**Team 3 summary**

**Ian Oliver:**

Going into the product with an exploratory proof of concept standpoint helped with shaping the overall technical design. This in turn helped with handling the technical challenges of using a preconfigured database. Naming using systems that are setup in a more advanced configurations than what our application was prepared to handle. Overcoming those challenges and learning about the different methods of communications.

A sweeping software update to several open-source candidates, making them unusable for our purposes also forced our team to find new solutions those packages left. Learning how to incorporate the new packages and utilize them in a functional way while still maintaining a minimal budget cost proved an interesting challenge as well. Though we learned later on that our backup choice in GUI software does not handle “live” updates as gracefully as we had hoped.

**Design Strengths:**

A major design strength is limiting the need for constant communications with a server. Using a Model-View system, data could be retrieved locally and stored in internal data structure for manipulation and use. Meaning the application only need to send and retrieve data packets for essentials and data updates.

Storing the data local to the viewer also allowed for greater control and freedom for displaying the information. Meaning graphs, gantt charts, pie charts, etc. would still be viewable even if the network was down.

Though intentionally designed but extremely underutilized, this approach also allows the application to fold into an Edge-to-Cloud computing model smoothly. No major changes to system or networking code would be necessary. Allowing for greater computing power in future use cases.

**Design Limitations:**

Current design limitations all fall into two categories. Communications to the database, and updating frames in the GUI. We discovered after having to fall back onto our plan C choices for development tools. That the native Java Swing components, while robust, are not very flexible. Which injected odd behaviors and side effects into the project which needed to minimized for functionality and overall user experience.

A similar experience was found in the networking and communications side of the project. As the more secure protocols for accessing and writing to the database proved unreliable. As such we were forced to use more brute force communications, which can potentially open up security risks.

**Future Improvements:**

Some future improvements for the project would be to update the GUI from Java Swing to JavaFx or C++ Qt. Giving greater control and limiting many of the issues found during the coding process. As well as, finding a more secure method of data retrieval in general. As either through inexperience, design limitations, or software limitations, there are security risks that can be mitigated but aren’t.

An architectural improvement that would align with future proofing both dataset size and Edge computing would be to have an external Database and an internal copy on top of the internal model. Either via something simple like SQLite or more complex. This would allow for the application to function in both a stand-alone capacity as well as in a distributed or edge computing capacity.

**Dave Leake**

It was interesting to determine a collaboration tool and environment. We found Asana and Github Served our purposes... up until we tried to develop. Azure VM solved that issue. Trying to backup, sync a Github DB would have been a pain.

The recap of the actual schedule, cost burn, and delivery schedule for all our work packages. (one page with Gantt \_ metrics) is included in Team 3 weekly report.

In conclusion, Azure was a strength, the Limitation of the Swing Java proved cumbersome and overall DB Design was a bit too complex for us to tackle in this 8-week project, had to really scope down the end product.

**Will Tchouente**

The scope of this project had to be changed for functionality as we would not have had a enough time or a working product had we stuck to the initial scope we intended. I came into this project having no prior experience working with SQL , and always writing my programs first and doing the documentation after it worked so this experience was interesting and I learned a lot throughout the process and a big takeaway for me was the way Ian worked with eclipse tools and made me realized just how much more I have to learn as I’ve always written the code without using the built in tools provided by eclipse.