Vision:

For MS3, we wanted to work further on quality of life support so that users don't need to type in a separated text box in the GUI and click send for their text. To make our app as similar to google doc as possible, we want to immediately reflect character by character changes from each user. The vision remained essentially the same as MS2 now shifting focus away from text styling but making sure the typing experience is smooth and error free. We still don't want user to run the program

Summary of Progress:

Our submission for MS3 now provides full functionality for LIVE collaborative text editing. Now without the need to always click the send button. In addition, we have also added live cursor handling so users can edit texts in the middle instead of only at the end. We also added the formatting for text to be aligned left, right, or at the middle. We decided not to move on with text styling due to the limitation of the frontend and our goal of a truly live editing experience that conflicts. (Bogue.widget.label does not support mixed formatting within the widget). The user also now has the option to save the fruit of their labours into a .txt file for later use. Lastly, we've added testing coverage to the codebase. Lastly, we also reorganized the project so the frontend code and non-testable server logic lives in the bin instead of lib. This helps prevent non-testable code from being counted in bisect tests.

Activity Breakdown:

Jacob Kupperman:
- Organize group meeting and lead PM check-in meeting
- Attempted to implement styling of text
- Revise user interface styling theme
- Refactor codebase so the frontend stays in lib instead of bin.

Stanley Amkhanitsky:
- Implemented text wrapping and text alignment (left, center, right)
- Implemented cursor movement via a button
- Added testing to modules

Muhammad Ali:
- Implemented final save logic. Now it saves in a txt file whenever saves is called so a file can be stored even after the work session finishes/server is shut down.
- Connected frontend save button to backend
- Added mli interfaces for backend files
- Refactored logic from spec_actions back into backend
- Adding implementation invariant and abstraction functions to mli files
- move server logic and frontend code into bin instead of lib.

Ratchaphon Lertdamrongwong:
- Implemented polling input to send characters immediately after the key is pressed.
- Minimize the input field to make user feel like typing directly into shared space

- Build on Stanley implementation of cursor button and connect it directly into the keyboard (instead of only needing to click the button)
- Added a caret character as the cursor so user can see where they are typing
- Auto reset the input field to prevent the issue of sending the wrong key to the server.
- Implemented a new state(dirty) to keep track of whether a change has been made to the document state

Productivity Analysis:

Our team was proud of what we created. We have achieved full functionality of what we aimed to create. Users can conveniently edit the document at the same time without needing to click send, deletion and cursor movement works smoothly even with multiple editors. For MS3, we switched to physical meetings which helped increase productivity and get everyone on the same page. It also allowed us to learn each other's strengths and weaknesses in order to distribute tasks appropriately. We had initially desired to set up styling that could change across the document. However, we unfortunately realized that that goal was quite challenging to achieve because it would require complex coordination of text placement since the text would be broken into separate objects according to where style changes occurred. Fortunately, we were able to keep to our goal of implementing file saving, in our case as a text file in the data folder.