Artifact 3

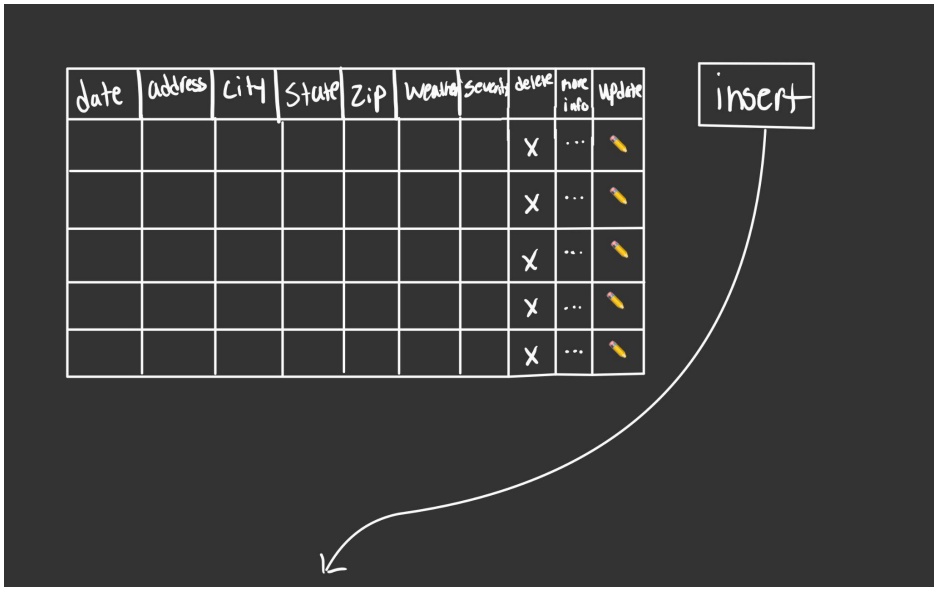
Feature 1
As a user I want to be able to update the severity of the accident, city, state, street address, and zip code of a record.

Feature 2
As a user I want to record an accident. I enter the street address, city, state, zip code, and severity of the accident.

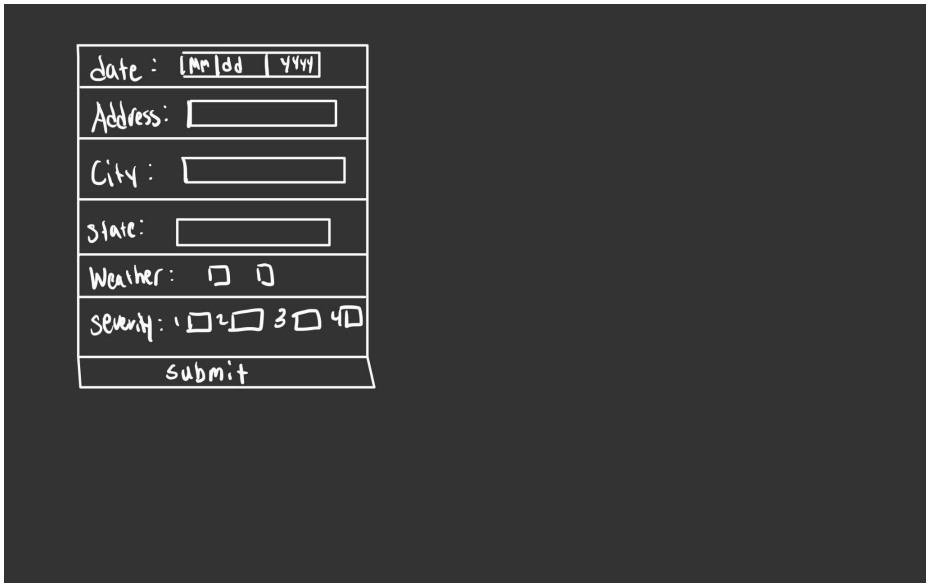
Feature 3
As a user I want to delete a record. I want to search for a specific accident and delete it from the database.

Feature 4
As a user I want to see all the records. I want to search for specific accidents by specifying certain fields such as the address or accident severity.

Update & Delete



Insert



Test cases

Feature 1 test cases: As a user I want to be able to update the severity of the accident, city, state, street address, and zip code of a record.

Test case 1: As a user, after I have searched for the records in CA, I want to be able to click on an edit button next to the record I want to update by editing the address or severity.

Correct Output: A pop up window will display showing the fields I can edit for the specified record. The form will be populated with the existing data.

Test case 2: As a user, I want to update the severity level by clicking on a dropdown button and selecting a new severity level (1-4).

Correct Output: The new severity level will be displayed on the dropdown button.

Test case 3: As a user, I want to click a submit button on the pop up window.

Correct Output: The pop up menu will close and a message “The record has been successfully updated”.

Feature 2 test cases: As a user I want to record an accident. I enter the street address, city, state, zip code, and severity of the accident.

- Test case 1: As a user I enter numbers in city field
  Correct Output: The user gets a message prompting them to input a string.
- Test case 2: As a user I enter a number in the state field
  Correct Output:The user gets a message prompting them to only string.

- Test case 3: As a user I enter a string in the state field
  Correct Output:The user gets a message prompting them to only input numbers in the field.

- Test case 4: As a user I enter a string in the severity field
  Correct Output:The user gets a message prompting them to only input numbers in the field.

Feature 3 Test Cases: As a user, I want to delete an entire record.

- Test case 1: as a user, I want to click on a button with a trash icon next to the record I want to delete
  Correct output: A form will pop up. The top of the form will have a message that reads "Are you sure you want to delete the following record?". The rest of the form will be populated with the existing data for that record and at the bottom of the form will be a yes and no button.
- Test case 2: as a user, I want to click on the "yes" button to delete the record.
  Correct output: The pop up form will close and the message "You have successfully deleted the record" will display for 3 seconds. The deleted record will no longer be visible in the list of displayed records.
- Test case 3: as a user, I want to click on the "no" button to not delete the record.
  Correct output: The pop up form will close and the record will still be visible in the list of displayed records.

Feature 4 Test Cases: As a user I want to see all the records. I want to search for specific accidents by specifying certain fields such as the address or accident severity.

- Test case 1: As a user I enter the address of an accident
  Correct output: The website displays a table with accidents that happen at that address.
- Test case 2: as a user, I select a severity number.
  Correct output: The website displays a table of all the accidents with the same severity.

Taskboard:

- **Back-end:** Backup the data (locally) automatically when you close the application. Create a copy of the data. Do not replace.
- **Back-end:** Import data on startup from a save file

- Insert (Severity, Street,City,State,Zip Code)
  - **Front-end:** Create a form which sends JSON data to the back-end.
  - **Back-end:** Receive the JSON data and store it into database;
- **Front-end:** Return results from a search as a table.
- **Front-end:** Create a button to update that specific record.

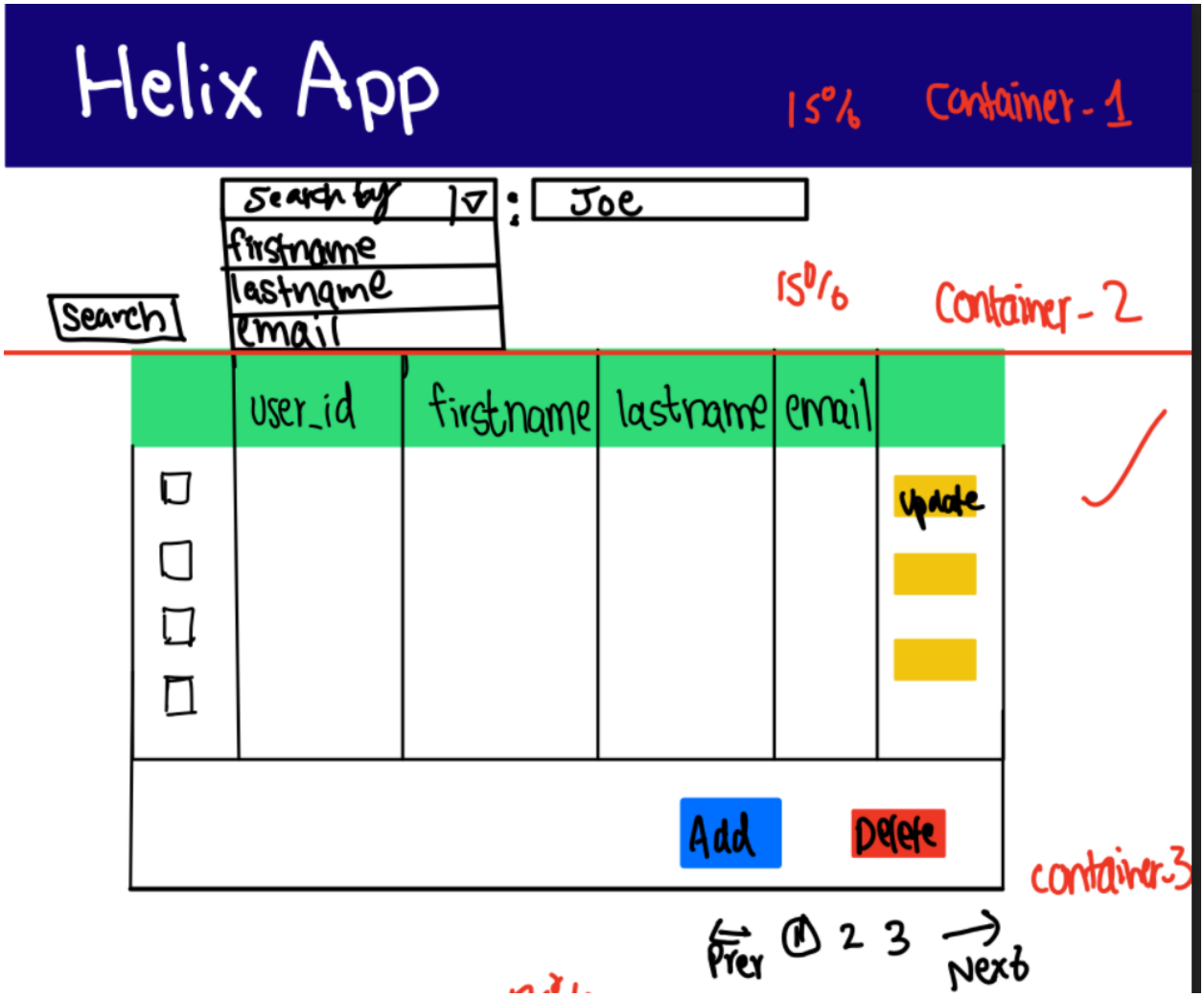
- Update (Severity, Street,City,State,Zip Code)

    - **Front-end:** Add an edit button which shows a form to edit that specific record. Sending a JSON request to update.
    - **Back-end:** Receive an update request and update the data in the database.
- Delete
    - **Front-end:** Create a button to delete that specific record. (Send the accident id?)
    - **Back-end:** Receive delete request from front end and delete the record with the corresponding id.
- GUI
    - **Front-end:** Create a layout for the search / update page. Add an insert button at the top. Add the search form. Add the table of results component.
    - **Front-end:** CSS / make it look pretty

- Validation
    - **Front-end:** Create validation for form data.
    - **Back-end:** Return data from valid parameters even if one parameter is incorrect.

Done List of Last Sprint

- Ivan Carrillo worked on the frontend submit form
- -Thuan Vu worked on display the parsed data on the frontend
- Thuan Vu & Yiu Ming Wong worked on connecting backend to frontend
- Thuan Vu worked on CsvToJson parser
- Yiu Ming Wong worked on searchquery
- Estela worked fixing searchquery and the demo
- Jacob worked on the date search query


Helix App
15%
Container - 1
Search by
Joe
firstname
lastname
email
Search
15%
Container - 2
user_id
firstname
lastname
email
Update
Add
Delete
container-3
Prev
1
2
3
Next