TAPS Maintenance and Help Guide

How to install the system:
This system is run in ReactJS. So NodeJS must be downloaded on the device in order to run the application. Once NodeJS is installed, npm must also be installed– this can be done by running the command: npm install (in the main branch from the terminal).

What the system should do:
This system takes in input files that have information related to the class schedule of the upcoming semester and information relating to potential Teachers Assistants. Once this user input is validated, the user should be presented the option to pre-set any TAs to a specific class that they would like to do. After any predeterminations are made, the system should run and generate the three best possible schedules that can be created with the algorithm that is used. This will take some time, anywhere from one minute to five minutes. Once the three best schedules are found, they should be displayed for the user next to each other. Beneath each schedule output, the system should print out what issues are generated by each of the possible schedules. The system should then allow the user to download each of the possible schedules if they would like to.

Problems and Potential Solutions:

1. Right now, the current semester and year that the application is generating schedules for is hard-coded as a variable in the createEligList.js component. Allowing the user to input this information would be important for this to work without constant care from a developer
2. Currently the algorithm takes a bit of time due to the amount of iterations we are asking it to run through and because of that the page becomes unresponsive while we wait for it to complete. It would be good to find a way if possible for this to be running in the background but still have the page responsive so the user doesn't think the website crashed. Perhaps implement a loading bar.
3. If the TAs in the TAList file aren’t in the transcripts of all the students' files then they are not added to the TAs variable. This could just be a case of bad input files that were given to us, because to be a TA one would assume that they would have to be enrolled at the University and thusly they would appear in the transcript file however we had some TAs that didn’t appear in the transcript file so be wary of that.
4. Potential Future Problem: Line 60 of ./inputHandlers/AllClassesCSV.js you’ll see that the for loop starts with i = 2 and also line 23 the headers are found at row index 1 instead of 0. This is because in the data that was given to us the first row was left blank. It is likely that this could be something that changes with future input files and if so change the header index to 0 and for loop var i to equal 1.

Suggested Additions:

1. The schedule csv should include more course information like the course name, instructor's name, date and time information, ect. as well as include the TA's name not just their UUID.
2. Local Storage for input files or quick drag and drop folder of files options (especially for testing purposes, because putting all 4 each time can be somewhat laborious).
3. Input option for predeterminations. Something along the lines of inputting an incomplete schedule.