The development process for this project was rather modular. Each team member was assigned a specific part of the app, such as chat, profile, docker, registration, log in. Once each member completed their part on their branch; taking into account the technologies used and code formatting previously agreed upon. Each Team member pushes the completed piece to their respective branches. At this point the designated GitHub person merged everything to create the final project. For the team member in charge of Docker, we created a skeleton of the app to make it easier on them. Otherwise each team member was in charge of their own microservice. We used the following Stacks: `Frontend Stacks:` React, Redux, Material UI. `Backend Stack:` Express, MongoDB, Docker, Redis, Websocket.

Some of the overall hardships were pretty standard in terms of group work. For example: fining a good time to meet, making sure everyone can play their strengths in the assignment, and overall cohesion of the code written (Front end and back end). Our group was able to complete this project almost entirely without meeting outside of class, due to conflicting schedules, though luckily this wasn't as big of a set back as other classes because of the microservice architecture. This is the first time the majority of us have ever used most of the tools required for this project, though this is a challenge regardless, its worth mentioning that the frameworks chosen allow for easy cohesion between parts of an application; eg. using material UI, allows for small edits to each page without affecting the unique styling of another service, all with keeping the same look and feel. In addition, because there is no central declaration for redux, we could all work independently. We felt that most of these tools allow for a fluid development process in groups, but their learning curve is very steep. Which was a challenge for most of us. Especially getting everything to work in docker. Because this can only really be done with the apps skeleton built, the team member in charge of configuring docker had an extra challenge. In the future we things docker should be done by the person responsible for their microservice.

This was an interesting project to say the least, and I think we can all say we learned some pretty cool tools in the process. Though they are very different from what we are use to; it was well worth it.