VR Implementation Document

for

InnerSpace

Version 1.0

Prepared by

Group #: N/A **Group Name: Studio Centauri**

Drashti Modi	**210365**	**drashti21@iitk.ac.in**
Jhalak Sharma	**210474**	**jhalak21@iitk.ac.in**
Khush Khandelwal	**210511**	**khushk21@iitk.ac.in**
Naman Kumar Jaiswal	**220687**	**namankj22@iitk.ac.in**
Parv Chhabra	**210706**	**pchhabra21@iitk.ac.in**
Pranav Susana	**210739**	**pranavs21@iitk.ac.in**
Shashikant Yadav	**210966**	**syadav21@iitk.ac.in**

Course: **DES643**

Mentor Professor: ***Dr. Amar Bahera***

Date: **19 April 2024**

1 Implementation Details

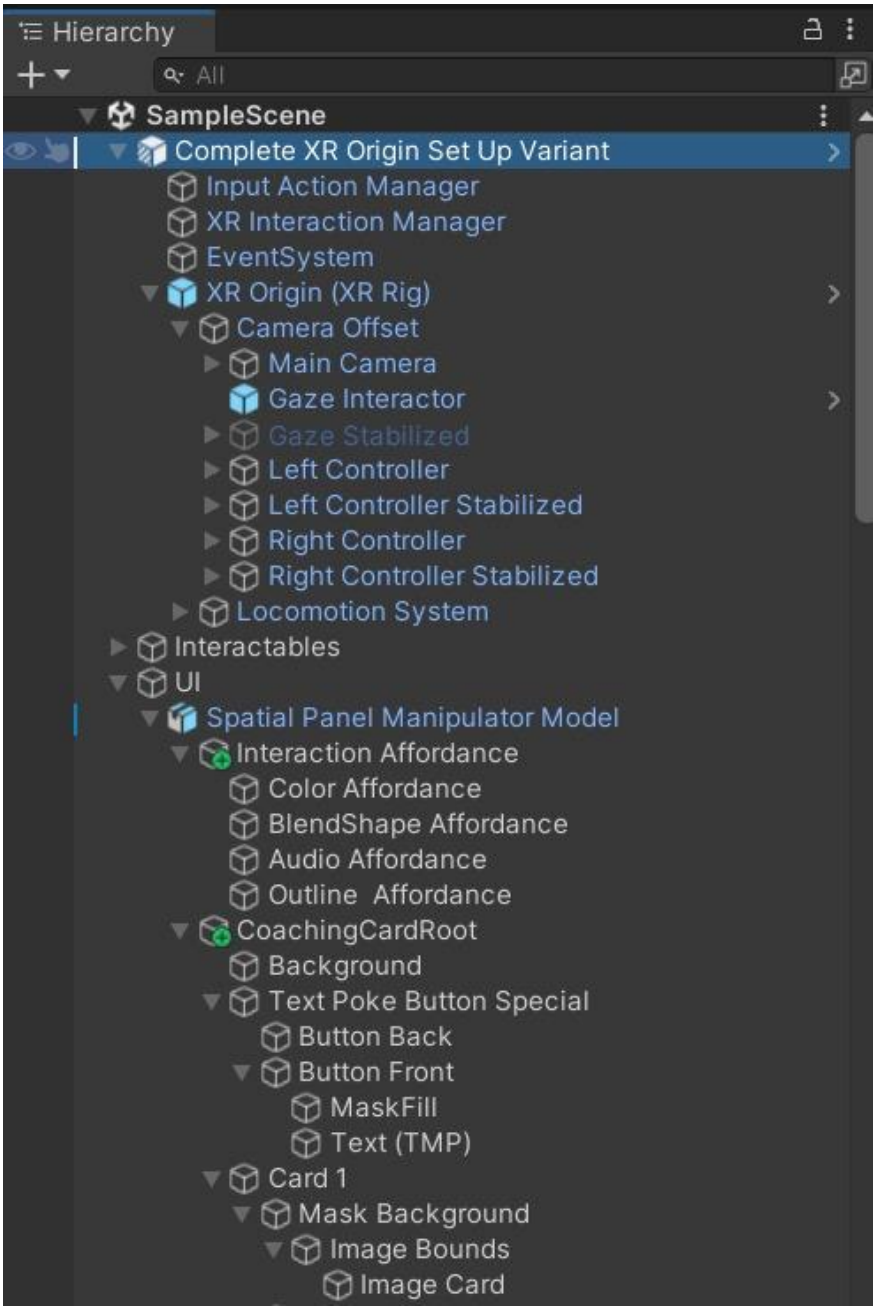
Embark on a journey of tranquillity and self-discovery with our immersive virtual meditation environment. Whether seeking solace in the serenity of nature or connecting with a community of like-minded individuals, our platform offers two unique modes: individual meditation and collective practice. Through guided gestures, tranquil visuals, and dynamic terrain interaction, users can deepen their meditation practice, foster inner peace, and experience a profound sense of unity with themselves and the universe.

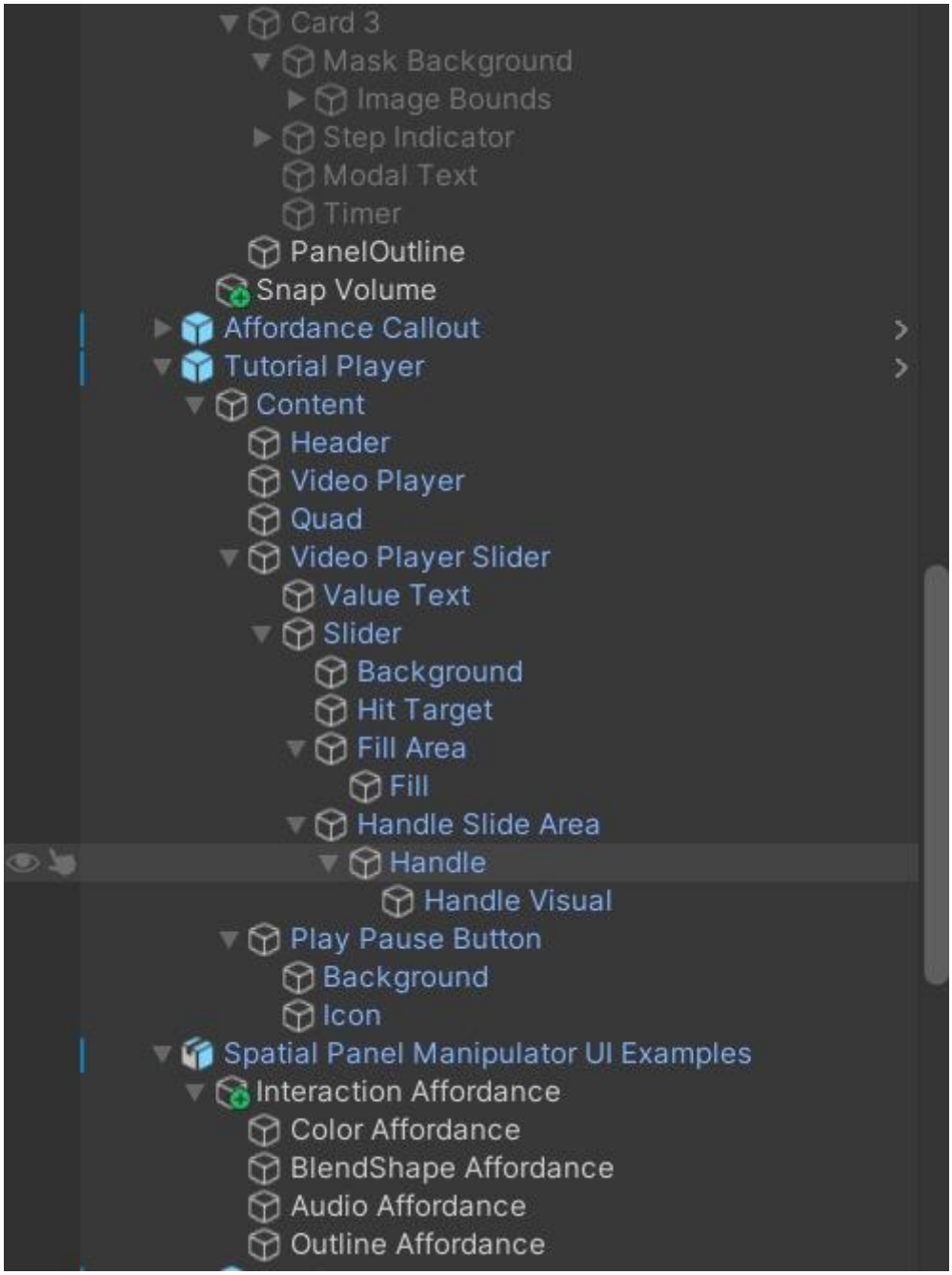
Major Features and Implementation:

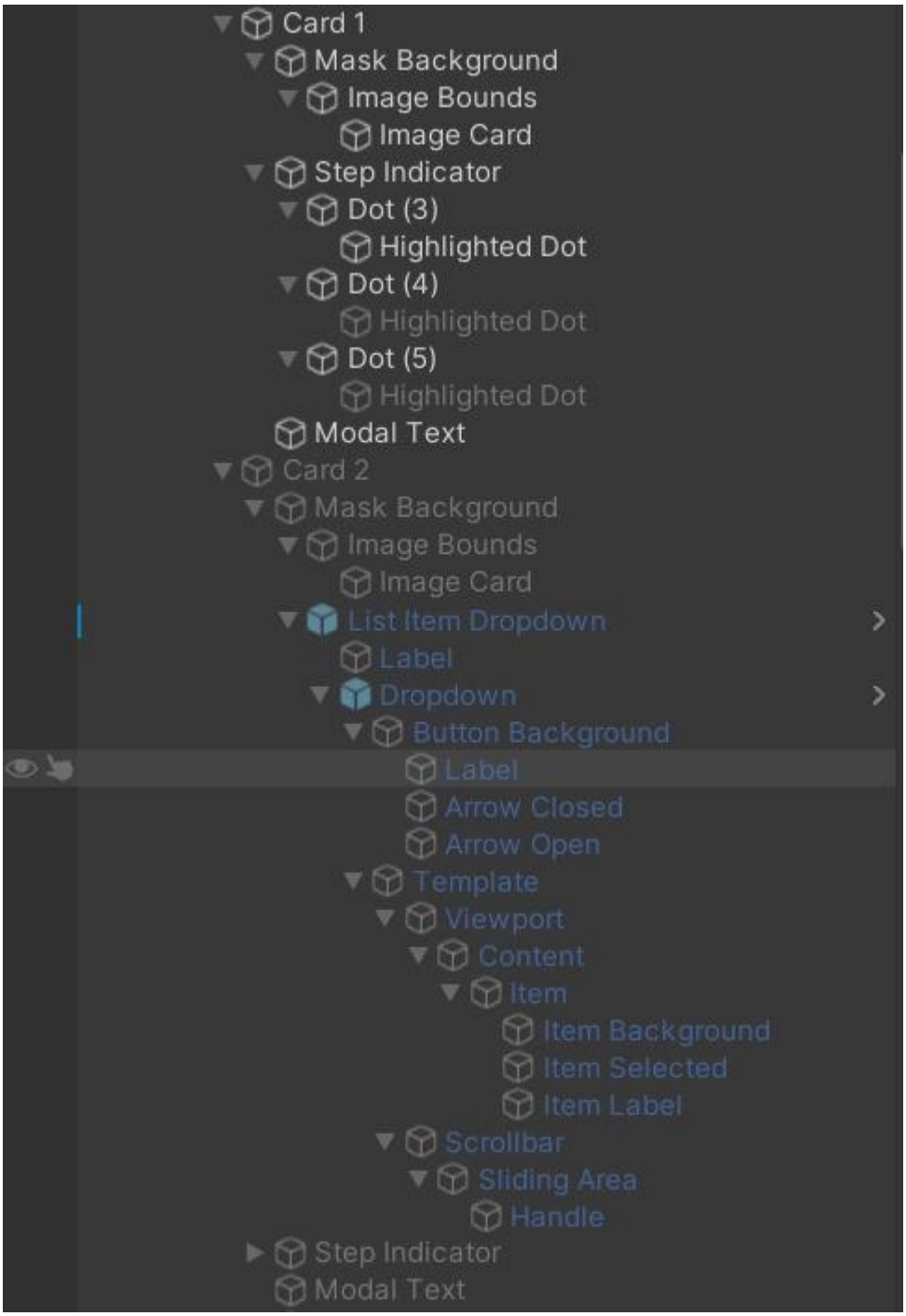
1. Dual Meditation Modes:
    - Our virtual meditation environment offers two distinct modes: individual meditation and collective meditation.
    - Individual Meditation: Users can immerse themselves in a serene natural setting, surrounded by lush landscapes and calming sounds. They can perform guided meditation gestures to deepen their practice.
    - Collective Meditation: Users can experience the sense of community by joining a virtual yoga class with other participants and an instructor. The collective energy enhances the meditation experience, fostering a deeper connection with oneself and others.
2. Guided Gestures for Meditation:
    - Users are guided through various meditation gestures that facilitate relaxation, mindfulness, and inner peace.
    - Gestures are implemented using intuitive controls, allowing users to seamlessly interact with the virtual environment through hand movements or controller inputs.
3. Tranquil Visuals and Sounds:
    - The environment is enriched with tranquil visuals, including serene natural landscapes, calming colours, and ambient lighting.
    - Soothing sounds of nature, gentle music, and guided instructions create an immersive auditory experience, enhancing the overall sense of relaxation and tranquillity.
4. Dynamic Terrain Interaction:
    - The virtual terrain dynamically responds to user interactions, providing a realistic and engaging meditation environment.
5. Alignment with Sahaja Yoga Principles:
    - Our virtual meditation environment is inspired by the principles of Sahaja Yoga, aiming to facilitate a deeper understanding of self and the universe.
    - Features such as stronger vibrational sensations, effortless transition into thoughtless awareness, and obstacle clearance align with the core tenets of Sahaja Yoga, promoting holistic well-being and spiritual growth.

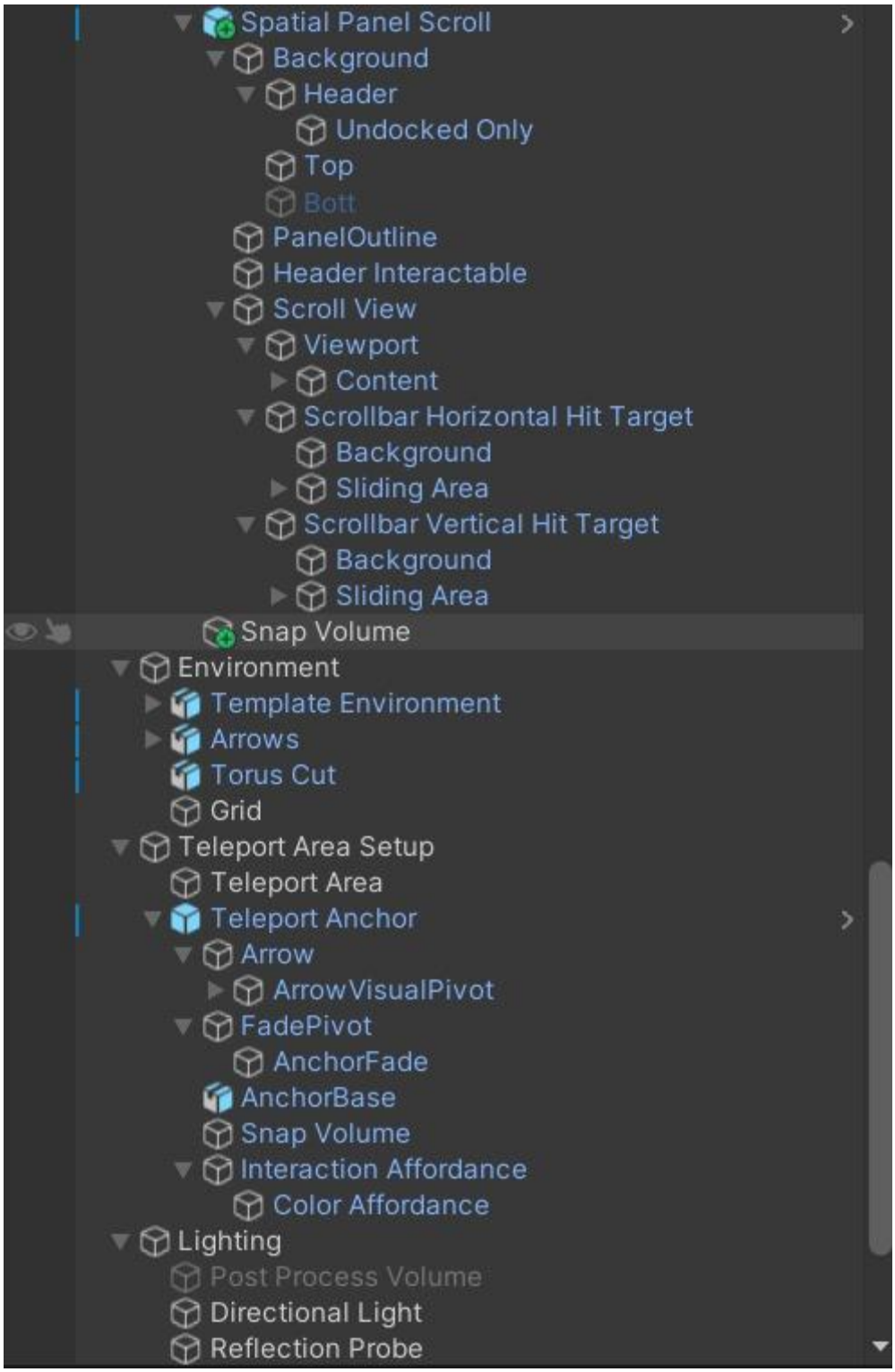
2 Codebase

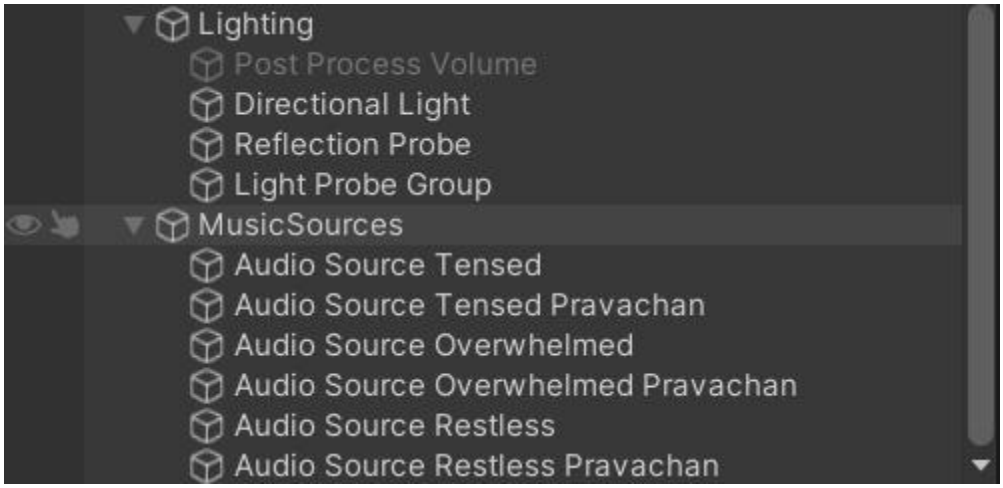
UNITY-HIERARCHY PANEL:




Card 3
Mask Background
Image Bounds
Step Indicator
Modal Text
Timer
PanelOutline
Snap Volume
Affordance Callout
Tutorial Player
Content
Header
Video Player
Quad
Video Player Slider
Value Text
Slider
Background
Hit Target
Fill Area
Fill
Handle Slide Area
Handle
Handle Visual
Play Pause Button
Background
Icon
Spatial Panel Manipulator UI Examples
Interaction Affordance
Color Affordance
BlendShape Affordance
Audio Affordance
Outline Affordance


Card 1
Mask Background
Image Bounds
Image Card
Step Indicator
Dot (3)
Highlighted Dot
Dot (4)
Highlighted Dot
Dot (5)
Highlighted Dot
Modal Text
Card 2
Mask Background
Image Bounds
Image Card
List Item Dropdown
Label
Dropdown
Button Background
Label
Arrow Closed
Arrow Open
Template
Viewport
Content
Item
Item Background
Item Selected
Item Label
Scrollbar
Sliding Area
Handle
Step Indicator
Modal Text


Spatial Panel Scroll
Background
Header
Undocked Only
Top
Bott
PanelOutline
Header Interactable
Scroll View
Viewport
Content
Scrollbar Horizontal Hit Target
Background
Sliding Area
Scrollbar Vertical Hit Target
Background
Sliding Area
Snap Volume
Environment
Template Environment
Arrows
Torus Cut
Grid
Teleport Area Setup
Teleport Area
Teleport Anchor
Arrow
ArrowVisualPivot
FadePivot
AnchorFade
AnchorBase
Snap Volume
Interaction Affordance
Color Affordance
Lighting
Post Process Volume
Directional Light
Reflection Probe



Game Objects:

- **Complete XR origin Setup Variant:** The Variant game object is used in Unity's XR origin setup to define the initial position and configuration of the XR camera and tracking space within a virtual reality or augmented reality environment. It allows developers to customise the starting point and orientation of the user's VR/AR experience, including aspects like camera placement, interaction points, and collision boundaries. Subobjects for this object are as follows.
    - **Input Action Manager:** This game object in Unity is used to define and manage input actions, including controls, bindings, and input behaviours, for player interaction with the game.
    - **XR Interaction Manager:** This game object in Unity is used to manage and control interactions between input devices (such as VR controllers) and interactive objects in a virtual reality (VR) or augmented reality (AR) environment.
    - **Event System:** This game object in Unity is used to handle input events, such as clicks, taps, and keyboard inputs, for user interaction with UI elements and game objects in the scene.
- **Interactables:** This game object in Unity is used to define objects within a scene that can be interacted with by the player or other game elements. It typically includes components such as colliders, rigidbodies, and scripts to enable interactions like picking up, moving, activating, or manipulating objects in the game world, adding depth and interactivity to the gameplay experience.
- **UI:** This is used to create and manage graphical user interface elements such as buttons, text, images, sliders, and panels within the game's interface. It allows developers to design and display interactive elements for player input, feedback, menus, HUD (Heads-Up Display), and other visual components to enhance the user experience and facilitate interaction with the game.
- **Spatial Panel Manipulator:** The Spatial Panel Manipulator game object in Unity is used to dynamically adjust and manipulate spatial UI elements, such as panels and canvases, based on user interactions or environmental changes in virtual reality (VR) or augmented reality (AR) environments. It enables developers to create responsive and immersive UI experiences that adapt to the user's perspective and actions within the 3D space.
- **Affordance Callout :** The Affordance Callout game object in Unity is used to visually communicate to players the interactive capabilities of objects or elements in the game, such as indicating where and how objects can be manipulated or interacted with. It helps improve user understanding and engagement by providing clear cues about gameplay mechanics and interactions.
- **Tutorial Player :** The Tutorial Player game object in Unity is used to guide players through tutorials or instructional sequences within the game, providing step-by-step instructions, hints, and demonstrations to help players learn game mechanics, controls, and objectives effectively. It enhances the onboarding experience and reduces player frustration by offering interactive guidance and feedback during gameplay.
- **Environment :** The Environment game object in Unity is used to define the overall setting, ambiance, and background elements of a scene, including skyboxes, lighting, terrain, and other environmental

features that contribute to the atmosphere and immersion of the game world. It plays a crucial role in creating a cohesive and visually appealing environment for the player's experience.

- **Lighting :** The Lighting game object in Unity is used to configure and manage the lighting conditions within a scene, including directional, point, spotlights, and ambient lighting settings, to illuminate objects, surfaces, and environments realistically and enhance visual quality and atmosphere in the game world. It plays a crucial role in creating mood, visibility, and overall aesthetics in the game.

SCRIPTS:

- **MusicBoxAudioSource.cs :** This script manages audio playback in a Unity VR game. It includes references to UI elements such as a volume slider and a dropdown menu for track selection. The script allows users to adjust volume and select between different audio tracks categorised as "Overwhelmed," "Restless," and "Tensed." It provides functions to handle slider and dropdown value changes, as well as play and pause functionality based on user input.
- **Timer.cs:** The Timer script is responsible for managing a countdown timer in a Unity environment. It utilises a TextMeshProUGUI component to display the remaining time in minutes and seconds format. The timer starts at 3 minutes (181 seconds) when initiated. It continuously updates the timer display in real-time during gameplay, hiding it when the time runs out.
- **UI Manager.cs:** This script handles user interface interactions in a Unity environment. It provides functions for quitting the application, logging debug messages, and toggling between different UI panels. Specifically, it controls the visibility of panels related to music and meditation, enabling smooth transitions between them.
- **AnchorVisuals.cs :** This script aids in managing animations for teleport anchor visuals in a Unity XR environment. It controls the appearance and behaviour of elements such as a fading glow and an arrow indicating the teleportation direction. The script handles animations when the teleport interactor enters or exits the anchor selection, as well as hiding the arrow visual during teleportation.
- **BezierCurve.cs :** This script allows for drawing a bezier curve between two specified transforms in a Unity environment. It calculates control points based on start and end positions, adjusting the curve based on factors like curve factors and segment count. Additionally, it provides functionality for animating the curve's appearance, including gradient colour changes over time.
- **BooleanToggleVisualsController.cs :** This script manages the visual states of a boolean toggle switch UI element in Unity. It adjusts the position of the toggle knob based on its boolean value, animating it to the corresponding position. Additionally, it provides hover effects by translating the knob's z-position and resetting it upon exit, enhancing the user interaction experience.
- **Callout.cs :** The Callout script manages callouts in a Unity VR environment, typically used to display information like tooltips for world objects or controllers. It handles the behaviour of the callout when the user gazes at it, including activating associated tooltip and curve visuals after a specified dwell time. Additionally, it supports disabling the callout visuals upon gaze end or at the start, providing control over their visibility.
- **CalloutGazeController.cs:** The CalloutGazeController script detects when an object is within the field of view of a gaze transform and fires corresponding events. It evaluates the direction of the gaze using the dot product of the gaze transform's forward direction and the vector pointing towards the object. Events are triggered when the object enters or exits the gaze's field of view, with additional logic to handle large movements triggering exit events.
- **DestroyObject.cs :** The DestroyObject script is designed to automatically remove a GameObject from the scene after a specified duration. Upon start, it initiates a countdown based on the provided lifetime value, then destroys the GameObject once the time elapses. This script is useful for temporary or transient objects that are no longer needed after a certain period, helping manage scene clutter and optimise performance.
- **LaunchProjectile.cs :** The LaunchProjectile script enables the application of forward force to a projectile prefab upon instantiation in a Unity environment. It allows for specifying the projectile prefab, the starting point for instantiation, and the launch speed. When the `Fire()` method is called, it creates a new instance of the projectile prefab and applies the specified launch force in the forward direction from the starting point.
- **RayAttachModifier.cs :** This script enables an interactable object to snap to the source of an XR Ray Interactor upon selection, mimicking the behaviour of Force Grab. When enabled on an interactable object, it listens for selection events and adjusts the object's position to match the position of the ray interactor. This script ensures that the interactable snaps to the interactor's source instead of remaining at a distance, enhancing the precision and control of interactions in VR environments.
- **Rotator.cs :** The Rotor script allows for the rotation of an object at a user-defined speed in a Unity environment. By specifying the angular velocity in degrees per second through the m_Velocity variable, this script

continuously rotates the object around its axis as per the defined speed. This can be useful for creating dynamic visual effects or simulating motion in virtual environments.

- **StepManager.cs :** The StepManager script is responsible for controlling the steps in a coaching card within a Unity environment. It manages a list of steps, each represented by a GameObject and a corresponding button text. The script allows for transitioning between steps, activating and deactivating step objects accordingly. Additionally, it handles specific actions for certain steps, such as playing audio and displaying a timer when reaching a particular step index. This script provides a structured approach to guiding users through a series of instructional steps.
- **VideoPlayerRenderTexture.cs :** This script is responsible for creating a RenderTexture to render video content onto a target renderer within a Unity environment. Upon startup, the script instantiates a RenderTexture with specified width, height, and depth settings. It then creates a material with an appropriate shader (typically an Unlit/Texture shader) and assigns the RenderTexture as the main texture of the material. Finally, it links the RenderTexture to a VideoPlayer component, allowing the video content to be rendered onto the target renderer, such as a mesh or a UI element. This setup facilitates the integration of video playback into Unity scenes with flexible rendering options.
- **VideoTimeScrubControl.cs :** The VideoTimeScrubControl script enables users to control video playback through a UI slider. It manages play/pause functionality, scrubbing to specific times, and updates UI elements to reflect the current playback state and time. Additionally, it includes an option to automatically hide the slider after a period of inactivity for a cleaner user interface.
- **XRKnob.cs :** This script, XRKnob, implements an interactable knob in a Unity XR environment. It tracks the rotation of the interactor and updates the knob's value accordingly. The knob's rotation can be clamped within specified angle limits, and it supports angle increments for precise control. Events are triggered when the knob is rotated, and it provides visual feedback with a handle object.
- **XRPokeFollowAffordanceFill.cs :** This script, XRPokeFollowAffordanceFill, enables animation for an interactive object when poked. It moves a designated transform to follow the poke direction and scales a UI mask based on the interaction strength. Options include controlling smoothing speed, clamping movement distances, and returning to the initial position when interaction ends.
- **TerrainCheck.cs :** This TerrainCheck script manages the camera's position to prevent it from colliding with the terrain. It smoothly adjusts the camera's position above the terrain if a collision is detected. Additionally, there are commented-out sections related to checking the distance between the terrain and the camera, as well as handling camera zoom functionality. These sections are not currently in use but seem to be related to potential future functionality.

3 Completeness

Guided meditation sessions are becoming increasingly popular worldwide, catering to individuals seeking collective serenity without the need for physical travel. Inspired by the Sahaja Yoga centre's principles, our virtual meditation environment offers a comprehensive experience that encompasses the profound benefits of collective meditation. With meticulous attention to detail, we have implemented intuitive gesture guidance and immersive natural landscapes, allowing users to seamlessly transition into a state of thoughtless awareness, deeper meditation, and complete harmony with themselves and the universe.

In our virtual space, users have the freedom to meditate alone amidst serene natural settings or participate in collective sessions that mimic the atmosphere of a yoga class. Through intuitive gesture guidance, users are effortlessly led through meditation techniques, promoting a quicker shift into a state of thoughtless awareness, and facilitating a deeper meditation experience. Additionally, the immersive natural environments create a serene backdrop, fostering a sense of tranquillity and connection with nature.

Furthermore, our platform ensures completeness by addressing all aspects outlined in the project requirements. From intuitive gesture guidance to immersive natural environments, we have meticulously implemented every feature to provide users with a holistic meditation experience. With intuitive controls, users can seamlessly navigate the virtual space, while immersive visuals and audio enhance the overall meditation experience.

In conclusion, our virtual meditation environment encapsulates the essence of collective meditation, offering users a profound journey towards inner harmony and self-discovery. Through meticulous implementation and attention to detail, we have created a platform that fosters deeper connections with oneself, others, and the universe. With all features implemented in-depth, users can cultivate balance, integration, and a sense of inner peace, making our platform a transformative tool for personal growth and well-being.