**Implementation Documentation**

* The game is implemented using unity.
* The scenes are as follows in order
  + Intro - Introductory scene
  + Login - Login functionality. After login the user details are stored in the game object “**UserData**”
  + gameselector - Lists out the games added to the user profile of the earlier authenticated user as buttons. The “**GameMetaData**” object in the scene has a script named “**GameMetaScript**” which stores all Game configuration data as follows
    - Gname - Game name
    - Gid - Game ID
    - Envid - Environment ID
    - RuleID - Game rule ID
    - Minp - Minimum number of players
    - Maxp - Maximum number of players
    - OwnerId - Owner id which is actually the game creator’s id. This ID will help us later to know to whom the game data to be sent.
  + Lobbyscene - Lobby scene for the multiplayer. The important components of the scene are as follows
    - LobbyManager - the network manager for the mutiplayer game. This users the “**UNet”** framework to connect and manage the multiplayer platform
    - LobbyManager Panel - The panel has to sections
      * Matchmaker - In this part any user starting a mutiplayer game or joining a game wouldn’t need to know of the IP address of the server. There is a centralized server in Unity’s domain which will take care of the Network IP match making. I have used the Unity’s free “**muliplayer**” service to implement this.
      * Manual Connection- This part is application for a multiplayer game using LAN connection. One of the players within the same network will either “**Play as a host”** or start a “**Direct Server**”. Player within the same LAN will be able to connect by filling the IP address and click on “**Join**”.
* The next two scenes are the main game scene.
  + The game scene are named as per the configuration. Example - if scene name is “**Scene11**” then **EnvID** =1 and **RuleID** =1.
  + As of now there are two games implemented. They are as follows
    - RuleID=1 is **Mazerunner. Winning criteria -** Starting from the outside of the maze whoever collects the object at the center of the maze first wins
    - RuleID=2 is **Crowd Evacuation game**. **Winning criteria -** Like a race, starting from the centre of the maze whoever exits the maze first wins.
* Player controls:
  + Up - forward
  + Down - Backward
  + Left - move left
  + Right - Move right
  + Hold shift - Run
* **Scripts-**
  + **ButtonTasks -** 
    - **Login() -** The login validation and storage of user details for the game
    - **storeGameDetails() -** storing all game datails
    - **goBack() -** action for backbutton in “lobbyscene” scene
  + **CollisionDetector**
    - **onTriggerEnter -** In case of maze runner game this method is used to detect the collection of first person to collect the prize
  + **FillDetailsScript**
    - Fills all game meta data from the server for an user.
  + **GameMetaScript** 
    - Stores all game configuration data
  + **GameTracker**
    - **OnTriggerEnter()-**
      * For crowd evacuation game - each exit is enclosed with 3 invisible colliders.
      * Each collider game object has this script assigned to it to detect the collision which will help us find who reached the exit
  + **mainCameraControls**
    - Third person shooter camera control
  + **MyBehaviorTree**
    - Behavior tree for automated agents for mazerunner game
  + **MyBehaviorTree1**
    - Behavior tree for automated agents for crowd evac game
  + **PlayerController1**
    - User Player controller
  + **PlayerSyncRotation**
    - Synchronizing the rotation component of an user controlled agent across the network
  + **SpawnPosGenerator** 
    - Creates multiple start position based on the number of agents joining the game
  + **startPositionScript**
    - This script assigns a randomly generated transform values to each of the previously spawned start positions
    - There are some invisible **“DummyPlanes”** in the scene to point out the random positions to our required start locations according to the game.
  + **WinningCriteria**
    - Decides when game is over