CART 451: Project Prototype

November 8th, 2022

Amanda Clément

ID 40136259

PROJECT DESCRIPTION

My project is an interactive data visualization that illustrates sentiment-based trends related to the COVID-19 pandemic. The data surrounding this global crisis, such as reported cases and death tolls, is predominantly numerical. This quantitative data, although crucial to our understanding of the virus' spread, can be difficult to engage with and to relate to, and gives no information about the public's emotional response to the pandemic. As such, my goal is to explore and analyze the collective sentiments associated with the COVID-19 crisis by extracting sentiment scores from Twitter posts. By organizing and filtering this data in certain ways, I wish to explore patterns that may emerge while simultaneously making the findings more approachable and digestible than numerical statistics.

PROGRESS

A list of what has been completed:

- The database connection is set up and queries are used to extract the necessary information (at this stage, I only get the text from each Tweet).
- The text from the Tweets is processed and passed through the sentiment analysis algorithm to assign a score to each one.
    - The highest number of Tweets ran through the algorithm is 10,000
- For each extracted sentiment (so for each Tweet in the sample), an ellipse is drawn to the canvas, coloured based on the associated sentiment (yellow for positive, blue for negative, grey for neutral or inconclusive).
- When the user presses down on an ellipse, the Tweet text is displayed in the center of the screen.
- The user can filter the visualization by sentiment using buttons on the HTML page.
- The user can click on an ellipse to view a snippet of the Tweet text.

- Some additional information is displayed to the screen to give the user more information about the sample (i.e. percentages of each sentiment, sample size).
- HTML elements are displayed on the page (will add more and make modifications as/if needed).
- CSS styling is applied.

A list of what still needs to be completed:

- Explore additional ways of using the Tweet data (e.g. size the ellipses based on the number of followers or favorites a user has)
- Determine if it's possible to include a larger sample size (>10,000 Tweets) without too much lagging, or to at least improve the performance at 10,000 Tweets.
    - Decide how the sample of Tweets is selected (e.g.. choose at random, based on the post date).
- Add more data filtering options: by geolocation (need to fetch the geolocation of each Tweet from the database), by date (need to fetch the data of each Tweet from the database), by keyword (need to allow users to query the database using a keyword).
- Add relevant information from government or news databases (e.g. number of reported cases in a given month, mandates that were in place that month) alongside the visualization.
- Add more text to the *About* page.
- Implement a loading page.
- Improve the language processing algorithm and run more tests.

COMPONENTS

A description of the responsibilities of each component and how they integrate into the prototype:

index.js

- Connects to the database (using MongoDB), which in turn uses DBSchema.js to set up the schema.
- Queries the database using Mongoose to extract the necessary information.
- Processes the text from each Tweet (e.g. changing to lowercase, removing stopwords, tokenizing, etc).

- Applies the sentiment analysis algorithm from the Natural package, which uses AFINN to assign a numerical value from -5 to 5 for the score.
- Stores the resulting sentiment scores as well as the Tweet text in separate arrays and sends this info along the /custom route via a POST request to be used in main.js.

DBSchema.js

- Sets up the schema for MongoDB/Mongoose.

main.js

- Fetches the array of sentiment scores and Tweet texts from index.js via the /custom route.
- Uses the fetched data to create the visualization by drawing ellipses to the canvas using p5, with each ellipse representing a Tweet's corresponding sentiment score. Also uses this data to display the Tweet text associated with an ellipse when the user presses down on it, and to display information about the data on the index.html page (e.g. percentage of positive scores, sample size used).
- Creates and manages some of the UI elements (e.g. buttons to filter the data).

index.html

- The page the user sees and interacts with.
- Includes some basic HTML elements which are displayed on the page.

style.css

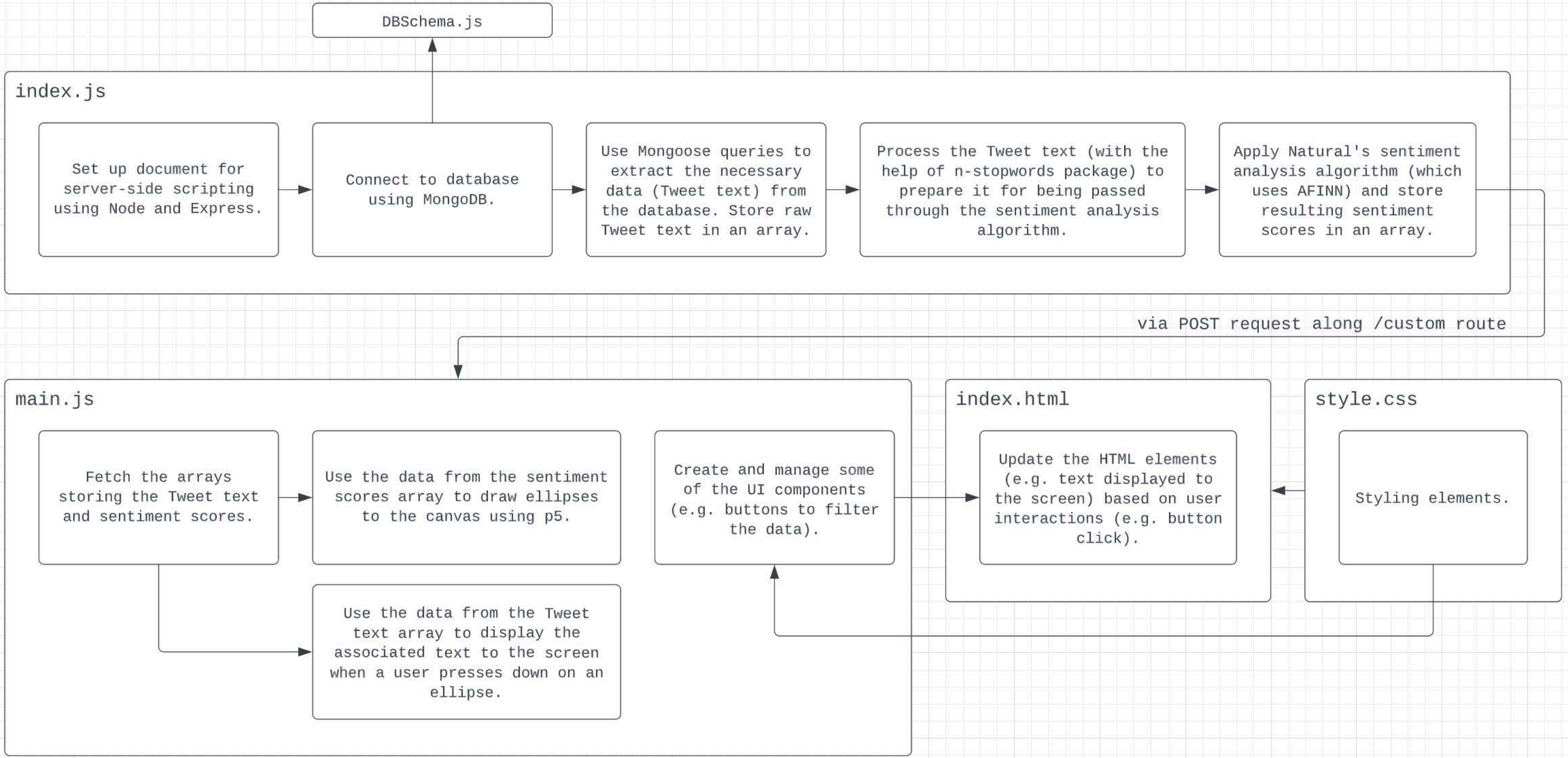
- Styles the HTML elements (some from the index.html file, others from the main.js file).

WHAT'S WORKING AND WHAT NEEDS TO BE RE-WORKED OR RE-CONSIDERED

The following features are working correctly: the database connection and querying to fetch the necessary data, the processing of the text from each Tweet, the passing of this text through the sentiment analysis algorithm to get a corresponding sentiment score, the canvas drawing for the visualization and the filtering of the visualization by sentiment, and the displaying of the associated Tweet text when the user hovers over an ellipse.

That being said, the sample size needs to be re-considered. As it stands, running the language processing and sentiment analysis algorithms on the text from each Tweet takes longer than expected. For instance, with a sample of 10,000 Tweets, it takes nearly four minutes for the visualization to load and the HTML page has noticeable lagging on my laptop. This speed problem only worsens if I choose to fetch more data from the database (e.g. number of likes on a Tweet for setting the ellipse size). The same applies if I include another dataset (e.g. government COVID-19 statistics). Unless I can find a solution to this performance issue, I need to be more selective about the data I include in my project. As such, I may not include all the features outlined in my proposal if they cause too much additional lagging.

Lastly, although I had originally planned on extracting an emotion (e.g. happiness, anger) from each Tweet rather than just a sentiment score (e.g. -3 for negative), I may have to scrap this idea. After doing more research, I discovered that emotion analysis is much more complex than sentiment analysis and may be too complex and time-consuming to effectively implement.


DBSchema.js
index.js
Set up document for server-side scripting using Node and Express.
Connect to database using MongoDB.
Use Mongoose queries to extract the necessary data (Tweet text) from the database. Store raw Tweet text in an array.
Process the Tweet text (with the help of n-stopwords package) to prepare it for being passed through the sentiment analysis algorithm.
Apply Natural's sentiment analysis algorithm (which uses AFINN) and store resulting sentiment scores in an array.
via POST request along /custom route
main.js
Fetch the arrays storing the Tweet text and sentiment scores.
Use the data from the sentiment scores array to draw ellipses to the canvas using p5.
Use the data from the Tweet text array to display the associated text to the screen when a user presses down on an ellipse.
Create and manage some of the UI components (e.g. buttons to filter the data).
index.html
Update the HTML elements (e.g. text displayed to the screen) based on user interactions (e.g. button click).
style.css
Styling elements.