Platform Name: VRCOPE
Project Documentation: VR Mental Health Experience
Team Members:

- Aditya Rachakonda
- Revanth Reddy Kancherla
- Deekshitha Karvan
- Keshav Rao
- Geethika Choudhary yadlapalli

---

1. Project Overview

- **Project Title:** Accessible VR experience
- **Objective:** Design a VR experience for improving mental health using VR technology.
  Technologies Used: Unity 3D, C#, VR hardware (Oculus)

---

2. Problem Statement

- **Mental Health Accessibility Issues:** Many individuals lack access to affordable mental health tools and therapy.
- **Solution:** Exposure to natural environments like snow-covered mountains, rainforests, and beaches has been shown to improve mental well-being by reducing stress, enhancing mood, and promoting relaxation. Scientific research confirms that viewing or immersing oneself in such environments even virtually can positively affect psychological states. A systematic review found that exposure to natural settings significantly decreases negative emotions and boosts relaxation, with specific environments like forests and coastal areas having the most pronounced effects (PMC). To increase accessibility, virtual reality (VR) technology has been successfully used to simulate such natural scenes. A study published in *JMIR Mental Health* demonstrated that participants who engaged in immersive VR environments resembling forests, beaches, and snowy landscapes experienced significant reductions in stress and anxiety even without a therapist present (JMIR). Furthermore, VR-based mindfulness exercises set in calming nature scenes were found to be more effective than audio-only formats (arXiv). These findings support the integration of VR

with simulated natural environments like snow mountains, rainforests, and beaches as a powerful, affordable, and scalable solution for improving mental health in individuals with limited access to traditional therapy.

---

3. Goals and Features

- **Main Goal:** To offer an immersive experience for mental health support.
- **Key Features:**
    - VR environments (beaches, forests, calming landscapes)
    - Interactive soundscapes (birds, ocean waves, ambient music)
    - Breathing exercises

---

4. Technology Stack

Development Environment for VR-Based Nature Therapy

Environment Creation

- **Game Engine: Unity 3D**
- **Development Language: C#**
- **Environment Design: Custom-built scenes featuring snow mountains, rainforests, and beaches**

VR Integration

- **XR Toolkit: OpenXR (for cross-platform VR compatibility)**
  Deployment Mode: Converts Unity-built environments into VR experiences

Audio System

- **Primary Tool: FMOD (for dynamic and spatial audio)**
  Use Case: Ambient nature sounds – flowing water, birdsong, ocean waves

Hardware Requirements

- **VR Headsets:**
    - Oculus Quest (for high-fidelity experiences)

Supported Platforms

- Primary: PC (Windows/Linux – compatible with most VR headsets)

---

5. Code Architecture

The breathing exercise script is designed to integrate breathing exercises into a Unity environment by animating a visual breathing cue.

Code Breakdown:

- public RectTransform innerCircle;
  Reference to the UI circle that scales up and down during breathing.

- public TMP_Text breathingText;
  Displays the current breathing phase as "Inhale" or "Exhale".

- public float inhaleDuration, exhaleDuration;
  Control how long each breathing phase lasts.

- public float minScale, maxScale;
  Define how small (exhale) and large (inhale) the circle should get.

- Start()
  Initializes total cycle time (inhale + exhale) and resets the timer.

- Update()
  Continuously tracks time using Time.deltaTime.
  Uses modulo (%) to loop the breathing cycle.
  Determines current phase:
    - Inhale:Increases scale using Mathf.Lerp(minScale, maxScale, p) Sets breathingText.text = "Inhale"
    - Exhale:Decreases scale using Mathf.Lerp(maxScale, minScale, p), Sets breathingText.text = "Exhale"

Usage:

- Attach the script to any GameObject.
- Drag and drop UI elements (RectTransform and TMP_Text) in the Inspector.
- Customize durations and scale values as desired.

2. The second script is designed to simplify **scene transitions** in a Unity project, allowing scenes to be loaded either by name or build index.

Code Purpose:

- Manages scene transitions within a Unity game or application.
- Useful for menus, level loading, and navigation between UI screens.

Code Breakdown:

- using UnityEngine;
  Core Unity engine features.
- using UnityEngine.SceneManagement;
  Required for scene loading functionality.
- public class SceneLoader : MonoBehaviour
  A MonoBehaviour script that can be attached to any GameObject (e.g., a UI button handler).

Methods:

- public void LoadLevel(string sceneName)
  - Loads a new scene by its name (as set in the Build Settings).
  - Can be triggered by UI buttons that pass a scene name.

- public void LoadLevel(int sceneIndex)
  - Loads a scene by its index in the Build Settings list.
  - Useful when scene order is fixed and known.

Usage:

- Attach the script to a GameObject (like a UI manager or button handler).
- Use Unity's Event System to call LoadLevel() methods from buttons.
- Make sure scenes are added to the **Build Settings** (File > Build Settings).

3. The third script is used to **smooth snow mountain meshes** in Unity by averaging vertex positions near a specified center point. It applies localized smoothing based on proximity and mesh topology.

Purpose:

- To smooth irregular or jagged regions of a mesh (e.g., snowy mountain tops).
- Particularly useful in terrain editing or runtime deformation effects.

Code Breakdown:

- public Vector3 worldCenter
  The world-space center around which smoothing will be applied.

- public float radius
  Only vertices within this radius (from worldCenter) will be smoothed.

- public int smoothingIterations
  Number of times the smoothing operation is applied for finer results.

- Mesh mesh and Vector3[] verts
  Internal storage of the mesh and its vertices.

- List<int>[] neighbors
  Stores the indices of adjacent vertices for each vertex, used for averaging.

Usage:

- Attach this script to any GameObject with a MeshFilter (e.g., a snow mountain model
- Set the worldCenter to the region you want smoothed (e.g., mountain peak).
- Adjust radius and smoothingIterations in the Inspector for desired results.

---

6. Testing & Optimization

- **Performance Testing:**
  Hardware Testing
- **Device Used: Oculus (Meta Quest)**
- **Purpose: Test real-world performance including:**
  - Frame rate consistency
  - Latency and motion tracking accuracy
  - Visual fidelity of natural environments (snow mountains, rainforest, beach)
  - Audio-visual synchronization under real-time usage

Prototype Testing

- **Tool: XR Device Simulator (Unity Editor)**
- **Purpose:**
  - Simulate VR interactions without physical hardware
  - Test controller input mapping, gaze tracking, teleportation mechanics

---

7. Deployment

Deployment Format:**Executable File (.exe)** for Windows-based systems

Target Platforms:
PC-based VR: Oculus Quest (via SteamVR or OpenXR)
Distribution Options:
Direct installation (.exe) for local or institutional use

---

8. Future Enhancements

- Procedural Generation of Personalized Environments Based on User Customization and Interactive Engagement

- Enhance user interaction to enable more real-time responsiveness, creating a highly immersive and dynamic experience.