completedLevelScore") + "00";
15         scoresLable.text = completionTime + "\n" + completionScore;
16         PlayerPrefs.SetInt("completedLevel", 0);
17     }
18
19     // Update is called once per frame
20     void Update()
21     {
22     }
23 }
24
25
```

# PROGRAMMING

(CANVAS UI – LEVEL COMPLETION)





```
acter.cs  C# LevelComplete.cs  C# MazeEndPortal.cs  C# MapHint3.cs  C# MazeHintLevel1.cs  C# MenuScript.cs x
```

Users > pramodhroy > Downloads > My project (6) > Assets > Scripts > C# MenuScript.cs

```
1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4  using UnityEngine.SceneManagement;
5  using TMPPro;
6
7  public class MenuScript : MonoBehaviour
8  {
9      public GameObject[] menus;
10     public TMP_Text levelCompleteContextLabel;
11     public TMP_Text scoresLabel;
12
13     void Awake()
14     {
15         Cursor.lockState = CursorLockMode.None;
16         foreach (GameObject menu in menus)
17         {
18             menu.SetActive(false);
19         }
20         if (!PlayerPrefs.HasKey("completedLevel"))
21         {
22             PlayerPrefs.SetInt("completedLevel", 0);
23         }
24         if (PlayerPrefs.GetInt("completedLevel") == 0)
25         {
26             menus[1].SetActive(true);
27             PlayerPrefs.SetInt("activeCharacter", 0);
28             PlayerPrefs.GetInt("activeCharacter");
29
30     }
31
32     public void quitGame()
33     {
34         Debug.Log("QUIT!");
35         Application.Quit();
36     }
37
38
39
```

# PROGRAMMING

(CANVAS UI – MENU CANVAS)



CharacterSelection.cs

LoadCharacter.cs

LevelComplete.cs

MazeEndPortal.cs

MapHint3.cs

Users > pramodhroy > Downloads > My project (6) > Assets > Scripts > MapHint3.cs

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class MapHint3 : MonoBehaviour
6 {
7     public Sprite mazeImage;
8
9     void OnTriggerEnter(Collider other)
10    {
11        Debug.Log("Entered box");
12        MazeHintLevel3.isPlayerInBox = true;
13        MazeHintLevel3.mazeSprite = mazeImage;
14    }
15
16    void OnTriggerExit(Collider other)
17    {
18        Debug.Log("Exited box");
19        MazeHintLevel3.isPlayerInBox = false;
20        MazeHintLevel3.mazeSprite = null;
21    }
22 }
23
24
```

# PROGRAMMING

(HINT DISPLAY)



```
7 public class MazeHintLevel1 : MonoBehaviour
8 {
9     public GameObject helpText;
10    public GameObject panel;
11    public Image mazeImage;
12    public TMPro.TMP_Text timerText;
13    public static Sprite mazeSprite;
14    public static bool isPlayerInBox = false;
15
16    private float currentTime = 0f;
17
18    void Update() {
19        if (isPlayerInBox && Keyboard.current[Key.E].wasPressedThisFrame) {
20            Debug.Log("pressed!");
21            mazeImage.sprite = mazeSprite;
22            panel.SetActive(true);
23        }
24        if (Keyboard.current[Key.E].wasReleasedThisFrame) {
25            Debug.Log("released!");
26            panel.SetActive(false);
27            mazeImage.sprite = null;
28        }
29        if (isPlayerInBox) {
30            if (!panel.activeSelf) {
31                helpText.SetActive(true);
32            }
33        }
34        currentTime += Time.deltaTime;
35        string formattedTime = System.TimeSpan.FromSeconds(currentTime).ToString(@"mm\:ss");
36        timerText.text = "Time - " + formattedTime;
37        MazeEndPortal.completionTime = formattedTime;
38    }
39
40
41
42
43
44 }
```

# PROGRAMMING

(MAZE MAP DISPLAY – ‘E’ KEY TRIGGERED)





PREMIUM

# FUNCTIONALITY

## VISION:

- EASY LEVEL – 10\*10 MAZE , DESERT THEME
- MEDIUM LEVEL – 15\*15 MAZE , FOREST THEME
- DIFFICULT LEVEL – 20\*20 MAZE , CITY THEME



# FUNCTIONALITY

## SOUND:

- FOOTSTEPS OF THE PLAYER ACCORDING TO MOVEMENT.
- SOUND EFFECTS WHEN PLAYER REACHES A POINT IN MAZE.
- BACKGROUND MUSIC THROUGHOUT THE GAME

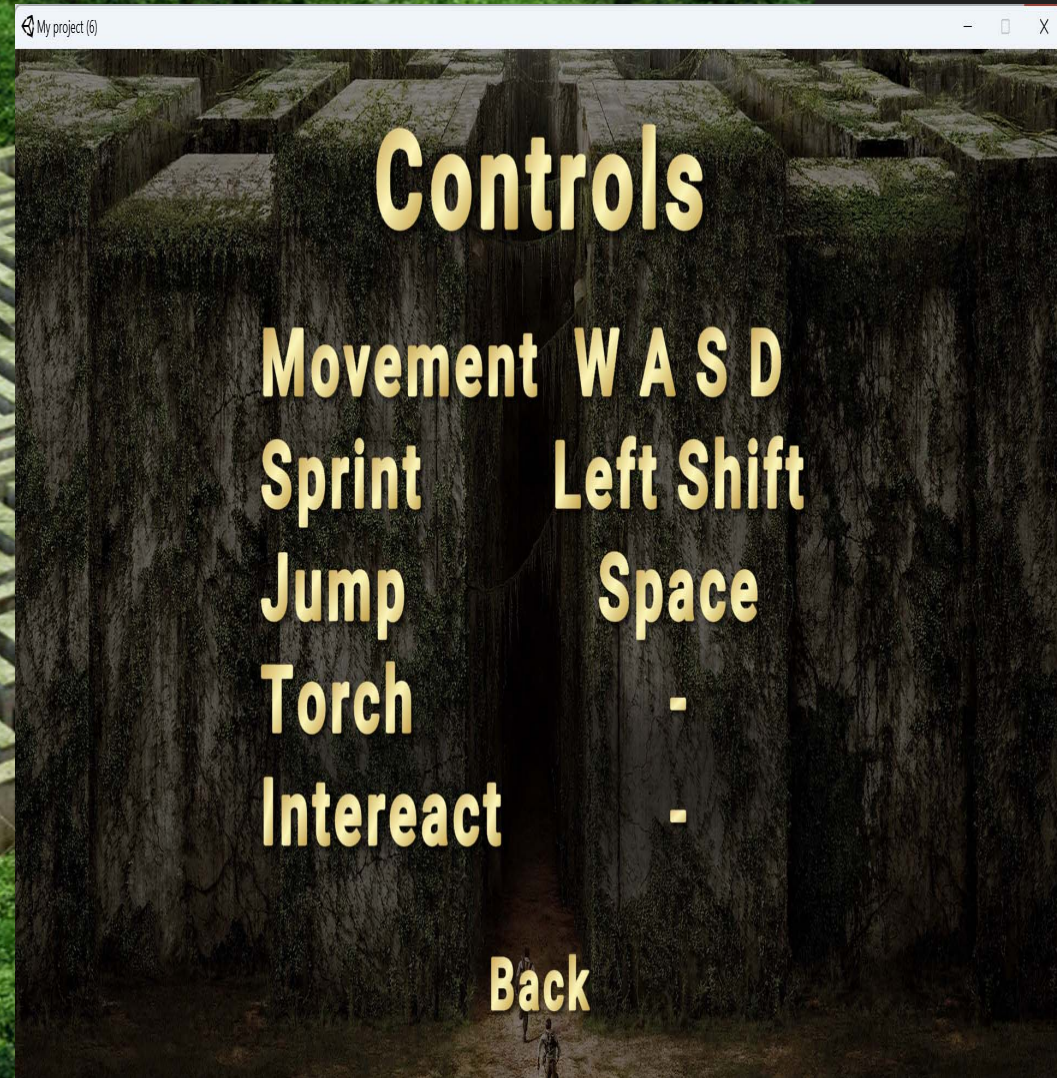


# FUNCTIONALITY

## ANIMATIONS:

### > PLAYER ANIMATIONS –

Player movement in all the directions, Jumping, Running and Climbing



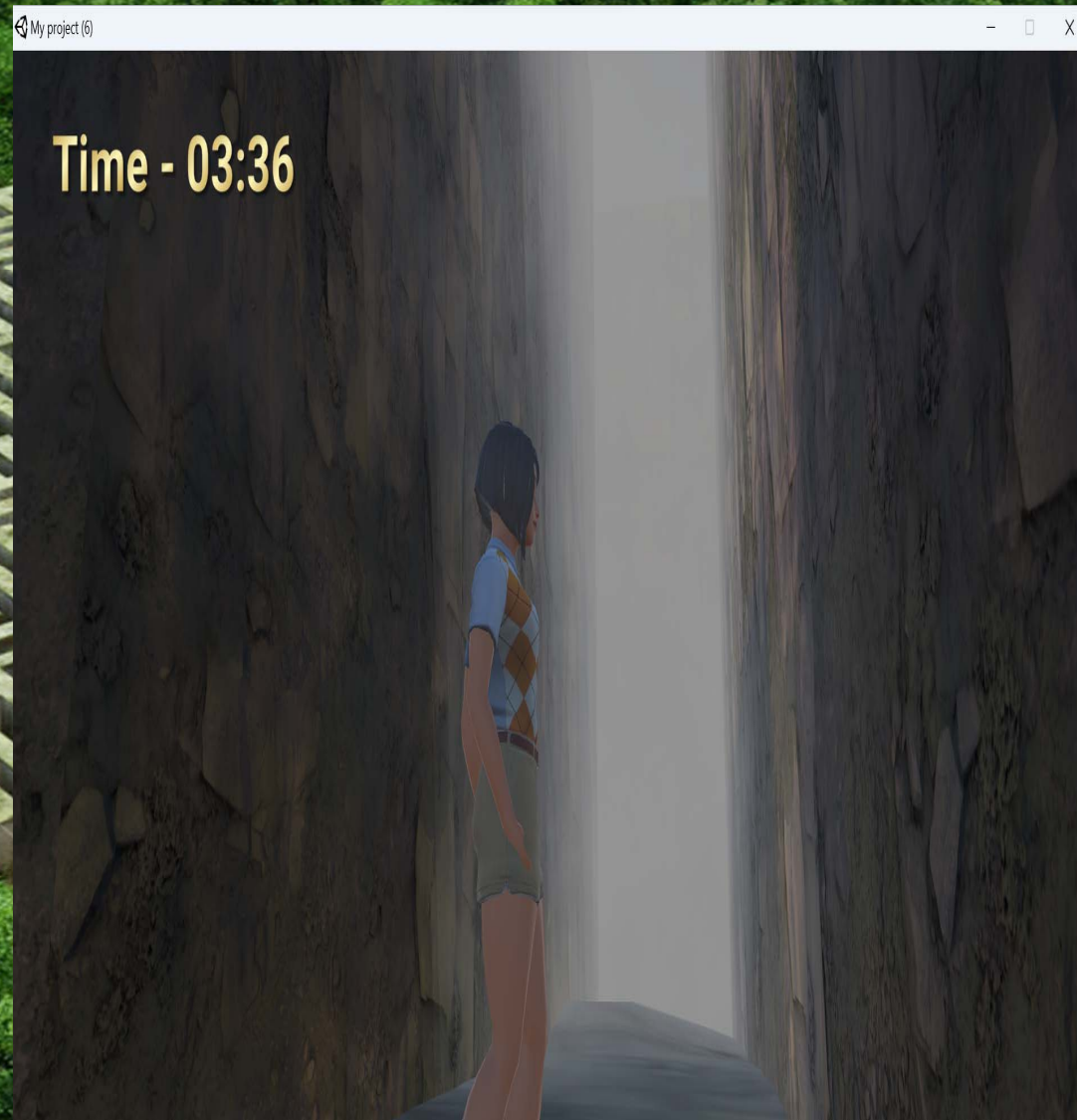


# FUNCTIONALITY

## ANIMATIONS:

### ➤ FOG ANIMATIONS –

Volumetric Fog is created using particle generator Plugin





# FUNCTIONALITY

## INTERACTIVITY (AVATAR SELECTION)

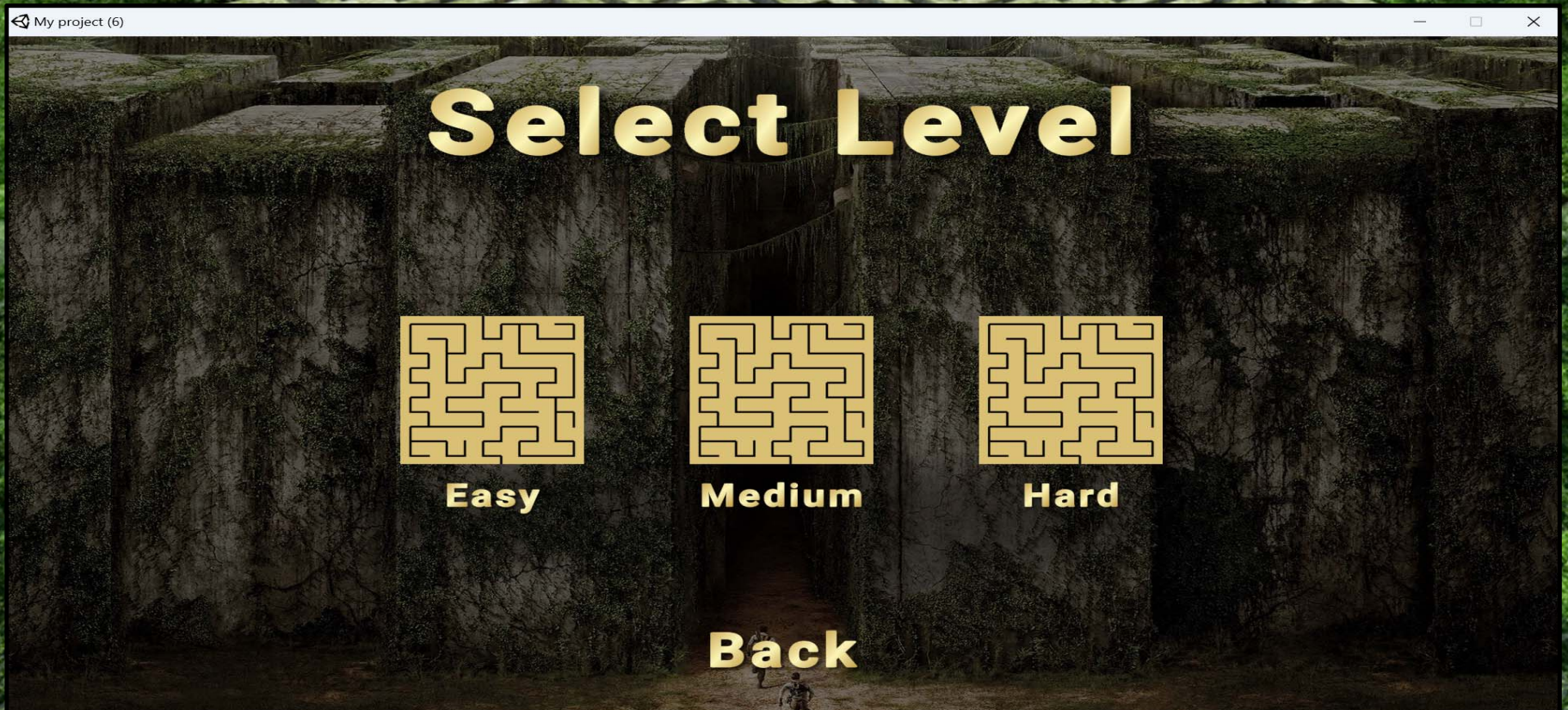




PREMIUM

# FUNCTIONALITY

• INTERACTIVITY (LEVEL SELECTION)





# FUNCTIONALITY

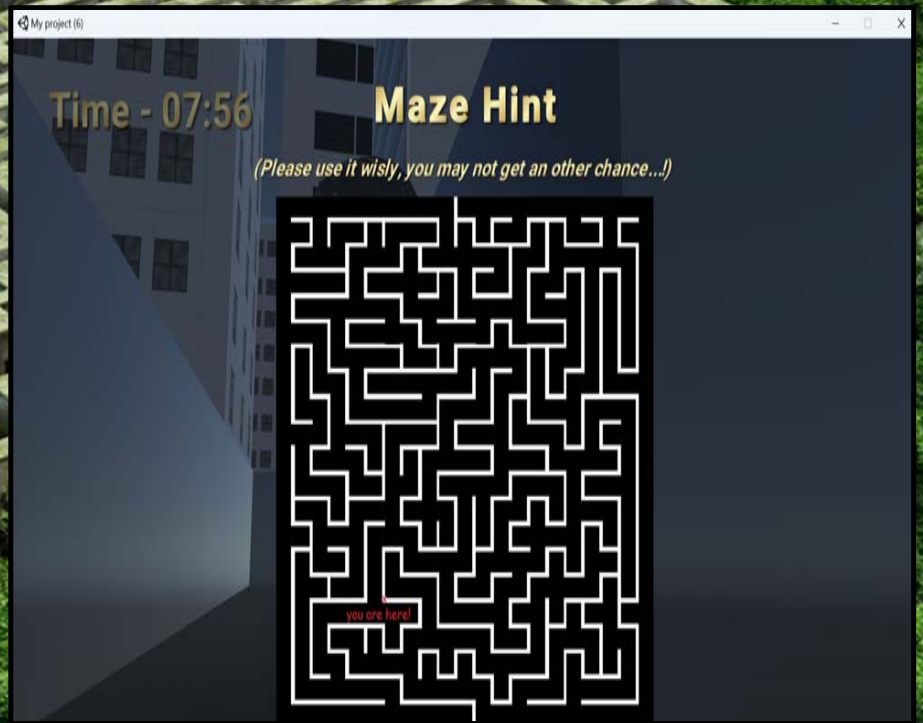
## INTERACTIVITY (HELP NOTIFICATION)





# FUNCTIONALITY

## INTERACTIVITY (MAZE MAP DISPLAY)





# FUNCTIONALITY

## INTERACTIVITY (TELEPORTATION)





# FUNCTIONALITY

## CHARACTERS/AVTARS

- FEMININE AVTAR
- MASCULINE AVTAR
- W, A, S, D and LEFT SHIFT KEY, SPACE BAR for movement.
- Interactive with environment
- If the player collides with walls/buildings – Proximity Sensor



# FUNCTIONALITY

➤ SENSORS (PROXIMITY SENSOR)





# FUNCTIONALITY

## > SENSORS (TIME SENSOR)





# FUNCTIONALITY

➤ SENSORS (KEYBOARD INPUT - TOUCH SENSOR)





# FUNCTIONALITY

PLAYER

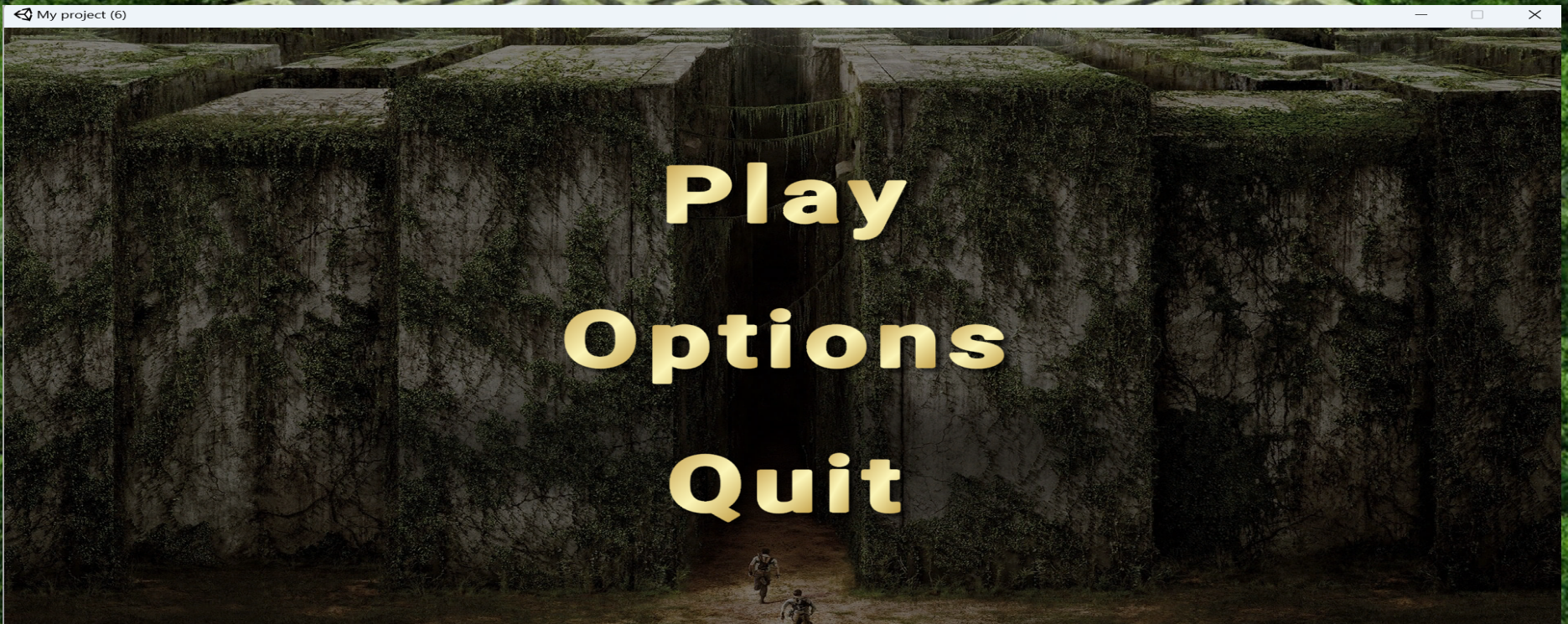
Third Person Controller





# FUNCTIONALITY

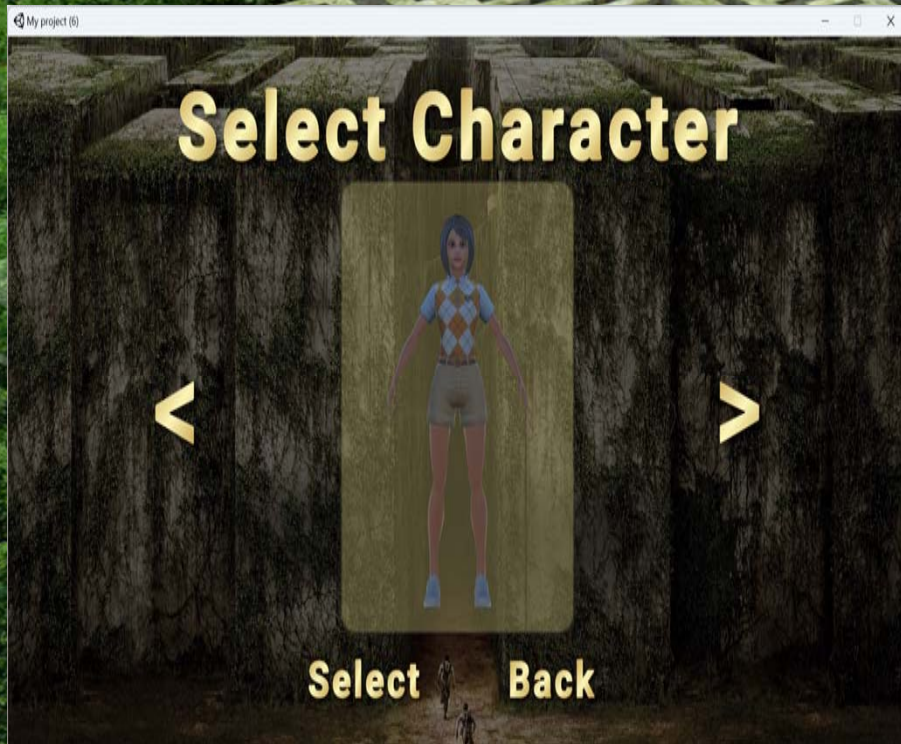
## INTERFACE ELEMENTS - MAIN MENU





# FUNCTIONALITY

## INTERFACE ELEMENTS – CHARACTER SELECTION





# FUNCTIONALITY

## INTERFACE ELEMENTS - OPTIONS





PREMIUM

# FUNCTIONALITY

INTERFACE ELEMENTS – HELP NOTIFICATION





PREMIUM

# FUNCTIONALITY

## INTERFACE ELEMENTS – MAZE MAP





PREMIUM

# FUNCTIONALITY

INTERFACE ELEMENTS – TELEPORTATION





# FUNCTIONALITY

## INTERFACE ELEMENTS – TIMER

A timer runs for each level separately to calculate the time taken by the player to solve the maze





# Why VR is appropriate Technology?

- Provides a highly immersive and interactive environment for the players.
- By using a VR headset and controllers, players can move around and interact with the environment as if they are actually present in it.
- Allows for a high level of customization and interactivity, which is necessary for creating a complex and engaging maze game like this one.
- it provides a unique and highly engaging experience to the players.



# FUTURE WORK

- Multiplayer Mode
- More Levels
- Adding Opponent
- Achievements and Rewards





# CONCLUSION

- In conclusion, the project is a maze game with three levels of increasing difficulty.
- The game allows players to select their avatar, adjust sound settings, and navigate through different terrains.
- The project is developed using Unity and written in C#.



PREMIUM

THANK YOU !!

