Visualyze (Crimetrack)

By Philippe, Piyush, Mack

Our motivation for developing this application is to help people easily visualize crime rates and statistics of municipalities and states in the form of a geographical heat map (choropleth). The application is designed specifically to allow maximum extensibility by utilizing the many field values featured in the database schema. This application is a three-tier application, involving [1] a **frontend** using React and MUI, a **backend** using Python and Flask, and a **database** which is operated by the backend using Sqlite.

Design

The state of our current application is a fully functional fullstack application; however there are several limitations in our current state of our application. Though our application can already fully map out the various regions, and appropriately showing the colors based on the average crime rate, our application Is still highly limited in terms of speed (for large databases), adjustable features, and data synchronization.

File Structure

Our source is hosted on GitHub. On the top level, we have the **frontend**, **backend**, and **sample dataset** folders. Notes and initialization scripts are placed on the top level. Due to in-development artifacts, our frontend folder is temporarily named `crime_database`, while our backend folder is temporarily named `scripts`. Imported data files will be stored on the backend subfolder `$BACKEND/db`, in which they will be converted and augmented by the backend.

Dataset Schema

Our dataset schema is decided based on a single Kaggle dataset by Michael Bryant. This dataset contains 146 different fields, but only a few of them are actively involved in the backend, while extraneous fields are here for future expansions. Sqlite will add four more different fields, in which it will use the community name and state information to obtain geographical boundaries as a Polygon, and then add a flag stating whether it is city-level or state-level.

It is important that some fields, such as the community name, state, population, land area, and important percentage-based datasets be not null to avoid complications from the backend and frontend. Concessions have to be made should an anomalous entry be entered.

Frontend

Our frontend currently uses React, supplanted with MUI. Our program currently has three sites:

Import Dataset

The user is able to import dataset through a file-input tag and through an `onDrag` event handler. It currently supports local Excel and CSV files.

My Datasets

The user is able to view all of the data after selecting or reselecting the databases processed and stored on the backend. The most important data are shown on the table, further data can be seen by hovering over the community name.

Visualization

The capstone of our project. The user is provided a map, implemented in React Leaflet. After selecting a file, the app will load the polygons to provide a crime frequency choropleth.