Project Report

Subject : Augmented Reality

Subject Code : COCSE27

Submitted By :
Sarthak Shukla [2019UCO1732]

Experiment -2

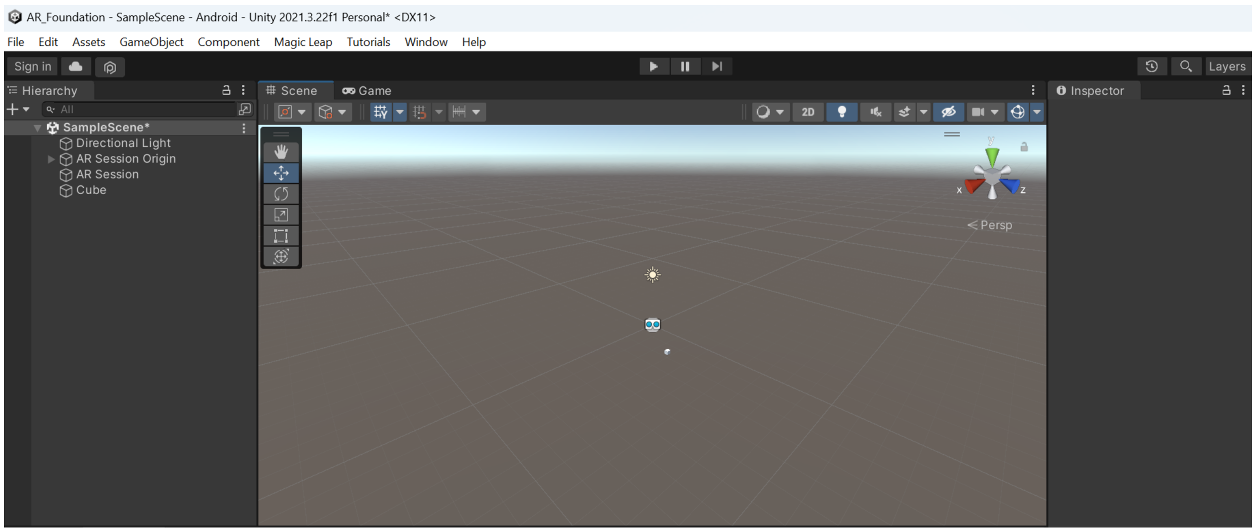
Aim: Build an Augmented Reality application having 3D object in it using Unity

Methodology:

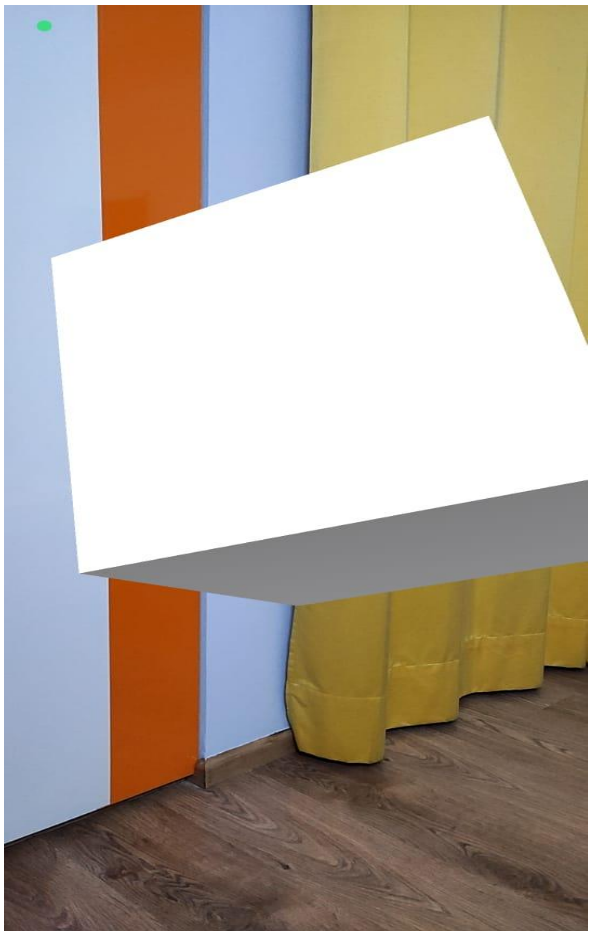
1. Launch Unity and create a new 3D project.

2. In the Hierarchy window, right-click and select "3D Object" -> "Sphere" (or any other 3D object you want to display).
3. In the Inspector window, select the Sphere object and set its position and rotation according to your preferences.
4. Add a camera to your scene by right-clicking in the Hierarchy window and selecting "Camera".
5. Adjust the camera's position and rotation to properly view the 3D object.
6. Add a directional light to your scene by right-clicking in the Hierarchy window and selecting "Light" -> "Directional Light".
7. Adjust the light's position and rotation to properly illuminate the 3D object.
8. Press the Play button to test your scene and make sure the 3D object is properly displayed.

Unity Console output:


AR_Foundation - SampleScene - Android - Unity 2021.3.22f1 Personal* <DX11>
File Edit Assets GameObject Component Magic Leap Tutorials Window Help
Sign in
Layers
Hierarchy
Scene
Game
Inspector
SampleScene*
Directional Light
AR Session Origin
AR Session
Cube
2D
Persp

Application output:



Experiment - 3

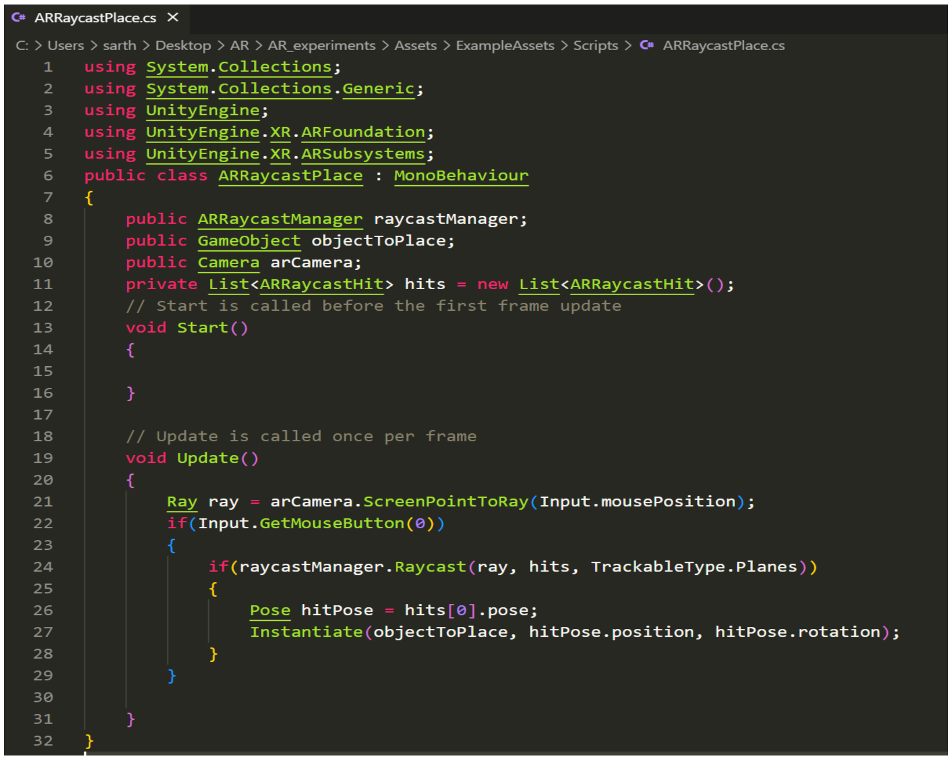
Aim: Build an Augmented Reality application having Placement Indicator in it to summon 3D objects in it using Unity.

Methodology:

1. Create a new Unity project and set up the necessary components for AR, such as AR Foundation and ARCore/ARKit depending on your target platform.
2. Import a 3D model or create a new one using Unity's built-in tools.
3. Create an indicator object that will be used to show the user where the 3D object will be placed in the real world. This could be a simple cube or sphere.
4. Add scripts to handle the placement of the 3D object using the indicator. One way to do this is to use the AR Raycast Manager to detect a surface where the indicator should be placed, and then instantiate the 3D object at that location.
5. Configure the user interface to allow the user to place the indicator and the 3D object. This could be done using touch gestures or a virtual button.
6. Test the application on a target device and make any necessary adjustments to improve the user experience and stability of the application.

7. Once the application is working correctly, build and deploy it to your target platform.

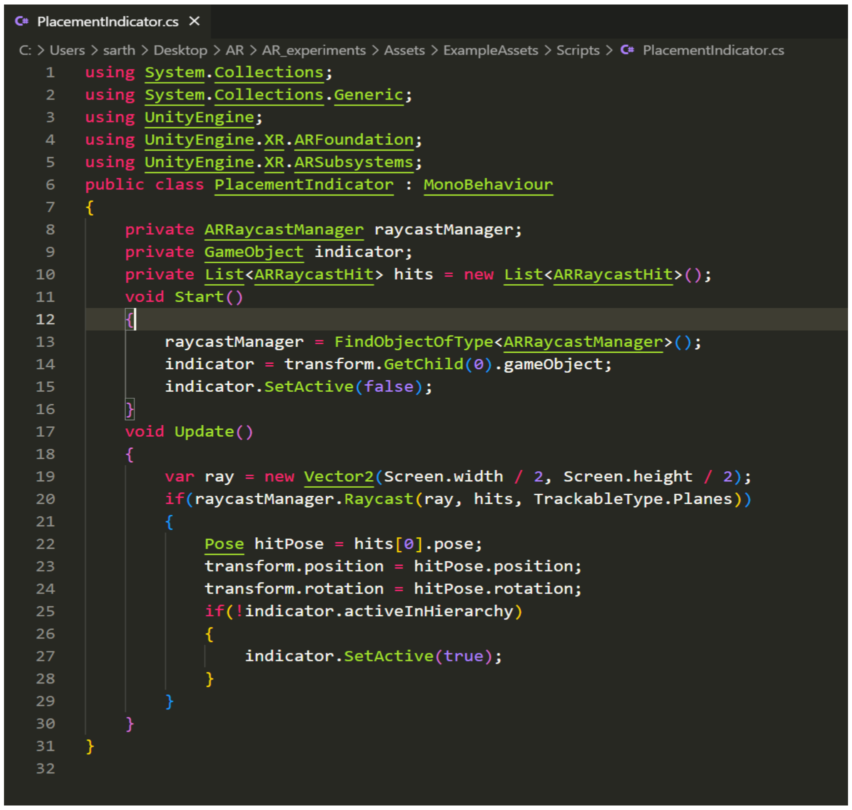
Unity Console output:



ARRaycastPlace.cs ✕

C: > Users > sarth > Desktop > AR > AR_experiments > Assets > ExampleAssets > Scripts > ARRaycastPlace.cs

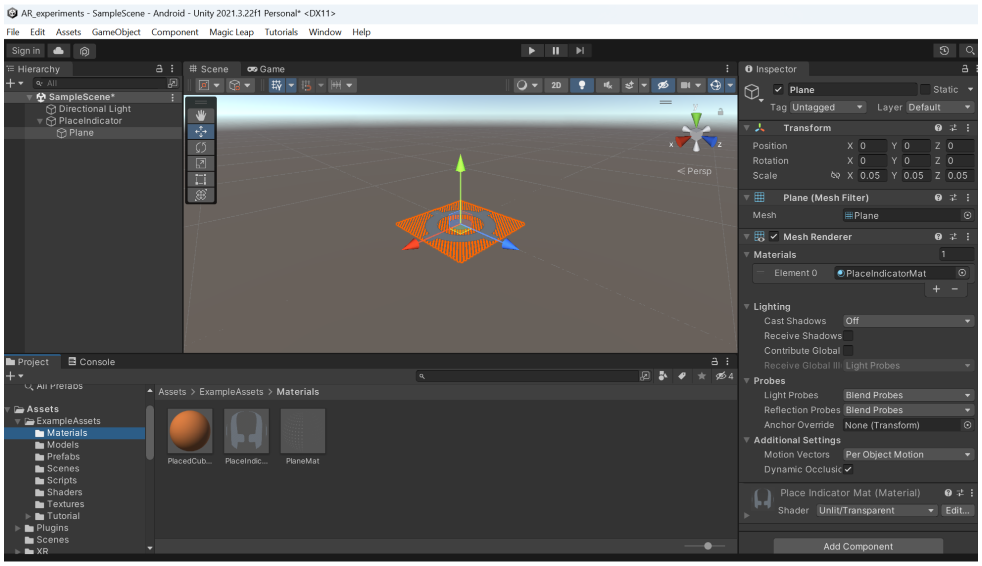
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.XR.ARFoundation;
using UnityEngine.XR.ARSubsystems;
public class ARRaycastPlace : MonoBehaviour
{
    public ARRaycastManager raycastManager;
    public GameObject objectToPlace;
    public Camera arCamera;
    private List<ARRaycastHit> hits = new List<ARRaycastHit>();
    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {
        Ray ray = arCamera.ScreenPointToRay(Input.mousePosition);
        if(Input.GetMouseButton(0))
        {
            if(raycastManager.Raycast(ray, hits, TrackableType.Planes))
            {
                Pose hitPose = hits[0].pose;
                Instantiate(objectToPlace, hitPose.position, hitPose.rotation);
            }
        }

    }
}
```



PlacementIndicator.cs

C: > Users > sarth > Desktop > AR > AR_experiments > Assets > ExampleAssets > Scripts > PlacementIndicator.cs

```csharp
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.XR.ARFoundation;
using UnityEngine.XR.ARSubsystems;
public class PlacementIndicator : MonoBehaviour
{
    private ARRaycastManager raycastManager;
    private GameObject indicator;
    private List<ARRaycastHit> hits = new List<ARRaycastHit>();
    void Start()
    {
        raycastManager = FindObjectOfType<ARRaycastManager>();
        indicator = transform.GetChild(0).gameObject;
        indicator.SetActive(false);
    }
    void Update()
    {
        var ray = new Vector2(Screen.width / 2, Screen.height / 2);
        if(raycastManager.Raycast(ray, hits, TrackableType.Planes))
        {
            Pose hitPose = hits[0].pose;
            transform.position = hitPose.position;
            transform.rotation = hitPose.rotation;
            if(!indicator.activeInHierarchy)
            {
                indicator.SetActive(true);
            }
        }
    }
}
```



Hierarchy:
- SampleScene*
  - Directional Light
  - PlaceIndicator
    - Plane

Project: Assets > ExampleAssets > Materials
- Assets
  - ExampleAssets
    - Materials
    - Models
    - Prefabs
    - Scenes
    - Scripts
    - Shaders
    - Textures
    - Tutorial
  - Plugins
  - Scenes
  - XR

PlacedCub... PlaceIndic... PlaneMat

Inspector

Plane — Static
Tag Untagged — Layer Default

Transform

	X	Y	Z
Position	0	0	0
Rotation	0	0	0
Scale	0.05	0.05	0.05

Plane (Mesh Filter)
Mesh: Plane

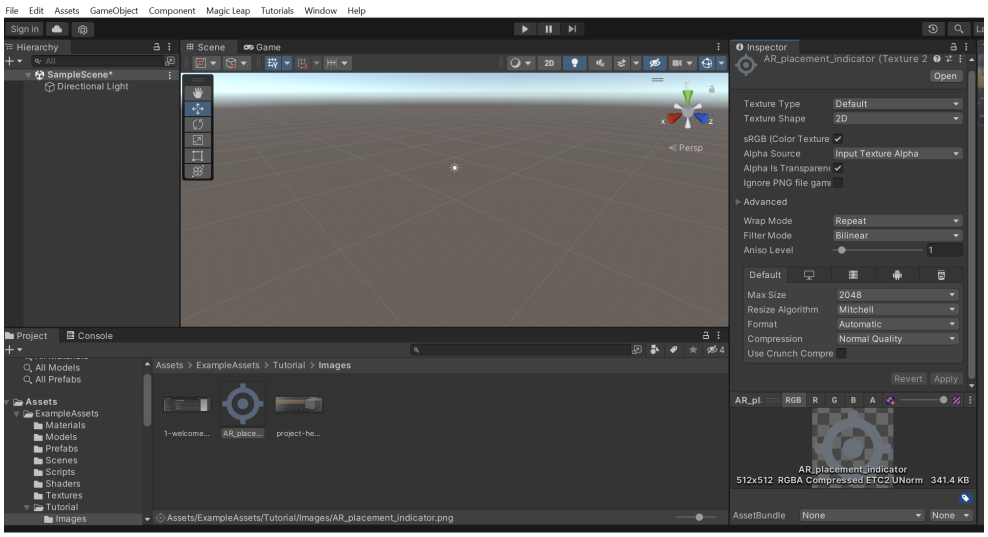
Mesh Renderer
Materials: 1
Element 0: PlaceIndicatorMat

Lighting
Cast Shadows: Off
Receive Shadows
Contribute Global
Receive Global Ill: Light Probes

Probes
Light Probes: Blend Probes
Reflection Probes: Blend Probes
Anchor Override: None (Transform)

Additional Settings
Motion Vectors: Per Object Motion
Dynamic Occlusic ✓

Place Indicator Mat (Material)
Shader: Unlit/Transparent — Edit...



SampleScene*
Directional Light

AR_placement_indicator (Texture 2

Texture Type	Default
Texture Shape	2D
sRGB (Color Texture	✓
Alpha Source	Input Texture Alpha
Alpha Is Transparenc	✓
Ignore PNG file gami	
Advanced	
Wrap Mode	Repeat
Filter Mode	Bilinear
Aniso Level	1
Default	
Max Size	2048
Resize Algorithm	Mitchell
Format	Automatic
Compression	Normal Quality
Use Crunch Compre	

AR_placement_indicator
512x512 RGBA Compressed ETC2 UNorm 341.4 KB

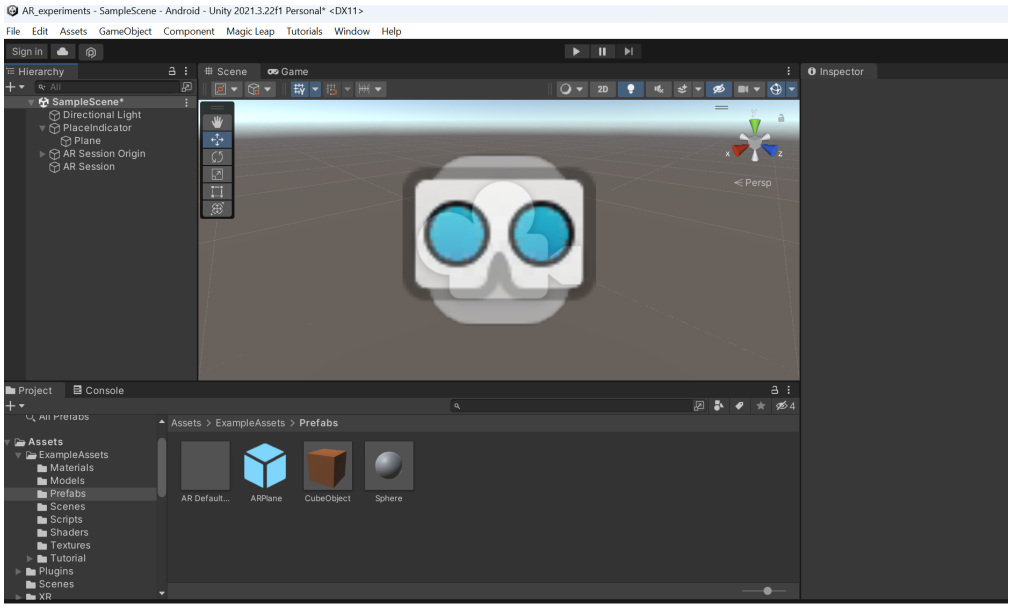
AssetBundle None None

Assets
ExampleAssets
Materials
Models
Prefabs
Scenes
Scripts
Shaders
Textures
Tutorial
Images

Assets > ExampleAssets > Tutorial > Images

1-welcome... AR_place... project-he...

Assets/ExampleAssets/Tutorial/Images/AR_placement_indicator.png

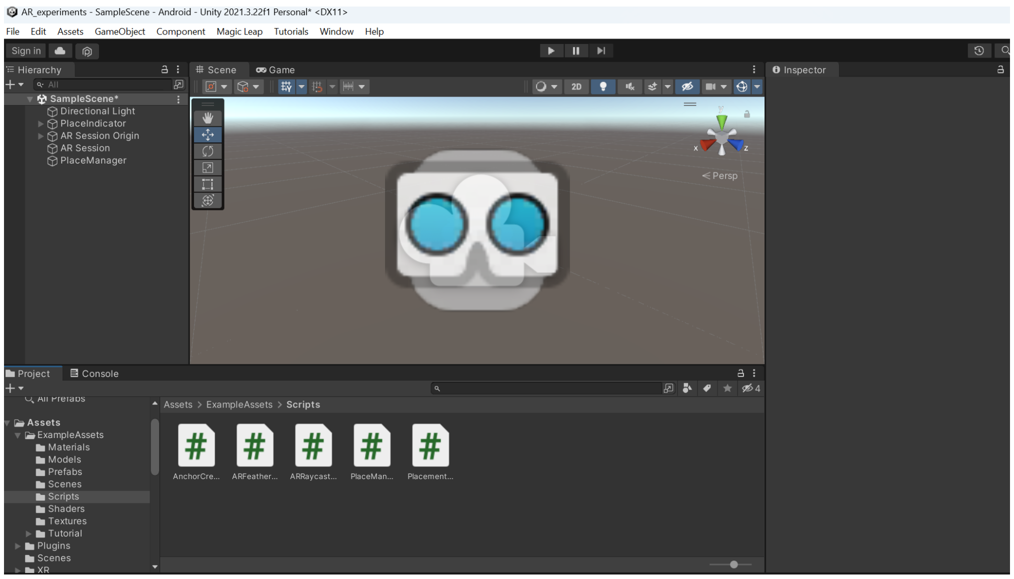

AR_experiments - SampleScene - Android - Unity 2021.3.22f1 Personal* <DX11>
File Edit Assets GameObject Component Magic Leap Tutorials Window Help
Sign in
Hierarchy
Scene
Game
Inspector
Persp
Project
Console
All Prefabs

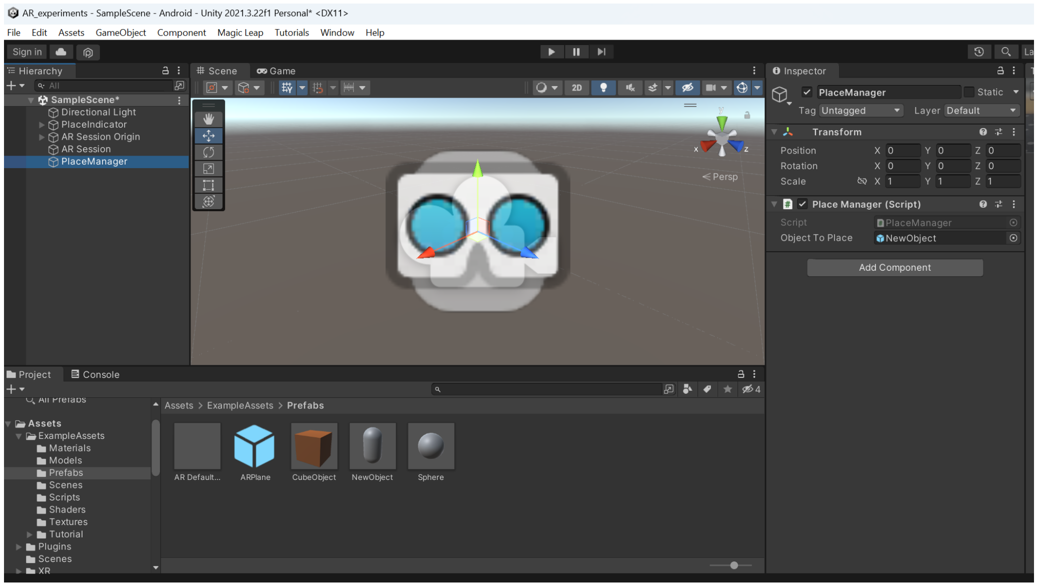
SampleScene*
Directional Light
PlaceIndicator
Plane
AR Session Origin
AR Session

Assets
ExampleAssets
Materials
Models
Prefabs
Scenes
Scripts
Shaders
Textures
Tutorial
Plugins
Scenes
XR

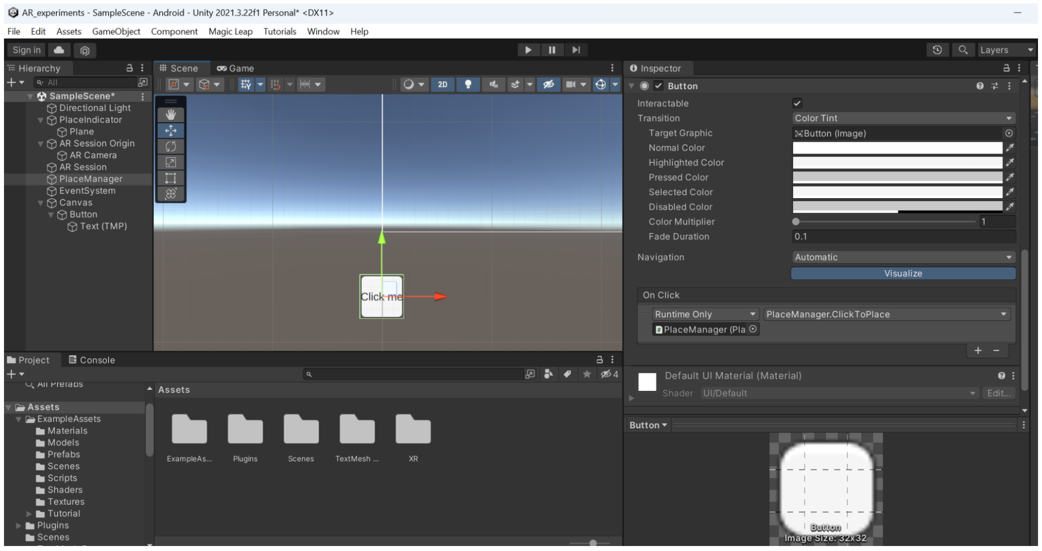
Assets > ExampleAssets > Prefabs

AR Default... ARPlane CubeObject Sphere





PlaceManager Static
Tag Untagged Layer Default
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Place Manager (Script)
Script PlaceManager
Object To Place NewObject



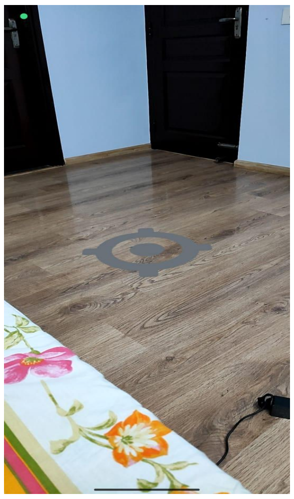
Button

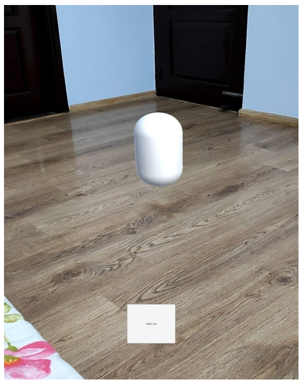
Interactable ✓
Transition Color Tint
Target Graphic Button (Image)
Normal Color
Highlighted Color
Pressed Color
Selected Color
Disabled Color
Color Multiplier 1
Fade Duration 0.1
Navigation Automatic

On Click
Runtime Only PlaceManager.ClickToPlace
PlaceManager (Pla

Default UI Material (Material)
Shader UI/Default

Application output:





Experiment - 4

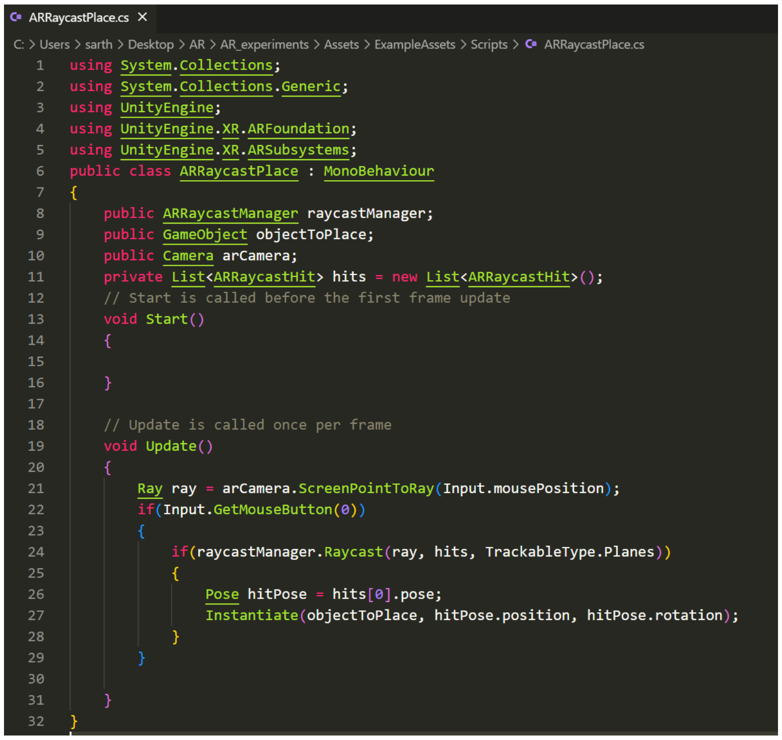
Aim: Build an Augmented Reality application using Unity for inserting multiple AR objects.

Methodology:

1. Create a new Unity project and select the platform for which you want to build the AR application.
2. Import the AR Foundation package and AR Kit/AR Core plugins to your project.
3. Add a camera to your scene and attach an AR Session component to it.
4. Create a prefab of the object you want to place in the AR scene.
5. Add an AR Raycast Manager component to your camera.
6. Create a script that will spawn the prefab when a user taps on the screen.
7. Add a plane detection component to the AR Session component.
8. In the script, create a method that detects the placement of the object on the detected plane.
9. Modify the script to enable the placement of multiple objects, by creating a list of objects and iterating through it to instantiate each object.
10. Add a UI element to your scene to allow users to switch between the objects they want to place.

11. Build and deploy your AR application on your desired platform.

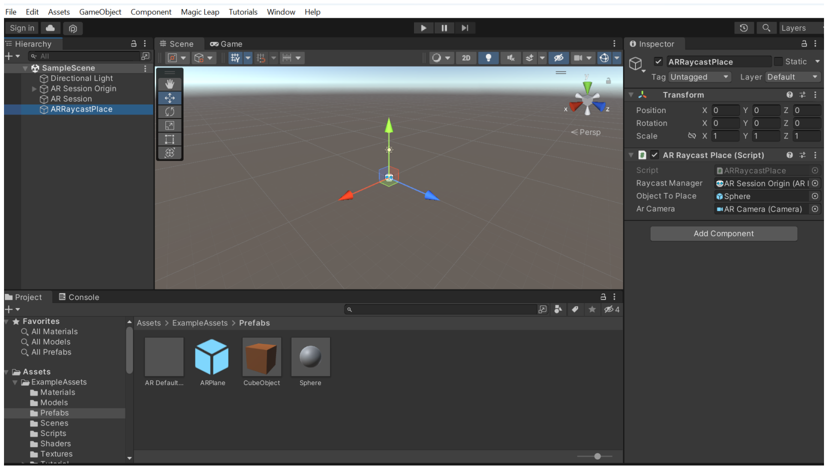
Unity console output:



ARRaycastPlace.cs

C: > Users > sarth > Desktop > AR > AR_experiments > Assets > ExampleAssets > Scripts > ARRaycastPlace.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.XR.ARFoundation;
using UnityEngine.XR.ARSubsystems;
public class ARRaycastPlace : MonoBehaviour
{
    public ARRaycastManager raycastManager;
    public GameObject objectToPlace;
    public Camera arCamera;
    private List<ARRaycastHit> hits = new List<ARRaycastHit>();
    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {
        Ray ray = arCamera.ScreenPointToRay(Input.mousePosition);
        if(Input.GetMouseButton(0))
        {
            if(raycastManager.Raycast(ray, hits, TrackableType.Planes))
            {
                Pose hitPose = hits[0].pose;
                Instantiate(objectToPlace, hitPose.position, hitPose.rotation);
            }
        }

    }
}
```

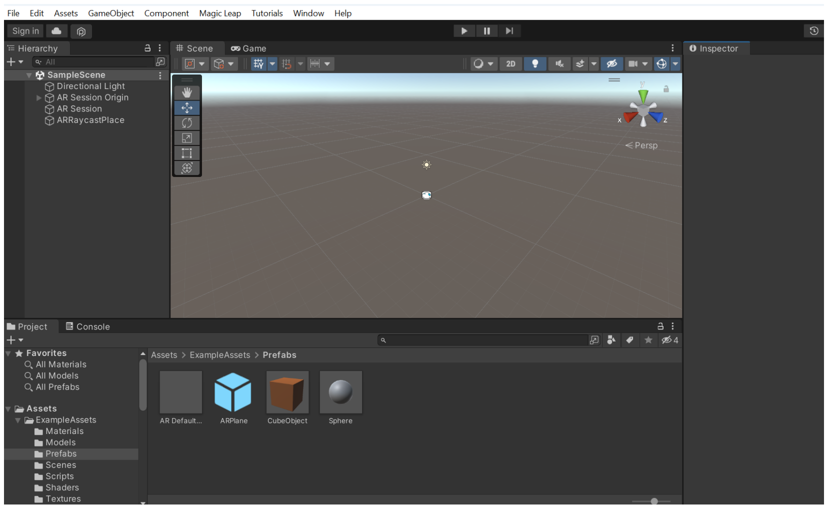


Inspector

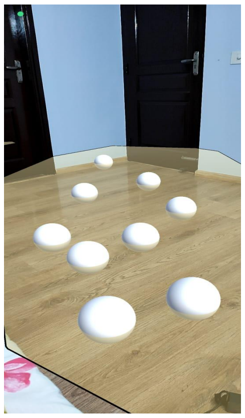
ARRaycastPlace Static
Tag Untagged Layer Default

Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

AR Raycast Place (Script)
Script ARRaycastPlace
Raycast Manager AR Session Origin (AR I
Object To Place Sphere
Ar Camera AR Camera (Camera)



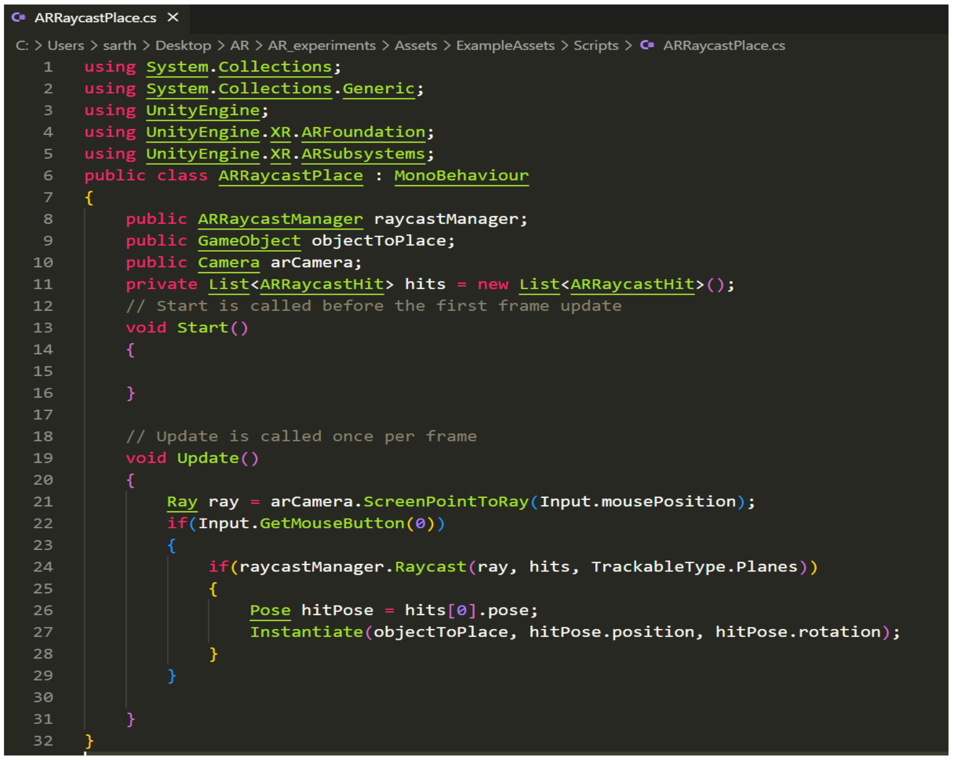
Application output:



Experiment - 5

Aim: Build an Augmented Reality application using Unity for summoning multiple AR objects.

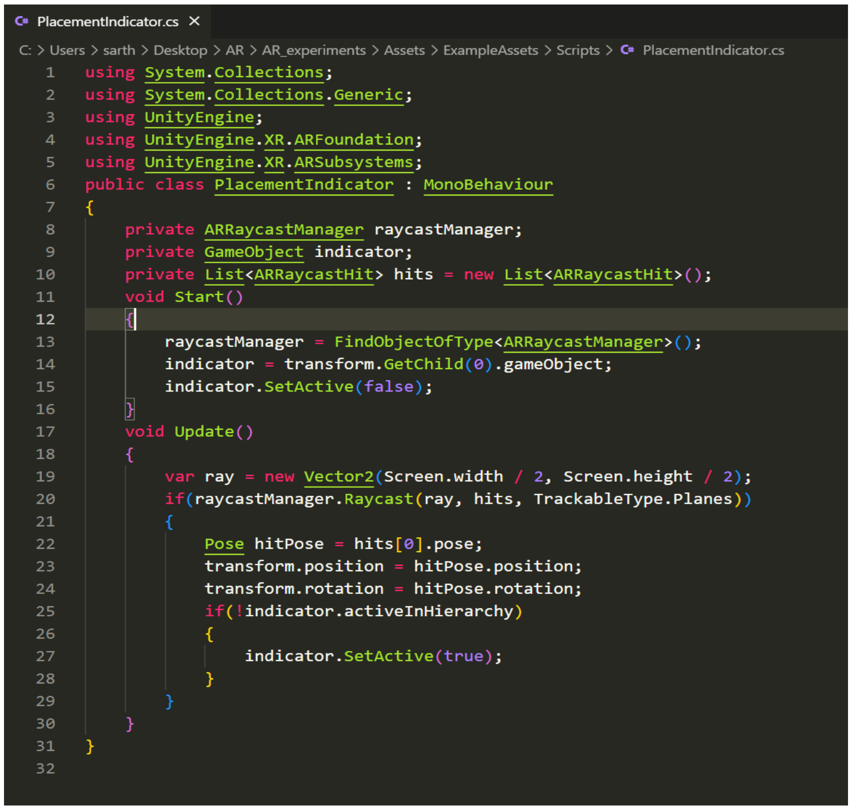
Unity Console output:



ARRaycastPlace.cs

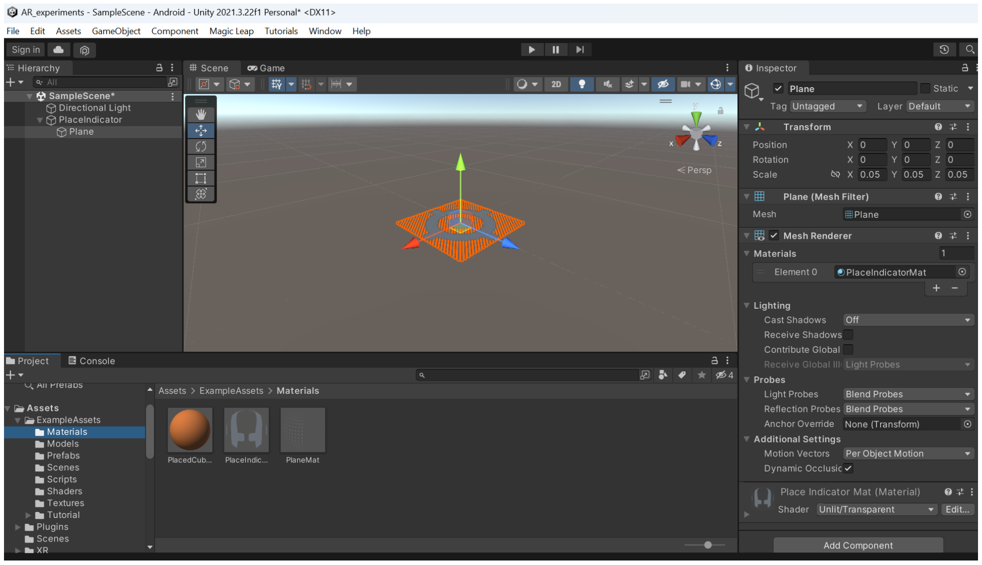
C: > Users > sarth > Desktop > AR > AR_experiments > Assets > ExampleAssets > Scripts > ARRaycastPlace.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.XR.ARFoundation;
using UnityEngine.XR.ARSubsystems;
public class ARRaycastPlace : MonoBehaviour
{
    public ARRaycastManager raycastManager;
    public GameObject objectToPlace;
    public Camera arCamera;
    private List<ARRaycastHit> hits = new List<ARRaycastHit>();
    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {
        Ray ray = arCamera.ScreenPointToRay(Input.mousePosition);
        if(Input.GetMouseButton(0))
        {
            if(raycastManager.Raycast(ray, hits, TrackableType.Planes))
            {
                Pose hitPose = hits[0].pose;
                Instantiate(objectToPlace, hitPose.position, hitPose.rotation);
            }
        }

    }
}
```


PlacementIndicator.cs

C: > Users > sarth > Desktop > AR > AR_experiments > Assets > ExampleAssets > Scripts > PlacementIndicator.cs

```csharp
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.XR.ARFoundation;
using UnityEngine.XR.ARSubsystems;
public class PlacementIndicator : MonoBehaviour
{
    private ARRaycastManager raycastManager;
    private GameObject indicator;
    private List<ARRaycastHit> hits = new List<ARRaycastHit>();
    void Start()
    {
        raycastManager = FindObjectOfType<ARRaycastManager>();
        indicator = transform.GetChild(0).gameObject;
        indicator.SetActive(false);
    }
    void Update()
    {
        var ray = new Vector2(Screen.width / 2, Screen.height / 2);
        if(raycastManager.Raycast(ray, hits, TrackableType.Planes))
        {
            Pose hitPose = hits[0].pose;
            transform.position = hitPose.position;
            transform.rotation = hitPose.rotation;
            if(!indicator.activeInHierarchy)
            {
                indicator.SetActive(true);
            }
        }
    }
}

```


AR_experiments - SampleScene - Android - Unity 2021.3.22f1 Personal* <DX11>

Inspector

Plane ✓ — Static

Tag: Untagged — Layer: Default

Transform

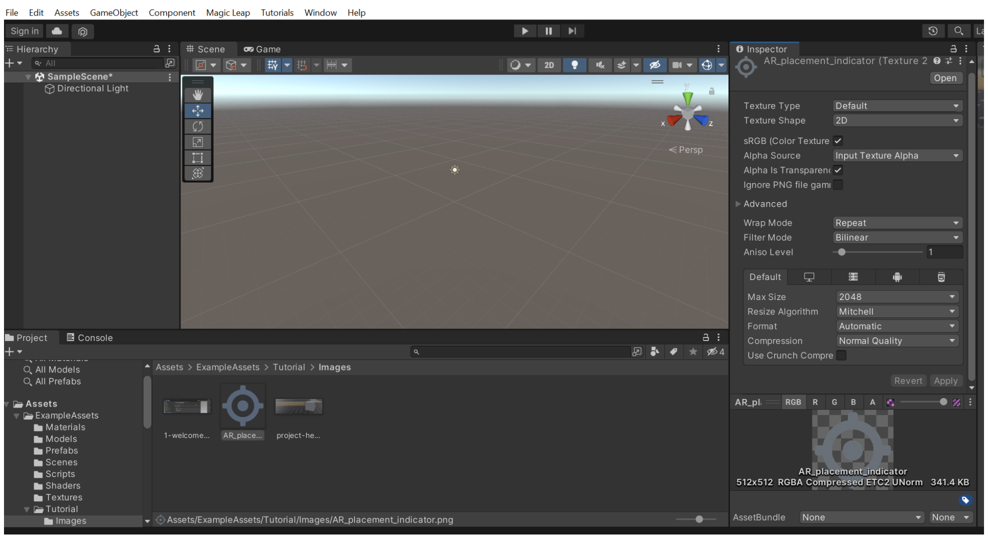
	X	Y	Z
Position	0	0	0
Rotation	0	0	0
Scale	0.05	0.05	0.05

Plane (Mesh Filter)
- Mesh: Plane

Mesh Renderer ✓
- Materials: 1
  - Element 0: PlaceIndicatorMat
- Lighting
  - Cast Shadows: Off
  - Receive Shadows: ☐
  - Contribute Global: ☐
  - Receive Global Ill: Light Probes
- Probes
  - Light Probes: Blend Probes
  - Reflection Probes: Blend Probes
  - Anchor Override: None (Transform)
- Additional Settings
  - Motion Vectors: Per Object Motion
  - Dynamic Occlusic: ✓

Place Indicator Mat (Material)
- Shader: Unlit/Transparent — Edit...

Add Component


File Edit Assets GameObject Component Magic Leap Tutorials Window Help
Sign in
Hierarchy
Scene
Game
Inspector
Persp
Project
Console

SampleScene*
Directional Light

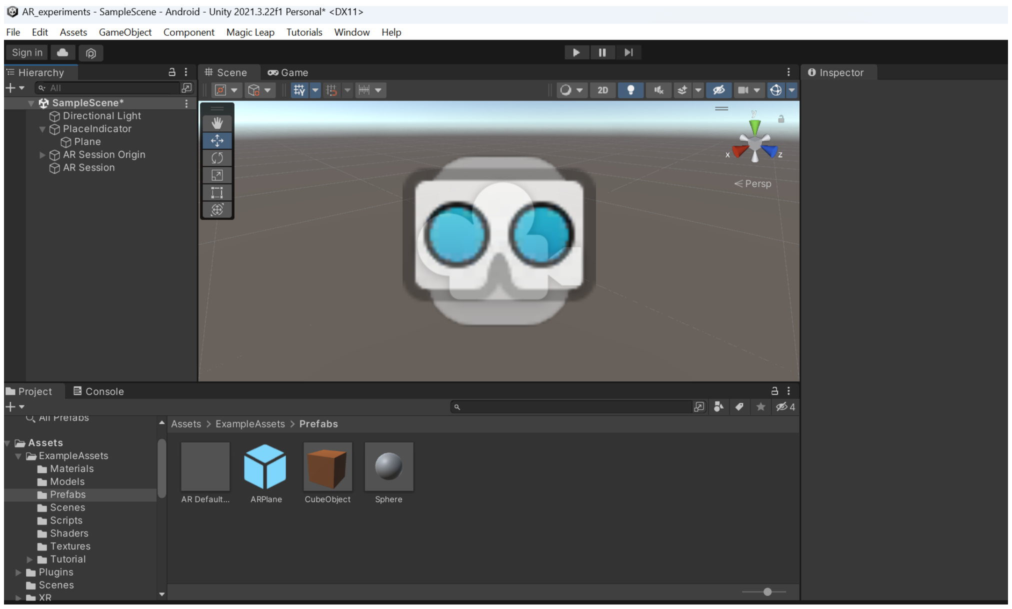
AR_placement_indicator (Texture 2
Open

Texture Type Default
Texture Shape 2D
sRGB (Color Texture ✓
Alpha Source Input Texture Alpha
Alpha Is Transparenc ✓
Ignore PNG file gam
Advanced
Wrap Mode Repeat
Filter Mode Bilinear
Aniso Level 1
Default
Max Size 2048
Resize Algorithm Mitchell
Format Automatic
Compression Normal Quality
Use Crunch Compre
Revert Apply

AR_placement_indicator
512x512 RGBA Compressed ETC2 UNorm 341.4 KB
AssetBundle None None

All Models
All Prefabs
Assets
ExampleAssets
Materials
Models
Prefabs
Scenes
Scripts
Shaders
Textures
Tutorial
Images

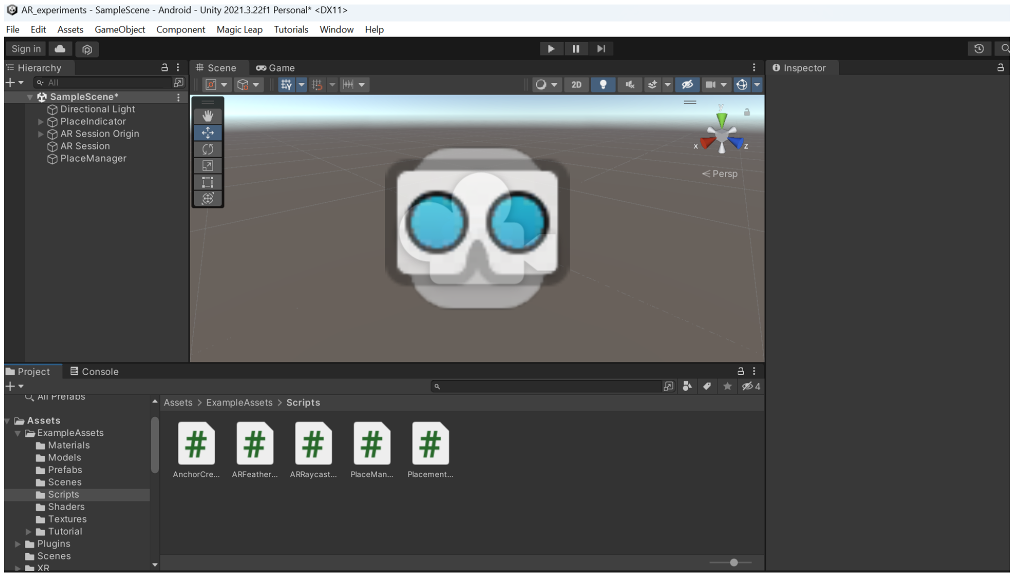
Assets > ExampleAssets > Tutorial > Images
1-welcome... AR_place... project-he...
Assets/ExampleAssets/Tutorial/Images/AR_placement_indicator.png

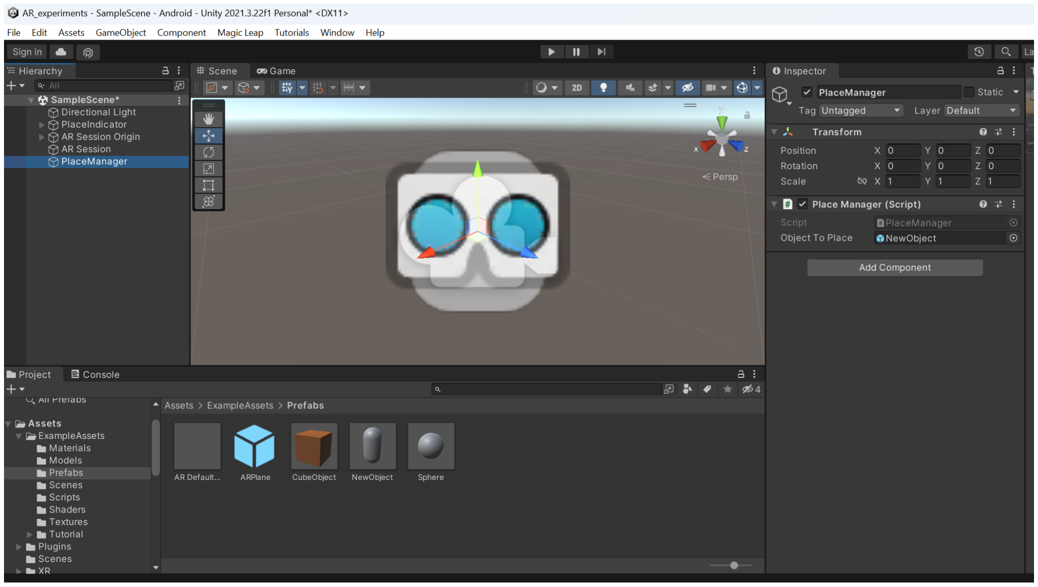

AR_experiments - SampleScene - Android - Unity 2021.3.22f1 Personal* <DX11>
File Edit Assets GameObject Component Magic Leap Tutorials Window Help
Sign in
Hierarchy
Scene
Game
Inspector
Persp
Project
Console

SampleScene*
Directional Light
PlaceIndicator
Plane
AR Session Origin
AR Session

All Prefabs
Assets
ExampleAssets
Materials
Models
Prefabs
Scenes
Scripts
Shaders
Textures
Tutorial
Plugins
Scenes
XR

Assets > ExampleAssets > Prefabs
AR Default... ARPlane CubeObject Sphere

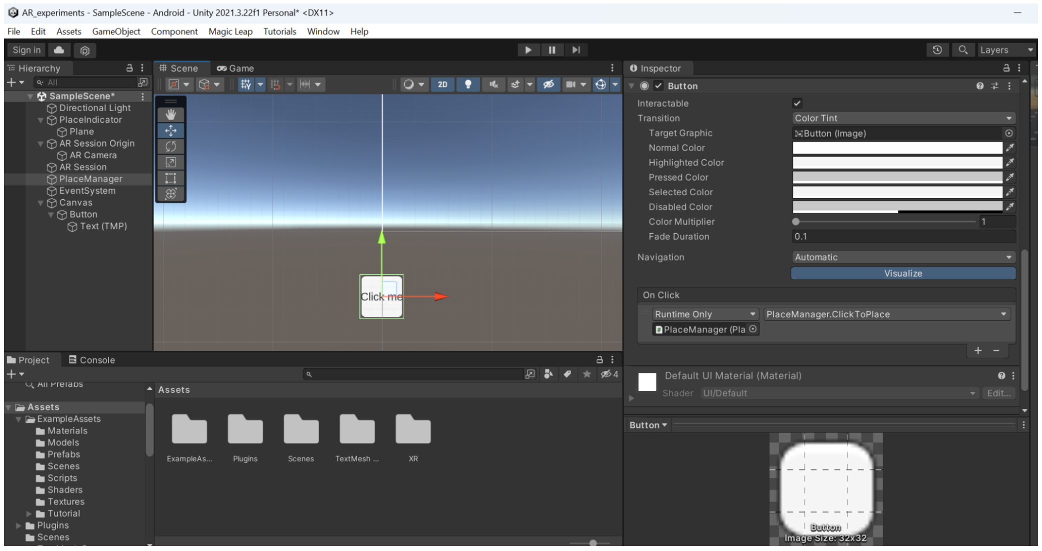




PlaceManager Static
Tag Untagged Layer Default

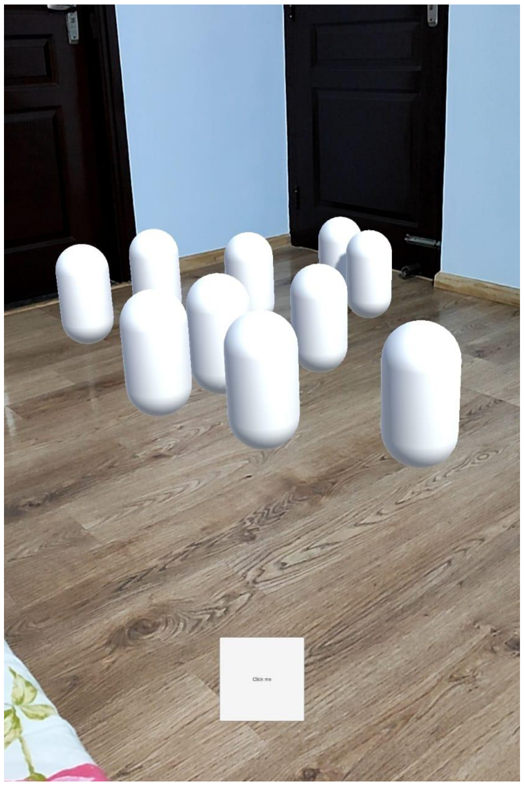
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Place Manager (Script)
Script PlaceManager
Object To Place NewObject



```
Button
Interactable          ✓
Transition            Color Tint
  Target Graphic      Button (Image)
  Normal Color
  Highlighted Color
  Pressed Color
  Selected Color
  Disabled Color
  Color Multiplier    1
  Fade Duration       0.1
Navigation            Automatic
                      Visualize
On Click
  Runtime Only        PlaceManager.ClickToPlace
  PlaceManager (Pla
Default UI Material (Material)
  Shader  UI/Default
```

Application output:



Experiment - 6

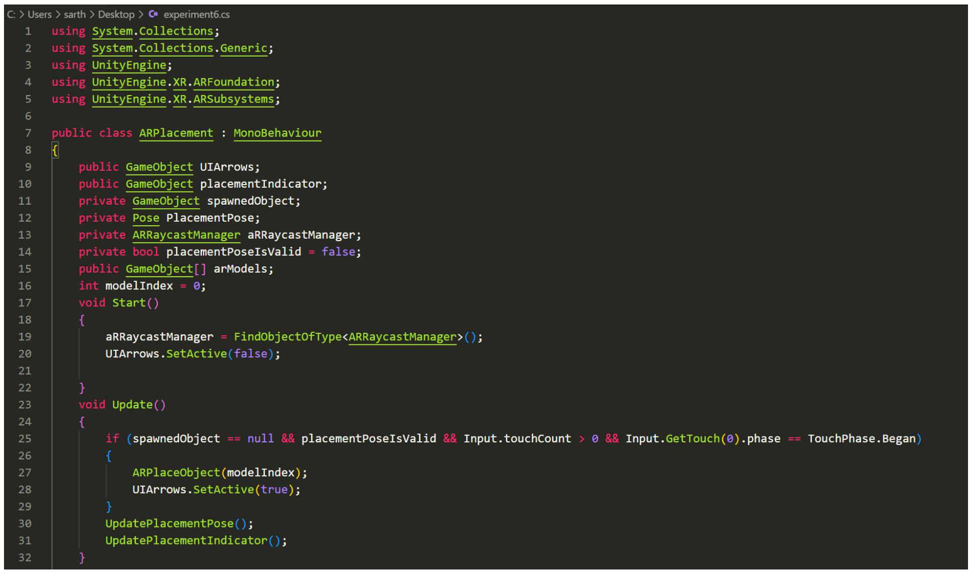
Aim: Build an Augmented Reality application using Unity to use arrows as placement indicators to summon multiple AR objects.

Methodology:

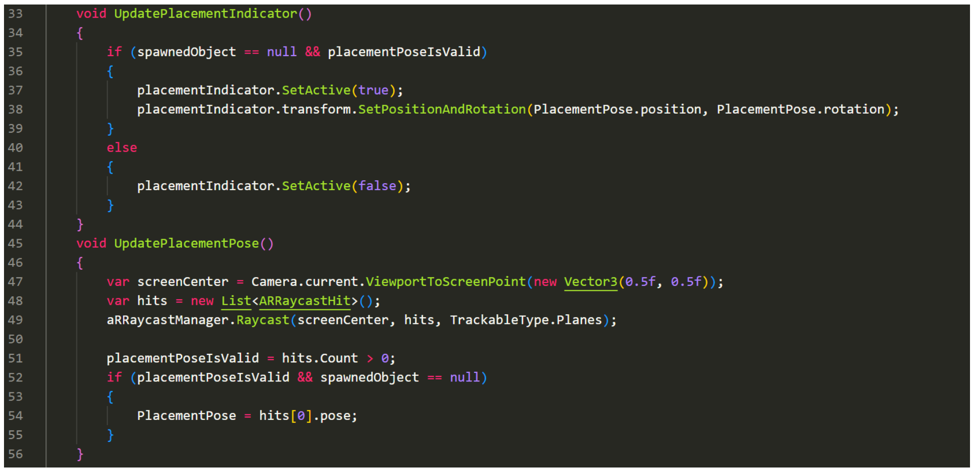
1. Create a new Unity project and import the necessary assets for your AR application.
2. Create a new scene and add a camera and directional light to the scene.
3. Import the 3D models of the arrows and the AR objects that you want to place.
4. Create a script that will instantiate the arrows when the camera detects a flat surface for placement. You can use raycasting to detect the surface and instantiate the arrows at the detected position.
5. Add a touch input function to the script to detect when the user touches the screen on an arrow. When an arrow is touched, you can remove the arrow and instantiate the corresponding AR object at that position.
6. Add a script to the AR objects that will allow them to be dragged and placed in different positions.

7. Test the application on an AR-compatible device to ensure that it is functioning properly.
8. Optimize the application by minimizing the number of polygons, using texture compression, and implementing other performance optimizations.

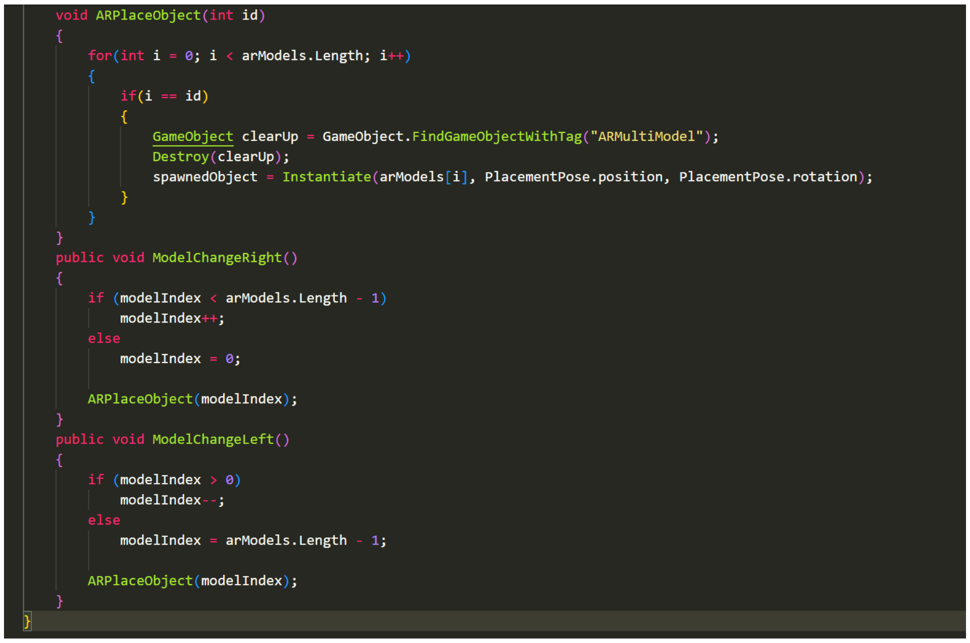
Unity console output:



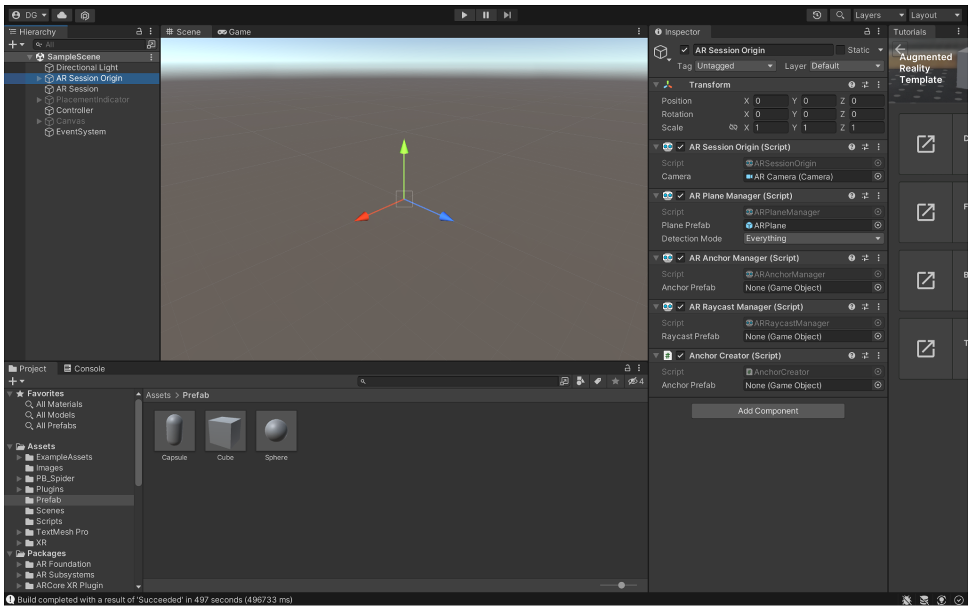
```
C: > Users > sarth > Desktop > experiment6.cs
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.XR.ARFoundation;
using UnityEngine.XR.ARSubsystems;

public class ARPlacement : MonoBehaviour
{
    public GameObject UIArrows;
    public GameObject placementIndicator;
    private GameObject spawnedObject;
    private Pose PlacementPose;
    private ARRaycastManager aRRaycastManager;
    private bool placementPoseIsValid = false;
    public GameObject[] arModels;
    int modelIndex = 0;
    void Start()
    {
        aRRaycastManager = FindObjectOfType<ARRaycastManager>();
        UIArrows.SetActive(false);

    }
    void Update()
    {
        if (spawnedObject == null && placementPoseIsValid && Input.touchCount > 0 && Input.GetTouch(0).phase == TouchPhase.Began)
        {
            ARPlaceObject(modelIndex);
            UIArrows.SetActive(true);
        }
        UpdatePlacementPose();
        UpdatePlacementIndicator();
    }
```



```
    void UpdatePlacementIndicator()
    {
        if (spawnedObject == null && placementPoseIsValid)
        {
            placementIndicator.SetActive(true);
            placementIndicator.transform.SetPositionAndRotation(PlacementPose.position, PlacementPose.rotation);
        }
        else
        {
            placementIndicator.SetActive(false);
        }
    }
    void UpdatePlacementPose()
    {
        var screenCenter = Camera.current.ViewportToScreenPoint(new Vector3(0.5f, 0.5f));
        var hits = new List<ARRaycastHit>();
        aRRaycastManager.Raycast(screenCenter, hits, TrackableType.Planes);

        placementPoseIsValid = hits.Count > 0;
        if (placementPoseIsValid && spawnedObject == null)
        {
            PlacementPose = hits[0].pose;
        }
    }
```



```
    void ARPlaceObject(int id)
    {
        for(int i = 0; i < arModels.Length; i++)
        {
            if(i == id)
            {
                GameObject clearUp = GameObject.FindGameObjectWithTag("ARMultiModel");
                Destroy(clearUp);
                spawnedObject = Instantiate(arModels[i], PlacementPose.position, PlacementPose.rotation);
            }
        }
    }
    public void ModelChangeRight()
    {
        if (modelIndex < arModels.Length - 1)
            modelIndex++;
        else
            modelIndex = 0;

        ARPlaceObject(modelIndex);
    }
    public void ModelChangeLeft()
    {
        if (modelIndex > 0)
            modelIndex--;
        else
            modelIndex = arModels.Length - 1;

        ARPlaceObject(modelIndex);
    }
}
```



```
Hierarchy
SampleScene
  Directional Light
  AR Session Origin
  AR Session
  PlacementIndicator
  Controller
  Canvas
  EventSystem

Inspector
AR Session Origin                Static
Tag Untagged          Layer Default
Transform
  Position   X 0   Y 0   Z 0
  Rotation   X 0   Y 0   Z 0
  Scale      X 1   Y 1   Z 1
AR Session Origin (Script)
  Script           ARSessionOrigin
  Camera           AR Camera (Camera)
AR Plane Manager (Script)
  Script           ARPlaneManager
  Plane Prefab     ARPlane
  Detection Mode   Everything
AR Anchor Manager (Script)
  Script           ARAnchorManager
  Anchor Prefab    None (Game Object)
AR Raycast Manager (Script)
  Script           ARRaycastManager
  Raycast Prefab   None (Game Object)
Anchor Creator (Script)
  Script           AnchorCreator
  Anchor Prefab    None (Game Object)
Add Component

Project
Favorites
  All Materials
  All Models
  All Prefabs
Assets
  ExampleAssets
  Images
  PB_Spider
  Plugins
  Prefab
  Scenes
  Scripts
  TextMesh Pro
  XR
Packages
  AR Foundation
  AR Subsystems
  ARCore XR Plugin

Assets > Prefab
Capsule  Cube  Sphere

Build completed with a result of 'Succeeded' in 497 seconds (496733 ms)
```

Application output: