26 November 2017

Vision Document - SE 3313A Project

Trent Chappus - 250845587
Ashley Ottogalli - 250845708
Connor McCauley - 250865901
Matthew Price - 250752191
Rachel Vanderloop - 250798250

For this project, our vision is to create an online multi-player competitive game for Android devices. This game, entitled *ScavENGer Hunt*, will be a “scavenger hunt” game that you can play in your own home. A match between four - six people (random or invited) will consist of 10 rounds. The round will start with the name of a common household item (such as “banana”, “pen”, “chair”, etc.) being displayed across the players’ devices. The first player to take a photo of the object wins the round. The round will go on until a player takes a matching photo, or until 60 seconds has elapsed - whichever comes first. The player that wins the most out of the 10 rounds wins the match.

The client app will be implemented using a standard Android development environment (Android Studio with standard Android UI). To initiate a match, the client will make a request to the server with certain parameters (such as whether the matchmaking should be random or not, or if the player provided an invite code to join an existing match). The server will decide to make a new match or query the current ongoing matches and provide the details of the match the player will join.

While a round is taking place, the player can press a button to open the camera to take a photo. The photo will be uploaded to a cloud storage location, and then the client will make a request to the IBM Watson Visual Recognition API with the URL to the photo. Upon receipt of the response from Watson, the client will check if the photo is of the current word. If so, it will send a request to the server indicating that the player has taken a matched photo.

The server will have a main thread that reacts to requests from clients. The main thread will handle matchmaking and relaying requests to corresponding threads. It will create a new thread for every match. The match threads will manage the list of players in the match, the time left in a round, and the current word for the round. It will create a new thread for every player as well; the player threads will keep the player’s current score and handle requests from its player. These threads will need to be synced with one another: the main thread will need to sync with the match threads to keep track of the ongoing matches and the match threads will need to sync with the player threads to handle the event that a player sends a request indicating that the player has won the round. In the event of a win, the player thread will sync with the match thread, which will then sync with the rest of the player threads indicating that a player has won the round.

When a player leaves a match, their player thread will be terminated. This will communicate to the match thread that the player has left, and can then sync this with the rest of the players. When a match reaches its end (after 10 rounds), it’s thread can terminate its player threads and then itself safely.