Patrick Fenn
Priscila Huante
Karen Gonzalez
Michael O'Dea

LAB 3 - Uber Calculations

Platform: Desktop

HTTP client: Node.js

Languages: Java, Javascript

Dataset:
https://www.kaggle.com/fivethirtyeight/uber-pickups-in-new-york-city?select=other-American_B01362.csv

Data We Have:

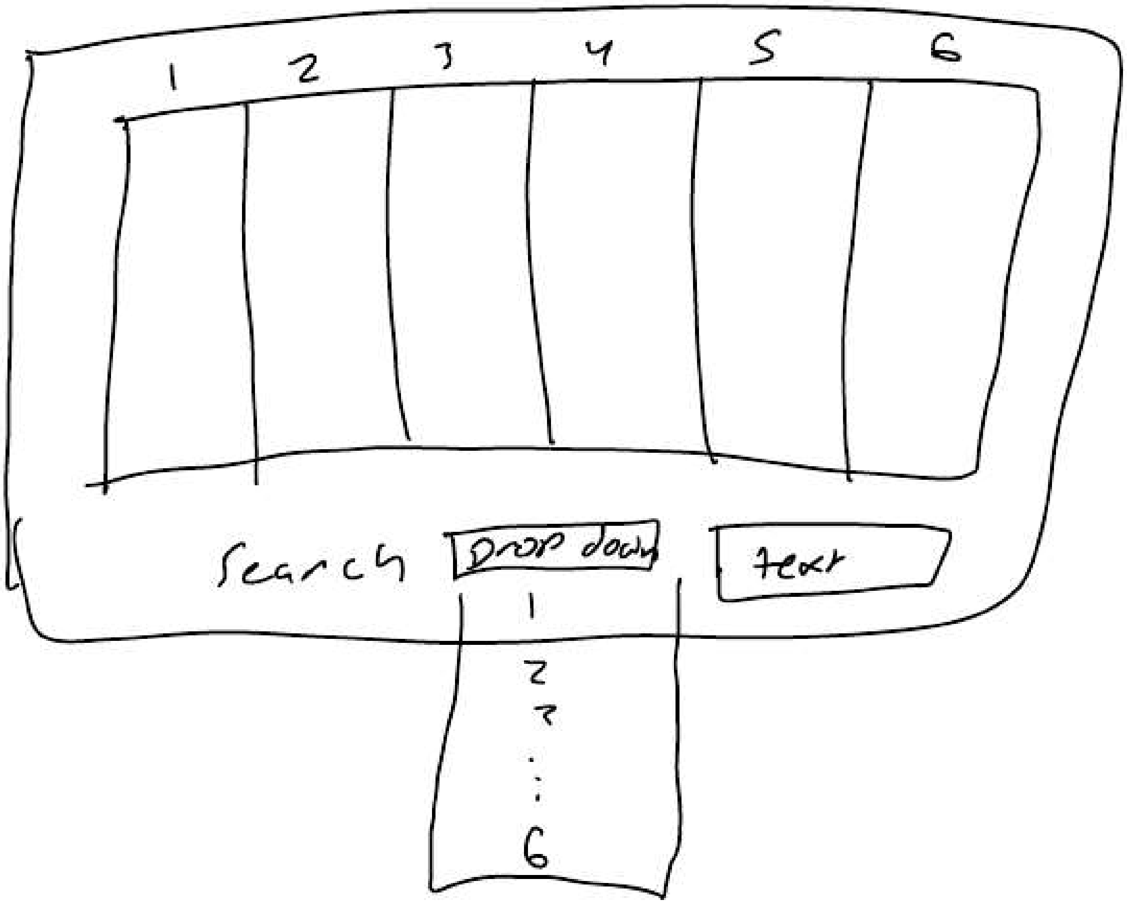
- Uber trip data from 2014 and 2015
    - The dates and times of each Uber trip
    - Pickup Addresses
    - Uber Base Numbers
- Non Uber trips from other companies from 2014 and 2015

Feature List:

- **other-Dial7_B00887.csv**
    - Feature 1: As a user I want to know how many times a pickup address has been used.
    - Feature 2: As a user I want to be able to view what is the busiest time to access a pickup address.
    - Feature 3: As a user I want to be able to view the earliest pick-up time.
    - Feature 4: As a user I want to view the busiest streets that get the most pickup requests
    - Feature 5: As a user, I want to be able to see which cities in NYC get the most pickup requests.

- Feature 6: As a user, I want to be able to see which states get the most pickup requests

GUI:



User Test Cases:

- **Feature 1 Test Cases:** As a user I want to know how many times a pickup address has been used.
  - **Test Case 1:** As a user on the GUI, I put in the address in a search bar to search for how many times an address is used to pick up somebody.
    - Output: The website displays the addresses going from most used at the top, to least used at the bottom.
  - **Test Case 2:** I decide to flip the order from most to least, and now want it to go from least used to most used, with least used at the top.

        - Output: The website reloads and now displays the addresses with the least used addresses near the top, and the most used addresses near the bottom of the table.
- **Feature 2 Test Cases:** As a user I want to be able to view what is the busiest time to access a Pickup address..
    - **Test Case 1:** The user searches an address after selecting this option in the dropdown menu.
        - Output: It searches the address and street column for the full address, and displays the matching entries.
- **Feature 3 Test Cases:** As a user I want to be able to view the earliest pick-up time on select days.
    - **Test Case 1:** The user selects a day in the dropdown menu.
        - Output: It displays the earliest pick-up time from the selected day.
- **Feature 4 Test Cases:** As a user I want to view the busiest streets that get the most pickup requests
    - **Test Case 1:** As a user, I select the street option in the dropdown menu, then select most frequent to least frequent.
        - Output: The website displays the streets going from most used at the top, to least used at the bottom.
    - **Test Case 2:** I decide to flip the order from most to least, and now want it to go from least used to most used, with least used at the top.
        - Output: The website displays the streets going from least used at the top, to most used at the bottom.
- **Feature 5 Test Cases:** As a user, I want to be able to see which cities in NYC get the most pickup requests.
    - **Test Case 1:** As a user, I select the city option in the dropdown menu and then select most to least frequent.
        - Output: The website displays the cities that had more pickup requests on the top, to least requests at the bottom.
- **Feature 6 Test Cases:** As a user, I want to be able to see which states get the most pickup requests.
    - **Test Case 1:** As a user on the GUI, I put the day for pickup requests to see which states are the most active for that day.
        - Output: The website displays the districts going from most used at the top, to least used at the bottom.
    - **Test Case 2:** I decide to flip the order from most to least, and now want it to go from least used to most used, with least used at the top.
        - Output: The website displays the districts going from least used at the top, to most used at the bottom.

To-Do List

Done list of last sprint

- Add textbox, buttons to GUI.
    - Finished by Patrick Fenn
- Implement server used for connection in Node.js.
    - Finished by Patrick Fenn
- Add logic for each button in the Controller class.
    - Finished by Patrick Fenn
- Create Html class to initiate connection from client to server.
    - Finished by Patrick Fenn
- Implement a function that sends and receives messages from the server.
    - Finished by Patrick Fenn

ToDo Task List for the next sprint:

- GUI for search
    - Add Table
    - Add Textbox
    - Add search label.
- Create a backend for data.
    - Implement Mysql
        - Add the Dial7 data set.
- Implement search feature
    - Search the provided array for matching cases.
        - Search for address must search both address and pickup street. Will need a function that joins the values and then performs the search.
- Create a layer to access the mysql data set.
    - Have a function that returns each column within the data set as an array.