Team: Caution

Features (user stories) to Implement in Next Sprint:

- **US Accidents:**
    - **Feature 1**: as a user, I want to be able to compare the number of accidents based on weather. **(Completed)**
    - **Feature 2**: as a user, I want to be able to compare the average severity of accidents based on weather. **(Completed)**
    - **Feature 3**: as a user, I want to be able to compare the number of accidents based on location. (near a highway vs. not near a highway)
    - **Feature 4**: as a user, I want to be able to compare the number of accidents based on the city. **(Completed)**
    - **Feature 5**: as a user, I want to be able to compare the number of accidents based on the distance from home. **(Completed)**
    - **Feature 6**: as a user, I want to be able to compare the number of accidents based on having stop signs and/or speed bumps at the accident. **(Completed)**
    - **Feature 7**: as a user, I want to be able to compare the number of accidents based on humidity level. **(Completed)**
    - **Feature 8**: as a user, I want to be able to compare the average severity of accidents based on humidity level.

Test Cases

- **Feature 1 Test Cases**: as a user, I want to be able to compare the number of accidents based on weather.
    - **Test Case 1**: as a user, in the Statistics section, I select "Number of Accidents on Weather" and click the "Compare" button.
      Correct Output: A statistic request is sent to the backend. The backend finds the number of accidents based on weather. The backend returns the values to the frontend. The frontend displays a graph relating weather conditions and number of accidents.
- **Feature 2 Test Cases**: as a user, I want to be able to compare the average severity of accidents based on weather.

    - **Test Case 1**: as a user, in the Statistics section, I select “Severity of Accidents on Weather” and click the “Compare” button.
      Correct Output: A statistic request is sent to the backend. The backend finds the different types of weather conditions and based on the conditions, looks at the severity of the accidents. The backend returns the values to the frontend. The frontend displays a graph relating weather conditions and severity of accidents.
- **Feature 3 Test Cases**: as a user, I want to be able to compare the number of accidents based on location. (near a highway vs. not near a highway)
    - **Test Case 1**: as a user, in the Statistics section, I select “Number of Accidents on Location (highway)” and click the “Compare” button.
      Correct Output: A statistic request is sent to the backend. The backend finds the number of accidents based on location (highway). The backend returns the values to the frontend. The frontend displays a graph relating highway existence and number of accidents.
- **Feature 4 Test Cases**: as a user, I want to be able to compare the number of accidents based on the city.
    - **Test Case 1**: as a user, in the Statistics section, I select “Number of Accidents on Cites” and click the “Compare” button.
      Correct Output: A statistic request is sent to the backend. The backend finds the number of accidents based on cities. The backend returns the values to the frontend. The frontend displays a graph relating cities and number of accidents.
- **Feature 5 Test Cases**: as a user, I want to be able to compare the number of accidents based on the distance from home.
    - **Test Case 1**: as a user, in the Statistics section, I select “Number of Accidents on Distance from Home” and click the “Compare” button.
      Correct Output: A statistic request is sent to the backend. The backend finds the number of accidents based on distance from home. The backend returns the values to the frontend. The frontend displays a graph relating distance from home and number of accidents.
- **Feature 6 Test Cases**: as a user, I want to be able to compare the number of accidents based on having stop signs and/or speed bumps at the accident.
    - **Test Case 1**: as a user, in the Statistics section, I select “Number of Accidents on Stop Signs/Speed Bumps” and click the “Compare” button.
      Correct Output: A statistic request is sent to the backend. The backend finds the number of accidents based on having stop signs and/or speed bumps at the accident. The backend

returns the values to the frontend. The frontend displays a graph relating having stop signs and/or speed bumps at the accident and number of accidents.

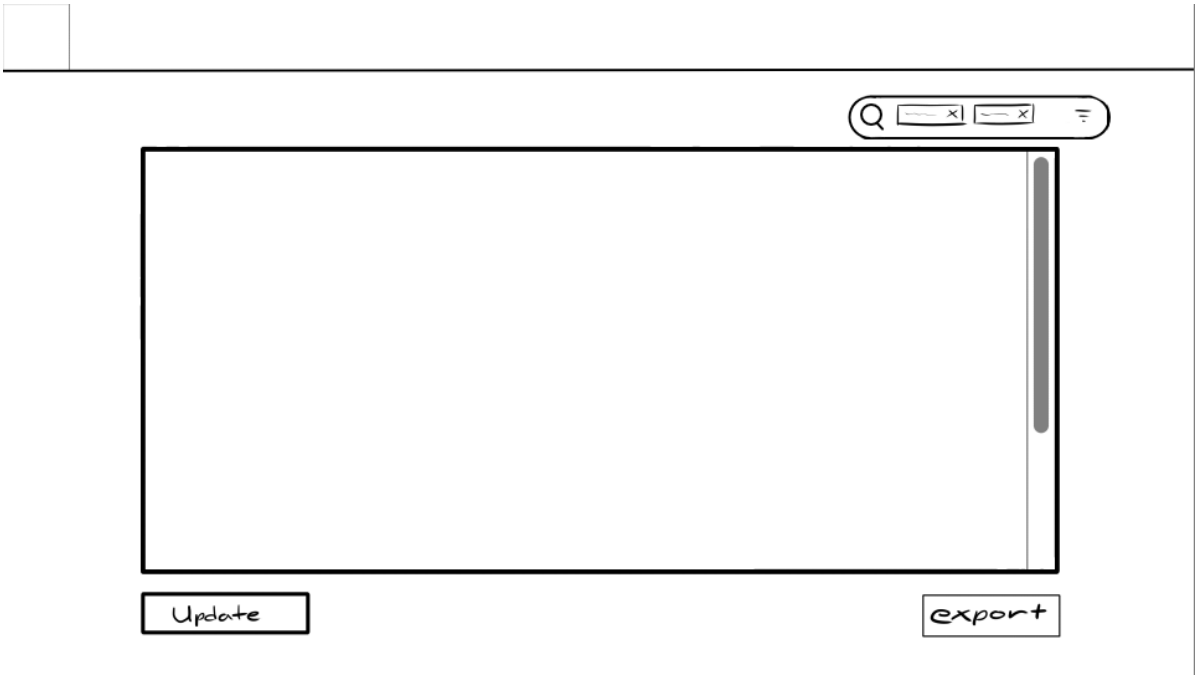
- **Feature 7 Test Cases**: as a user, I want to be able to compare the number of accidents based on humidity level.
    - **Test Case 1**: as a user, in the Statistics section, I select “Number of Accidents on Humidity” and click the “Compare” button.

      Correct Output: A statistic request is sent to the backend. The backend finds the number of accidents based on humidity. The backend returns the values to the frontend. The frontend displays a graph relating humidity and number of accidents.
- **Feature 8 Test Cases**: as a user, I want to be able to compare the average severity of accidents based on humidity level.
    - **Test Case 1**: as a user, in the Statistics section, I select “Average Severity on Humidity” and click the “Compare” button.

      Correct Output: A statistic request is sent to the backend. The backend finds the average severity based on humidity. The backend returns the values to the frontend. The frontend displays a graph relating humidity and average severity.

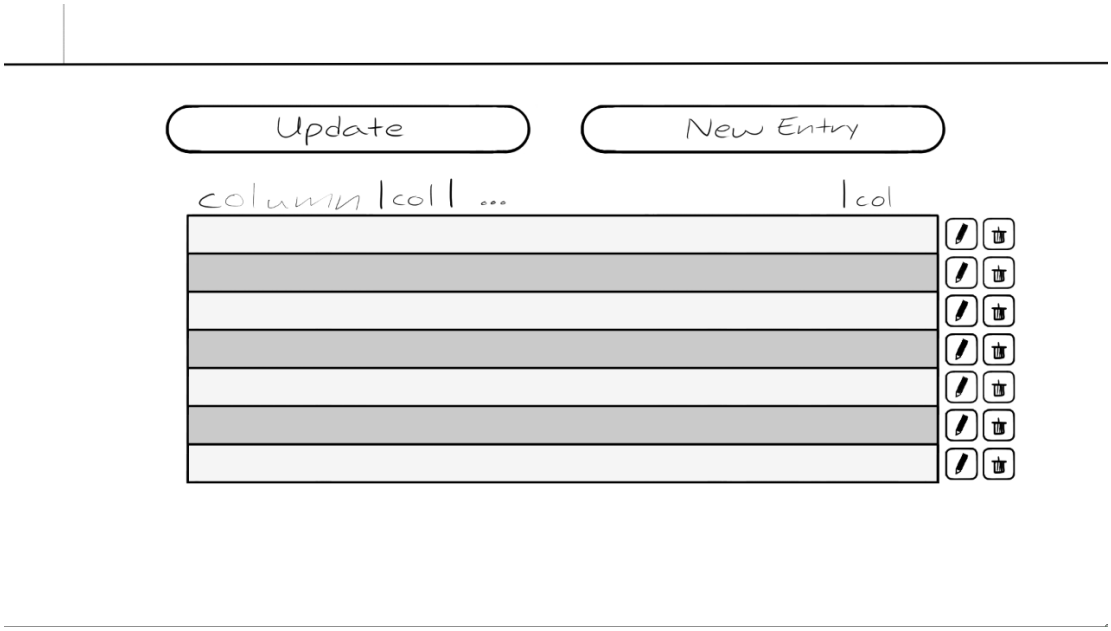
Design:

- **Search Page:**

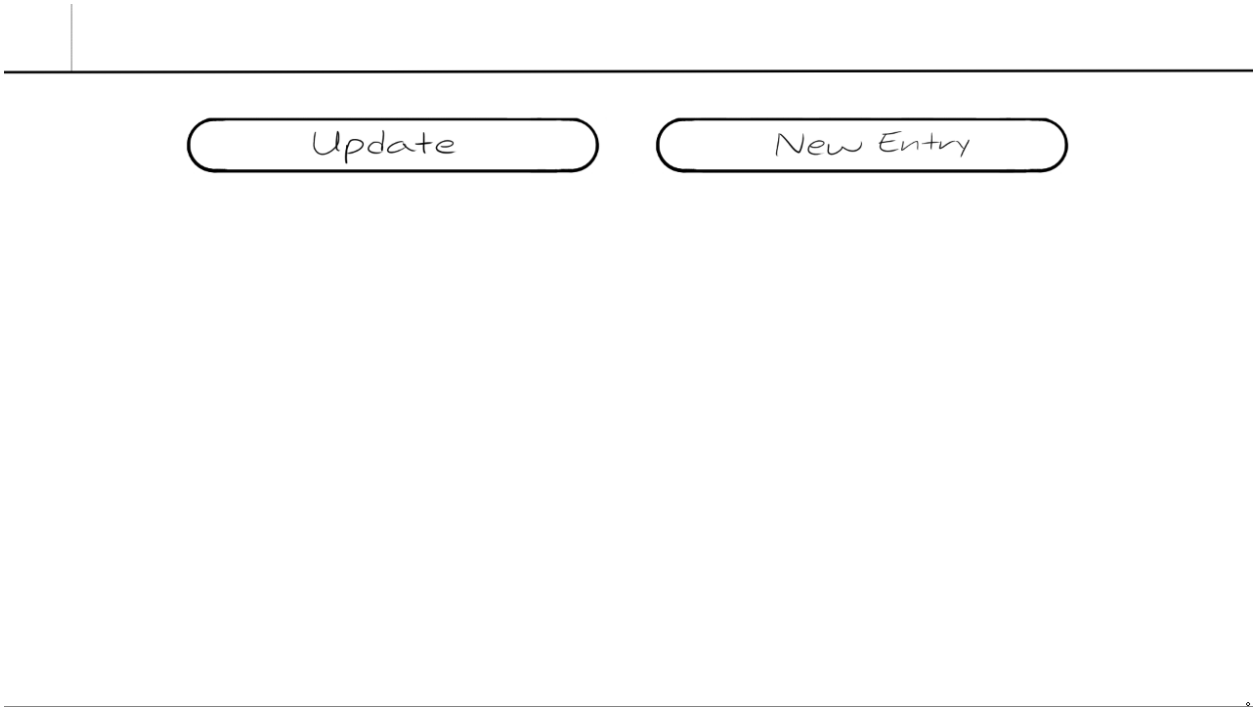


New Entry:

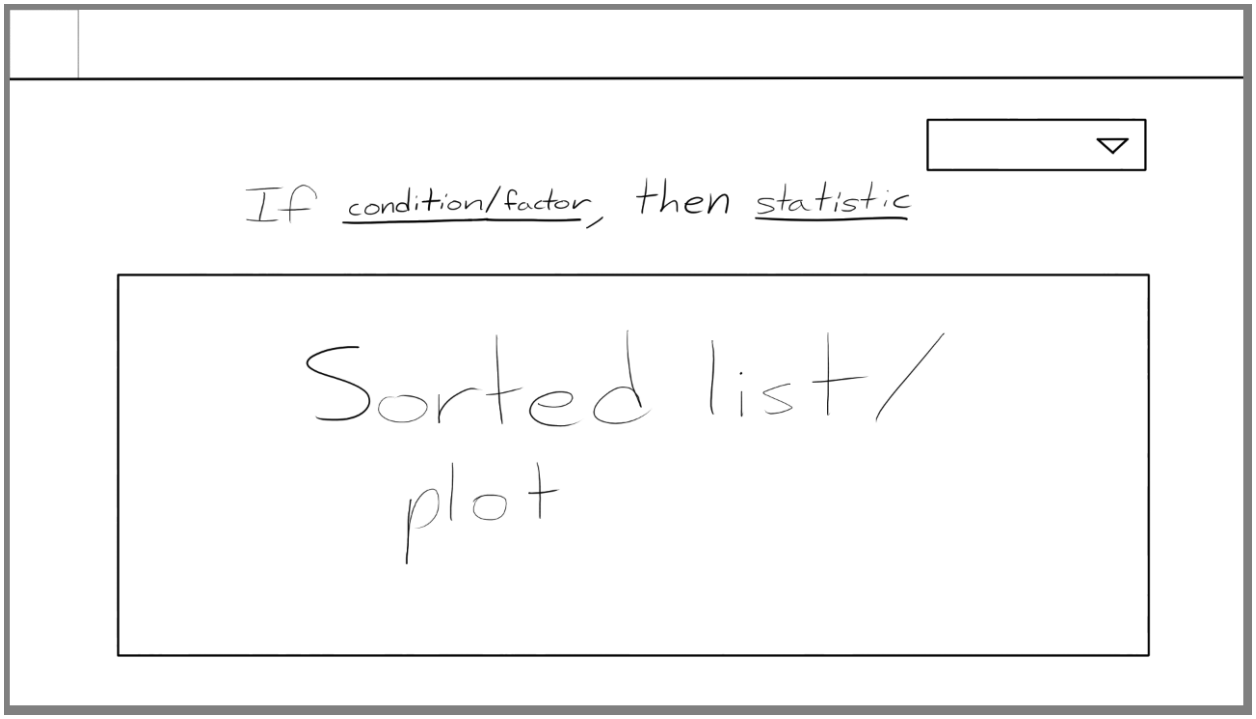
Update:



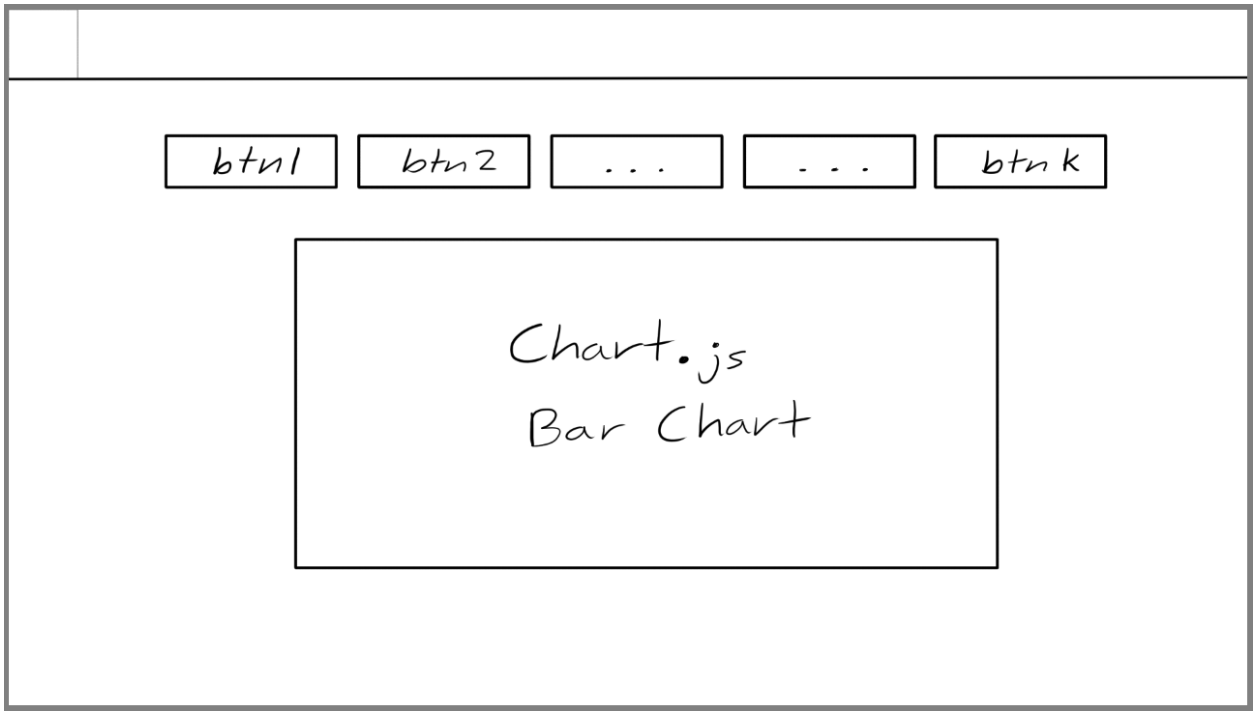
Insert/Update:



Analytics1:



Analytics2 (revised):



TO-DO LIST:

Done list of last sprint:

- Backend development to get frequency of elements in a filter and putting that into a 2D array for analytics
    - Finished by Kenny and verified by everyone
- Frontend development for connecting and making buttons for analytics using 2d array returned from the server
    - Finished by Danial and Kenny and verified by everyone
- Query analytics in backend for new UI
    - Finished by Albert and verified by everyone
- Display analytics graph in frontend in new UI
    - Finished by Albert and verified by everyone
- Front-end Updating of Tables to allow sorting from greatest to least
    - Finished by Rahul and Danial and verified by everyone
- Created Chart.js template to be used on front-end for bar-charts
    - Finished by Rahul and Matthew and verified by everyone

To-Do for next sprint:

- Clean up the front end with a more user friendly UI matching the theme of Caution
    - Acceptance Criteria: Functionality still works but the UI is easier to navigate and look at which can be done by following the mockups more closely
- Have the front end work on different tabs
    - Acceptance Criteria: Be able to click buttons at the top to change pages/tabs for adding new values, searching, analytics, etc.
- Finish the rest of the analytics for displaying statistic plots/tables
    - Acceptance Criteria: When clicking the compare button we are able to see from the plot/table the statistics and text area updated
- Above the plots/tables have explanations of the actual analytics
    - Acceptance Criteria: User will be able to read and understand how the analytics coordinates with the theme of the web application