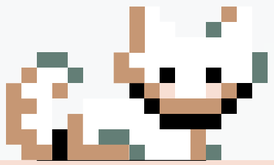
PROJECT MENTORS :

- ARYAN VERMA
- ABHISHEK MEENA

VISION VORTEX
INTRO TO AR/VR IN GAME DEVELOPMENT

GAME DEV CLUB,
IIT KANPUR



PROJECT OBJECTIVES

Exploring AR development [**WEEK 1-5**]

- Learning interactions
- Tapping using Google ARCore
- Object placement using GEO AR

Learning VR Development [**WEEK 6-8**]

- Focusing on mechanics like locomotion
- Maximizing optimization

PROJECT TIMELINE

WEEK 01

Basics of Unity and C# language

WEEK 02

Intro to AR Foundation

WEEK 03

AR Placement Indicator Using Raycast

WEEK 04

GeoSpatial Creator in Unity

PROJECT TIMELINE

WEEK 05

Setting up Unity for VR Development

WEEK 06

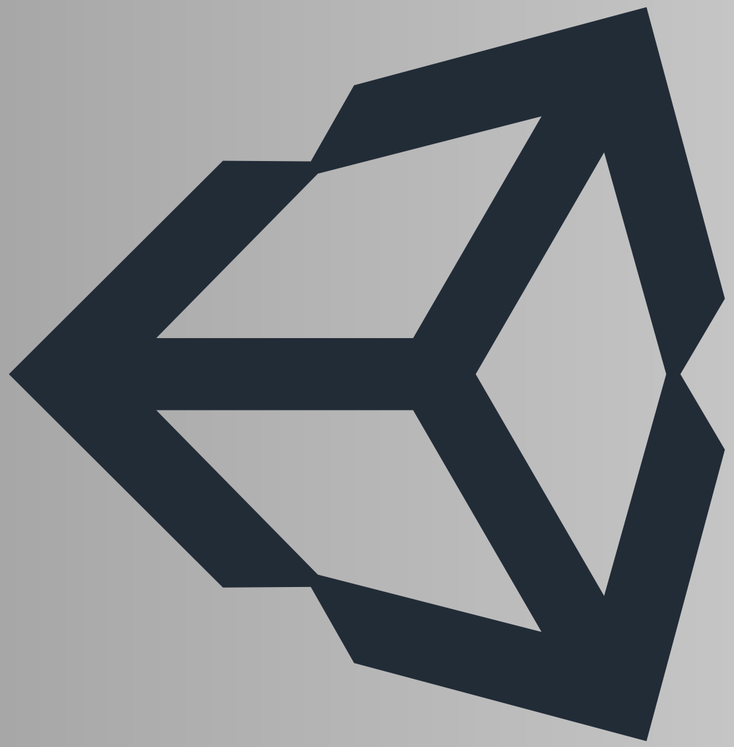
Input and Hand Presence with Continuous Movement

WEEK 07

Teleportation in VR

WEEK 08

Hover, Use, Grab Interactable



WEEK 1

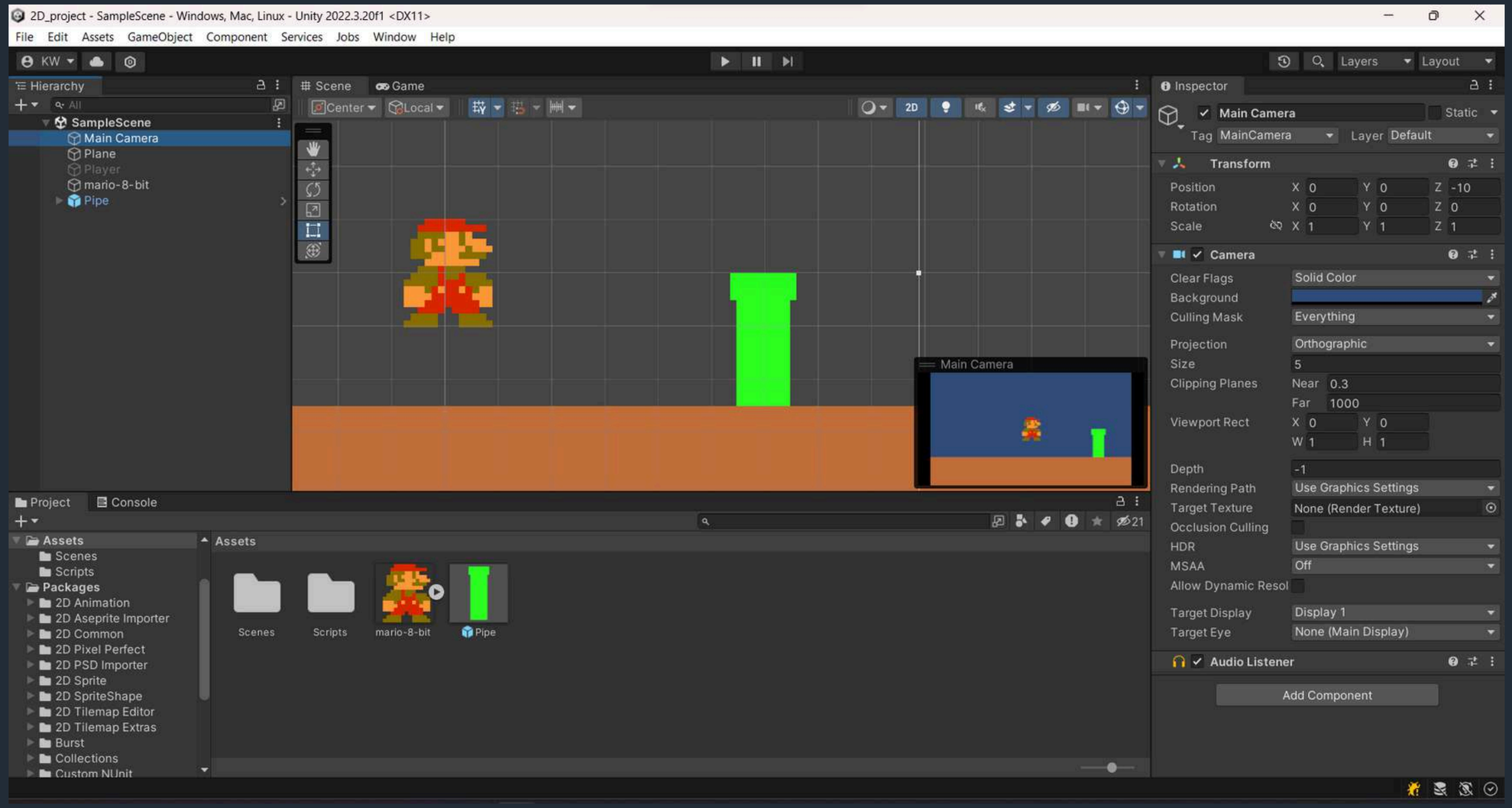
BASICS OF UNITY AND C# LANGUAGE

INTRODUCTION

Unity is a powerful game development platform that is used to create 2D, 3D or AR/VR games and other interactive experiences.

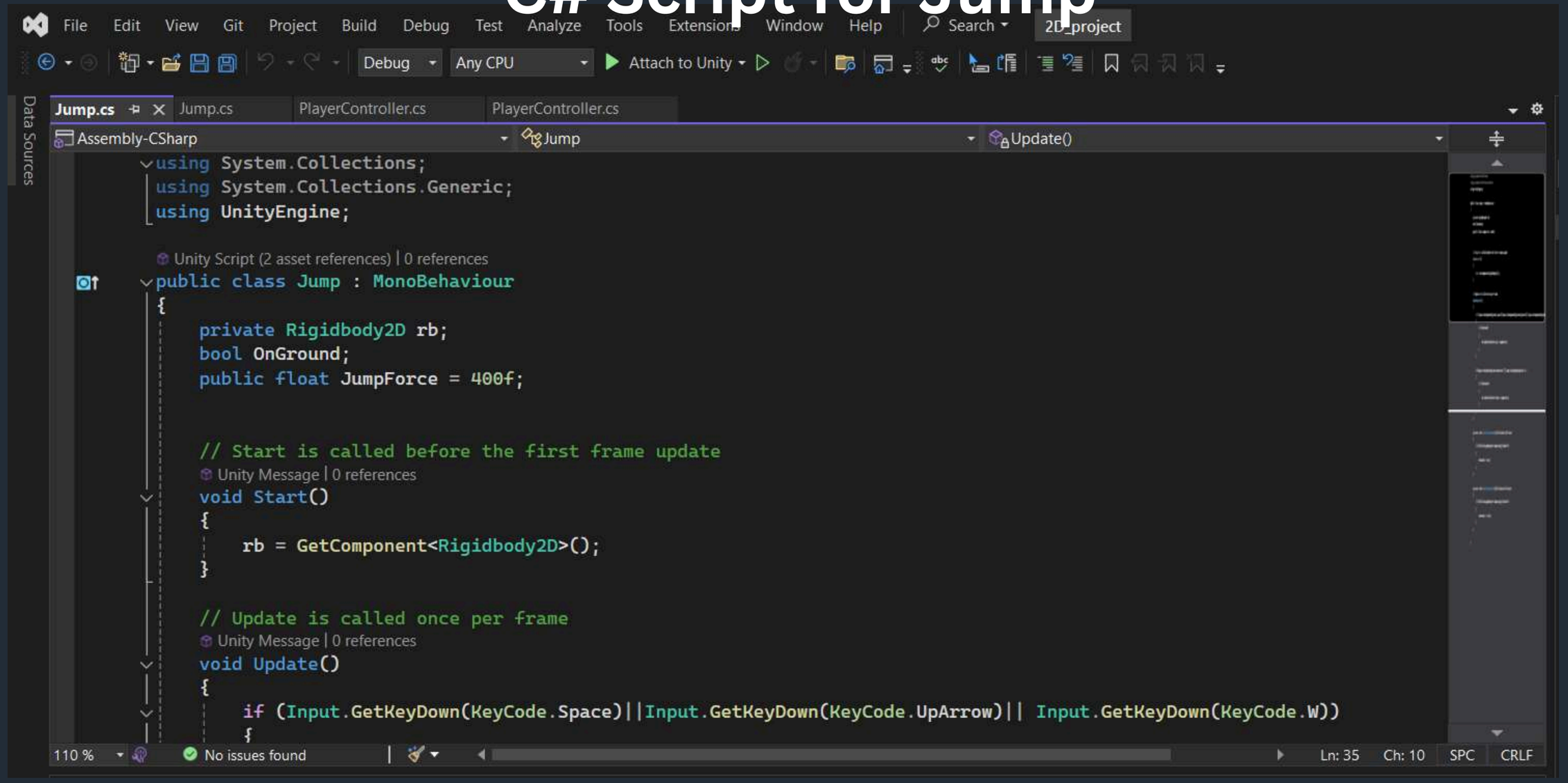
Unity Interface Overview:

1. Unity Hub:
   - Manage different versions of Unity you have installed.
   - Create, view and manage all your Unity projects.
2. Unity Editor Layout:
   - Scene View: The main workspace where you create and arrange your game objects.
   - Game View: A preview of what the player will see during gameplay.
   - Hierarchy: A list of all game objects in your current scene.
   - Inspector: Displays properties and settings for the selected game object.
   - Project Window: Shows your project's assets (models, scripts, textures, etc.).
   - Console: Displays messages, warnings, errors, or debug log from the editor and scripts.



Unity 2D Project

C# Script for Jump



Creating Our First Game

2D PLAYER CONTROLLER

- Added two 2D square sprites as Game Objects. One is used as the plane, other as the player.
  - Plane: Transform (Position, Scale), RigidBody 2D (Static), Box Collider 2D.
  - Player: Transform (Position, Scale), RigidBody
- Create 2 scripts:
  - Player Controller: Detect input from A/Left Arrow and D/Right Arrow Keys. Upon key press, add a movement speed to the player.
  - Jump: On detecting input from W/Up Arrow key, add a jump force to the player. No jump force if player does not collide with ground.
- Running the Game:
  - Play Mode: Test your game and interact with your Game Objects.
  - Stop Mode: Click the Play button again to stop Play Mode (Don’t forget to do this before editing).

WEEK 2

INTRO TO AR FOUNDATION

INTRODUCTION

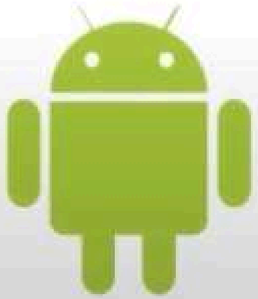
AR Foundation enables us to create multi-platform augmented reality (AR) apps with Unity. In an AR Foundation project, we choose which AR features to enable by adding the corresponding manager components to our scene. When we build and run our app on an AR device, AR Foundation enables these features using the platform's native AR SDK, so we can create once and deploy to the world's leading AR platforms. AR Foundation provider plug-ins rely on platform implementations of AR features, such as Google's ARCore on Android and Apple's ARKit on iOS. Unity normally imports assets automatically when they are dragged into the project but it is also possible to import them under script control.

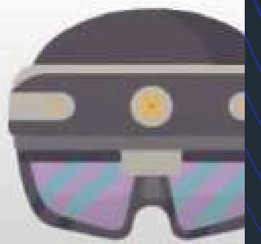
AR Foundation

Subsystems

-XRSessionSubsystem
-XRCameraSubsystem
-XRPlaneSubsystem
-XRImageTrackingSubsystem

Providers

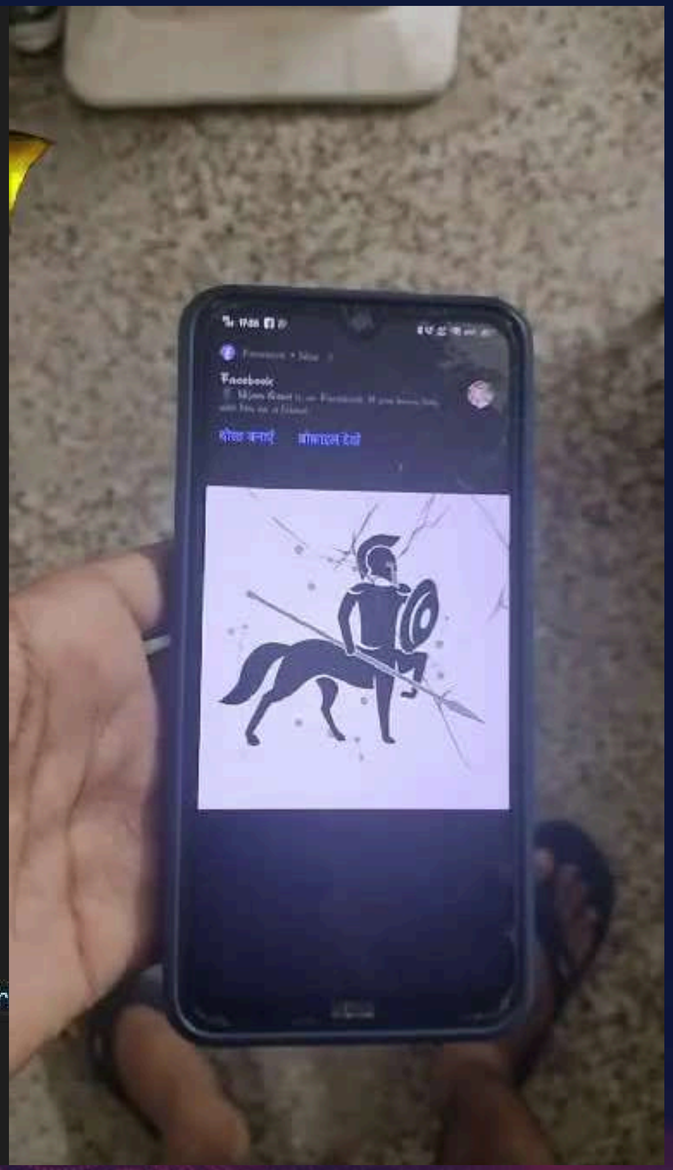


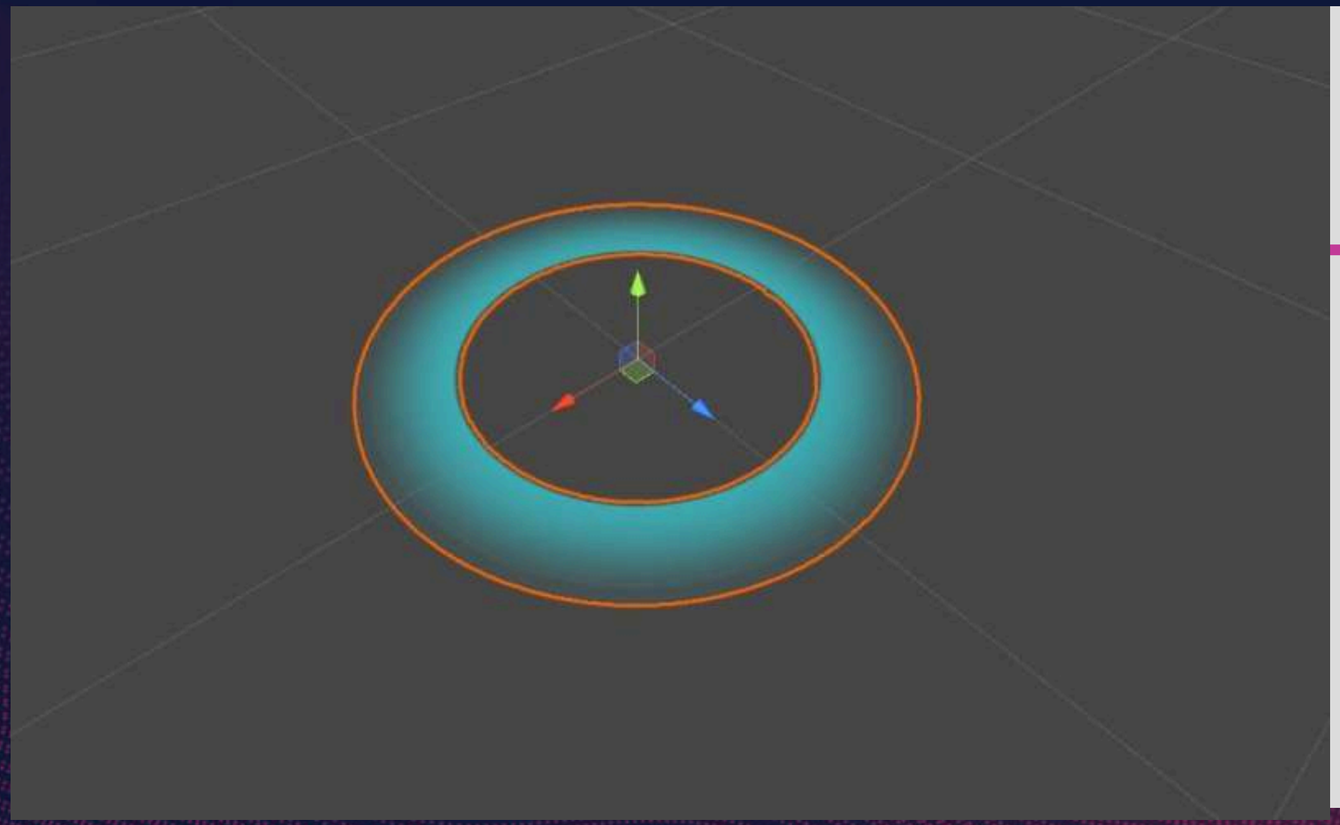


PROJECT USING AR FOUNDATION

- Platform-specific packages: Apple ARKit for iOS, Google ARCore for android.
- Configure Build Settings (switch platform) and change Player Settings.
- Add AR Session and AR Session Origin in Hierarchy to set up the AR functionality.
- Create a simple prefab and script to place object attach Script to AR Session Origin
- Build and Deploy patch and run device using USB cable.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class dragon_controller : MonoBehaviour
{
    [SerializeField] private float speed;

    private FixedJoystick fixedJoystick;
    private Rigidbody rigidBody;

    private void OnEnable()
    {
        fixedJoystick = FindObjectOfType<FixedJoystick>();
        rigidBody = GetComponent<Rigidbody>();
    }

    private void FixedUpdate()
    {
        float xVal = fixedJoystick.Horizontal;
        float yVal = fixedJoystick.Vertical;

        Vector3 movement = new Vector3 (xVal, 0, yVal);
        rigidBody.velocity = movement*speed;

        if( xVal != 0 && yVal != 0)
        {
            transform.eulerAngles = new Vector3(transform.eulerAngles.x, Mathf.Atan2(xVal, yVal) * Mathf.Rad2Deg, tr

        }
    }
}
```





WEEK 3

AR PLACEMENENT INDICATOR USING RAYCAST

INTRODUCTION

First we create a new project in unity using ARCore Mobile template.

ARCore Mobile Template
The ARCore Mobile Project Template configures project settings for Unity applications that wish to use Augmented Reality on mobile devices

Raycast
A raycast is conceptually like a laser beam that is fired from a point in space along a particular .Any object making contact with the beam can be detected and reported.

AR Placement Indicator
This is a template project for Unity's AR Foundation system. It's ready to build and includes an AR placement indicator. This is a reticle that tracks the center of your screen in the AR world. So it snaps to surfaces such as tables, the floor, walls, etc. It can be used to spawn in objects, or any other feature you wish to include in your AR app.

Target cursor plane detection

This script allows an AR application to detect planes in the AR environment and place an indicator at the position and orientation of the detected plane, centered in the middle of the screen. The indicator is initially inactive and becomes active once a plane is detected. This is useful for visualizing where virtual objects can be placed in the AR scene.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

using UnityEngine.XR.ARFoundation;
using UnityEngine.XR.ARSubsystems;
public class PlaceIndicator : MonoBehaviour
{
    private ARRaycastManager raycastManager;
    private GameObject indicator;

    private List<ARRaycastHit> hitList = new List<ARRaycastHit>();
    void Start()
    {
        raycastManager = FindObjectOfType<ARRaycastManager>();
        indicator = transform.GetChild(0).gameObject;
        indicator.SetActive(false);

    }

    // Update is called once per frame
    void Update()
    {
        var ray = new Vector2(Screen.width/2, Screen.height/2);

        if(raycastManager.Raycast(ray, hitList, TrackableType.Planes))
        {
            Pose hitPose = hitList[0].pose;

            transform.position = hitPose.position;
            transform.rotation = hitPose.rotation;

            if(!indicator.activeInHierarchy)
            {
                indicator.SetActive(true);
            }
        }
```

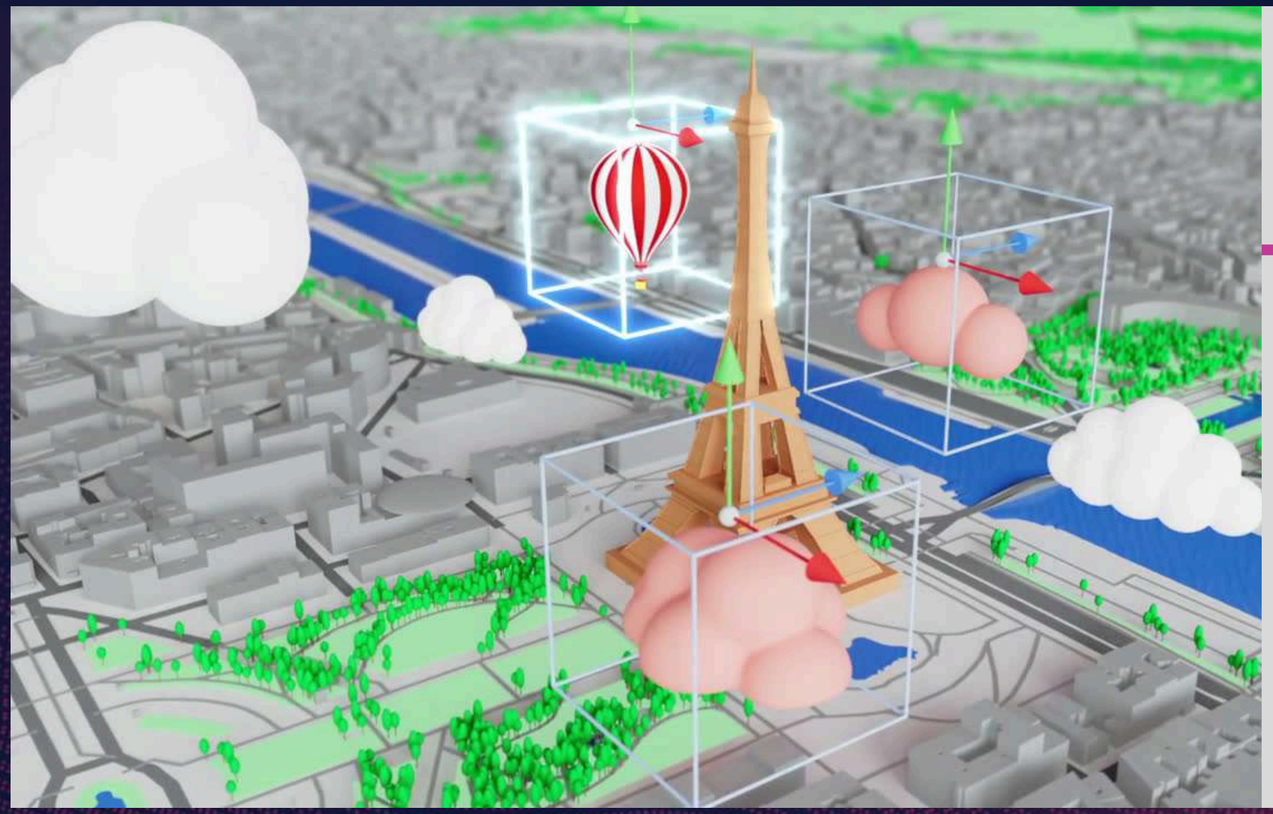
Placing Objects

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PlaceManager : MonoBehaviour
{
    private PlaceIndicator placeIndicator;

    public GameObject objectToPlace;
    private GameObject newPlacedObject;
    void Start()
    {
        placeIndicator = FindAnyObjectByType<PlaceIndicator>();
    }

    public void ClickToPlace()
    {
        newPlacedObject = Instantiate(objectToPlace, placeIndicator.transform.position, placeIndicator.transform.rotation);
    }
}
```

This script manages the placement of objects in an AR environment:

- It uses a PlaceIndicator to determine where to place the objects.
- The Start method initializes the PlaceIndicator.
- The ClickToPlace method instantiates a specified object (objectToPlace) at the position and rotation of the PlaceIndicator.

This script allows for dynamic object placement in an AR scene by linking object instantiation to the position and orientation determined by the PlaceIndicator.



WEEK 4

GEOSPATIAL CREATOR IN UNITY

INTRODUCTION

The ARCore Geospatial Creator for Unity allows developers to preview and add geospatial content within the Unity Editor by utilizing Google Maps data in a new 3D tiles format. This enables visualization of where content will be placed in the real world while building the app.

Cesium for Unity is used to visualize massive high-resolution real-world photogrammetry and 3D geospatial content at runtime using 3D Tiles. It is integrated with Unity's Game Objects, Components, Character Controllers, and more. Has support for VR platforms such as Quest 2 and Quest Pro.

World space AR Project

1. Installed AR Foundation and ARCore Extensions

Opened Unity Hub, created a new project, and selected the 3D template.
Opened the Package Manager (Window > Package Manager).
Searched for and installed AR Foundation.
Also searched for and installed ARCore XR Plugin.

2. Set Up ARCore Extensions

Downloaded ARCore Extensions for AR Foundation from the Google GitHub repository.
Imported the ARCore Extensions package into the Unity project.
Added the ARCore Extensions GameObject to the scene by going to GameObject > ARCore Extensions > ARCore Extensions.

3. Configured AR Settings

In the Project Settings (under Edit > Project Settings), navigated to the XR Plug-in Management section.
Enabled ARCore for Android.
Navigated to XR > ARCore Extensions Config in the Project Settings.
Set up the API key for ARCore services as required.

World space AR Project

4. Set Up Google Maps API

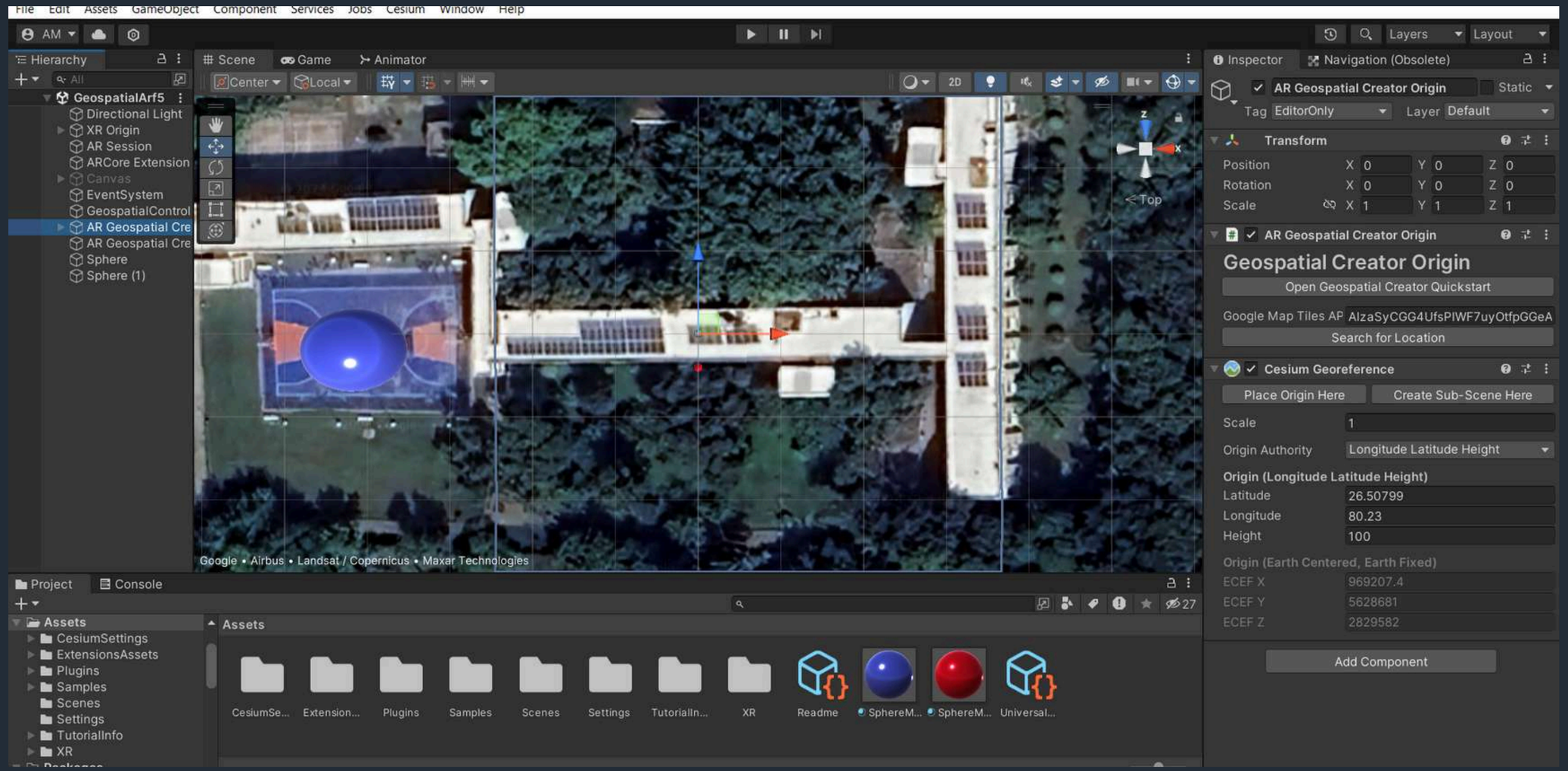
Went to the Google Cloud Console.
Created a new project or selected an existing project.
Enabled the Maps SDK for Unity API.
Generated an API key and restricted it to the Maps SDK for Unity API.
Configured the Unity project to use the API key by adding it to the relevant ARCore Extensions component or through the code.

5. Installed and Configured Cesium for Unity

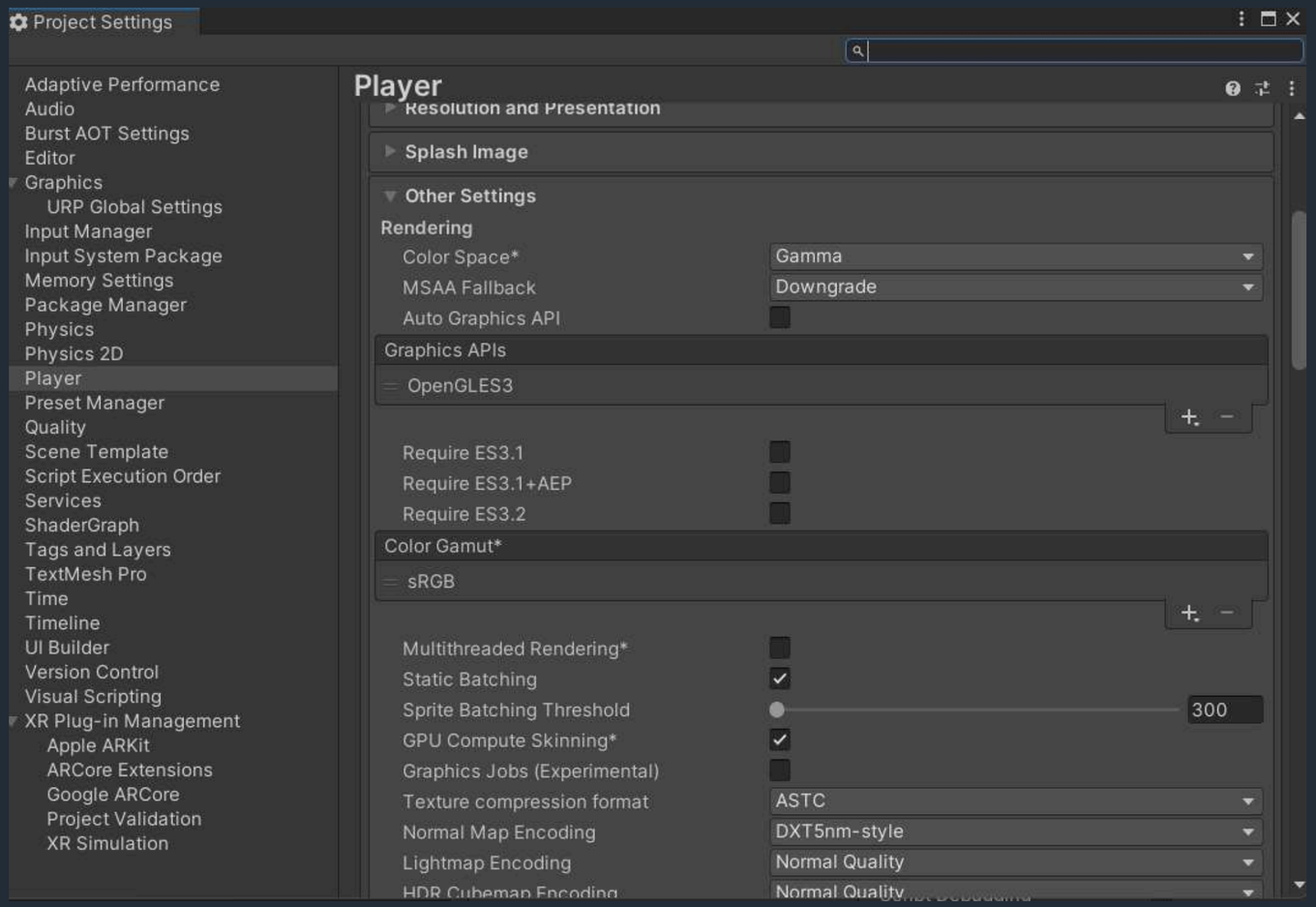
Downloaded the Cesium for Unity package from the Cesium for Unity GitHub repository.
Imported the Cesium package into the Unity project.
Added the Cesium World Terrain to the scene by navigating to GameObject > Cesium > Cesium World Terrain.
Configured the Cesium settings, including adding the API token obtained from the Cesium ion platform.

6. Set Up Geospatial Anchors

Created an empty GameObject and named it AR Session Origin.
Added an AR Session Origin component to the AR Session Origin GameObject.
Added an AR Session component to the scene.
Also added an ARCore Extensions component to the AR Session Origin.



Unity Layout for GeoSpatial Creator



Player Settings for ARBuild

WEEK 5

SETTING UP UNITY FOR VR DEVELOPMENT

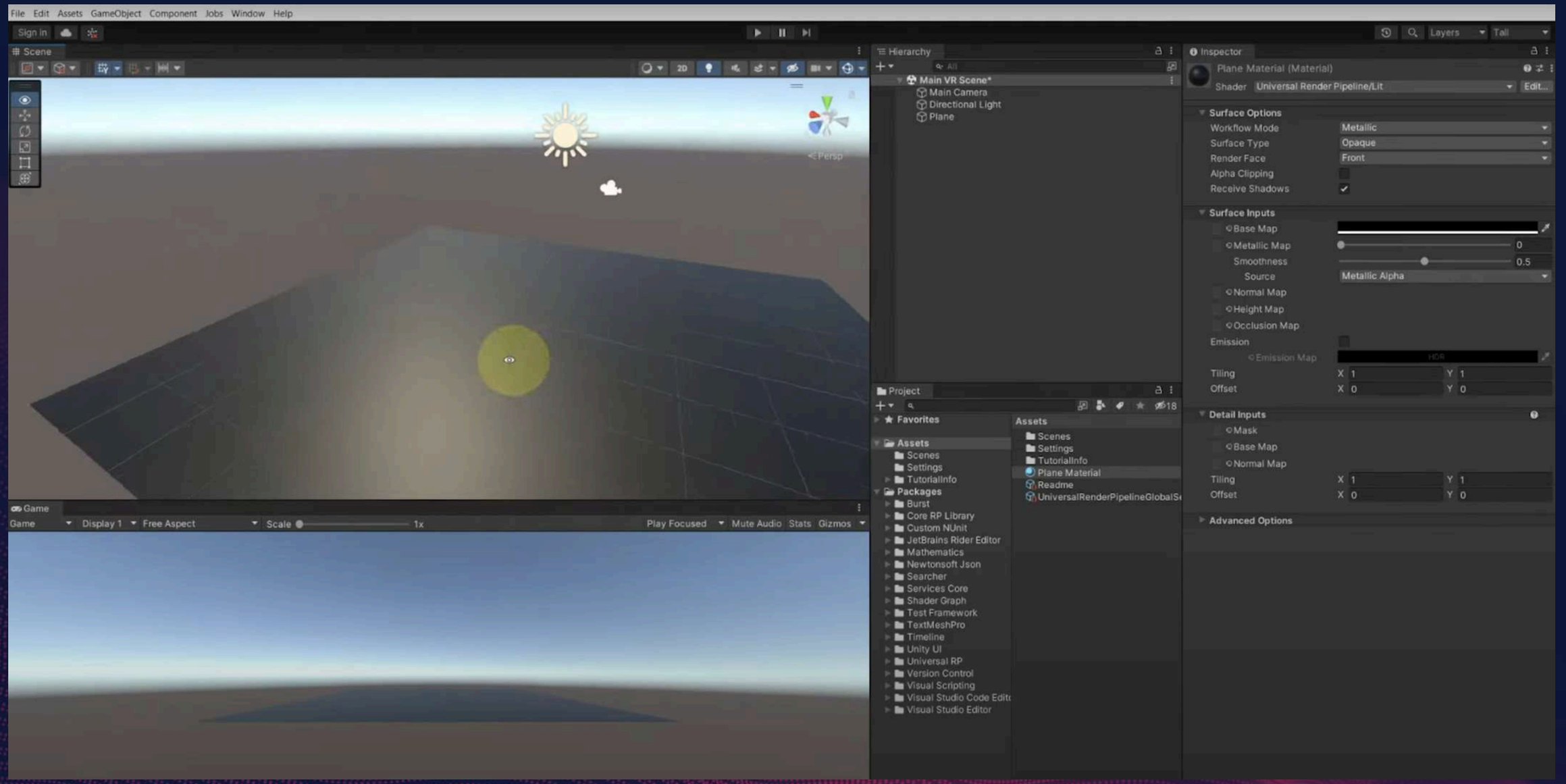
INTRODUCTION

- **Downloading and setting up Unity for VR development, including installing the latest version, adding Android support, and creating a new project with optimized graphics for VR.**Install XR plugin management, select Open XR for desktop and Android, and set interaction profiles for devices to support.

- Testing VR games in Unity by connecting a VR headset to the computer, adjusting camera movements based on headset rotation, and using tools like Unity XR Interaction Toolkit for faster game development.

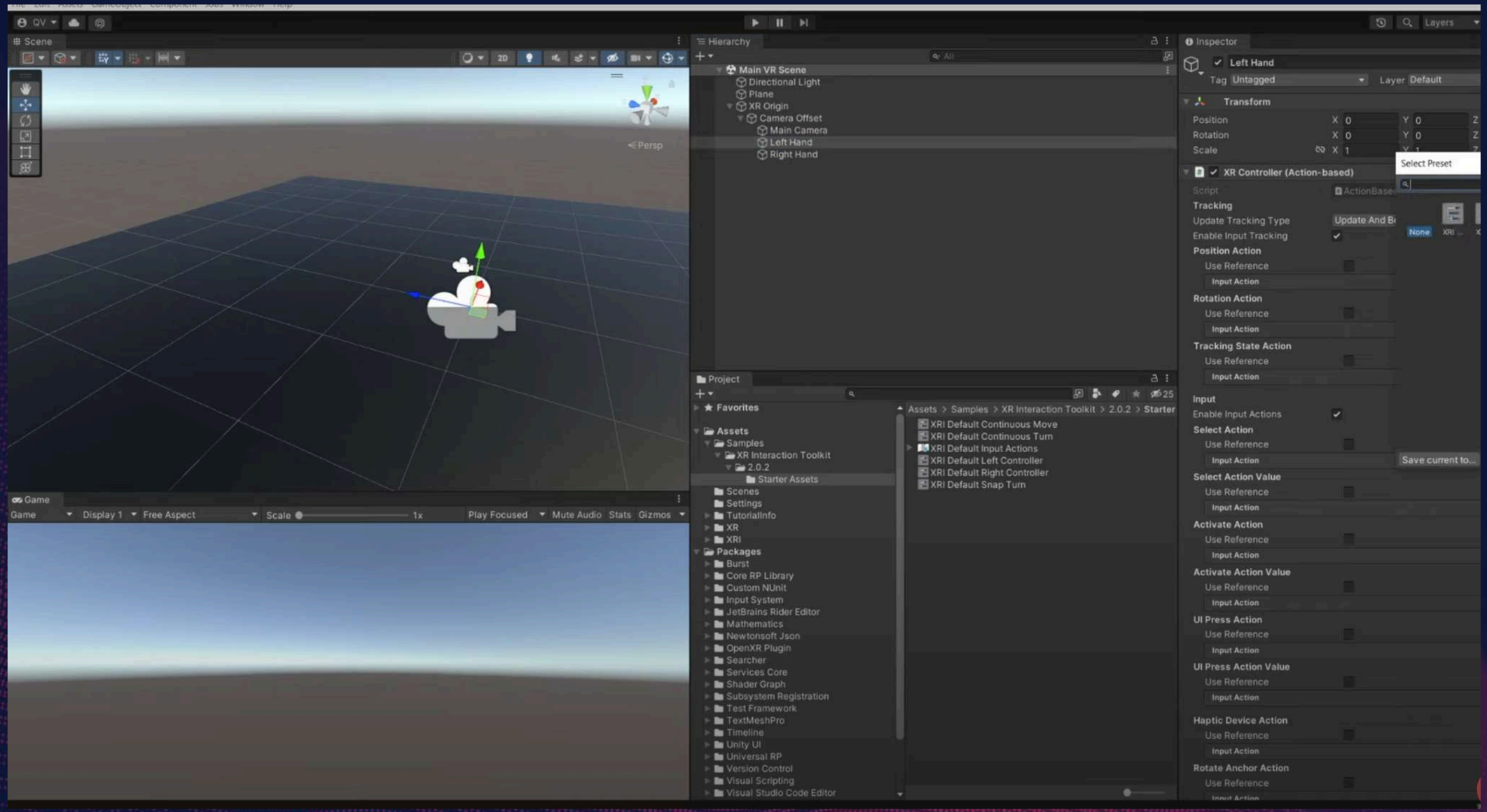
- Installing the Unity. XR Interaction Toolkit and setting up XR origin and camera for the XR rig setup in Unity. Adding controllers, creating empty game objects, and setting up XR controller action-based components for the XR rig in Unity.

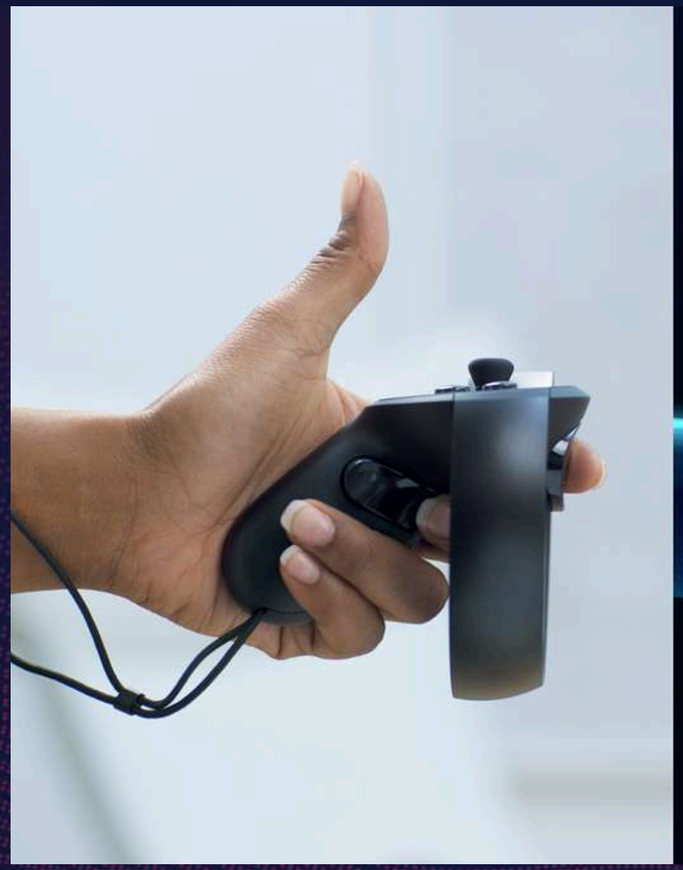
- Now we add a cube to the game window, adjust its properties, and preview the changes. Adding and customizing a cube in a game window for testing purposes, including adjusting its collider and scale, then previewing the changes by playing the game.

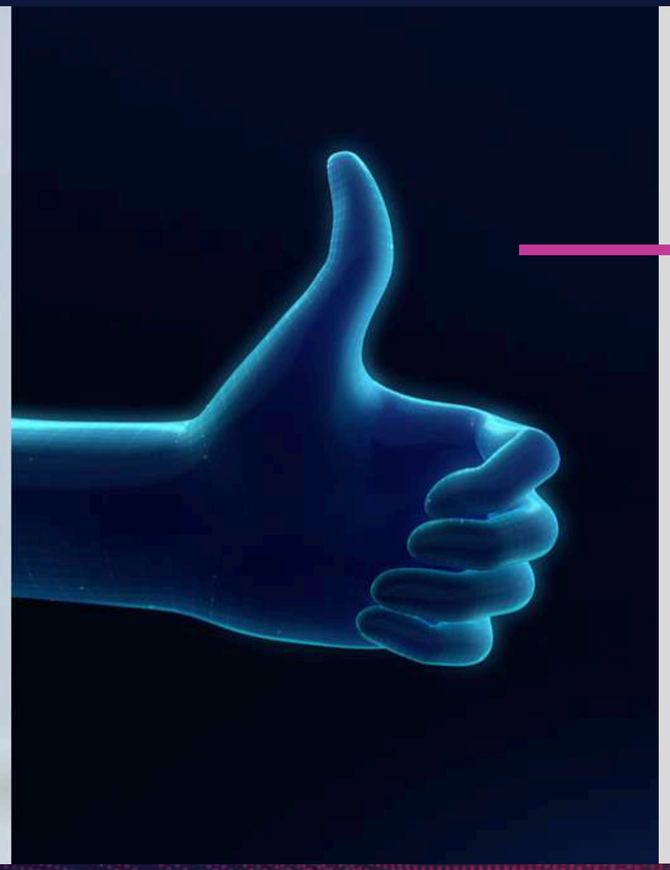


Creating a platform with the cube added

After that we efficiently set up input actions for controllers using preset components from the Unity XR toolkit to streamline the XR rig setup process.





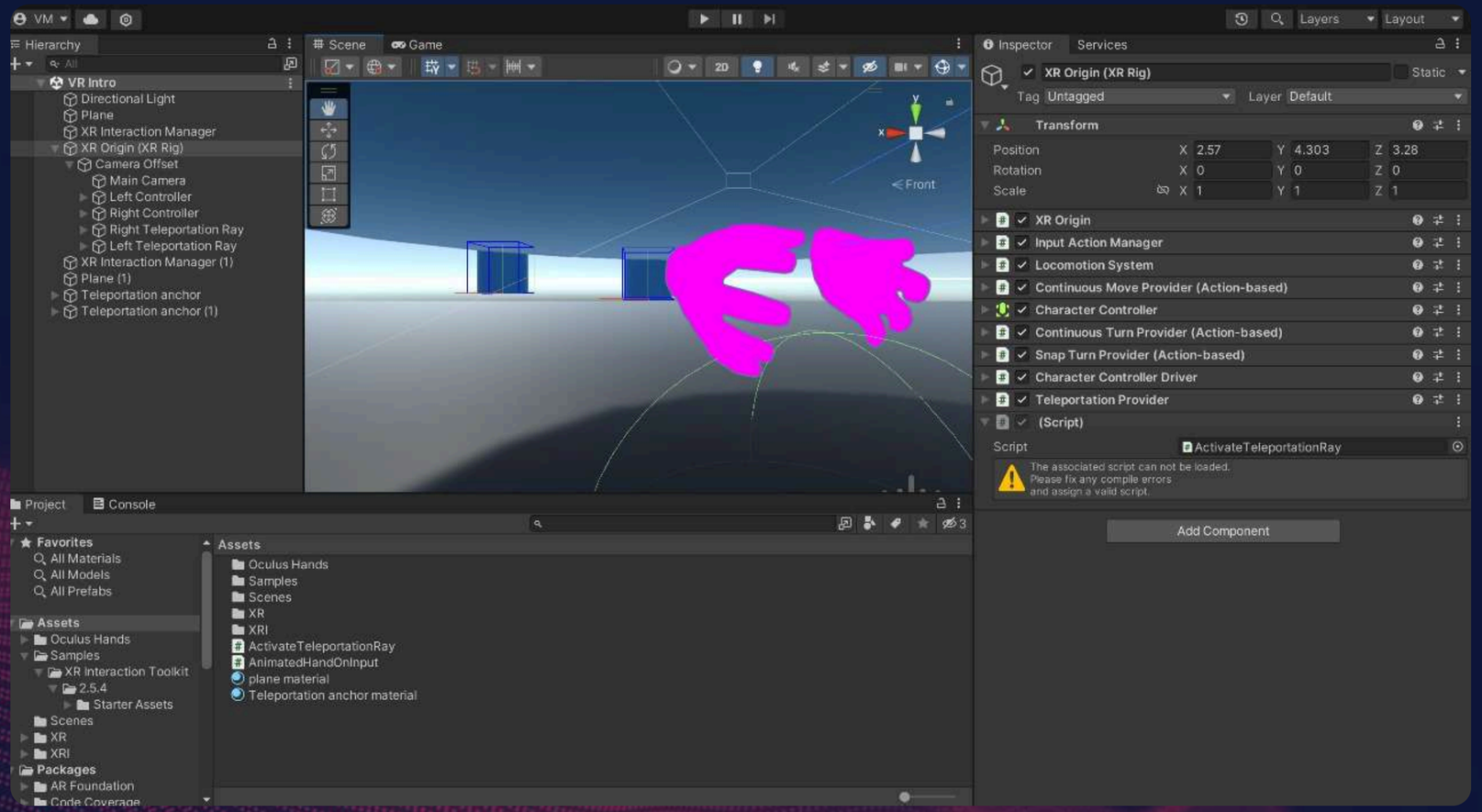


WEEK 6

INPUT AND HAND PRESENCE WITH CONTINUOUS MOVEMENT

INTRODUCTION

- We then proceed to input hand animations into the Unity by Importing and replacing cube with animated hand models in Unity for VR development.
- Exploring the animator panel to control hand animations based on controller input in VR development.
- Creating a custom component to handle input controls for animating the hands in Unity for VR development.



- Using Unity Engine input system, we can create and set input actions in the Unity editor, either by creating actions from scratch or using pre-made actions for better organization and functionality.

- Testing and displaying input values in Unity can be done efficiently using console logs. Utilizing input values to animate objects in Unity involves setting parameters in the animator component.

- Animating objects in Unity with input values. Setting parameters in the animator component. Utilizing input actions to trigger animations.

Continuous movement in the VR game:

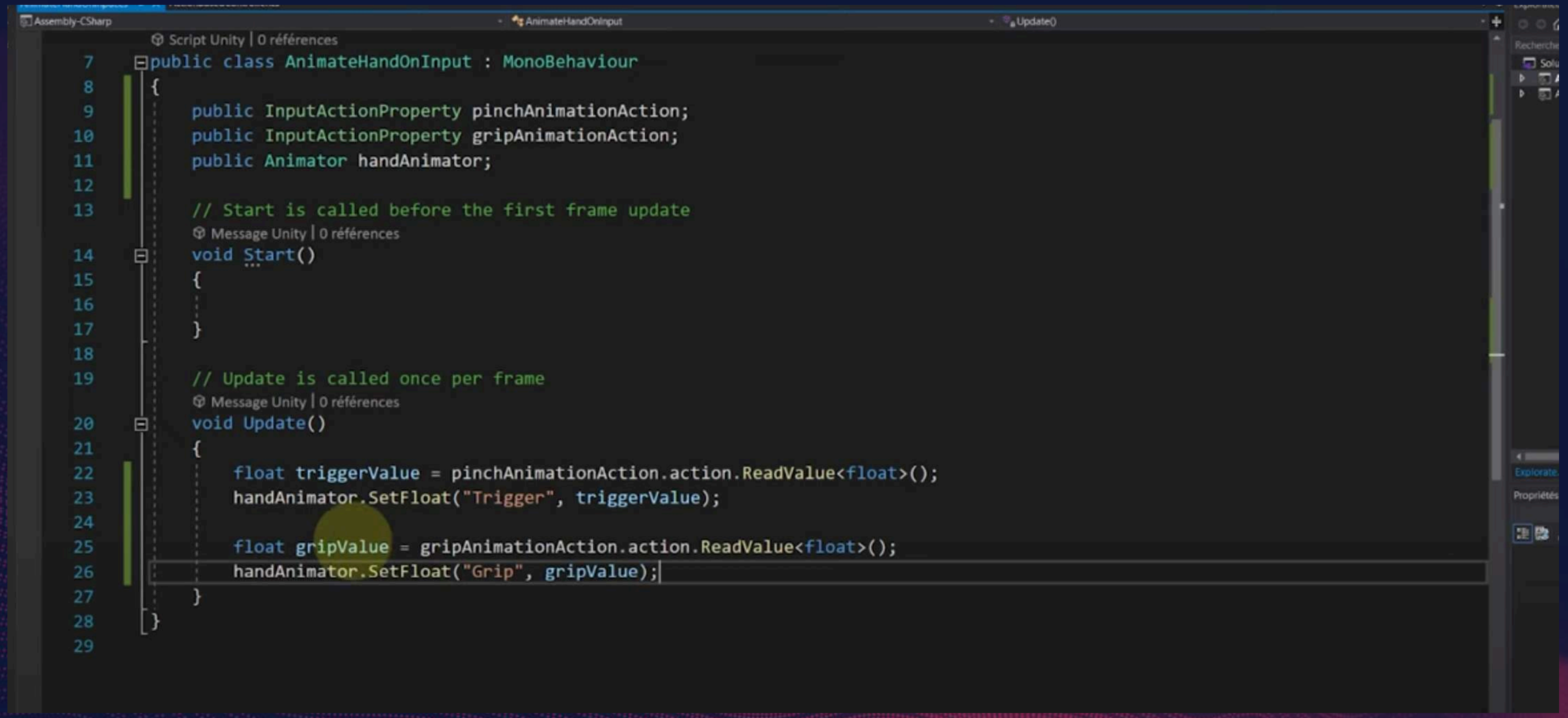
- Now we implement continuous movement in a VR game using Unity's XR Toolkit, including setting up character controllers, gravity, and rotation options.Now we begin setting up continuous movement in VR games using Unity XR toolkit, explaining components like locomotion system, continuous move provider, and character controller for precise movement and obstacle interaction.

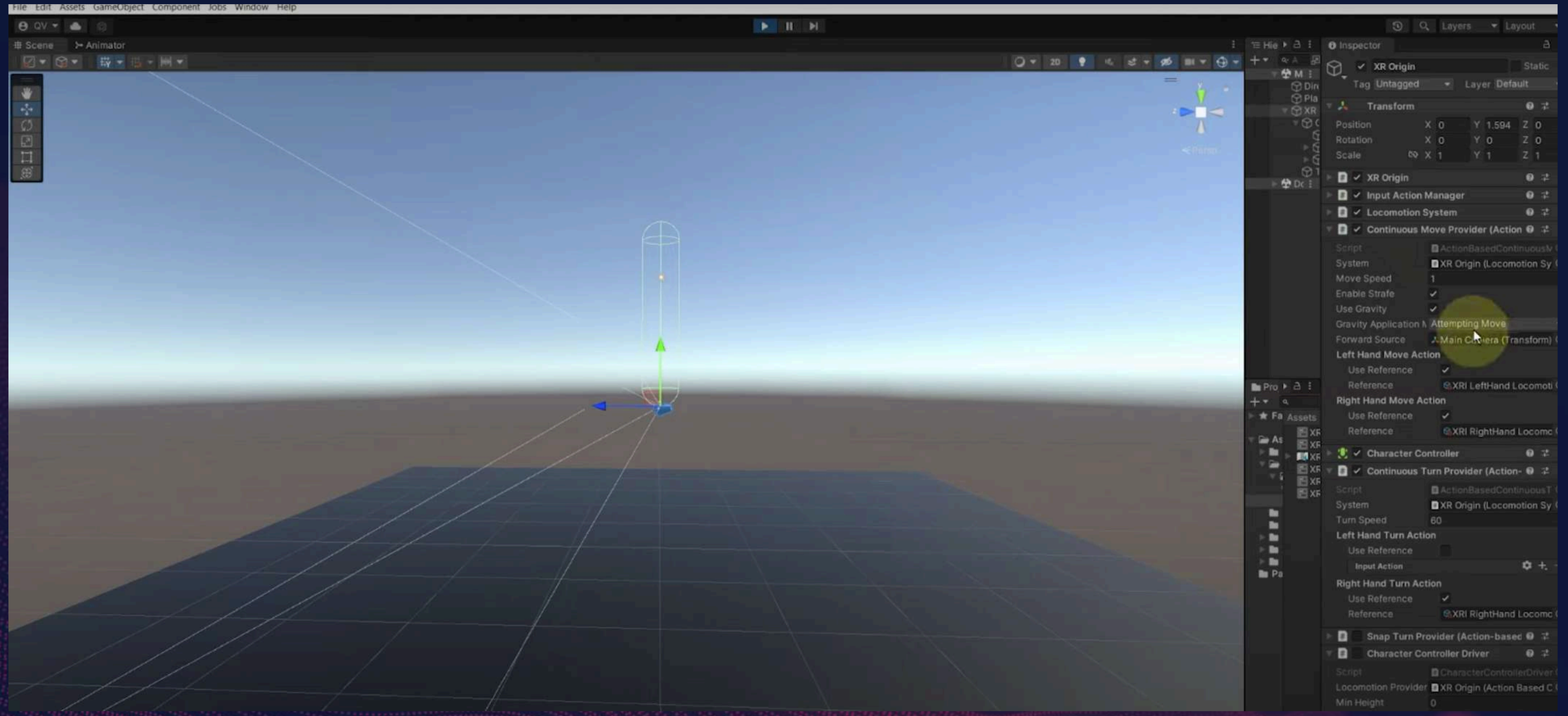
- Configuring continuous move provider and character controller components for precise movement and obstacle interaction. Then we were explained how to set up locomotion and rotation controls in VR, demonstrating how to enable continuous and snap turns for player movement.

- Configuring locomotion controls for left and right hands using thumbsticks and bindings for movement in VR. Demonstrating the setup of continuous and snap turn options for rotating the player in VR, with a focus on customization and testing preferences.

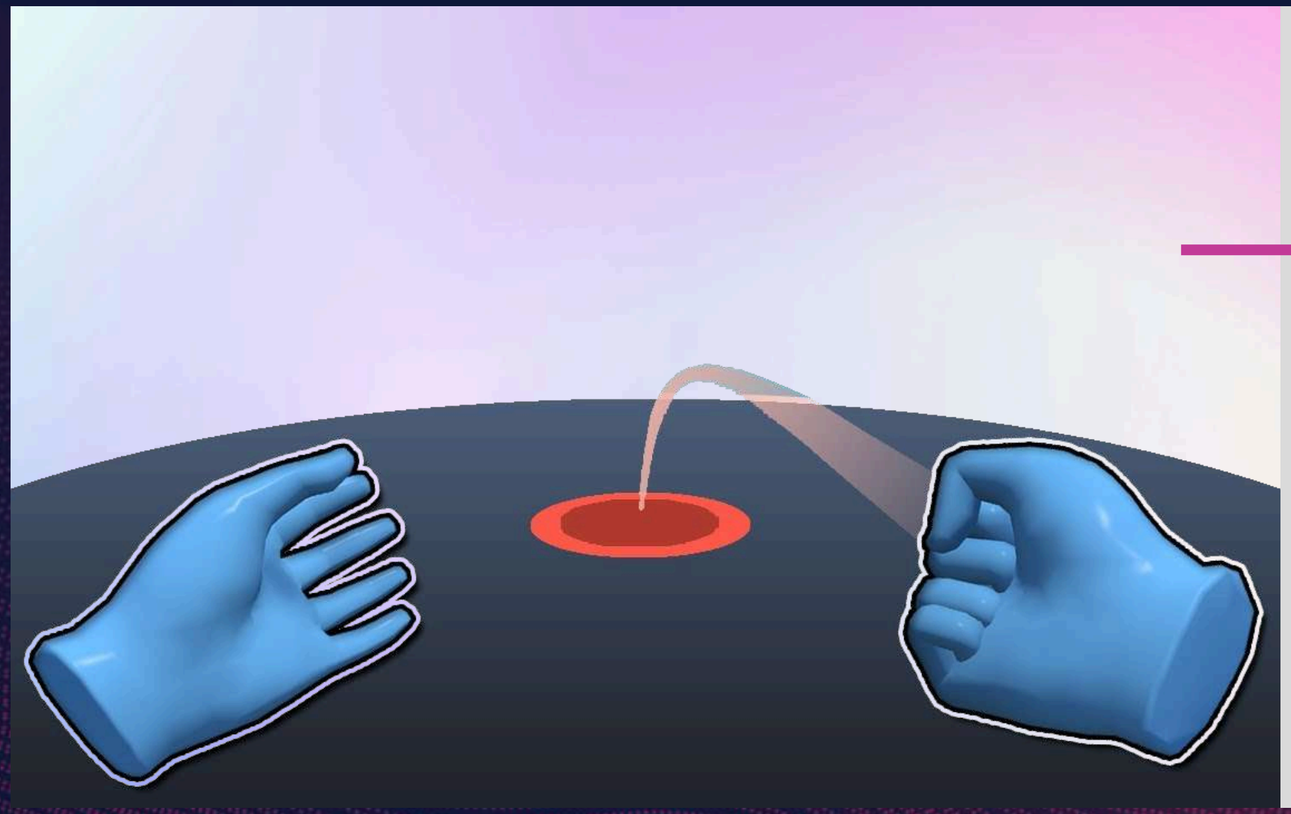
- Continuous movement in a game can be achieved by updating the character controller with the Unity XR toolkits, ensuring the capsule follows the player's height and position during movement. Adjusting the min and max values in the character controller driver component allows for height clamping customization, optimizing player experience in the game.

Writing the script for the hand movements of gripping and pinching to make it move.





Setting up the continuous movement and turning for the movement of the player using the thumbsticks.



WEEK 7

TELEPORTATION IN VR

INTRODUCTION

- Now to add a thrill and uniqueness in our game, we add the ability to teleport to our character.To set up teleportation for VR in Unity by creating a teleportation system using ray interaction and teleportation zones.

- Creating a teleportation system in Unity involves setting up a teleportation anchor, adjusting materials, and configuring teleportation behavior for precise player movement. Customizing materials and shaders for the teleportation anchor to enhance visual appeal and functionality.

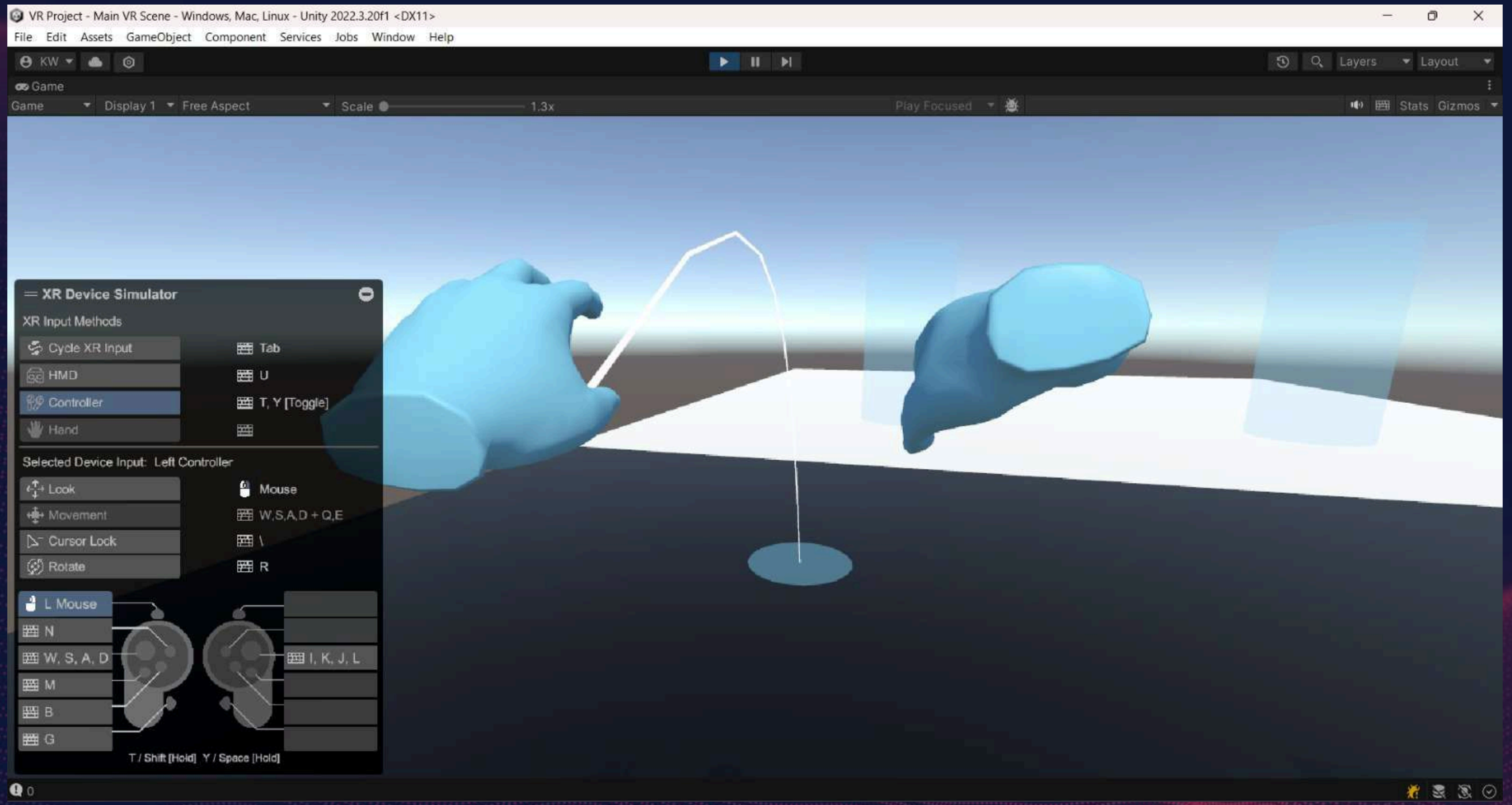
- Enhancing the teleportation system by curving the teleportation ray and adding a reticle for improved visual feedback in Unity.Experimenting with visual settings can enhance the look of the teleportation ray.

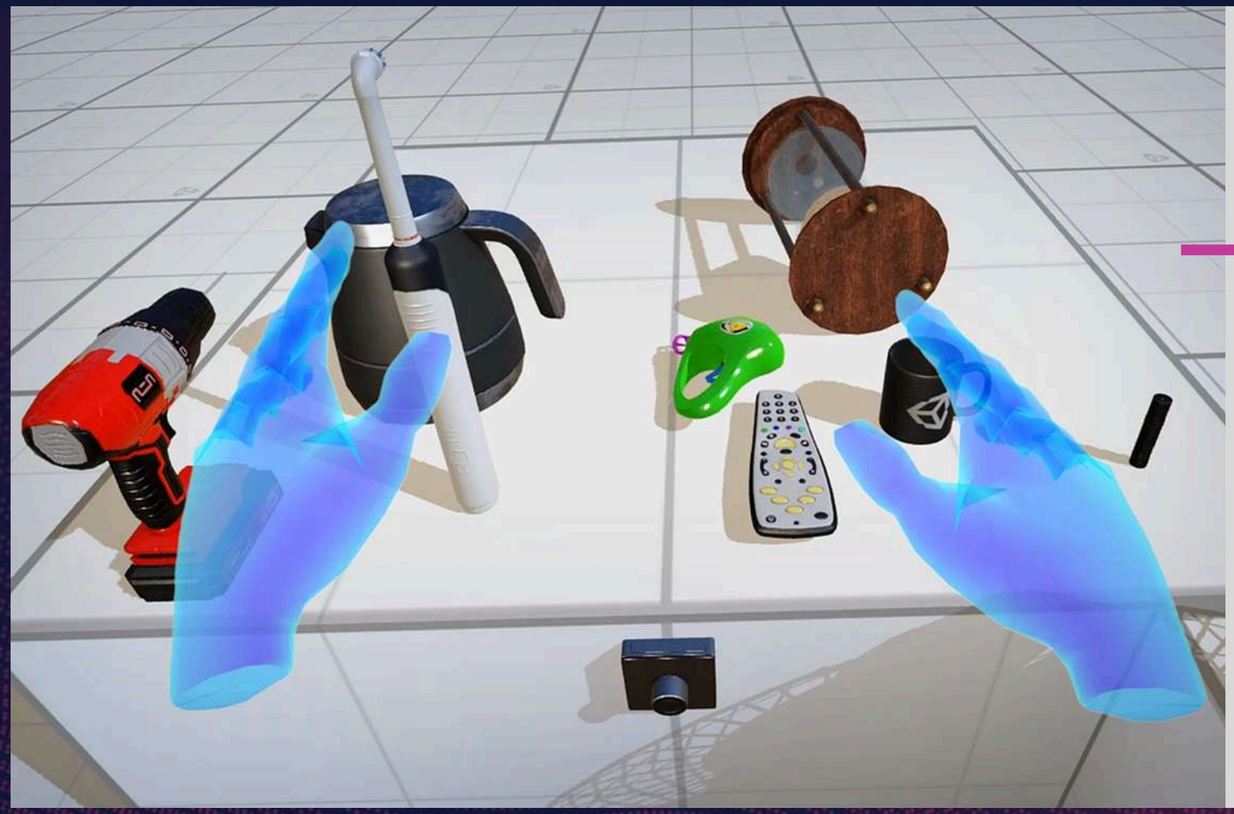
- Customizing the appearance of the teleportation ray by adjusting settings like width and color can improve the overall visual experience for users.Utilizing custom shaders and exploring different visual effects can further enhance the teleportation system's aesthetics and functionality.

#CODE

- In the Hierarchy, right-click and create an empty GameObject called `TeleportationArea`. Add a `Box Collider` component to it and set it as a trigger. This area can be used to define where the player can teleport
- Select the `XR Rig` in Hierarchy. Add the `Teleportation Provider` component (found under XR > Teleportation). Select the `Teleportation Area` GameObject which is created. Add the `Teleportation Area` component to it.
- Create an empty GameObject called TeleportationAnchor. This will serve as the destination for the teleportation. Select the Teleportation Anchor GameObject and add the Teleportation Anchor component to it.Select the XR Rig again, and add the Teleportation Input Handler component. Link input action asset (e.g., XRControls) to the Teleportation Input Handler.
- n the XR Toolkit, create a new Input Action Asset (e.g., XRControls) for teleportation inputs. Create an action for "Teleport" and map it to the desired button on the controller (e.g., pressing the grip button). Add Teleportation Input Handler. Select the XR Rig again, and add the Teleportation Input Handler component. Link input action asset (e.g., XRControls) to the Teleportation Input Handler.
- Select left or right controller GameObject (the ones used for hand tracking). Add the Teleportation Interactor component to each controller. In the Teleportation Interactor component, set the action to use the previously defined teleportation input action.

Using this teleport feature we can teleport ot the marked location with a single click which gives us an amazing feature





WEEK 8

HOVER, GRAB, USE INTERACTABLES

HOVER, GRAB AND USE INTERACTABLE

- We focus on VR interaction now, on how to interact, grab, and use objects in Unity XR Toolkit. Setting up an interactor component to define the type of interaction near hands using a sphere collider for zone detection. Creating an interactable object by adding components like rigid body and XR simple interactable to enable responses to interactions.
- Customizing interactions by changing object materials based on events like over-enter, over-exit, and select, showcasing the versatility of interactable responses.
- Creating a super simple pistol mechanism in VR by adding colliders, rigid body, and XR grab interactable components to the pistol model.Demonstrating how to set up the attach transform for the pistol to ensure it snaps to the correct position when grabbed, and how to link firing a bullet using a custom script.Setting up the script to spawn and fire bullets in Unity using XR interaction with proper parameters and velocity calculations.
- Resolving bugs related to teleportation timing, collision with character controller, and setting up layers and physics interactions for a seamless gameplay experience.

OFFSET AND DISTANCE GRAB:

Now we proceed towards offset grabbing and array interaction.
Attach Transform for Left Hand

- Created a new script that extends the XR Grab Interactable component.
- Overrode the OnSelectEntered function to set the attachment transform based on the tag of the interactor object. Set up the left and right attach transforms in Unity. Fixed the issue with the left hand attachment transform. Introduced the concept of offset grabbing, where the object keeps its position when grabbed.

Offset Grab Interactable

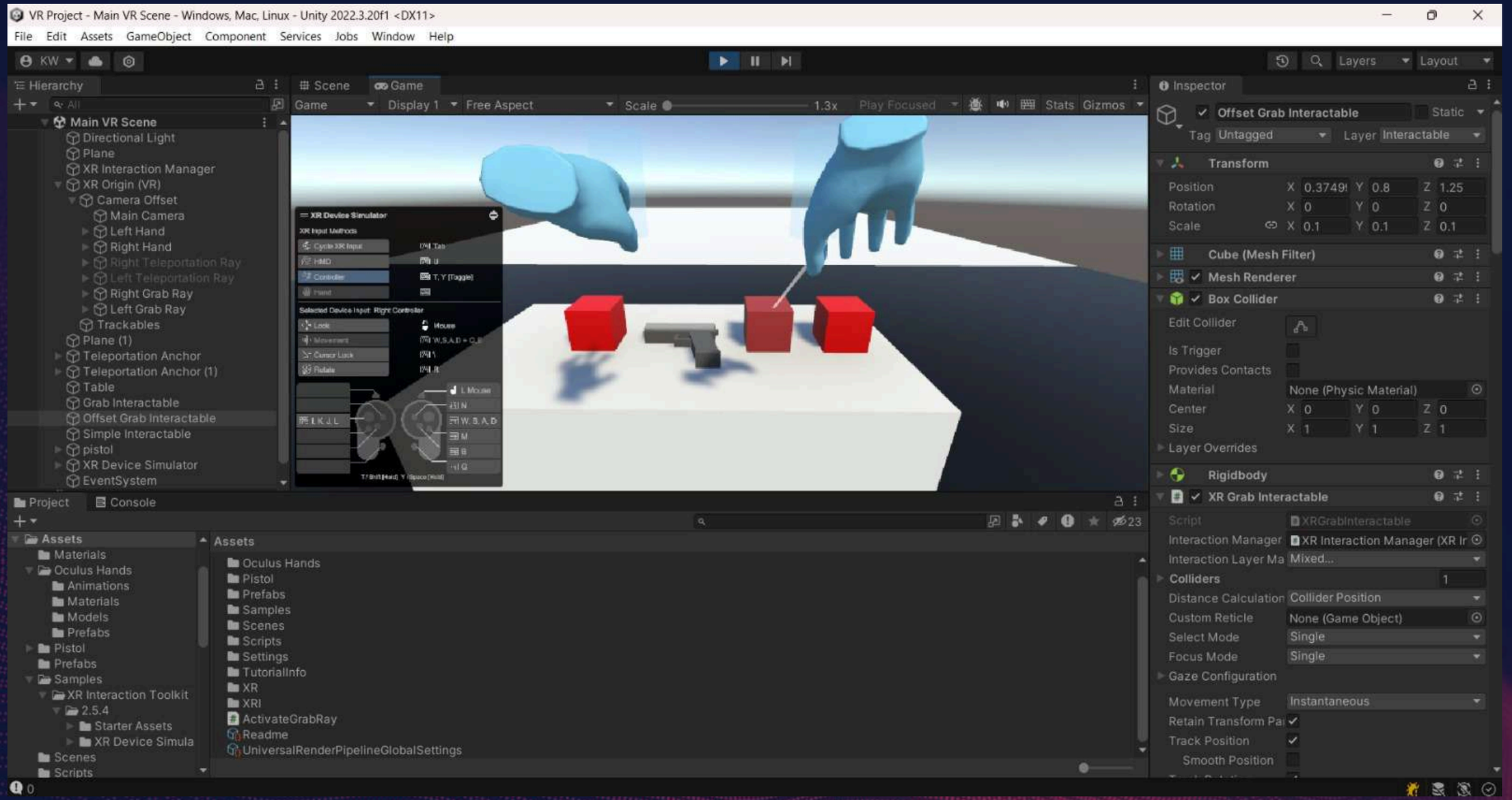
- The Unity XR Toolkit 2.1 update has added the offset grab interactable feature by default.
- Duplicate a cube, move it to the side, and rename it "Grab Interactable Offset". Create a new component called "XR Offset Grab Interactable" that extends the XR Grab Interactable component. In the XR Offset Grab Interactable script, create an attach transform if it doesn't already exist.
- Override the OnSelectEnter and OnSelectExit methods to set the position and rotation of the attach transform to the position and rotation of the interactor object. Save the script and go back to Unity. The object will keep its initial position and be grabbable with both hands.
- The distance grab interactable feature allows objects to be grabbed from a distance. To implement this, create a new component called "XR Distance Grab Interactable" that extends the XR Grab Interactable component.
- In the XR Distance Grab Interactable script, add a public float variable called "maxDistance" to specify the maximum distance from which the object can be grabbed. Override the CanStartSelect method to check if the distance between the interactor object and the attach transform is less than or equal to the maxDistance.
- If the distance is less than or equal to the maxDistance, return true to allow the object to be grabbed. Otherwise, return false. Save the script and go back to Unity. The object can now be grabbed from a distance.

OFFSET AND DISTANCE GRAB:

- Distance Grab
  Distance grabbing is a technique used to make it easier to grab objects that are far away in VR.
  To set up distance grabbing in Unity using the XR Toolkit:
- Create two new XR Ray Interactors, one for each hand.
- Set the tag of each ray to "Right End" and "Left End" respectively.
- Adjust the appearance of the rays by changing the colour gradient and line width.
- Set the max raycast distance to 4 in the XR Ray Interactor component.
- When using distance grabbing with the XR Offset Grab script, the object would grab and keep the same position. To fix this, add a check in the XR Offset script to see if the interactor is an XR Direct Interactor.
- If it's not, set the position of the attached transform back to its default position.
- This ensures that distance grabbing doesn't affect the XR Offset Grab behaviour.
- To keep the distance between the player and the object when grabbing from a distance, uncheck the "Force Grab" option in the XR Ray Interactor component.
- This allows the player to use the thumbstick to change the position and rotation of the object while maintaining the distance.

- Interaction Layer Mask
- To separate different types of interactions, create a layer for each type in the Interaction Layer Mask. Assign the appropriate layer mask to each interactor and interactable object.
- This ensures that interactors can only interact with objects that have the corresponding layer mask.
- Created separate interaction layers for teleportation, direct interaction, and array interaction.
- Set the interaction layer mask for the teleportation plane, anchors, and grab interactable objects accordingly.
- Verified that the interactable objects respond correctly to the different interaction types.
- Created a script called "Activate Grab Ray" to deactivate the grab ray when holding an object.
- Added references to the grab ray game objects and XR Direct Interactors in the script.
- Deactivated the grab ray when the number of objects held by the interactor is zero.
- Assigned the script to the left and right grab rays and XR Direct Interactors. Verified that the grab ray is now deactivated when holding an object.

Using of the grab function to grab the object and shoot targets



THANK YOU

PROJECT MEMBERS:

- **KHUSHAL WADHWA**
- **VIKAS MEENA**
- **BASUDEV MOHAPATRA**
- **DAKSH PRATAP SINGH**
- RISHI
- VICKY MEROTHA