Ng Yan Bo (A0186871N)

Personal Contribution to group project:

Development Work:
React Frontend Development
Flask Middleware
Data Access Layer
Various functions for Data Transformation and Manipulation (Handlers)
Integrations between Modules

Infrastructure:
Dockerising - PostgreSQL, React, Python and Flask
Technical Architectural Design

Reflection:

Patently, transforming payloads into a compliant one for any form of integration makes me want to tear my hair out. The oxymoronic part is that I actually find solace in the cathartic experience that follows when my API returns a Status Code of 200.

On the last day of the project, with less hair than ever before, I feel barely adequate to be able to draw some connections between different data type across languages.

- Associative array/ Dictionary is now a JSON object in JS and python introduced tuples.

I discounted it by thinking I could just do a JSON.dump at the end but getting to that final stage took really time for my part.

Work from home in this Covid situation is a SI employees' nightmare. It requires me to work more than ever before with the shrinking budget and the same amount of work to be done. 1 developer doing 150% - 180% more output.
I am really jaded after 10+ hours of dev and collaborative work/day and the quality of code is not really great for this project.
I did more development than really coding ML stuff but I think that's the reality for all projects. 40% cleaning data, 40% integration work, 20% ML

Frontend development work reminded me of why there is a specialization for such a skill and further reminded me of why I stayed away in the first place. However, I understood the difficulties behind it and grew a new appreciation for my front-end engineering counterpart that resides halfway across the globe.
I realized the importance of API design and planning before I get to dev work proper as well. Thought this is a smaller project so should just do away with the formalities but doing so will create structure and drastically improve productivity of work.

Finally, really thankful that I had really talented teammates that picked up programming in the short span of a month and provided good quality code for integration!

Key Takeaways:

- On hindsight, dockerise and deploy the nginx along with the golden template first. SSH into the instance to develop instead of developing on individual computers. Dependency packages differ for every OS and had to end up scrambling for almost one week for the dependencies to work

- In the next module, set up CICD pipelines right away, deploy the relevant cloud infrastructure and configure the network security groups properly. Will get some more knowledge in between mods (if I make it).
  Attempted this semester on an entire new cloud platform Azure but abandoned it to do development work instead. Enjoyed using it so I will advocate Azure instead of AWS for my upcoming delivery project in the domain of ML-ops

- I was really intrigued by Xgboost. It does gradient boosting by combining an ensemble of models and in our case, we used different output estimators for our decision tree (k-nn, Linear Regression and Extra Trees) and combined it to boost the accuracy of the overall model with Xgboost. I will try and use it for an upcoming data project that has big unstructured datasets. I have heard it mentioned several times. I think a lot of my peers use this but have yet to fully understand gradient boosting.

Most Important Takeway:

Generally the creation and the discussion was the most essential part, having able to verbalize and hear my teammates thoughts on the subject was paramount to my growth. Incorporating various elements taken from our courses to come up with a solution and applying the concepts solidified our understanding and provides us with a strong foundation for supervised learning architecture design