Walking Safely Project Portfolio

CSCI 491 Web Development

Instructor

Daniel DeFrance

Group 11

Chris Miller
Connor Morrison
Manuel Salazar
Jacob Weikert

Montana State University

Nov 26, 2019

1. Link To Project

Here is a link to a demonstration of our app:

https://www.youtube.com/watch?v=jYG7u3mEVag

Directions to deploy our app locally:

1. Unzip the “Walking Safely” folder.
2. Cd into the root of the folder.
3. Run “npm install”

Ex:

```
C:\Users\connor.morrison\Desktop\WalkingSafely>npm install
```

4. Once the npm install is done run “run-p server start”

Ex:

```
C:\Users\connor.morrison\Desktop\WalkingSafely>run-p server start
```

2. Objectives

Our project was divided into stages. By doing that, we were able to set out objectives for every stage in an easier way. However, the main goal of the entire project is written below:

- **Provide users a web-based tool that lets them know what areas of specific cities are unsafe to walk in.**

Stage One (Initiation & Planning)

- Identify the project’s main purpose.
- Assign roles to group members.
- Establish technologies/frameworks/programming languages for the project.

Stage Two (Execution, Debugging & Integration)

- Deliver separate working pieces of the project.
- Merge them into one working entity of the project.
- Test and debug.

Stage Three (Delivery & Closure)

- Come up with a working prototype of the project, taking care it resembles the main objective as much as possible.
- Deliver the project on time.

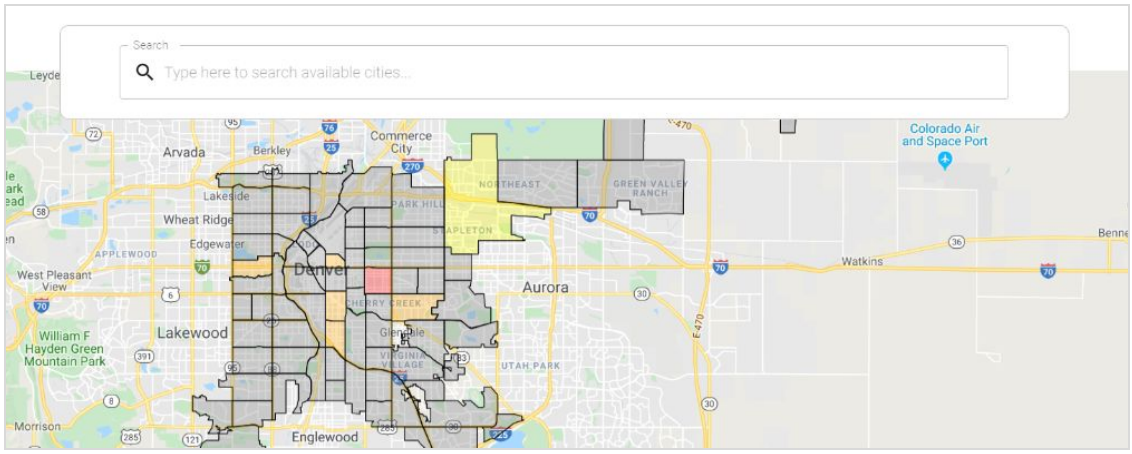
3. Summary

How It Works?

Our website provides users detailed information regarding crimes within an area (the area should be small enough to be considered walkable).

It takes a location, like an address, and turns that into latitude and longitude coordinates. This is compared to polygons of neighborhoods and municipalities, and changes the color based on the number of reports in the area. This lets users know what areas are unsafe.

As shown in the image below, areas are drawn based on polygon information of neighborhoods and municipalities. Colors are then used to indicate different levels of crime within given neighborhoods (yellow is low severity, while red is high).



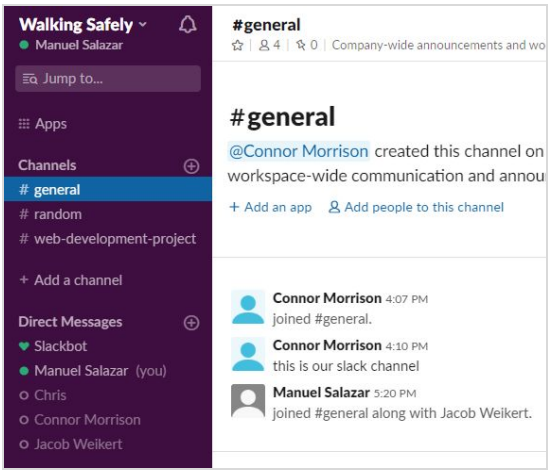
Trade Offs

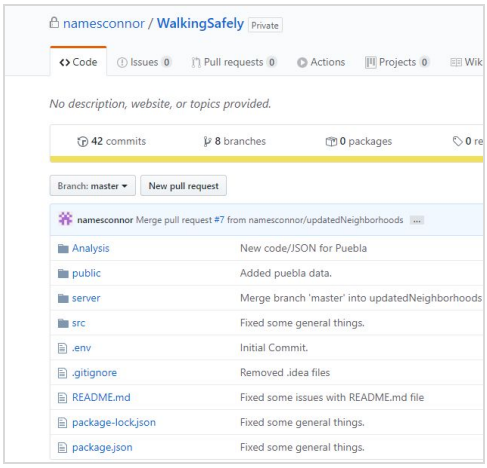
- Instead of a clustering algorithm to display crime rates, it was decided to use colors based on the number of crimes that occurred within a neighborhood and municipality. This is because the colors allow the information to be easier to digest to users then a bunch of numbers scattered across the screen.
- Municipalities were used for Mexico instead of neighborhoods, as the data simply does not exist.

Tools Used

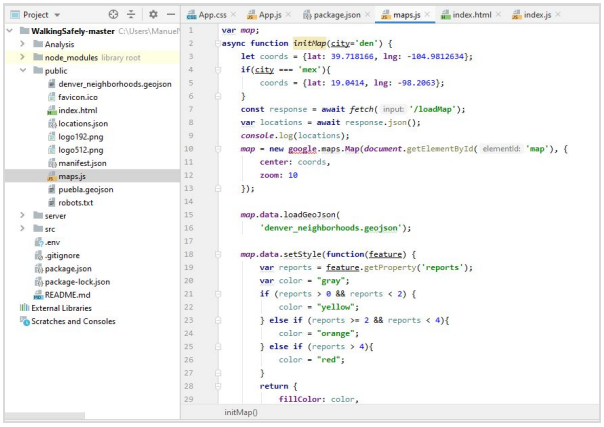
In order to properly implement our project, we used the following tools/software:

- Slack, for group discussions and general communication, and GitHub for code managing.

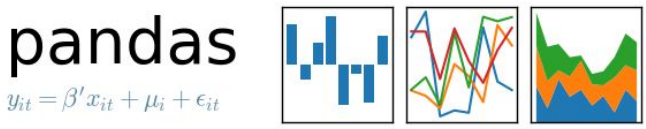


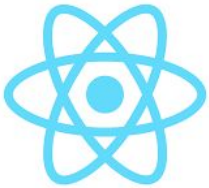


- React, JSX, HTML, JavaScript, and CSS for front-end.
- The back-end is written in Node.js. We also use Express.js to allow for routes to make easier api calls. We used Python to develop the data used to build the crime latitude and longitudes found within the map.



- Several libraries were also used, such as Google Maps API, Pandas Py-Data and GeoPy, as well as a heaping of node modules that can be found in our package.json.





4. Members

Chris Miller

➢ Created the react front end for the form that allows users of the site to submit reports of crime or suspicious activity in their area. Created, along with Connor the express routes that the form data is POSTed to which returns proper HTTP Request status codes. The data taken in from the form is processed and the address given by the user is converted to coordinates by a google maps API. These coordinates are then placed in the geojson file Connor worked with which automatically updates the live map with a new reported crime, changing the color of the appropriate area based on how many reported crimes there are.

Connor Morrison

➢ Created the React front-end and Node.js back-end for the app. Found geojson data for each of the cities (Denver and Puebla) to build the lines distinguishing the neighborhoods and municipalities around them. Created, alongside Chris, a form to

allow for people to put in an address to be converted to coordinates. This is put into a JSON file that is used to change the frequency of attacks per neighborhood, which is represented on the map via different colors. This map is built using the Google Maps JS API.

Jacob Weikert

- Assisted with the decision making processes used to display the crime rates in a proficient way to the user, allowing for ease of use and removal of information deemed irrelevant for our objective. (ie. Use colors and neighborhoods to display different crime areas instead of the clustering algorithm originally proposed. This will allow users to see the differences between crime-rate areas more quickly, as they can use colored areas instead of a bunch of numbers scattered across the screen)
- For cities in the USA -- Developed backend workings in Python using Pandas to filter out information that was not associated with our final objective (ie. traffic accidents -- not a crime), as well as information that was considered to be no longer relevant (ie. data that is 3+ years old). The geo-coordinates of the results were then passed into a json object and sent forward to another member to be further processed for the front end.

Manuel Salazar

- The Mexican government offers data regarding crimes for every state in Mexico. I got the data from the government's website. I cleaned the data by parsing it to Numpy arrays, and then I converted it to Pandas DataFrames. I ended up having an array of names of municipalities of Puebla only, and another one containing the number of crimes for each municipality. Following those steps, I used GeoPy to transform each municipality name into a latitude and longitude number. I converted those numbers onto a JSON DataFrame. Connor and Chris figured out how to use that DataFrame and put it into polygons.

5. Conclusion

The conclusion we came to is that mapping apis like Google have a large amount of complexities to allow for basically anything you want to accomplish. We first tried marker clustering, but this was buggy and/or did not work on any of the apis we supplied. We instead changed the color of polygons determined from a geojson file, which we felt better suited our project. We also implemented something called “React-router”, which allows for our app to be a single page application, without having to reload the page everytime someone clicks a link. We also learned that deploying a React/Node.js app is much more difficult that deploying a LAMP stack application. If given enough time, we would probably deploy the app using a service like Heroku. If we had to do this again, we would probably try to make it mobile friendly first. The next step would be to test our web application with

users and to add this to a mobile application store to allow for all people who want to walk safely to enjoy.

6. References

[1] Python Module, https://pandas.pydata.org/pandas-docs/stable/reference/index.html

[2] Denver polymap, crime dataset, https://www.denvergov.org/opendata/dataset/city-and-county-of-denver-crime

[3] Project Github, https://github.com/namesconnor/WalkingSafely

[4] Material-ui documentation, https://material-ui.com/

[5] Mexico Dataset, https://www.gob.mx/sesnsp/acciones-y-programas/incidencia-delictiva-87005?idiom=es

[6] GeoPy Documentation, https://geopy.readthedocs.io/en/stable/