RAJALAKSHMI ENGINEERING COLLEGE

An AUTONOMOUS Institution

Affiliated to ANNA UNIVERSITY, Chennai

AUGMENTED/VIRTUAL REALITY PROJECT REPORT

AR – CHARACTER MODEL

VR – FLYING CASTLE

Submitted by

NAVEEN S 221701502

in partial fulfilment for the course

***C*S23B34 Immersive Experience in AR/VR**

for the degree of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND DESIGN

RAJALAKSHMI ENGINEERING COLLEGE

RAJALAKSHMI NAGAR

THANDALAM

CHENNAI-602 105

APRIL 2025

RAJALAKSHMI ENGINEERING COLLEGE

CHENNAI – 602105

BONAFIDE CERTIFICATE

Certified that this project report **"AR – CHARACTER MODEL, VR – FLYING CASTLE"** is the bonafide work of **NAVEEN S (221701502)**, who carried out the project work for the subject CS23B34 – Immersive Experience in AR/VR under my supervision.

SIGNATURE

Prof. Uma Maheshwar Rao

Head of the Department

Computer Science and Design

Rajalakshmi Engineering College

Chennai – 602105

SIGNATURE

Dr. L. Priya., Ph.D.,

IIIC Coordinator

Rajalakshmi Engineering College

Chennai - 602105

Submitted to Project and Viva Voce Examination for the subject

CS23B34 – Immersive Experience in AR/VR held on_____.

Internal Examiner

External Examiner

ACKNOWLEDGEMENT

Initially we thank the Almighty for being with us through every walk of our life and showering his blessings through the endeavour to put forth this report. Our sincere thanks to our Chairman **Mr. S. Meganathan, B.E, F.I.E.,** our Vice Chairman **Mr. Abhay Shankar Meganathan, B.E., M.S.,** and our respected Chairperson **Dr. (Mrs.) Thangam Meganathan, Ph.D.,** for providing us with the requisite infrastructure and sincere endeavouring in educating us in their premier institution.

Our sincere thanks to **Dr. S. N. Murugesan, M.E., Ph.D.,** our beloved Principal for his kind support and facilities provided to complete our work in time. We express our sincere thanks to our **Prof. Uma Maheshwar Rao** Associate Professor and Head of the Department of Computer Science and Design for his guidance and encouragement throughout the project work. We convey our sincere thanks to our internal guide and Project Coordinator, **Dr. L. Priya, Ph.D.,** IIIE Coordinator, Rajalakshmi Engineering College for his valuable guidance throughout the course of the project.

NAVEEN S (221701502)

TABLE OF CONTENT

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The evolution of technology has consistently reshaped how we interact with the world. Among the most transformative advancements are Augmented Reality (AR) and Virtual Reality (VR), technologies that merge the physical and digital realms to create immersive experiences. These innovations have opened up a multitude of possibilities in fields such as education, healthcare, gaming, architecture, and beyond. Coupled with powerful development tools like Blender and Unity Engine, AR and VR enable the creation of dynamic, interactive environments that captivate users and redefine engagement.

1.1 Augmented Reality (AR)

Augmented Reality enhances the real world by overlaying digital information such as text, images, and 3D objects onto the physical environment. This blending of virtual and physical spaces is achieved through devices like smartphones, tablets, and AR glasses, which utilize cameras, sensors, and software to detect and interact with the surroundings.

Key Features of AR

Real-Time Interaction: AR allows for real-time engagement with digital elements superimposed on the real world.

Device Versatility: From mobile apps to advanced AR headsets like Microsoft HoloLens, AR technology is accessible across various platforms.

Practical Applications: AR is used for navigation, retail, education, and entertainment, offering users a new dimension of interaction.

1.2 Virtual Reality (VR)

Virtual Reality immerses users in a fully digital environment, creating simulated worlds that may mirror reality or depart entirely from it. VR is experienced through headsets such as Oculus Quest or HTC Vive, which use visual, auditory, and sometimes tactile feedback to transport users into virtual spaces.

Key Features of VR

Immersive Environments: VR offers users a 360-degree experience, isolating them from the physical world.

Interactive Simulations: Virtual worlds respond to user input, enabling interaction with the environment.

Applications Across Industries: From gaming and entertainment to professional training and therapy, VR's potential is vast.

1.3 Blender Software

Blender is an open-source, all-in-one software tool for creating 3D content, including modeling, animation, rendering, and simulation. It is widely used in AR and VR projects due to its versatility and compatibility with other platforms. Blender is essential for creating the 3D assets required in AR/VR environments. Once assets are designed, they can be exported into game engines like Unity for further development. Blender's integration with AR/VR workflows makes it a cornerstone tool for immersive content creation. Unity provides specialized frameworks like AR Foundation and XR Interaction Toolkit to streamline the development of AR and VR applications. These frameworks offer built-in functionalities for environment mapping, object interaction, and multi-device support, significantly reducing development time and complexity.

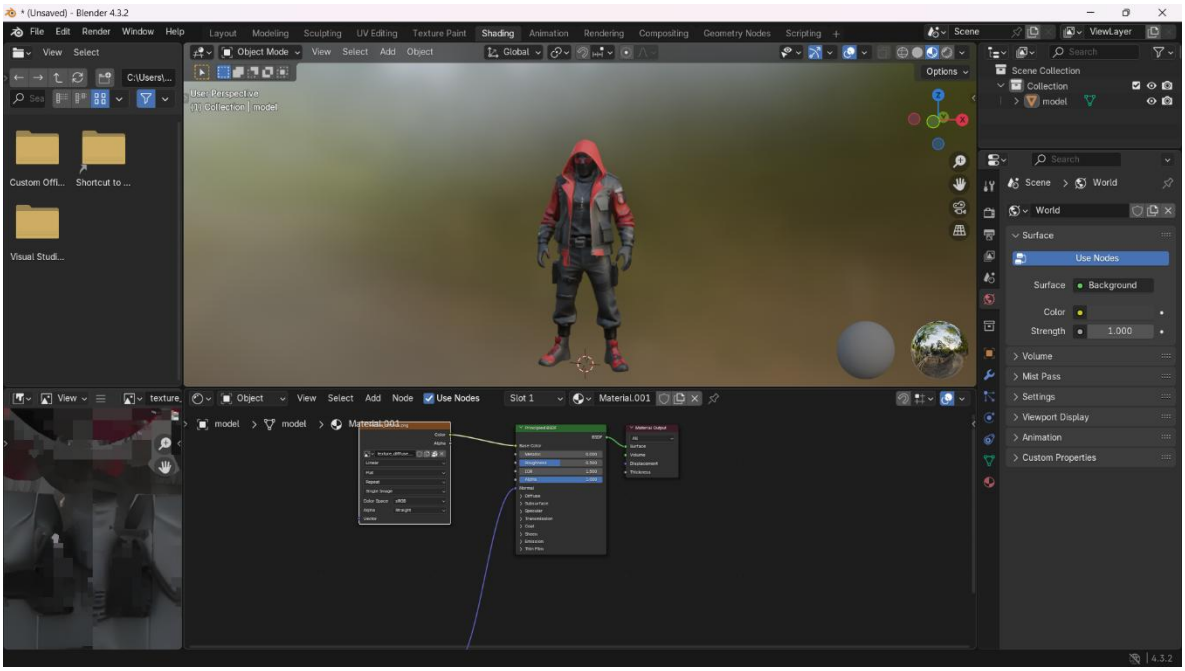

Fig 1.1 Sample Screenshot of animation in Blender

Key Features of Blender:

1. **Modeling**:
    - Blender offers a comprehensive set of modeling tools for creating both low and high-poly models.
    - It supports polygonal modeling, sculpting, and procedural modeling using modifiers.
2. **Sculpting**:
    - Blender provides advanced sculpting tools that are often used for detailed character creation or organic modeling.
    - Dynamic topology and multi-resolution sculpting allow for high-detail work.

3. **Texturing and Shading**:
    - You can create and apply textures using Blender's node-based material system (Cycles and Eevee engines).
    - It also supports PBR (Physically Based Rendering) workflows, allowing realistic material and texture creation.

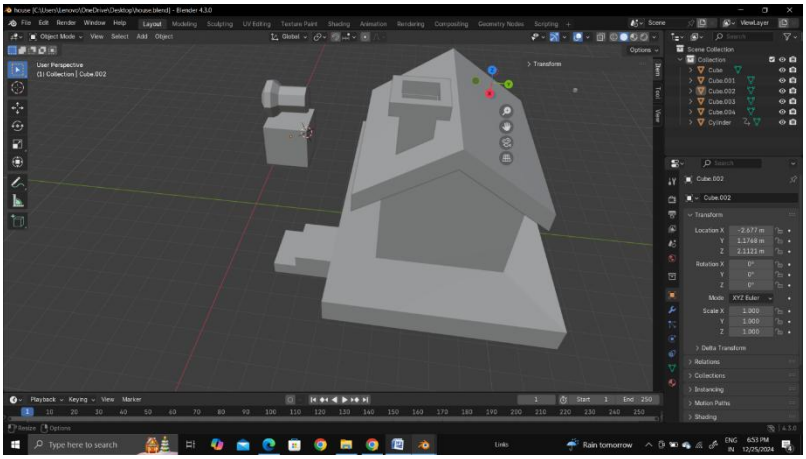

Fig 1.2 Sample Output Screenshot of Texturing in Blender

4. **Rigging and Animation**:
    - Blender has powerful rigging tools that allow for the creation of skeletal structures, inverse kinematics (IK), and character animation.
    - Non-linear animation (NLA) editor and action editor give control over complex animation tasks.
    - Blender also supports shape keys and physics simulations (cloth, smoke, etc.).

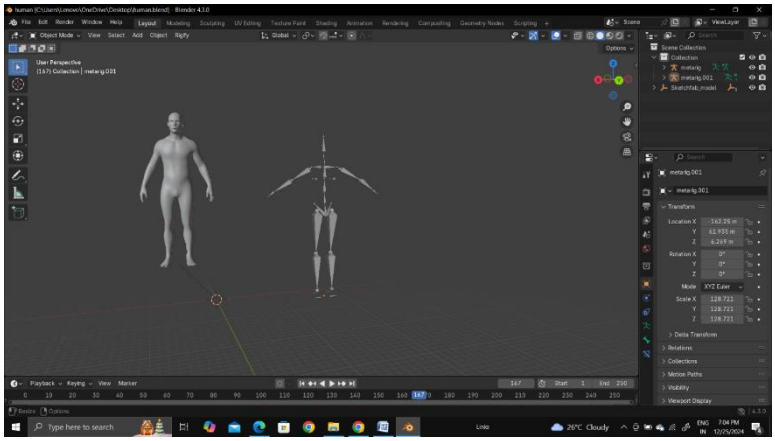

Fig 1.3 Sample Output Screenshot of Rigging in Blender

5. **Rendering**:
    - **Cycles**: A path-tracing renderer that produces high-quality images but requires more computational power.
    - **Eevee**: A real-time renderer that provides high-quality results with faster rendering times, ideal for real-time projects and previews.

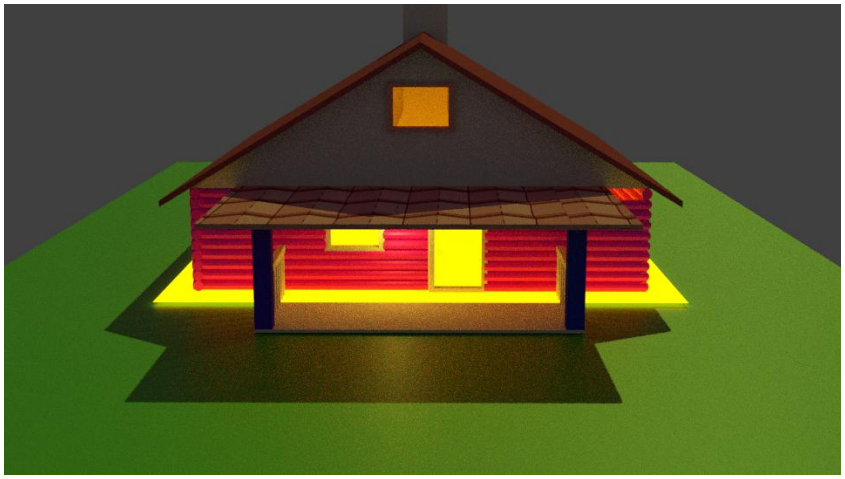

Fig 1.4 Sample Output Screenshot of Rendering in Blender

1.4 Unity Engine

Unity is a robust game engine and development platform that is widely used to create interactive applications across multiple platforms, including AR and VR. Its extensive features, ease of use, and strong community support make it an industry favorite.

Core Features of Unity

- **Cross-Platform Compatibility**: Unity supports deployment to mobile devices, desktops, consoles, and AR/VR devices.
- **Real-Time Development**: Unity's tools allow developers to visualize and test changes instantly.
- **Scripting with C#**: Developers can create complex behaviors and interactions using Unity's scripting environment.
- **Asset Store**: A vast library of pre-made assets, scripts, and tools simplifies the development process.

Augmented Reality, Virtual Reality, Blender, and Unity Engine collectively represent the foundation of modern immersive technology development. Together, they enable the creation of engaging, interactive, and impactful experiences that push the boundaries of what is possible in the digital world. Mastering these technologies equips developers to innovate across industries and deliver transformative applications.

1.5 Vuforia Engine

In Vuforia Engine, the **Image Target** option is used to detect and track images in the real world to display augmented reality (AR) content. Here's how to set up and use Image Targets in Vuforia:

Step-by-Step Guide to Using Image Target in Vuforia

1. Create a Vuforia Developer Account

- Go to the Vuforia Developer Portal and create a free account or log in if you already have one.t

2. Create a Vuforia License Key

- Once logged in, go to the **License Manager** section.
- Create a new license key for your project.
- Make a note of the license key because you'll need it later in your project.

3. Download and Set Up Unity (if you haven't already)

- Download and install Unity, which is the primary development environment for Vuforia.
- Open Unity and create a new project or open an existing one.

4. **Import Vuforia Engine into Unity**

- In Unity, go to the **Unity Asset Store** and search for **Vuforia Engine**.
- Download and import the Vuforia Engine package into your Unity project.

5. Enable Vuforia in Unity

- Go to **Edit > Project Settings > Player**.
- In the **Player Settings** window, click on the **XR Settings** tab.
- Check the **Vuforia Augmented Reality** box to enable Vuforia.

6. Set Up Image Targets in Vuforia

- Open the **Vuforia Configuration** by going to **Window > Vuforia Configuration**.

- In the Vuforia Configuration window, paste your license key (the one you generated earlier).

7. Create an Image Target

- In Unity, create an **Image Target** by right-clicking in the **Hierarchy** window and selecting **Vuforia > Image Target**.
- You should now see an **Image Target** object in your scene.

8. Upload Your Target Image to Vuforia

- Go to the Vuforia Developer Portal and navigate to the **Target Manager**.
- Create a new database and upload the image you want to use as the target.
- Make sure the image is of high quality and has good contrast for better tracking.
- After uploading, download the database and import it into Unity.

9. Configure the Image Target

- In Unity, select the **Image Target** object in your scene.
- In the **Inspector** window, under the **Image Target Behaviour** component, select the image target from the **Database** drop-down.
- Choose the specific image you uploaded in the **Image Target** section.

10. Add AR Content to the Image Target

- You can now add any 3D model, text, or other AR content as children to the Image Target object in the hierarchy.
- For example, to display a 3D model when the image target is detected, drag and drop a 3D model under the Image Target.

11. Test the Scene

- Click **Play** in Unity to test the scene. If everything is set up correctly, the AR content will appear when the image target is detected by the camera.

12. Build and Deploy the Project

- Once you're satisfied with the results in the Unity Editor, you can build and deploy the project to a mobile device.
- For mobile deployment, go to **File > Build Settings**, select your platform (Android or iOS), and then click **Build**.

CHAPTER 2

Augmented Reality Project – Character Model

2.1 Project Overview

The AR Character Viewer is an augmented reality (AR) application developed using Unity and Vuforia, designed to display and interact with a 3D character model in real-world environments. The 3D model, created in Blender, is textured, lit, and placed within a custom environment inside Unity. Using Vuforia's advanced tracking features such as Image Target, Ground Plane, and Mid-Air, users can view and manipulate the character model through interactive UI controls for scaling, rotation, and movement. This project demonstrates how AR can be used to present digital characters in physical spaces, opening up possibilities for virtual exhibitions, games, and educational tools.

2.2 Development Process

2.2.1. Requirement Analysis & Concept Design

- Defined the goal: to allow users to view and interact with a 3D character model using AR.
- Chose Vuforia for tracking and Unity as the development platform.
- Planned for multiple tracking types (Image Target, Ground Plane, Mid-Air) for versatility.
- Designed interactive buttons for scaling, rotating, and moving the 3D model.

2.2.2. Environment Setup

- Installed Unity (latest stable version) and integrated the Vuforia Engine SDK.
- Created a new Unity 3D project and configured the AR Camera.

• Registered the project on the Vuforia Developer Portal and created target databases.

2.2.3. Image Target, Ground Plane, and Mid-Air Setup

• Generated databases for image targets and enabled Ground Plane and Mid-Air in Vuforia configuration.

• Imported the databases into Unity and assigned them to the relevant AR components.

• Placed Image Target, Plane Finder (for Ground Plane), and Mid-Air Anchor in the scene for testing different tracking types.

2.2.4. 3D Model Integration

• Imported the character model (in .fbx format) from Blender.

• Added textures and materials for a realistic appearance.

• Adjusted lighting and environmental elements to enhance realism.

• Embedded the 3D model into the Image Target, Ground Plane, and Mid-Air GameObjects.

2.2.5. Interactivity

• Added UI buttons for scaling, rotating, and moving the character.

• Attached scripts to the buttons to allow real-time interaction with the model.

• Ensured that user input felt intuitive and responsive.

2.2.6. Testing & Debugging

• Tested the app across multiple devices and lighting conditions.

• Adjusted the character’s placement and orientation for different tracking methods.

• Debugged issues with touch interaction, object alignment, and lighting consistency.

2.2.7. Build & Deployment

- Configured build settings for Android.
- Built the APK and installed it on a test device.
- Conducted user testing and implemented improvements based on feedback.

2.3 Step-by-Step Procedure to Implement the Project

Step 1: Install and Set Up Tools

1. Install Unity Hub and a compatible version of Unity.
2. During installation, add support for:
   - Android Build Support
   - OpenJDK
   - Android SDK & NDK
3. Create a new Unity Project (3D template).

Step 2: Set Up Vuforia in Unity

1. Go to developer.vuforia.com and create an account.
2. Generate a license key for your project.
3. In Unity:
   - Go to Edit > Project Settings > Player > XR Settings.
   - Enable “Vuforia Augmented Reality Support.”
4. Import the Vuforia Engine package into the project.

Step 3: Add Image Target, Ground Plane, and Mid-Air

1. On the Vuforia developer portal:
   - Create a target database and upload an image to use as an Image Target.
   - Download the database and import it into Unity.
2. In Unity:
   - Delete the default Main Camera.

- Add an AR Camera (Vuforia > AR Camera).
- Add an Image Target and assign the database.
- Add Plane Finder and Ground Plane Stage for ground detection.
- Add a Mid-Air Anchor for mid-air placement.

Step 4: Import and Set Up 3D Character Model

1. Import your 3D character model (.fbx) from Blender into Unity.
2. Drag the model into the Image Target, Ground Plane, and Mid-Air anchors.
3. Position, scale, and rotate the model to ensure correct alignment.
4. Apply materials and textures for realism.

Step 5: Add Interactivity

1. Create UI buttons for Rotate, Scale, and Move.
2. Write or use scripts to control the model's transform based on button inputs.
3. Optionally integrate Unity's Lean Touch asset for intuitive gestures.
4. Ensure that the controls work consistently across all tracking modes.

Step 6: Build and Deploy

1. Go to File > Build Settings, select Android, then click “Switch Platform.”
2. Set up Player Settings (app name, orientation, icon, etc.).
3. Connect your Android device and enable Developer Mode with USB Debugging.
4. Click “Build and Run” to deploy and test the app on your phone.

Step 7: Testing and Final Touches

1. Test each AR mode (Image Target, Ground Plane, Mid-Air) in different environments.
2. Adjust character alignment and lighting for maximum realism.

3. Confirm the functionality of all interaction buttons.

4. Optimize performance and package the final APK for distribution or demonstration.

2.4 Output and Discussion

Fig 2.1 Output Screenshot of Image target

Fig 2.2 Output Screenshot of Mid Air

Fig 2.3 Output Screenshot of Ground Plane

The project effectively demonstrates how Vuforia's image recognition can be used to simulate virtual product try-ons. By targeting a static image of a hand, the app bypasses the complexities of hand tracking while still achieving a realistic wristwatch preview.

2.5 Conclusion

The development of the AR-based virtual watch try-on application demonstrates the practical use of Augmented Reality and image targeting to enhance product visualization and user experience. By leveraging Unity and Vuforia, the project successfully overlays a realistic 3D watch model onto a predefined hand image, simulating the effect of wearing a wristwatch in real time.The application not only showcases the potential of AR in the retail and fashion industries but also highlights how immersive technology can be used for interactive and engaging customer experiences. Although currently limited to static image targets, the project lays a strong foundation for future enhancements such as real-time hand tracking, gesture controls, and broader product customization.

CHAPTER 3

Virtual Reality Project – Flying Castle

3.1 Project Overview

The VR Flying Castle project is a virtual reality experience developed in Unity, designed to immerse users in a fantastical floating environment. The main objective of this project is to let users explore and interact with a magical castle suspended in the sky using VR controllers. Through the use of Unity's XR Interaction Toolkit, users can grab and interact with objects placed throughout the flying castle, enhancing the sense of presence and immersion. This project showcases how VR can be used to bring imaginative settings to life with intuitive and interactive elements.

3.2 Module Descriptions

3.2.1. Player Interaction Module

Purpose: Enables players to interact with the floating castle environment using VR controllers.

Key Features:

• Grabbing 3D objects using XR Grab Interactable.

• Releasing objects using controller input (grip/trigger).

• Provides haptic feedback during interactions.

Tools Used: XR Interaction Toolkit, Unity Input System, C#

3.2.2. Object Physics & Interactivity Module

Purpose: Simulates object behavior and responsiveness within the floating castle environment.

Key Features:

• Rigidbody and colliders applied to interactive objects.

• Realistic response to user input and gravity.

• Physics-enabled elements for floating, bouncing, or falling effects.

Tools Used: Unity Physics Engine, Rigidbody, Collider

3.2.3. Environment & Visual Design Module

Purpose: Builds the flying castle and creates a visually stunning environment.

Key Features:

• Modeled flying castle with floating platforms and detailed architecture.

• Skybox, clouds, and ambient lighting for aerial atmosphere.

• Optional animated elements like floating lanterns or birds.

Tools Used: Unity 3D Models, Lighting System, Skybox, Particle System

3.2.4. Interaction Feedback Module

Purpose: Enhances user engagement through visual and tactile responses.

Key Features:

• Haptic feedback when grabbing or interacting with objects.

• Optional visual cues for interactable elements.

• Audio feedback for grabs, releases, or environmental effects.

Tools Used: AudioSource, XR Haptics, Unity UI Toolkit (optional)

3.2.5. Game Management & Scene Control Module

Purpose: Manages the game state and scene transitions.

Key Features:

• Controls scene initialization and object states.

• Optional reset mechanism for repositioning objects.

• Supports expanding the experience with puzzles or exploration goals.

Tools Used: Custom GameManager script, C#, Singleton pattern (optional)

3.3 Step-by-Step Procedure to Implement the Project

Step 1: Set Up Unity Project

• Create a new 3D Unity project.

• Install the XR Interaction Toolkit and relevant XR plugin (e.g., Oculus XR).
• Enable XR Plugin Management and configure settings for your headset (e.g., Oculus Quest).

Step 2: Design the Environment

• Build the flying castle environment with 3D models (castle, floating islands, etc.).
• Add lighting, skybox, and particle effects for an immersive atmosphere.

Step 3: Add XR Rig

• Add XR Origin (Action-Based) to the scene.
• Set up VR camera and controller inputs.
• Optional: Include teleportation or smooth locomotion systems for movement.

Step 4: Add Interactable Objects

• Import or create 3D objects for interaction (e.g., magical artifacts, furniture, tools).
• Add XR Grab Interactable component to each object.
• Enable Rigidbody and appropriate Colliders to support physics and grabbing.

Step 5: Implement Object Behavior

• Use physics to simulate realistic movement (e.g., gravity on dropped objects).
• Add custom scripts for any special effects or object-specific behaviors.

Step 6: Feedback Systems

• Add haptic feedback when objects are grabbed or dropped.
• Add optional sound effects or visual indicators for interaction.
• Enhance realism with ambient sounds like wind, castle creaks, or magical hums.

Step 7: Game Management

• Create a GameManager script to manage interactions, resets, or scene state.

• Add functionality for scene transitions or object respawning if needed.

Step 8: Build & Test

• Optimize scene for performance on your VR device.

• Build and deploy to the Oculus Quest (or your target VR headset).

• Test all interactions and visuals for polish and realism.

3.4 Output and Discussion

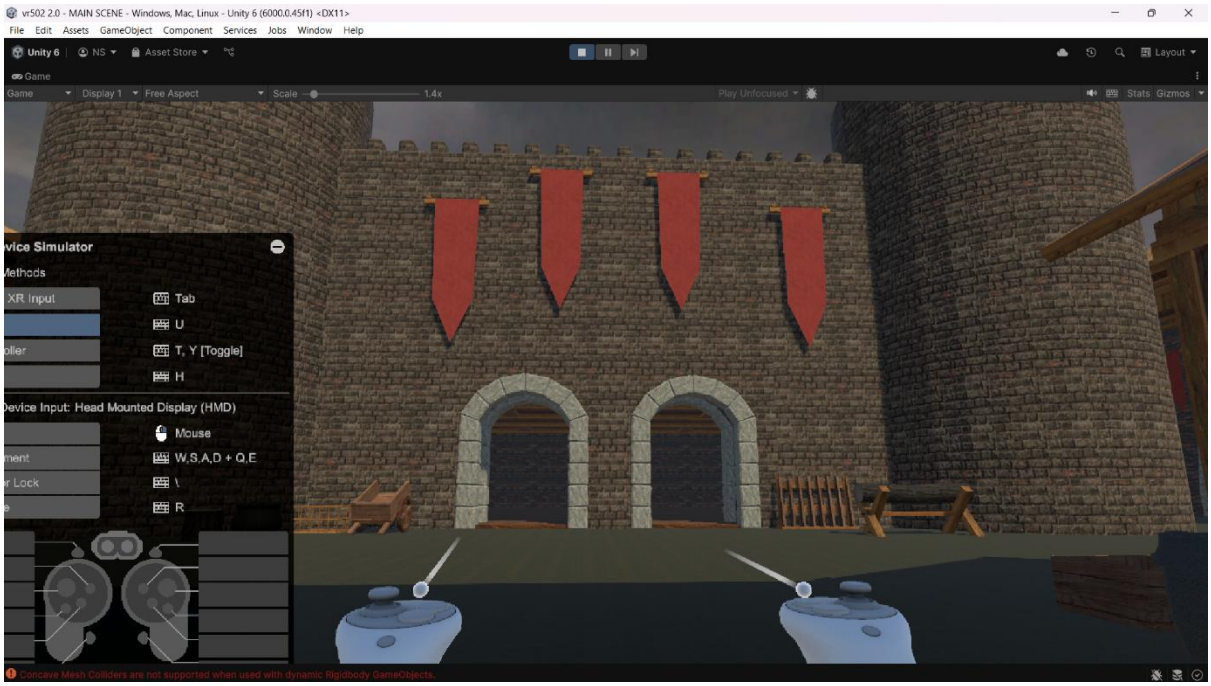

Fig 3.1 Output Screenshot of Flying Castle

The output of the VR Flying Castle project shows a well-designed virtual environment with a medieval castle, integrated VR controllers, and interactive elements. The scene confirms that XR input and object interaction using XR Grab Interactable and colliders are working correctly. The active Unity Device Simulator indicates the project is being tested successfully. A minor warning about mesh colliders suggests some physics adjustments may be needed, but overall, the implementation is effective and aligned with the project goals.

3.5Conclusion

The VR Flying Castle project successfully showcases the immersive potential of virtual reality in a fantasy-themed environment. Developed in Unity using the XR Interaction Toolkit, the experience allows users to explore and interact with a detailed 3D castle scene using VR controllers. Key features such as XR Grab Interactable components and colliders were effectively implemented to enable natural object interaction and enhance user immersion. While minor challenges like collider optimization were encountered, the final output is visually engaging, functional, and responsive. This project demonstrates the power of VR in creative world-building and interactive design, paving the way for future developments such as flying mechanics, teleportation, or narrative-driven gameplay experiences.