The following document is a step-by-step guide on replicating our system set-up.

Hardware requirements:

- Maker Pi Pico from Cytron technologies
- GSR sensor v1.2

Software requirements:

- CircuitPython 9.2.7 running on the Pico microcontroller
- Python running in Visual Studio Code
- Unity Game engine running the Editor version 2019.4.30f1
- Main.py code
- File_maker.py code
- The VR environment (including RainScript.cs code and the rain Prefab)

To set up the microcontroller, use Thonny and make sure you are utilising the correct serial port to establish communication with the microcontroller. Using Thonny, save the main.py on the microcontroller while ensuring that it is the only code file named main.py inside the Pico board. The name main.py ensures that the code can run on its own as soon as the Pico board is powered on. Once the file is saved, safely unplug the Pico board and close Thonny. Re-plug the microcontroller (make sure it is the same USB port so that the serial port number remains the same) and whenever you are ready to start, just press the recording button (GP20) to start capturing and sending GSR data. Please ensure that Thonny is not running for the rest of the process, to not disrupt the serial communication.

In Visual Studio Code, open the File_maker.py script. Firstly, change the configuration SERIAL_PORT to the same one that Thonny is using to establish a serial communication. Make sure to change the DATA_ASSET_DIR directory to the Asset folder of the Unity environment on your pc, to allow the CS code to read the files. The code is written in a way that if a ‘GSRData’ folder is not present in the Assets folder, it should create its own one before saving any data. After these steps, simply running the File_maker.py, the script should create a baseline (after about 30 seconds of no outputs being printed while the baseline is being computed), after which the normalized_change should be printed out into Visual Studio Code and the output (0 or 1) should be getting saved into your pre-selected directory in the format data[number].txt, starting with file name data0.txt.

Finally, for the Virtual Environment in Unity, you have to open the project using the correct version of the Unity game engine (2019.4.30f1). Once the project file is open, you need to open the right scene to be able to immediately run the program. The correct scene is named ‘GSR feedback’ and within the Unity UI it can be found in the Project section. Within the Project section of the UI, go into the Assets folder and navigate to Scenes folder in which you can find the ‘GSR feedback’ scene and open it. As the data files with the output are getting saved into the Unity directory, no changes should be needed for the Rainscript to be able to read and access the data[number].txt files. However, if problems acessing the directory arise, the RainScript.cs (attached to the RainPrefab gameObject) contains the line public string dataDirectory = “GSRData”; which should be able to access this file, if it is present in the Asset folder of this project.

Now everything should be set-up and you should be ready to run the program. To simply run the program after the set-up is finished:

1. Plug-in the microcontroller (into the usb where you have previously filled out the serial communication)
2. Press the GP20 button to start the recording of the GSR data

3. Press play in Visual Studio Code to start the computation of the baseline, and subsequent and recording and saving the processed output into the text files.
4. Start the Unity environment in order to see the weather change according to the feedback given.
5. Use W,A,S,D and mouse to move and look around. Move up the stairs of the platform and left-click on the pink meditation text (when instruction text is visible in middle of the screen) to start the guided meditation task.