Kevin Swan and Jithin S.
Capstone Final Writeup

Each of us had our own unique challenges and experiences for this project, so we have separated them.

Kevin:

For this project, I used discord.py which is an API wrapper for Discord to work with the Discord API. Using this, I was able to create a bot on our Discord server that stores and displays the server's message history. On the back-end, my bot was able to take key information from the server's messages including the message's ID, content, author, and time that the message was sent and store it on our MySQL database. This data was then parsed and able to be viewed on Discord itself when a user on the server inputs the *!history* command.

A challenge I had with this project was reading through the discord.py documentation and learning all the limitations of it. One limitation I faced was when I was trying to display the message history in an embed like we had done for other commands, and could not because it seems there is only a certain number of rows you are allowed to have in an embed. Using Python was also new for me as I have not used it much in the past and was not familiar with too much of the syntax.

Another battle scar I had was when initially implementing commands. I had done it in a way that caused the bot to run slower than it could have. I had been using an if-statement to see if the user's inputted command had shown up in the message content. This proved to be inefficient, and was fixed after using discord.py's *@client.command* feature which allows you to easily implement efficient commands. Still, overall, this project seemed more daunting in the beginning than it really ended up being.

In the future, I would like to add a command that clears the message history. This way, when the *!history* command is used it can return messages that are more relevant to the user's needs at the time.

Jithin:

Smite API had a bunch of issues, and I learned a lot about integrating APIs from the Capstone. For the SMITE API it was a bit different, I had to create sessions for the API that only lasted 15 mins, so I would have to make a new session every 15 mins. The documentation for it was very poor, to create sessions I had to use md5 to generate a hash to have access to the endpoints to call, this was the first hurdle I had to overcome, it was a lot of testing on checking the hash was made properly to call on all the endpoints. On top of that due to COVID they had to restrict API access to certain hours of the day to relieve stress from their game servers so that restricted the hours I can collect data from the matches. Something I wish I did was change how I collected the data, like a second table in the database that had the parsed data for my recap command, that way the data is pulled much faster instead of 4-5 mins it takes to pull 10,000 plus data and then parse through it.