**UNITY BASED 2D CYBERCITY(BASH BYTE)**

***A project based learning report submitted by***

**DAVID SANJEET KUMAR (URK20CS2041)**

***in partial fulfillment for the award of the degree of***

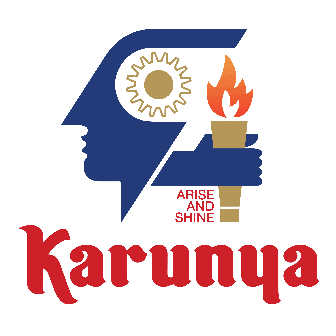
**BACHELOR OF TECHNOLOGY**

***in***

**COMPUTER SCIENCE AND ENGINEERING**

***under the supervision of***

**Nirmal Varghese Babu (Assistant Professor)**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**KARUNYA INSTITUTE OF TECHNOLOGY AND SCIENCES**

(Declared as Deemed to be University -under Sec-3 of the UGC Act, 1956)

**Karunya Nagar, Coimbatore - 641 114. INDIA**

**April 2023**

**UNITY BASED 2D CYBERCITY(BASH BYTE)**

***A project based learning report submitted by***

**DAVID SANJEET KUMAR (URK20CS2041)**

***in partial fulfillment for the award of the degree of***

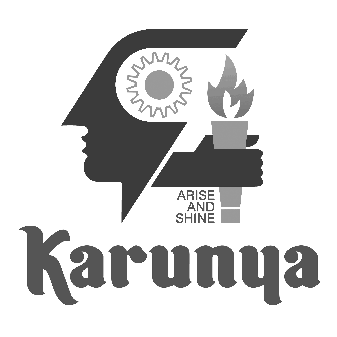
**BACHELOR OF TECHNOLOGY**

***in***

**COMPUTER SCIENCE AND ENGINEERING**

***under the supervision of***

**Nirmal Varghese Babu (Assistant Professor)**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**KARUNYA INSTITUTE OF TECHNOLOGY AND SCIENCES**

(Declared as Deemed to be University -under Sec-3 of the UGC Act, 1956)

**Karunya Nagar, Coimbatore - 641 114. INDIA**

**APRIL 2023**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**BONAFIDE CERTIFICATE**

This is to certify that the project report entitled, “Unity based 2d Cybercity:Bash Byte” is a bonafide record of project based learning work done during the even semester of the academic year 2022- 2023 by

**DAVID SANJEET KUMAR (Reg. No: URK20CS2041)**

in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering of Karunya Institute of Technology and Sciences.

Submitted for the Viva Voce held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Signature of the Guide**

**ACKNOWLEDGEMENT**

First and foremost, I praise and thank ALMIGHTY GOD whose blessings have bestowed in me the will power and confidence to carry out my project.

I am grateful to our beloved founders Late**. Dr. D.G.S. Dhinakaran, C.A.I.I.B, Ph.D** and **Dr. Paul Dhinakaran, M.B.A, Ph.D**, for their love and always remembering us in their prayers.

I extend my thanks to our Vice Chancellor **Dr.P. Mannar Jawahar, Ph.D** and our Registrar **Dr. Elijah Blessing, M.E., Ph.D,** for giving me this opportunity to do the project.

I wish to extend my thanks to our Pro-Vice Chancellor (RC) **Dr. E.J. James** and **Dr. Ridling Margaret Waller** Pro-Vice Chancellor (QS), for providing me this opportunity to do the project.

I would like to thank **Dr. Prince Arulraj, M.E., Ph.D.,** Dean, School of Engineering and Technology for his direction and invaluable support to complete the same.

I would like to place my heart-felt thanks and gratitude to **Dr. J. Immanuel John Raja, M.E., Ph.D.,** Head of the Department, Computer Science and Engineering for his encouragement and guidance.

I feel it is a pleasure to be indebted to, **Nirmal Varghese Babu,** AssistantProfessor, Department of Computer Science and Engineering and for her invaluable support, advice and encouragement.

I also thank all the staff members of the Department for extending their helping hands to make this project a successful one.

I would also like to thank all my friends and my parents who have prayed and helped me during the project work.

**ABSTRACT**

This 2D cybercity platformer game created in Unity offers players a thrilling adventure.The game features an immersive retro-futuristic aesthetic, with vibrant pixel art and a dynamic synthwave soundtrack that transports players to a world where rebellion is the only way to survive.As players jump and dodge their way through the city, they must also battle against a system that seeks to suppress their rebellion. Using advanced artificial intelligence techniques, the game challenges players with unpredictable enemies and obstacles, providing a constantly evolving and dynamic gaming experience.Players will need to master a range of skills, including precision jumping, combat, and puzzle-solving, as they progress through the game.The game's platformer mechanics are designed to offer a challenging but fair experience, rewarding players who take risks and think outside the box.

Along the way, players will encounter NPCs who will provide guidance and assistance on their quest.The game's narrative explores themes of social injustice, power, and corruption, inviting players to engage with complex issues in a meaningful way.With its engaging gameplay, retro-futuristic aesthetic, and thought-provoking narrative, this 2D cybercity platformer game in Unity offers an exciting and immersive gaming experience that will appeal to players of all skill levels.

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| Acknowledgement  Abstract  Content  **1. Introduction** | | **i** |
| **ii** |
| **iii** |
| **1** |
|  | 1.1 Introduction | 1 |
|  | 1.2 Objectives | 2 |
|  | 1.3 Motivation | 2 |
|  | 1.4 Overview of the Project | 2 |
|  | 1.5 Chapter wise Summary | 3 |
| **2. Analysis and Design** | | **4** |
|  | 2.1 Overall analysis | 4 |
|  | 2.2 Functional Requirements | 5 |
|  | 2.1 Non-Functional Requirements | 6 |
|  | 2.3 Architecture | 7 |
| **3. Implementation** | | **10** |
|  | 3.1 Modules Description | 10 |
|  | 3.2 Implementation Details | 11 |
|  | 3.3 Tools used | 16 |
| **4. Test Results** | | **17** |
|  | 4.1 Testing | 17 |
|  | 4.2 Results | 20 |
| **5. Conclusion and Further Scope** | | 24 |
|  |  |  |
| **References** | | **25** |
|  |  |  |
|  |  |  |
|  |  |  |

**CHAPTER 1**

**INTRODUCTION**

**1.1 INTRODUCTION**

  A thrilling game that is quickly acquiring a lot of popularity among gamers is the 2D Cybercity platformer. The user controls a character in this game, which is made for a 2D platform, as they go around a cybercity setting. The Unity game engine, a well-known game creation platform, is the platform on which the game is intended to be played. The purpose of this paper is to give a general overview of the Unity development process for a 2D cybercity platformer game. The numerous facets of game production are covered in the study, including game design, programming, artwork, sound, and testing.

**1.2 OBJECTIVE**

The primary goal of the project is to create and solidify the game mechanics and game core foundations in a 2D-3D space.

**1.3 MOTIVATION**

The game is developed using the Unity game engine and offers players a fast-paced and action-packed gameplay experience similar to hack and slash. The game includes a variety, each with its unique challenges and obstacles. The player must navigate through the level, jumping across platforms, avoiding traps, and taking down enemies to reach the end goal. The level is designed to be challenging but fair, with a focus on skill-based gameplay.

**1.4 OVERVIEW OF THE PROJECT**

Cybercity also features a robust upgrade system, allowing players to enhance their character's abilities. Players can earn experience points by defeating enemies and completing objectives, which can then be used to upgrade their character's stats and abilities.The character is armed with various abilities, to help them traverse the terrain and defeat enemies. The game includes a diverse cast of enemies, each with their unique abilities and weaknesses. The player must learn to adapt to each enemy's play-style to defeat them effectively. The game also includes various sound effects, such as a busy street and explosions, to further enhance the player's experience.

**1.5 CHAPTER WISE SUMMARY**

The chapters in the report are categorized as follows: Chapter 2 consists of System Analysis and Design, Chapter 3 describes the Implementation. Chapter 4 contains Test results and finally, conclusion and Further Scope is in Chapter 5.

**CHAPTER 2**

**ANALYSIS AND DESIGN**

**2.1 Game Design**

The first step in developing a game is to design the game. This involves creating a plan for the game, which includes the game mechanics, game objectives, game environment, and the characters in the game. For the 2D cybercity platformer game, the game environment is a cybercity. The player will control a character to navigate through the city and complete objectives. The game objectives may include collecting items, defeating enemies, and reaching a certain destination. The game mechanics will include running, jumping, and attacking. The game characters will include the player's character and the enemies in the game. The programming aspect of game development involves using programming languages to code the game mechanics and game logic. Unity uses C# as its programming language, and the game mechanics for the 2D cybercity platformer game will be coded in C#. The programming will involve coding the player movement, enemy AI, item collection, and other game mechanics. The programming will also involve coding the game logic, which will include the win and lose conditions, the game objectives, and the game progression.

**2.1.1 Level Design**

The game features several levels, each with its unique challenges and enemies. The levels are designed to provide a balanced difficulty curve, with the early levels being relatively easy and gradually increasing in difficulty as the player progresses through the game. The levels are also designed to be visually appealing, with a distinct cybercity aesthetic that immerses the player in the game's world.

**2.1.2 Artwork**

The artwork is an essential part of game development, and it involves designing the game environment, characters, and other game assets. For the 2D cybercity platformer game, the artwork will include designing the cybercity environment, designing the player's character, designing the enemy characters, designing the items in the game, and designing the game user interface. The artwork will be created using software such as Adobe Photoshop, Adobe Illustrator, or other similar software.

**2.1.3 Audio Effects**

Sound is an essential part of game development, and it involves creating sound effects and background music for the game. The sound effects will include sound effects for player movement, enemy AI, item collection, and other game mechanics. The background music will be designed to create an immersive experience for the player and to enhance the game's atmosphere.

**2.2 FUNCTIONAL REQUIREMENTS**

* Combat and Interaction - The player must be able to interact with the game world through combat, dialogue, and other means, with a range of weapons, abilities, and tactics at their disposal.
* Level Design - The game must have a range of levels with unique and challenging obstacles, puzzles, enemies, and boss fights that test the player's skills and abilities.
* Artificial Intelligence - The game must feature advanced AI techniques that create dynamic and unpredictable enemies and obstacles, adapting to the player's actions and challenging them in new and unexpected ways.
* Narrative and Dialogue - The game must have a compelling narrative and dialogue that engages players with complex themes and characters, and motivates them to progress through the game's levels.
* User Interface - The game must have a user-friendly interface that provides clear and concise information about the player's status, objectives, and progress, and allows them to access game settings, options, and controls.

**2.3 NON-FUNCTIONAL REQUIREMENTS**

* Performance and Scalability - The game must run smoothly and efficiently on a range of hardware and software configurations, with minimal loading times and high frame rates.
* Reliability and Availability - The game must be reliable and available, with minimal downtime or interruptions, and a robust system for error reporting and recovery.
* Security and Privacy - The game must be secure and protect user data, with a range of authentication, encryption, and data protection measures in place.
* Accessibility and Usability - The game must be accessible and usable by players of all abilities, with features such as adjustable difficulty levels, subtitles, and audio descriptions.
* Compatibility and Interoperability - The game must be compatible and interoperable with other software and hardware, such as operating systems, game controllers, and virtual reality headsets.
* Maintainability and Extensibility - The game must be easy to maintain and extend, with a modular and scalable design, clear documentation, and support for third-party plugins and assets.

Overall, this 2D cybercity platformer game in Unity must meet a range of functional and non-functional requirements to provide a seamless and enjoyable gaming experience that engages and challenges players.

**2.4 ARCHITECTURE**

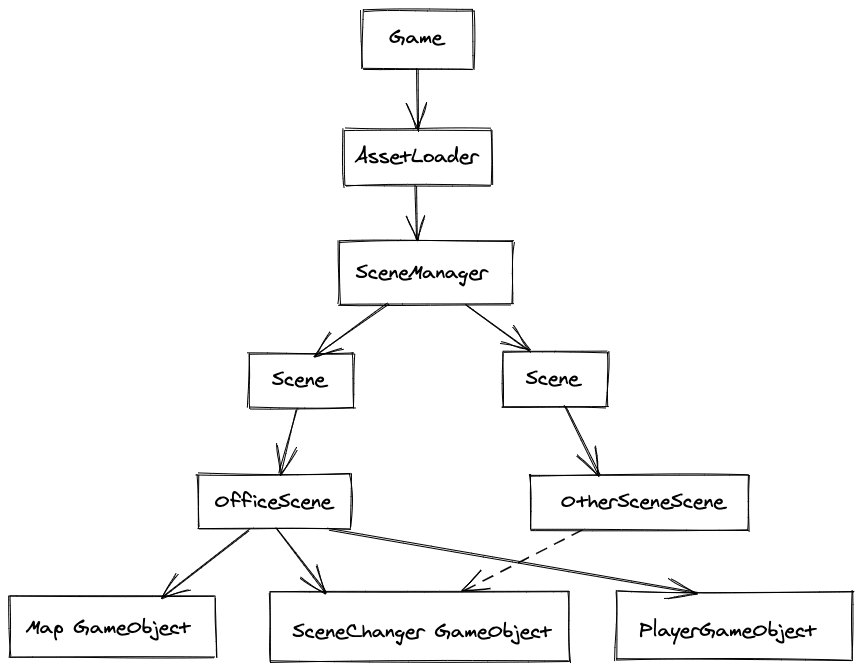
****

Fig 2.1 RPG Game Overview**CHAPTER 3**

**IMPLEMENTATION**

**3.1 MODULES DESCRIPTION**

To run Cybercity smoothly, your computer needs to meet the following minimum system requirements:

* Operating System: Windows 7 or later, or macOS 10.12 or later.
* Processor: Intel Core i3 2.4 GHz or equivalent.
* Memory: 4 GB RAM.
* Graphics Card: NVIDIA GeForce GTX 460 or AMD Radeon HD 5670 or equivalent.
* Storage: 2 GB available space.

While Cybercity can run on a variety of hardware configurations, meeting these system requirements ensures that the game will run smoothly and without any performance issues.

Movement and Navigation - The player must be able to move and navigate through the 2D cybercity environment with smooth and responsive controls. It is also worth noting that Cybercity is designed to be a controller-friendly game. While it can be played with a keyboard and mouse, using a gamepad will provide a more intuitive and enjoyable gameplay experience. If you plan to play the game with a controller, make sure that it is compatible with your computer and has enough buttons to support all of the game's actions.

The 2D cybercity platformer game in Unity can be divided into several modules, each of which serves a specific purpose in the overall game design. These modules include the game mechanics module, the graphics module, the audio module, the level design module, and the user interface (UI) module.

**3.1.1 Game mechanics module**

Thegame mechanics module is responsible for defining the rules and logic of the game. This includes the character movement, jumping, attacking, and interacting with objects in the game world. It also includes the physics engine, which governs the behavior of objects in the game world, and the collision detection system, which ensures that the player character and other objects in the game world do not pass through each other.

**3.1.2 Graphics module**

The graphics module is responsible for creating the visual elements of the game. This includes the character sprites, background graphics, foreground graphics, and any special effects, such as particle systems. It also includes the lighting system, which affects the mood and atmosphere of the game, and the camera system, which determines the player's viewpoint and how the game world is displayed on the screen.

**3.1.3 Audio module**

The audio module is responsible for creating and playing the sound effects and music in the game. This includes background music, sound effects for character movement and interactions, and voiceovers for any characters or narrative elements in the game.

**3.1.4 Level Design module**

The level design module is responsible for creating the various levels and environments in the game. This includes designing the layout of platforms, obstacles, enemies, and other interactive elements in the game world. It also involves determining the pacing and difficulty of each level, and ensuring that the overall game progression is well-balanced and engaging.

**3.1.5 UI module**

The UI module is responsible for creating the various user interfaces in the game. This includes the main menu, pause menu, game over screen, and any other screens or overlays that the player may encounter during gameplay. It also includes the heads-up display (HUD), which displays the player's score, health, and other relevant information on the screen during gameplay.

Overall, each of these modules is crucial to the design and implementation of the 2D cybercity platformer game in Unity. By creating a well-designed game mechanics module, graphics module, audio module, level design module, and UI module, game developers can create a cohesive and engaging gaming experience that players will enjoy.

**3.2 IMPLEMENTATION DEATAILS**

**3.2.1 Startup Script**

* Setup And Startup of the Entire world:

public class Startup : MonoBehaviour

{

    public bool dontDestroyOnLoad = false;

    // Use this for initialization

    void Awake()

    {

        if (dontDestroyOnLoad)

        {

            DontDestroyOnLoad(gameObject);

        }

        if (GameObject.Find("Startup") != null && GameObject.Find("Startup").tag == "Startup")

        {

            Destroy(gameObject);

        }

    }

}

* Parallax Effect:

/\*Allows the controller to move each layer based on the parallaxAmount!\*/

public class ParallaxLayer : MonoBehaviour

{

    [Range(-1f, 1f)]

    public float parallaxAmount; //The amount of parallax! 1 simulates being close to the camera, -1 simulates being very far from the camera!

    [System.NonSerialized] public Vector3 newPosition;

    private bool adjusted = false;

    public void MoveLayer(float positionChangeX, float positionChangeY)

    {

        newPosition = transform.localPosition;

        newPosition.x -= positionChangeX \* (-parallaxAmount \* 40) \* (Time.deltaTime);

        newPosition.y -= positionChangeY \* (-parallaxAmount \* 40) \* (Time.deltaTime);

        transform.localPosition = newPosition;

    }

}

**3.2.2 Player Script**

* To adds player functionality to a physics object:

[RequireComponent(typeof(RecoveryCounter))]

public class NewPlayer : PhysicsObject

{

    [Header ("Reference")]

    public AudioSource audioSource;

    [SerializeField] private Animator animator;

    private AnimatorFunctions animatorFunctions;

    public GameObject attackHit;

    private CapsuleCollider2D capsuleCollider;

    public CameraEffects cameraEffects;

    [SerializeField] private ParticleSystem deathParticles;

    [SerializeField] private AudioSource flameParticlesAudioSource;

    [SerializeField] private GameObject graphic;

    [SerializeField] private Component[] graphicSprites;

    [SerializeField] private ParticleSystem jumpParticles;

    [SerializeField] private GameObject pauseMenu;

    public RecoveryCounter recoveryCounter;

    // Singleton instantiation

    private static NewPlayer instance;

    public static NewPlayer Instance

    {

        get

        {

            if (instance == null) instance = GameObject.FindObjectOfType<NewPlayer>();

            return instance;

        }

    }

* The Movement speed and ledge falloff time:

 [Header("Properties")]

    [SerializeField] private string[] cheatItems;

    public bool dead = false;

    public bool frozen = false;

    private float fallForgivenessCounter; //Counts how long the player has fallen off a ledge

    [SerializeField] private float fallForgiveness = .2f; //How long the player can fall from a ledge and still jump

    [System.NonSerialized] public string groundType = "grass";

    [System.NonSerialized] public RaycastHit2D ground;

    [SerializeField] Vector2 hurtLaunchPower; //How much force should be applied to the player when getting hurt?

    private float launch; //The float added to x and y moveSpeed. This is set with hurtLaunchPower, and is always brought back to zero

    [SerializeField] private float launchRecovery; //How slow should recovering from the launch be? (Higher the number, the longer the launch will last)

    public float maxSpeed = 8; //Max move speed

    public float jumpPower = 18;

    private bool jumping;

    private Vector3 origLocalScale;

    [System.NonSerialized] public bool pounded;

    [System.NonSerialized] public bool pounding;

* Player Movement and Controls:

protected void ComputeVelocity()

    {

        //Player movement & attack

        Vector2 move = Vector2.zero;

        ground = Physics2D.Raycast(new Vector2(transform.position.x, transform.position.y), -Vector2.up);

        //Lerp launch back to zero at all times

        launch += (0 - launch) \* Time.deltaTime \* launchRecovery;

        if (Input.GetButtonDown("Cancel"))

        {

            pauseMenu.SetActive(true);

        }

        //Movement, jumping, and attacking!

        if (!frozen)

        {

            move.x = Input.GetAxis("Horizontal") + launch;

            if (Input.GetButtonDown("Jump") && animator.GetBool("grounded") == true && !jumping)

            {

                animator.SetBool("pounded", false);

                Jump(1f);

            }

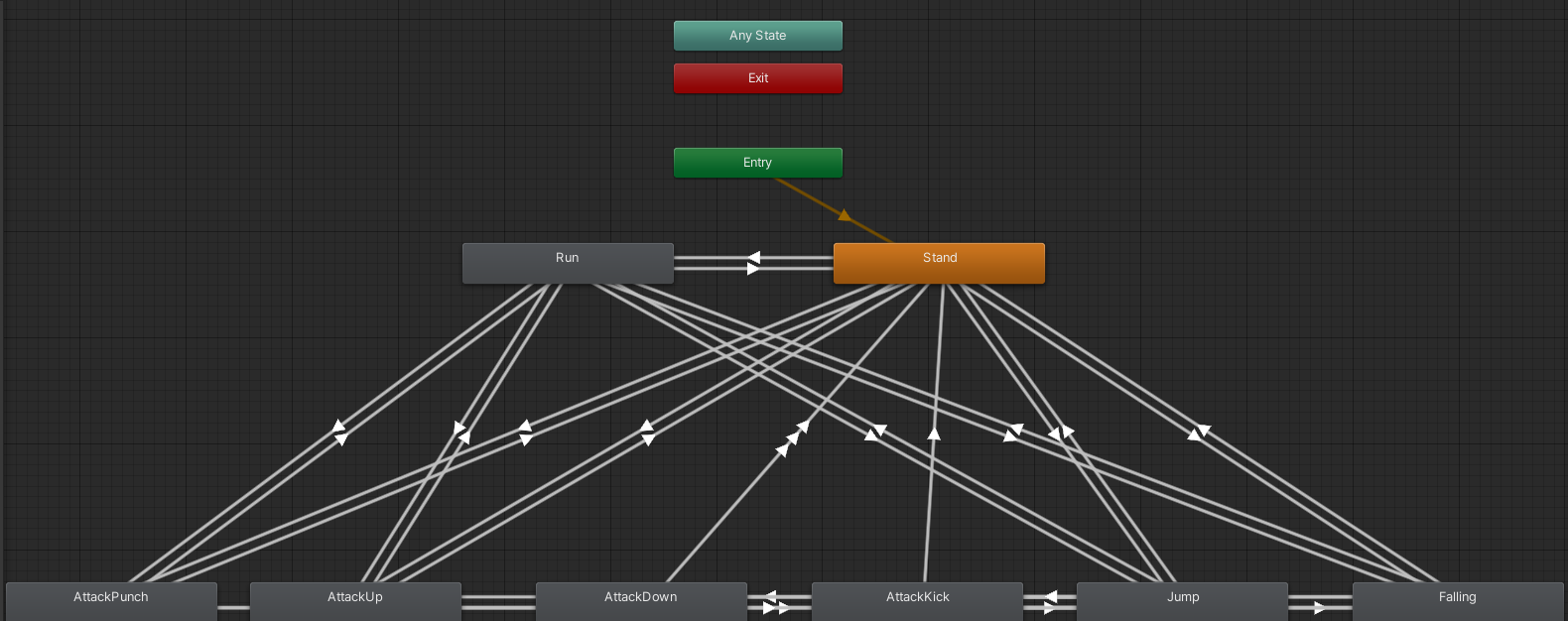


Fig 3.1 Player Decision Tree and actions

**3.3 TOOLS USED**

**3.3.1 UNITY 3.3.0**

Unity is a cross-platform game engine and development environment used to create video games, virtual and augmented reality experiences, simulations, and other interactive content. It is known for its ease of use, versatility, and flexibility, making it a popular choice for both indie developers and large game development studios.

Unity provides a wide range of features and tools for game development, including a visual editor, asset management, physics engine, scripting tools, and support for multiple platforms such as PC, mobile devices, consoles, and virtual reality headsets. Its visual editor allows designers to create and manipulate game objects, materials, and animations, while the scripting tools allow developers to program the game's functionality using C#, JavaScript, or Boo

**3.3.2 Adobe Photoshop**

Adobe Photoshop is a popular software program used for editing and manipulating digital images. It is a powerful tool for graphic designers, photographers, and digital artists. Photoshop is part of the Adobe Creative Suite and is available for both Mac and Windows operating systems.

Photoshop allows users to manipulate images in a wide variety of ways, including color correction, retouching, and adding effects. The program has an extensive array of tools and features, such as layers, masks, filters, and blending modes. It also supports a wide range of file formats, including JPEG, PNG, and TIFF.

**3.3.3 Filmora**

Filmora is a video editing software designed for beginner and intermediate video editors. It offers a range of features and tools that make it easy to create high-quality videos without extensive technical knowledge.

One of the key features of Filmora is its user-friendly interface. The program has a simple and intuitive interface that is easy to navigate, with drag-and-drop functionality and a range of pre-made templates and effects that allow users to quickly create professional-looking videos.

**3.3.4 VSCode**

Visual Studio Code (VSCode) is a source code editor developed by Microsoft. It is a free, open-source software that is widely used by programmers and developers around the world. VSCode is compatible with multiple programming languages, including JavaScript, Python, C++, and many more.

One of the key advantages of VSCode is its powerful editing capabilities. The software provides features such as code completion, syntax highlighting, and auto-formatting, which can help programmers to write code faster and more accurately. VSCode also includes built-in debugging tools, which allows programmers to debug their code more efficiently.

**CHAPTER 4**

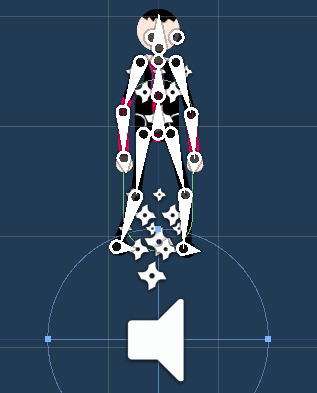
**VERIFICATION**

**4.1 TESTING**

The following modules were coded and tested during the course of this project

**4.1.1 Loading Assets**

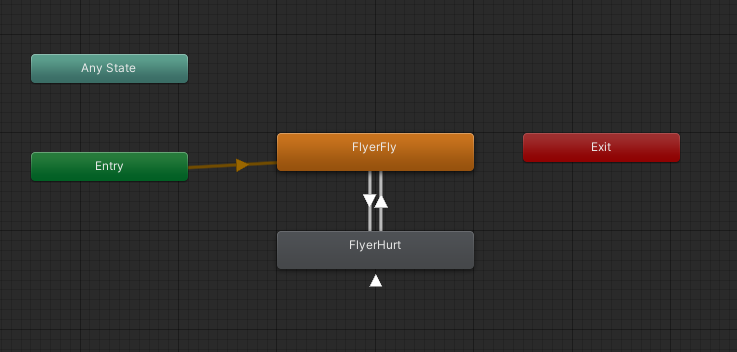
* **Player Inspector:**

****

**Fig 4.1 Player Sprite**

****

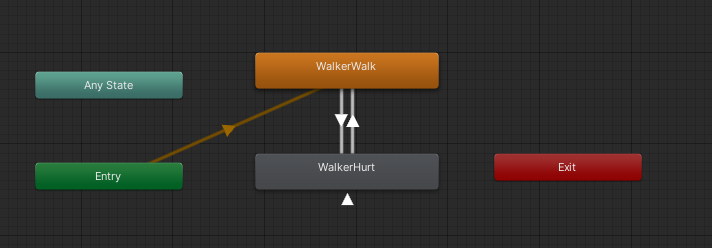
**Fig 4.2 Flying Enemy Sprite**

****

**Fig 4.3 Flying Enemy Action tree**

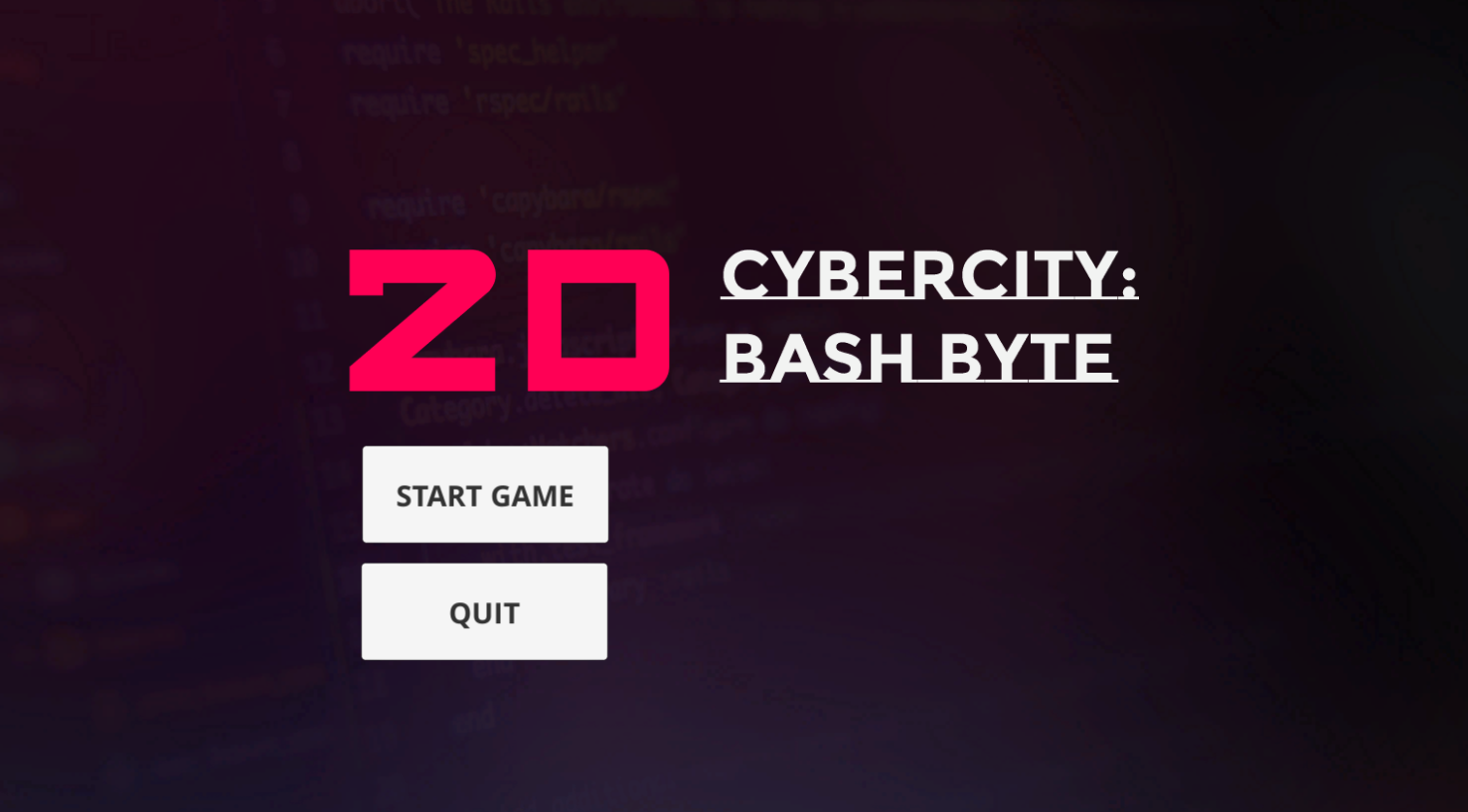
****

**Fig 4.4 Walker Enemy Sprite**

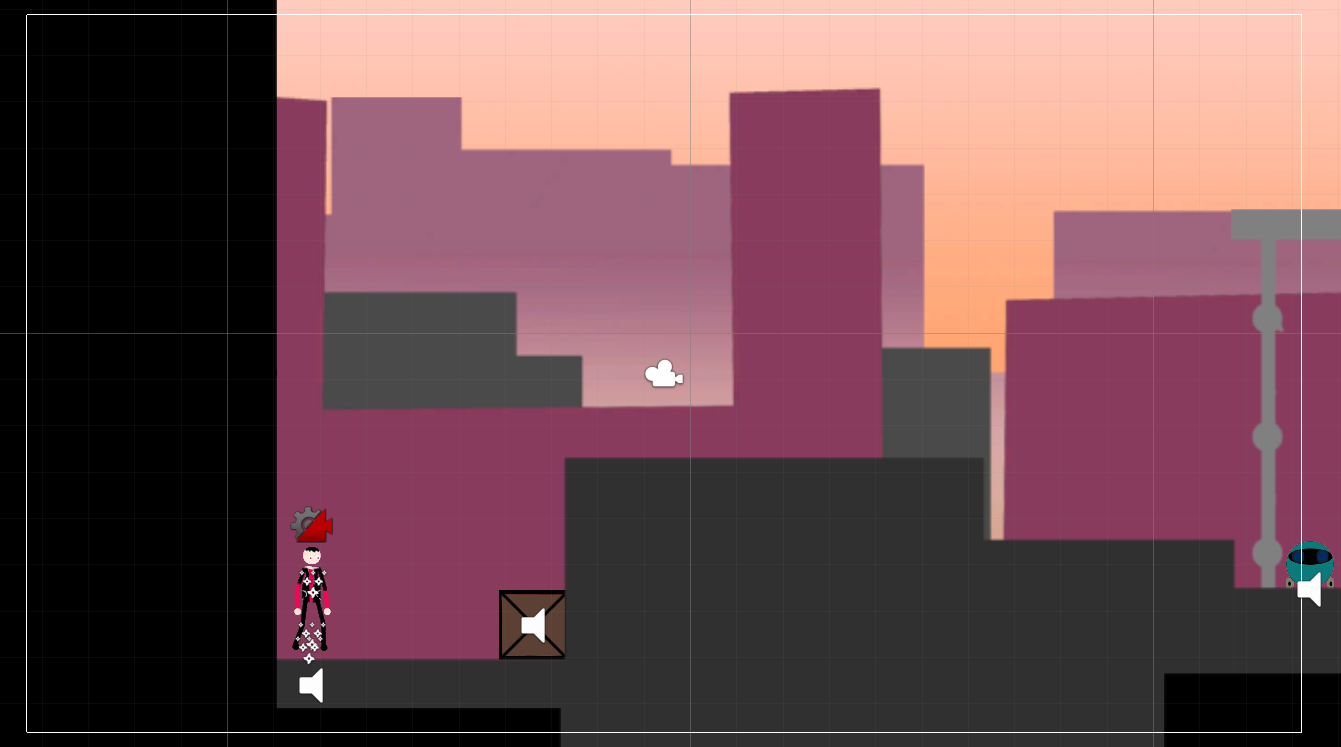
****

**Fig 4.5 Wa;ker Enemy Action tree**

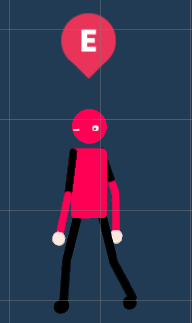
* 1. **RESULTS**
* **Loading all Assets into the game environment:**

****

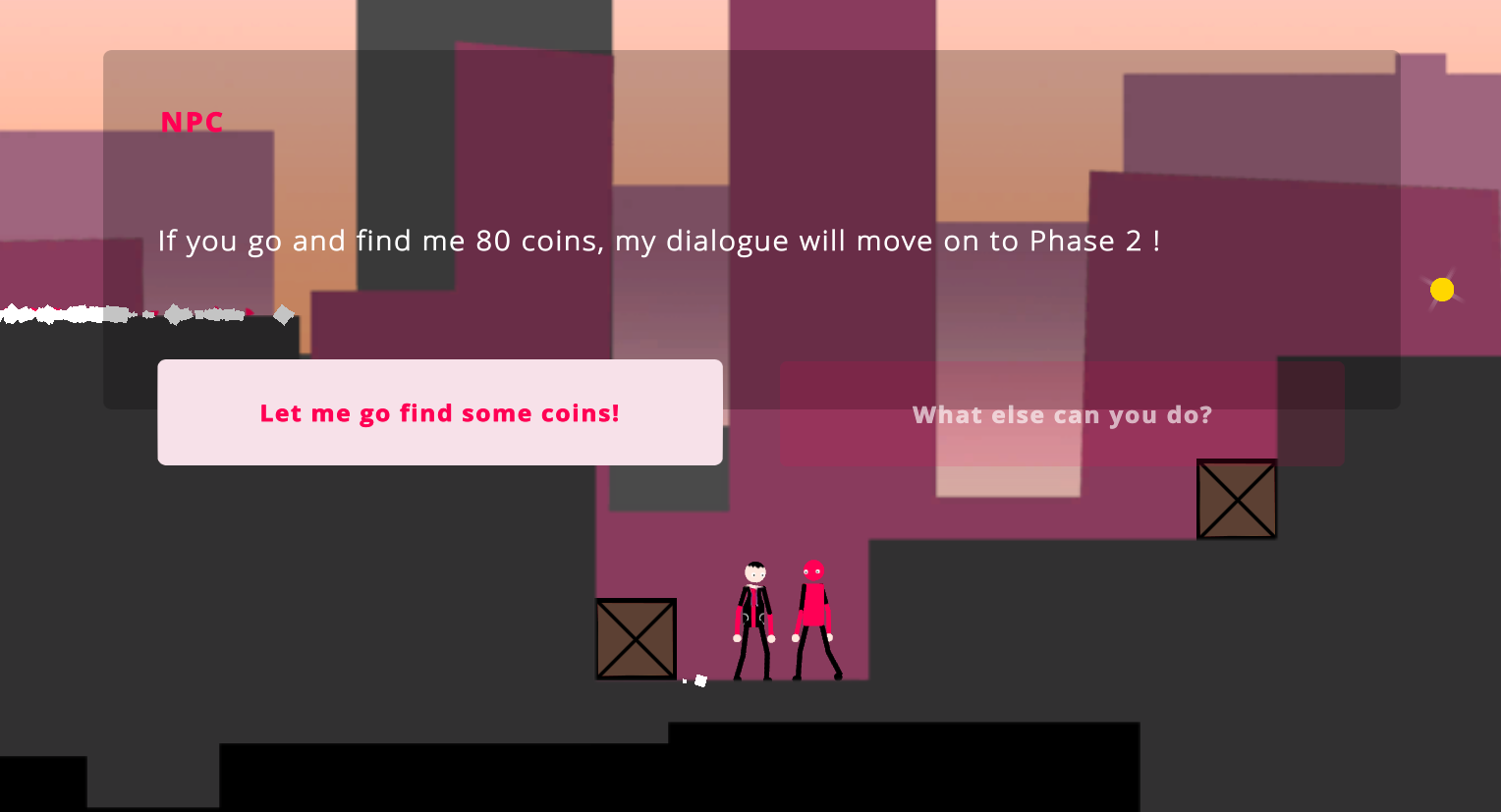
**Fig 4.5 Main UI**

****

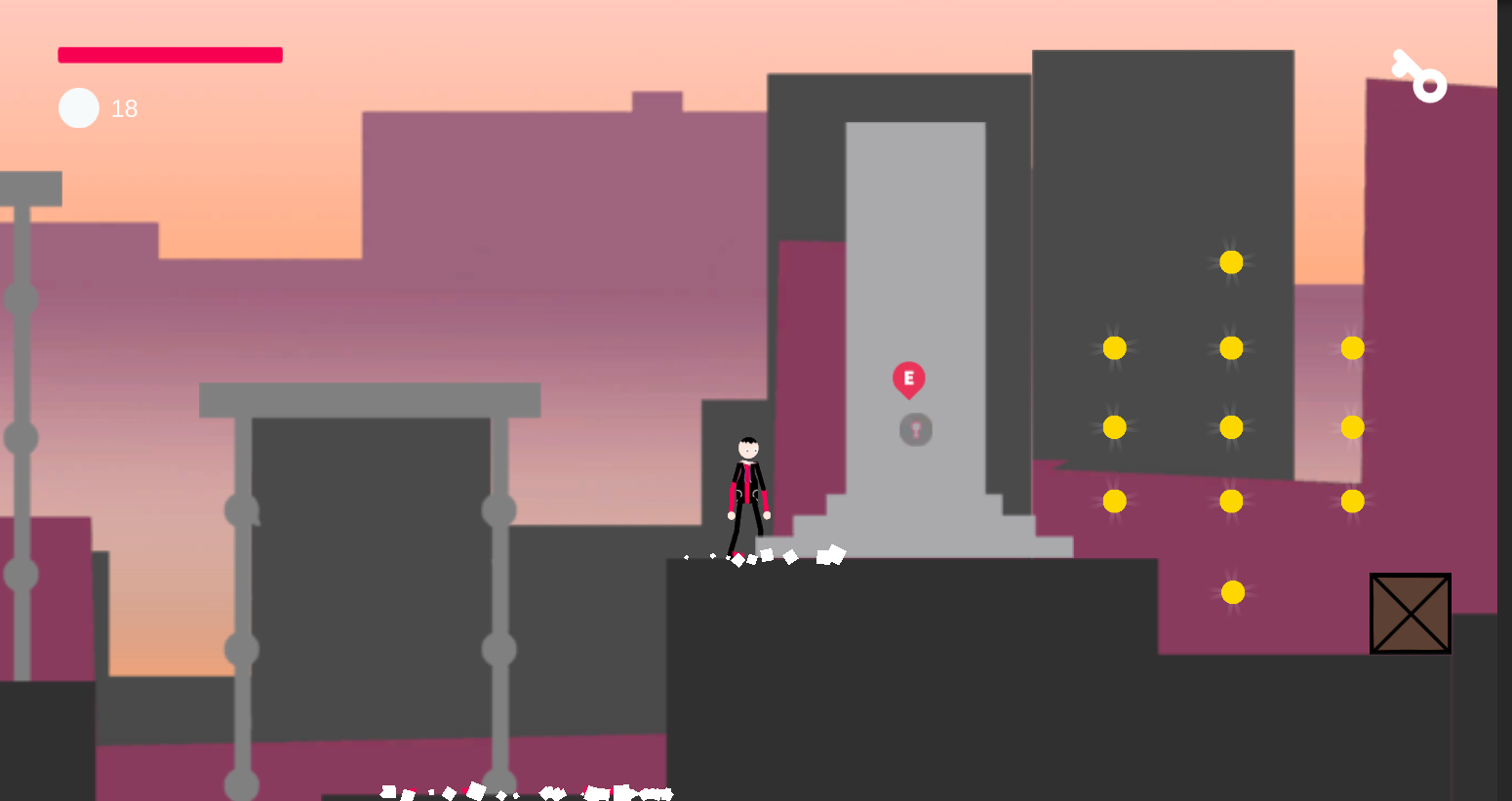
**Fig 4.6 Main Camera view with Gizmos**

****

**Fig 4.7 NPC Sprite**

****

**Fig 4.8 Dialogue With NPC**

****

**Fig 4.8 Interactable Objects**

**CHAPTER 5**

**CONCLUSIONS AND FURTHER SCOPE**

**5.1 CONCLUSION**

In conclusion, developing a 2D cybercity platformer game in Unity involves several aspects, including game design, programming, artwork, sound, and testing. The game design involves creating a plan for the game mechanics, game objectives, game environment, and game characters. The programming involves coding the game mechanics and game logic using Unity's programming language, C#. The artwork involves designing the game environment, characters, and other game assets using software such as Adobe Photoshop or Adobe Illustrator. The sound involves creating sound effects and background music to enhance the game's atmosphere. The testing involves testing the game to ensure that it is functional and that there are no bugs or glitches. Developing a 2D cybercity platformer game is a complex process that requires a team of developers with a diverse set of skills.

**5.2 FURTHER SCOPE**

The future scope for converting a 2D cybercity platformer game in Unity to 3D involves a few key considerations. The first consideration is the development team's expertise and experience with 3D game development. The team must have a strong understanding of 3D modeling, texturing, and animation techniques.

Overall, converting a 2D cybercity platformer game in Unity to 3D is a complex undertaking that requires a strong development team with experience in 3D game development. With careful planning and execution, the game's conversion can provide an exciting opportunity to create a new and improved version of the game that will delight players with its enhanced visuals and immersive experience.

**REFERENCES**

[1] Unity Technologies. (2021). Unity – Game Engine. Retrieved September 17, 2021, from <https://unity.com/>

[2] Brackeys. (2021). How to Make a 2D Platformer in Unity – Introduction. Retrieved September 17, 2021, from <https://www.youtube.com/watch?v=UbPiCgCkHTE>

[3] Unity Technologies. (2021). 2D Game Development. Retrieved September 17, 2021, from <https://unity.com/solutions/2d-game-development>

[4] Unity Technologies. (2021). Unity User Manual (2021.1 LTS). Retrieved September 17, 2021, from <https://docs.unity3d.com/Manual/index.html>

[5] Tornado Bandits Studio. (2021). Cyber Shadow: The Making of a Modern 2D Classic. Retrieved September 17, 2021, from <https://blog.tornadobandits.com/cyber-shadow-the-making-of-a-modern-2d-classic/>

[6] Doherty, D. (2020). How to Create a 2D Platformer in Unity: The Ultimate Guide. Retrieved September 17, 2021, from <https://www.gamedesigning.org/learn/unity-2d-platformer-tutorial/>

[7] CodinGame. (2021). Unity 2D Game Tutorial – Tutorial Guide for Beginners. Retrieved September 17, 2021, from <https://www.codingame.com/blog/unity-2d-game-tutorial/>