Documentation

Documentation for the project codebase is available here:
https://harrismcc.github.io/classroom-voter/

Sprint Report

Ishaan: My contributions for this sprint focussed on the professor UI. I merged the professor and Client UI's into one program, called login. The Login UI lets you enter a username and password, and have it sent to the server. The server responds with whether you have a correct username and password, and whether you are a client or professor. Depending, you'd be directed to the proper UI.

Jay: I don't actually have any code in master for this sprint - a set of miscommunications meant that I ended up doing work for things other people pushed to main first, and my other main project isn't production ready yet - I was sanity checking our encryption code from A2 and making sure it behaves with the data packets we're sending, but I want to clean that code up before I publish the branch. I spent two hours in meetings, two or three more coding, and another hour or two looking at the database structure and familiarizing myself with the server side code.

Douglas: My contributions for this sprint focused on server side communication, database design, and the authentication protocol. The server is able to authenticate both students and professors using their username (email address) and their passwords. When receiving a login request the server acquires the reading lock to the database and queries the credentials of the given username. Then the server checks to make sure that the incoming credentials match the credentials stored in the database. In addition to the server work I completed I also set up a basic email script and user database initialization. In this system we assume that we have a list of students emails, the classes they are enrolled in, and a temporary password. With this information we add them to the database and send them an email containing the login credentials.

The total number of hours that I spent working on the project was around 10 hours (including meetings). My productivity for this sprint was good. It was a little slow when trying to figure out how to send emails and settling on a database design. However, the group did a really good job planning out our design and coming up with message conventions. Thus, when it came time to piece all of our code together the integration process was smooth.

Harris: This sprint was for me was focused almost entirely on the database. Although we had first planned on storing all of our user/class/poll data in a large json file and creating a python class to read/write from it - making a pseudo-database, it became apparent that this approach had significant drawbacks. One of which being that concurrent or successive, but out of order, read/writes of the file could cause data to be erased. This is because in order to read the db the entire thing has to be put in memory, then to write a single entry the entire db has to be replaced with the one stored in memory. For this reason, we decided to use SQLlite3 to create a SQL

database instead. This allowed for much faster read/write/lookup times which was an added bonus. The disadvantage was that, on my end, there was a lot more work in developing a wrapper for the db, since our objects are set by default to use json formatting. This meant that I spend most of the time writing methods to break down json objects into db tables and vis-a-versa.

My productivity for this sprint was good, even though a chunk of my efforts went to the ultimately futile json solution - it was necessary to try it in order to realize that it wasn't the correct path.