Resource Preparation:

Github:

Most models have screenshots attached to them for clearer understanding of the process.

Prefabs:

- Button
- Human Figures

Scripts:

- All included in the Scripts Folder

Materials and Shaders:
- Elevator Material
- Elevator Floor Material

INIT:

Add Google Daydream API to your project
https://github.com/googlevr/gvr-unity-sdk/releases

A Microsoft Azure Key has already been set up in your project via script, if you need to request a new key - refer to:
https://azure.microsoft.com/en-us/try/cognitive-services/?api=speech-services

Plug in the Heart / Pulse monitor Assembly to a power source like a usb port / charger

Scene Setup:

As children of Main Scene:

Please add the following to the main scene hierarchy by searching the names in the asset search bar:

GvrControllerMain
GvrEventSystem
GvrEditorEmulator
GvrControllerPointer

Then:

Create an Empty Object > Rename it to Elevator

Create a new empty object as a child of Elevator > Rename it to “Elevator Structure”:

As children of Elevator Structure:

Create an Empty Object > Rename it to ‘Player’
Add ‘Main Camera’ to Player Object
Add GvrControllerPointer to Player Object

Create a Small Square Mesh of about 0.05 uniform scale units > call it ElevatorTeleportPos

Duplicate it and make it roughly .2 scale points > call it ElevatorFloor

Adjust ElevatorFloor to be a little lower than the ElevatorTeleportPos in Y direction.

Using ElevatorFloor as a rough guide for the base of the elevator, add 3 walls and a ceiling by using basic box meshes. Make their height roughly 2 scale units.

Using 3 rectangular shape blocks create a front to your elevator, and group them under a new Empty object > Rename it to: ElevatorFront

Add a new rectangular mesh object > Rename to: “ElevatorDoor”, it should be slightly larger than the inside the ElevatorFront frame.

Create a new Rectangle mesh object and call it to ButtonPanel, scale it down so its about 0.35 High and .2 wide.

Add 2 button assets as its children of ButtonPanel and adjust their scale and position to be located one above another sticking out of the ButtonPanel box facing into the elevator.

Rename the button on the top to ButtonUp

Rename the button on the bottom to ButtonDown

Add in 6 Prefab models of your choice from the HumanModels Asset Folder, arrange them **around** the ElevatorTeleportPos mesh, with one model standing in front of the button Panel.

These assets have completely modifiable poses that can be set by expanding the asset hierarchy inside unity and tweaking their positions.

Rename the Model in front of the Button Panel ‘Butler’

Inside the Butler Asset that you now have in the scene there are 2 parts - add to the 'highpoly' a capsule collider on a Z Axis and fit the collider box around the mesh

(this will allow the PhysicalRay to activate the object)

As children of Main Scene:

Lvl1:

Create a new Empty Object > Rename it to "LvL1"

Duplicate ElevatorFloor and stretch it out to 0.4 scale units to become the walkway by moving it out in front of the elevator. Rename it to LvL1Floor.

Duplicate ElevatorFront, Rename it to > LvL1Front add it as a child of LvL1, shift it so it acts as a front to the Elevator, enlarging it just a little to make it look like a building frame, and will help the visual representation of an elevator foyer.

Duplicate ElevatorDoor from Elevator and add it as a child to ElevatorFront. Rename to > L1FrontDoor

Duplicate ButtonPanel from the Elevator object, and move it to be a child of LvL1.
Rename to > L1ButtonPanel
Add it to one of the sides of the LvL1Front object, remove one of the buttons and centre the remaining button. Rename Button to L1Button This acts to call the elevator when pressed.

Lvl2:

Duplicate LvL1 object except for the Elevator (just delete it after duplication) > rename to LvL2.

Rename L1ButtonPanel to L2ButtonPanel
Rename L1Button to L2 Button
Rename L1FrontDoor to L2FrontDoor

Adjust the Y position of the object to be 3 scale points to be directly above LvL1.

Controllers and UI:

Add Empty object > Call it ComplexityControls.
Add Empty object > Call it HeartRateMonitor.
Add Empty object > Call it SpeechRecon

Add Canvas element with Text Objects named:
ComplexityDisplay

HeartDisplay
SoundDisplay

This concludes the Scene asset Set up

Scripts Setup:

Within LvL1:

LvL1Floor > Script > TeleportationSimple > Add to script:
Player: Player
Butler Ready: Your Butler Asset contains 2 parts - add the 'highpoly' to this script section
Elevator Door: ElevatorDoor
Lvl Door: L1FrontDoor

L1FrontDoor > Script > DoorScript > Add to script > Door Body: L1FrontDoor
Adjust Open Distance and Open Speed to match your preference in the scene
Keep direction X

L1Button> Script > Pushable > onReleased > L1FrontDoor > *DoorScript.openDoor*
L1Button> Script > Pushable > onReleased > ElevatorDoor> *DoorScript.openDoor*

You can change the colour of the highlighted button in the colorable script that comes with the button

Within Elevator:

Elevator Parent Object > Script > Elevator Auto Door > Add to script:
Elevator Door: ElevatorDoor
Lvl 2 Door: DoorTop
Lvl 1 Door: L1FrontDoor

ElevatorTeleportPos > Script > Teleportation Elevator > Add to script>
Player: Player
Elevator Door: ElevatorDoor
Lvl 1 Door: L1FrontDoor
Lvl 2 Door: L2FrontDoor

Butler Ready: Your Butler Asset contains 2 parts - add the 'highpoly' to this script section
ElevatorTeleportPos > Script > colorable

Feel free to change the colours to taste

ElevatorFloor > Script > ElevatorScript > Add to script: Elevator
Adjust open Distance and Speed to fit the scene

ElevatorDoor > Script > DoorScript > ElevatorDoor
Adjust open Distance and Speed to fit the scene

_Within Player:
Main Camera > Clipping Planes > 0.01

_Within Butler:
Your Butler Asset contains 2 parts - add script to 'highpoly' > Butler Script

+ Colorable Script > Change Colours to taste
+ On Released add SpeechRecognitionREST > SpeechRecognitionREST.Trigger

(this will act as the main speech recognition point, where the result is then passed to the elevator and depending on whether the result is "UP" or "DOWN" the elevator will follow to the next floor)

_Within ButtonPanel:

ButtonUp > Script > Pushable >onReleased > ElevatorFloor > *ElevatorScript.goingUp*
ButtonDown > Script > Pushable >onReleased > ElevatorFloor > *ElevatorScript.goingDown*

Within LvL2:

LvL2Floor > Script > TeleportationSimple > Add to script:
Player: Player
Butler Ready: Your Butler Asset contains 2 parts - add the 'highpoly' to this script section
Elevator Door: ElevatorDoor
Lvl Door: L2FrontDoor

L2FrontDoor > Script > DoorScript > Add to script: L2FrontDoor
Adjust open Distance and Speed to fit the scene

L2Button> Script > Pushable > onReleased > L2FrontDoor > *DoorScript.OpenDoor*
L2Button> Script > Pushable > onReleased > ElevatorDoor> *DoorScript.OpenDoor*

Within Main Scene:

ComplexityControls >Script>Complexity Controls>*Model Count*>Complexity Controls(Text)
ComplexityControls >Script>Complexity Controls > *Human_1* through to Human_6 > Populate with Human Models added to the Elevator by as assigned by number

SpeechRecon > Script > SpeechRecognition REST > SoundDisplay (text)

ComplexityControls > Script > Swipe Action

This concludes the scripting setup

Program workflow:

1. User starts on the far end of the Level 1 hallway opposite to the elevator

2. The controller UI contains a basic description of how to navigate the scene

3. The call elevator button lights up when hovered and changes colour when pressed

4. When call elevator button is pressed the elevator doors open and the user has the ability to teleport into the middle of the elevator. Onto a colour changing panel.

5. The user is able to increase the number of models in the elevator by using swipe gestures on the controller

6. Once inside, depending on how many human models are in the elevator, the user is able to either activate the buttons on the elevator panel, or if the model number is greater than 3 - then the user can 'activate' one of the models blocking the view of the elevator button panel by hover selecting and clicking on them in the same method as a button.

7. Upon being activated, the model 'ideally' will turn to face the user and await a voice command - the user will then need to say something that involves the word "Up" or "Down", this will be screened by a script which will filter the information received from the speech to text microsoft engine.

Currently the Butler Model only becomes highlighted and changes colour when pressed

8. When a button or a command has been received - either Up or Down, the elevator will close its doors, and after a short time teleport itself to the designated level - either level_1 or level_2.

9. The elevator will rise to the top, with a speed set in a script, the doors will open and the player can freely teleport out.

10. Upon teleporting out from the elevator to the floor of the selected level, the elevator doors will close completing the loop. The player can then call the elevator again, and repeat the process but by going to the lower floor.

Explanation of the app mechanics:

Teleportation:

The floor uses a mesh collider that reacts to the physicalRayCast that extends from the controller - wherever the ray hits the mesh is sent to the player position, if the player presses on the touchpad while sending a rayCast to the floor or the elevator floor where a teleport script has been implemented, the player object receives that collision point information and uses it as its own, on the x and z coordinates (y coordinate will always be the same as to not shift the player height during the teleport)

Elevator Teleport Area:

The elevator floor has a small collision mesh right above the floor mesh, that collision mesh is there to allow the player to be teleported directly into the middle of the elevator and create a semi-movable space where the player can navigate by teleporting, but also stops the player camera from intersecting into the human models positioned around the collision mesh, creating a tight fit of the elevator experience.

The Teleport area needs to be able to be highlightable to let the player know that this is an interactive section of the app. So we can apply the colorable script to it as well.

Button Controls:

The Buttons are already premade assets that come with scripts colorable and pushable, these scripts make the button change colour when hovered over by the raycastController and can be pressed in when using the controller touch button

The outside button calls the function of elevator doors to open.

The inside buttons make calls to the function elevator up or elevator down, which make calls to the elevator door to close and then for the elevator object to change position after a designated amount of time. Upon that time being elapsed, the elevator is teleported and the doors on the second floor open, by having the door.open function called.

Scene Complexity and interaction:

The human models are static and simply get turned off in the renderer while the required script checks if they need to be turned on. It's a simple script which simply checks whether the number of models needs to be more or less - if level of complexity if 2 then there will be 2 models that are rendered in.

The important distinction is the 'butler' model which stands in the way of the button panel, forcing the player to instead interact with the model.

The script attached to the 'butler' is similar to that of a push button - he is colorable to let the player know that they can interact with the model and upon being "pressed" the Butler turns to face the player camera object - making it look like the player has gotten the Butler's attention.

Once the Butler is active and looking at the player, the player is given a cue overlay to tell the butler which floor they are going to - the commands should be filtered to find the words - "Up", "Down", or something involving "Two", or "One" or "Ground" these should be enough to filter a basic idea of where the player is trying to get to.

Triggering the same functions as the elevator panel buttons - up or down.

Speech to Text Recognition:

The speech recognition engine used in this app is Microsoft Azure, the engine needs an authentication key and can record and translate audio to text given that the application has been granted permission from the phone to use its microphone.

HeartMonitor:

A custom unit has been built to allow bluetooth communication between a pulse sensor and the application, the phone will need to be paired to the device, which will then begin to display a normalised pulse value to the user HUD.

Optional and add on ideas include - Using the heightened pulse to measure anxiety and display messages to the HUD about slowing breathing or doing small calming exercises to relieve tension until the pulse rate has dropped to normal.

Additional Concepts:

Add basic UI information to daydream controller

- Use of Teleportation
- Speech Recognition
- Complexity Level Number
- Use of Swipe Gestures
- Contextual Information for buttons and interaction

Recommended Timeline of Development:

Week 1:

Complete Scene and Script Setup

Week 2:

Testing and troubleshooting of Speech Recognition

Week 3:

Testing and troubleshooting of HeartMonitor module

Week 4:

Overall scene testing and cohesion

TroubleShooting and Testing without a VR Headset:

Speech Recognition:

MicTest Script:

- *Using a basic MicTest Script to check that audio is being received*

Speech Recon Authentication Test:

- *using the authentication script to check for key authentication and server connection*

Speech Recon Keyword Test:

- *using a file on disk and script to check for speech recognition*

Things to figure out: How to filter the response string for triggers

JSON RESPONSE STRING:

```json
{
  "AudioFileResults": [
    {
      "AudioFileName": "Channel.0.wav",
      "AudioFileUrl": null,
      "SegmentResults": [
        {
          "RecognitionStatus": "Success",
          "ChannelNumber": null,
          "Offset": 400000,
          "Duration": 13300000,
          "NBest": [
            {
              "Confidence": 0.976174,
              "Lexical": "what's the weather like",
              "ITN": "what's the weather like",
              "MaskedITN": "what's the weather like",
              "Display": "What's the weather like?",
              "Words": null,
              "Sentiment": {
                "Negative": 0.206194,
                "Neutral": 0.793785,
                "Positive": 0.0
              }
            }
          ]
        }
      ]
    }
  ]
}
```

This response string will need to be filtered for words like - “Up” “Down” “One” “Two”

And make the necessary functions - work with the speech recognition

DayDream:

The app can be tested without the daydream controller:

Switch platform to Windows, then using LeftAlt+Mouse activates game camera in daydream mode

LeftShift+Mouse and LeftMouseButton as Basic Click on the touchpad of the daydream controller

Complexity testing can be done by directly inputting a number into the complexity controls script - variable > Models