Technical Design Document

Objects Pre-existing in Project Hierarchy on the Unity Editor:

1. LogInCanvas: Canvas object containing a Panel object. The important components the Panel contains:

1. **Username_InputField (TMP):** The user enters their username. (Presently not used anywhere).
2. **Role_Dropdown**: Choice of Participant or Researcher.
3. **Password_InputField (TMP)**: If dropdown choice is Researcher, this InpuField becomes visible, and the valid password is required to log in.
4. **RelayID_InputField (TMP)**: If dropdown choice is Participants, this InputField becomes visible, and the person who logs in as researcher needs to provide the Relay ID that is generated and presented on the loading screen.
5. **LogIn_Button**: Calls the relevant method on the attached script to log the user in.

Scripts Attached:

1. **LogInManager**: Performs form-checking, handles either local host-client, or Relay host-client connection based on the relevant choice on the UI. If local host checkbox not selected, if logging in as Researcher (host), the script sets up a Relay server and presents a Relay ID key. This key needs to be entered by the Participant user (client), to connect to the server set up by the host. In both local host and Relay cases, the NetworkManager script is called to start the host (Researcher) and client (Participant), depending on the role selected using the dropdown menu. This also instantiates a Player prefab object passed to the NetworkManager object through the Unity editor, for each user.

2. EventSystem: Unity object relevant to UI interaction. (Mentioning it for completeness.)

3. NetworkManager: Object with relevant components attached that handle network related functionalities.

Objects Attached:

1. **UTP**: Object with the UnityTransport script attached, having selected the Unity Transport protocol type, that facilitates local (or direct IP if enabled) host-client communication.
2. **UnityRelayUTP**: Object with the UnityTransport script attached, having selected the Relay Unity Transport protocol type, that facilitates network communication using the Unity Relay service.

Scripts Attached:

1. **NetworkManager**: *Unity Netcode* script that handles networking functionalities.
2. **UNetTransport**: Unity Netcode related script. Its functionalities are not used, in favor of using either the functionalities enabled by either the UTP or UnityRelayUTP objects.

4. PanoramaSphere: Sphere primitive object. Has material component holding the virtual environment texture. The material uses a Shader which renders textures on the inside and also inverts pixels on the X axis (because rendering he texture on the inside also mirrors it).

Objects attached:

1. **ResearcherCamera** object, which is used by the Player script, and is the non-VR camera that the Host (Researcher) views.

5. XR Interaction Manager: Object related to the XR Interaction Toolkit which handles VR-related functionalities. (Mentioning it for completeness.)

6. XR Rig: Related to the XR Interaction Toolkit which handles VR-related functionalities.

Objects attached:

1. **Camera Offset**: Controls the starting Transform configuration of the VR camera.

    ### Objects attached:

    1. **Left/Right Hand Controllers:** Not used by our system.
    2. **ParticipantCamera**: VR camera.

Objects Instantiated at Runtime:

1. PlayerNTW: Prefab object corresponding to each user logged in.

Scripts attached:

1. **Player:** Holds information including username, and whether the user is logged in as Researcher or Participant. Depending on user type, the appropriate UI is instantiated. Loads environment files from JSON and assigns them to the appropriate Gameobjects so that they are presented to the user.
   Handles networking-related functionality to facilitate message exchange between users and their display on the relevant UI, as well as Researcher camera updating based on Participant camera rotation.
2. **NetworkObject:** Netcode-related script that allows the Player script to have networking functionalities.

Objects attached:

1. **Researcher_Canvas_NTW:** Prefab Gameobject instantiated if the user is logged in as Host (Researcher). Contains UI elements relevant to displaying available messages that the Researcher can send, loaded via JSON at startup. Also contains a chat log that displays current session chat history.

   ### Scripts attached:

   1. **SetupResearcherUI_NTW:** Loads messages and their pre-defined responses from JSON, and instantiates the relevant UI component prefabs to present messages at the appropriate positions and with the relevant functionality, i.e. selecting a message, optionally editing it, optionally adding or removing responses, and then pressing the relevant button to send it to the Participant. Also enables sending a request to the Participant UI to present a questionnaire.
   2. **ChatLogBehaviour:** Also attached to the **Participant_Canvas_NTW** Gameobject. Handles correct placement of messages at the appropriate vertical position, as well as horizontal position based on which user it has been sent by. If sent by the other user, the message is displayed on the left, while if it was sent by the same user, it is displayed on the right. On the Participant UI, if the message was sent by the Researcher and it contains available responses, those are presented at the appropriate location as buttons that can be interacted with to send a message back to the Researcher.

2. Participant_UI_Smartphone_NTW: Simulated-smartphone prefab object instantiated if the user is logged in as Client (Researcher), and is then attached to the **ParticipantCamera** Gameobject which is attached to the **XR Rig** Gameobject.

Objects attached:

1. **Participant_Canvas_NTW - World Space:** UI Canvas Gameobject containing relevant UI elements to present the chat interface on which Researcher messages and their responses are displayed, as well as the response the Participant selects.

Scripts attached:

1. **ChatLogBehaviour:** Same as described under the **Researcher_Canvas_NTW** entry.
2. **ChatMessagesGamepadInteraction:** Handles Participant interaction with the UI using a gamepad controller.
3. **ParticipantUIQPresentation:** Loads questionnaire from “questionnaire.json” file and presents it on the participant UI upon request from the Researcher instance. Once responses have been submitted, sends a message through the network to notify the Researcher UI of this.
4. **SaveCollectedDataLC:** Saves questionnaire responses to a file upon request from the ParticipantUIQPresentation script.