Project Plan & Overview

General Architecture:

- Front end app (no-login, given unique player id).
- Idea to have one main start button (that begins the pairing) .
- Matched with other player OR bot.
- Chat for set amount of time (e.g, 5 minutes).
- Prompted to guess at the end of time limit.
- Can also guess before time limit.
- Score generated based off number of messages sent and/or time taken
- The score is not persistent / not saved to db, for the time being, but can be changed if needed.
- Score is displayed in a post-chat screen to be captured by the user, if desired.
- After game, provide optional dialog box that asks for reasons behind the users choice

In terms of the network/backend architecture, the app is essentially just a housing for displaying the messages that come in from the bot / the messaging service.
In terms of the interaction, the plan was to wait for a certain amount of time to see if a human could be paired and then make a random choice between the bot/human. If no human can be paired, just talk to the bot.
At the minute, the plan is to use IBM watson, which allows us to essentially just make api requests using a certain key and url that relates to a certain “assistant”. This assistant will be trained on a certain topic (i.e. food, cars, etc), as it is not really feasible to implement a bot that can have an open ended conversation. In the future, it may be better to have it randomly select a bot/topic from a database which will store the url & key of the particular assistant.
For the minute, we will assume that if paired with a human, each human will respond promptly (has an interest in playing the game / will not leave an excessive amount of time between responses).

Workload Split:

More than likely, it will probably be one member of the development team working on the front end (app) and two members working on the backend (on the bot side / the human chat side). However, this is subject to review and will change throughout the project as necessary in order to hopefully deliver what is needed/wanted from the client.

Backend / Matchmaking:

This is just to discuss the backend implementation of the project as this would seem to be the real meat of the development process.

From the discussions we've had, the backend would seem to be split into two sides (human to bot & human to human). They would also seem to be relatively separate in terms of their implementation.

Human to bot:

In terms of this side of the backend, the user will communicate directly with IBM's servers / the bot we choose through api calls and will converse independently of the matchmaking / chat service. The advantage of this is that the implementation of the bot is pushed off onto the bot providers side, meaning we can focus on how the users see this interaction. The timekeeping and eventual guessing of whether or not the agent is a bot or human will take place locally, within the app, and if required the results can be saved to a database (this may be done using firebase or could be done locally) for score keeping (depending on how the rest of development goes).

Human to Human:

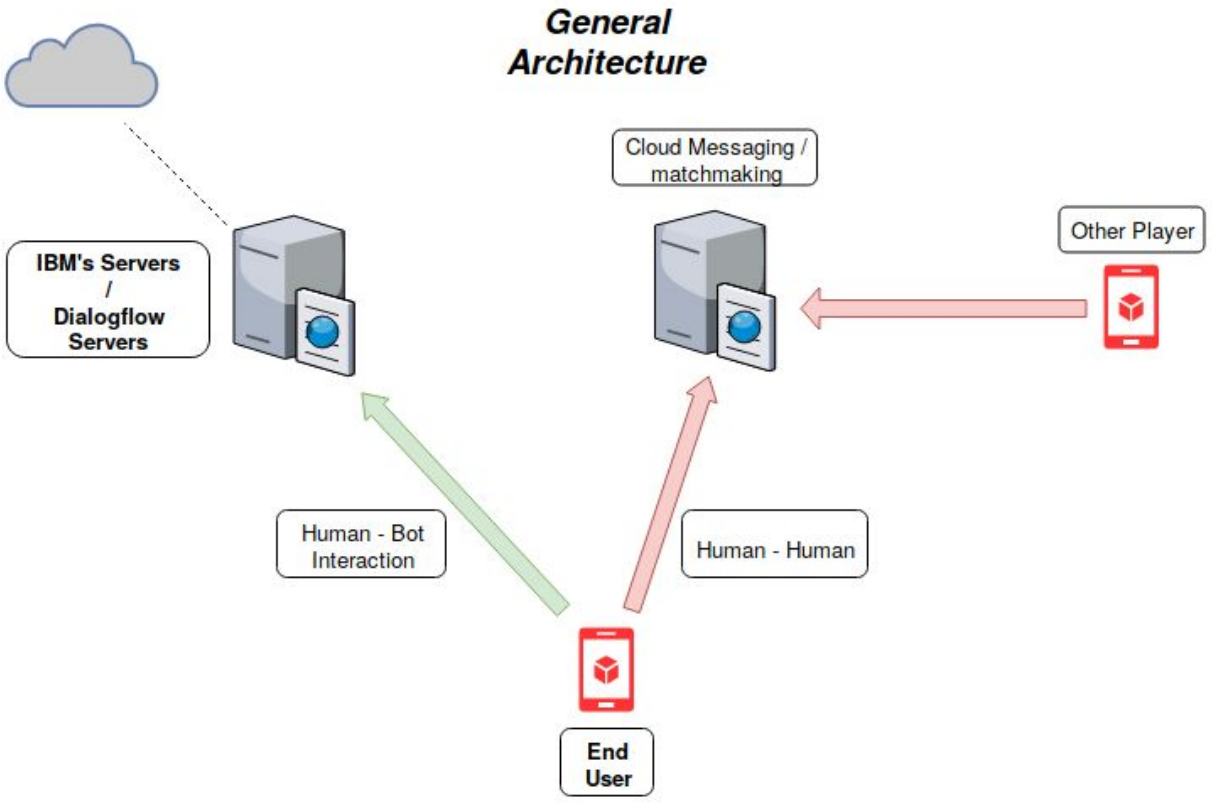
When two human players are matched between one another, the messaging will ideally be handled in a peer-to-peer system, allowing the server to handle matchmaking for other players, and storing any information that is collected in games, which will hopefully result in a more stable and consistent operation for the server.

Matchmaking:

The current idea is to implement matchmaking using the Google Play Game Services, as it will work well with Android Studio and also removes the need for us to build an entirely new matchmaking system from the ground up for our project. Only a small subsection of Google Play Game Services' tools will be required to implement the matchmaking we need, as there will obviously be no need to invite friends to a game session, for example.

**Any feedback / critiques from you guys is definitely appreciated and we'll do our best to address any issues. Some of this information may change by the next meeting, as we're in the process of doing up some material for the lecturer and may think of some issues / ideas in the coming week(s). While the discussion around design is greatly appreciated, we're eager to get started on the development work, just to stay on top of other work from college, so we were thinking of getting fully started next week.*

Simplistic view of architecture:



https://firebase.google.com/docs/cloud-messaging/

https://www.ibm.com/watson/

https://firebase.google.com/products/

https://developers.google.com/games/services/