2019-12-06 Biweekly Report

Tasks Completed

1. Reconsider target platform to be laptop / PC based (for prototype), with macOS as the primary platform.
2. Devise initial Systems Architecture diagrams for client and server for the systems.
3. Created sketches for UI and emulated environment.
4. Create initial Unity models (see screenshot attached).
5. Research DevOps for the server.
6. Decide on voice chat solutions (client-to-server, not peer-to-peer).
7. Research on Watson Personality Insight.

Problems and Difficulties Encountered

- We encountered significant difficulities when trying to develop for Oculus Go and other types of embedded VR head gear due to lack of documentation and integration testing difficulities; we switch to laptop / PC based environments as there are more libraries, tooling and more documentation available.
- Real-time voice chat turned out to be much more complicated than initial anticipated:
    - Recording voice, sampling, encoding and compression, as well as playback, are not too difficult.
    - However, the networking and real-time part of voice chat between multiple meeting participants is much more complicated.
        - We needed to pick either peer-to-peer to client-to-server model.
        - We wanted to go with peer-to-peer model initially because of its low latency and not needing a relay/middleman server to basically transmit the same audio frames twice.
        - However, due to IPv4 address space being limited (in the public IP space), a lot of networks are actually segregated from the public IP space through Network Address Translation (NAT) and firewalls. Clients typically are behind NAT and other types of firewalls, restricting information regarding their internal IP address as well as which ports can be exposed and forwarded by NAT and firewalls, subject to vastly different constraints.
        - We then decide to go with client-to-server model primarily because of the networking constraint.
        - We checked existing voice chat such as Discord - from what information is publically available, we find that Discord also use a client-to-server model.

Tasks Planned for the Next Two Weeks

1. Unity emulated environment modelling.
2. Unity UI.
3. Voice-chat and meeting session protocol design and documentation.
4. Meeting session management server-side implementation.
5. Voice-chat server-side implementation.