Smart OBD Milestone 5: Testing

Testing methodology: The following tests are tasks we will ask an "end-user" or tester to perform using the existing SmartOBD site. We will record whether they found that data easily and if the data found was indeed correct. Tasks involve uploading data, sorting data logs, and account management. We have a similar process outlined for our local executable used to upload car data to the site.

Website user experience

Features to be tested:

- Checking for specific full data log
    - Can the user navigate to data for multiple dates?
        - Using dropdown on full-log page
    - Can the user switch car profiles?
        - Do all car profiles display correctly
        - Can the user easily switch between them on multiple pages?
- Make an account and log in
    - Can the user make an account successfully?
        - Were there any errors in the data submitted?
    - Can they login into account from the last step
    - Did they gain access to any previous data?
- Check that the user can view asynchronous data
    - Data is updated consistently
        - User set refresh or fast as possible?
    - Data is accurate to real-world
        - Is data truly live and is in use from for end-user

Local executable user experience

- User can successfully select car by entering information
    - User starts program
    - User inputs username
        - Program checks for vehicle count
        - If more than one, user inputs make and model of car they wish to access
    - Program successfully queries the database and returns the correct table to enter data from OBD-II port
- Program successfully catches errors in between user input and database
    - If user inputs incorrect username or database query fails, program exits before references anything else (could potentially corrupt data in database or connection to car)
- Program is able to successfully run either asynchronous *or* a full query and upload to the database
    - User chooses which functionality they wish to use
    - The program gets the table to be updated

        - Uses userGet function
    - The program then either continuously updates a single row of the table (asynchronous)
    - Or it will query the vehicle with every compatible command and populate a new row in the respective table