AUGMENTED REALITY
VIRTUAL REALITY

PROJECT

FURNITURE AR

TEAM MEMBERS:

Hasini Ganta– 106123049

Nehasri Talari– 106123131

Code: C# script

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

using UnityEngine.UI;
using UnityEngine.EventSystems;

using UnityEngine.XR.ARFoundation;
using UnityEngine.XR.ARSubsystems;
using Unity.XR.CoreUtils;
public class FurniturePlacementManager : MonoBehaviour
{
   public GameObject SpawnableFurniture;
   public XROrigin sessionOrigin;
   public ARRaycastManager raycastManager;
   public ARPlaneManager planeManager;
   private List<ARRaycastHit> raycastHits = new List<ARRaycastHit>();

   private void Update()
   {   if (Input.touchCount > 0)
      {
         if (Input.GetTouch(0).phase == TouchPhase.Began)
         {
         bool collision = raycastManager.Raycast(Input.GetTouch(0).position, raycastHits,
                                                 TrackableType.PlaneWithinPolygon);

            if (collision && isButtonPressed() == false)
            {
               if (SpawnableFurniture != null)
               {
                  Debug.Log("Instantiating furniture...");
                  GameObject _object = Instantiate(SpawnableFurniture);

                  // Temporarily place object at origin to confirm it's visible
                  _object.transform.position = new Vector3(0, 0, 0);
                  _object.transform.rotation = Quaternion.identity; // Optional: set a default rotation

                  Debug.Log("Furniture placed at origin.");
               }
               else
               {
                  Debug.LogWarning("SpawnableFurniture is null! Assign a prefab in the Inspector.");
               }
            }
            // Disable planes once furniture is placed
            foreach (var plane in planeManager.trackables)
            {
               plane.gameObject.SetActive(false);
            }
            planeManager.enabled = false;
         }
      }
```

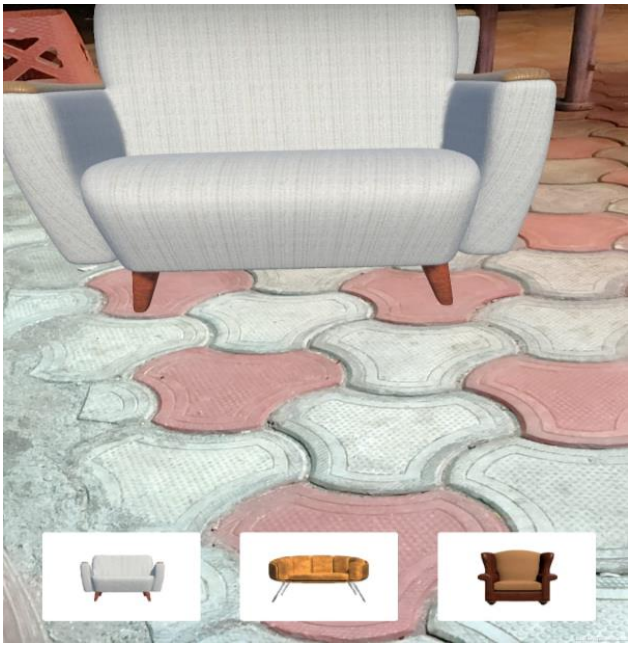
```
}
 public bool isButtonPressed()
  {
    if (EventSystem.current.currentSelectedGameObject?.GetComponent<Button>() == null)
    {
      return false;
    }
    else
    {
      return true;
    }
  }

  public void SwitchFurniture(GameObject furniture)
  {
    SpawnableFurniture = furniture;
  }
}
```

Sample Test Case

Test Case: Verify furniture placement on detected planes.

1. Run the application on an AR-compatible device.
2. Select a furniture item to be placed (make sure the item is set via SpawnableFurniture).
3. Point the device at a flat surface, and wait for the plane detection to register the surface.
4. Tap on the detected plane to place the furniture item.

- Expected Result: The selected furniture model should appear on the plane, visible and correctly oriented. All detected planes should then be hidden from view after placement.



Instructions for Execution

1. Prerequisites

- **Supported Device**: Ensure the mobile device supports Google ARCore (Android) or ARKit (iOS). Most newer devices from major brands support these AR frameworks. The device used by us is Realme and Oneplus Nord series.
- **Unity Setup**: Install Unity with **AR Foundation**, **ARCore XR Plugin** (for Android).
- **Development Environment**: Install **Android Build Support** (for Android).

2. Setting Up the Project in Unity

1. **Open Project**: Open the Unity project containing the AR app.
2. **Configure Build Settings**:
    - Go to **File > Build Settings**.
    - Select **Android** as the target platform and click **Switch Platform**.
    - **For Android**: Set the Minimum API Level to at least **Android 7.0 (Nougat)**, which is required for ARCore.
3. **Player Settings**:
    - In **Player Settings > XR Plug-in Management**, enable **ARCore** (for Android) .
    - Adjust package name, app version, and other settings in **Player Settings > Other Settings**.
4. **Add Scenes**: Ensure the main AR scene is added to the build in the Build Settings window.

3. Generating the APK (for Android).

1. **Build for Android (APK)**:
    - In the Build Settings window, select the scene and select **Build**
    - Choose a folder to save the APK file and start the build process.
    - Once the APK is generated, it will be saved in the chosen location.

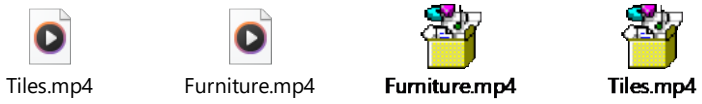
4. Transferring the APK

- **For Android**:
    - Transfer the APK file to the Android device using USB, email, or a cloud service (e.g., Google Drive).
    - Tap on the APK file on the device to install the app.

5. Running the App on the Mobile Device

1. **Open the App**: Launch the app on the mobile device.
2. **Permissions**: Allow any requested permissions, especially for camera access, which is necessary for AR functionality.
3. **AR Experience**: Point the device camera at a flat surface to allow plane detection. Once detected, follow the instructions to tap and place furniture items within the AR environment.

DEMO VIDEOS



6. How It Differs from Existing Apps

Existing apps do not allow to switch between the furnitures at the same time. Once we select furniture and visualize it in our environment, we need to go back to look at another piece of furniture in existing e-commerce websites. We incorporated a Switch furniture functionality to switch between objects that can be placed in the environment. We can also place multiples objects in the same environment which is very helpful in retail for comparison of the furniture we buy.

- Ease of Use: Simplified tap-to-place functionality, enabling users to quickly and easily visualize furniture in real time.
- Selective Placement: Allows users to choose from multiple furniture options using the SwitchFurniture function, making it flexible for different items.
- Improved UI Responsiveness: The app includes touch detection that prevents accidental object placement when interacting with UI buttons.

Team Members' Contribution

Each of us tried to implement one project on augmented reality for various senarios. We created an environment where we can place tiles, then we explored furniture and once it started working we focused on how to impove it by including the button functionality and switching between objects. Then we focused on the plane detection in the environment to place the object. We helped each other at various stages of code, learning new concepts and trying to incorporate in our project.

THANK YOU

Problem Statement

The topic assigned to us is Retail/Digital Marketing. In the digital marketing and retail sectors, enhancing customer engagement and reducing hesitation in purchasing decisions remain significant challenges. In online shopping, especially for furniture, customers often struggle to imagine how a product will look in their own space. This uncertainty can lead to hesitation and sometimes even returns. To help solve this problem, this project uses Augmented Reality (AR) to let customers tap on a furniture item and see it in their own environment through their phone or tablet. By allowing customers to visualize products in real life, this feature aims to make online shopping easier, increase confidence in purchases, and improve the overall shopping experience on e-commerce websites.

Design

The core design of this AR furniture placement feature focuses on detecting flat surfaces (like floors or tables) in the user's environment and allowing furniture to be placed on them. Key design elements include:

- Surface Detection: Uses AR plane detection to find flat surfaces where furniture can be placed.
- Furniture Placement: Allows users to tap on the detected plane to instantiate the selected furniture model.
- User Interaction: Ensures that interactions are intuitive and that users can select from different furniture items using a button-based UI.

Tools Used

- Unity: Primary tool for building the AR application, providing the environment for scripting and UI design.
- AR Foundation: Unity's AR framework, which handles plane detection and user interactions for AR applications.
- C#: Used for scripting the AR logic, including raycasting, object instantiation, and touch interactions.
- ARRaycastManager & ARPlaneManager: Manages raycasting for detecting planes and handles the display of plane surfaces for object placement.