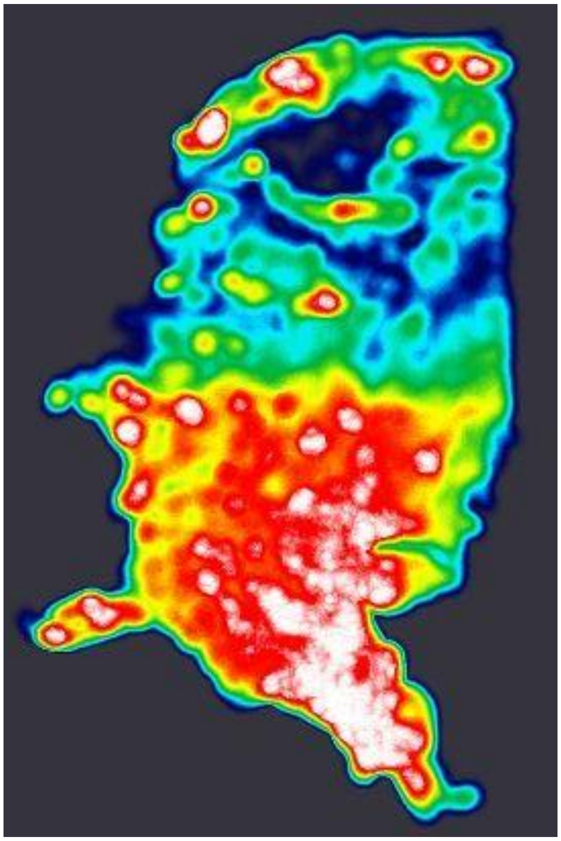


Civil Map

Design Document

TEAM 14

Ai Li Yew

Jing Zheng

Michael Mertz

Weifeng Huang

Qian Zhang

Civil Map

Index

Purpose

The Chicago Police department is interested in understanding trends and making correlations based on the data they have collected over the past few years. Thus, we are developing an application which will conduct data aggregation and analysis on the collected data. Then, providing heat map visualizations that allow comparison of multiple clustering results, this will enable the police department to identify and visualize relationships and patterns more efficiently.

Functional Requirements

1. As a user, I would like to be authenticated before access the dashboard.
2. As a user, I would like to switch between different types and subtypes such as "crime" and "arrest".
3. As a user, I would like to see a dashboard containing crime/arrest subcategories on the main page.
4. As a user, I would like to view the different categories' data as a heat map.
5. As a user, I would like to see crime density from the heat map based on color.
6. As a user, I would like to see crime occurrence pattern from the heat map.
7. As a user, I would like to see the trending of a certain crime type and frequency.
8. As a user, I would like to be able to see different types of points with the longitude, latitude and time.
9. As a user, I would like to the maps to have basic map functionality like Google maps where you can see street addresses, Satellite images, zoom in, zoom out and selectable locations.
10. As a user, I would like a different zoom in view from the zoom out view.
11. As a user, I would like to be able to export the maps I have aggregated in PDF.
12. As a user, I would like to filter crime data by time range(crime occurs within certain days or between certain customized dates)
13. As a user, I would like to filter crime data by area given center and radius or "zoom level".
14. As a user, I would like to have either thematic/reference type of maps as visualization of crime data after filtering.
15. As a user, I would like to be able to do multiple aggregations (filters).
16. As a user, I would like to be able to upload heatmap points, which builds a temporary entity with mapping section id. (section id will contain heat map points for a filter or multi-filter e.g. all the arrest an offender has had, the heat map points could either be from an third party application or

from our own application).

17. As a user, I would like to see the our application's map rendered with the heat map points from the section id (as clarified in user story No.16).
18. As a user, I would like the geocoding engine to check if it is a purified address (eg. 3500 S Michigan is actually 3500 S Michigan Ave) and if this address exists in the database.
19. As a user, I would like the geocoding engine to be able to save the non-purified address as an alias of the address.
20. As a user, I would like to like the purified address and all its aliases to have one id number.
21. As a user, I would like to be able to cache the address if repeated calls are made to the address.
22. As a user, I would like to be able to have a polygon table of all polygon types that exist for Chicago ("neighborhood", "beat").
23. As a user, I would like to be able to have a table of all clean address ID's and the polygons types they are associated with.
24. As a user, I would like to be able to define polygons of different districts such as US postal, zip code, police district etc.
25. As a user, I would like to pre-calculate all the zones that the address(longitude and latitude) belongs to.
26. As a user, I would like to send a list of clean address ID's to the api and have a map created from these points.
27. As a user, I would like to see what polygon contains the uploaded / inputted point/address.

Non-Function Requirements

1. As a developer, I would like the server to be able to deny any unauthorized requests
2. As a developer, I would like the application to be maintained easily
3. As a developer, I would like the framework to be reusable
4. As a developer, I would like the developed APIs to be general to other crime data
5. As a developer, I would like the APIs to be deliverable
6. As a developer, I would like the data to be securely protected
7. As a developer, I would like the application to be hosted on a cloud provider
8. As a developer, I would like the database to be hosted on a cloud provider.
9. As a developer, I would like to be able to extract and read the data provided by the company.
10. As a user, I would like the application to be user-friendly
11. As a user, I would like the application to respond in a timely manner.
12. As a user, I would like the application to be available 99.99% of times that I try to access it.
13. As a user, I would like an easy to use interface.
14. As a user, I would like to have an instruction manual.

15. As a user, I would like the application to return meaningful error message for any inappropriate user action.

- Scalability

  Since we separate frontend and backend, backend will be developed in a way that it provides API for access and thus may be connected and used by platforms other than the websites, such as apps on smartphones, software on PC.
  Because we have full access of our SQL database, we can modify any features to the data classes later easily.

- Usability

  We should develop a friendly interface as frontend as well as eliminate unhelpful features that mostly exist among our competitor applications to make the whole website easy for average users to understand and use.
  Based on the most recent voted predefined map of the users, we will automatically generate a set of predefined maps and quick access of different type of filters, which will make filtering process faster and easier.

- Response time

  The backend system will use object relational mapping classes to manipulate data. We have to decide essential classes to make, which will not hurt the comprehensiveness of our product nor the speed of retrieving data. For retrieving address, the average response time is supposed to be less than 5 seconds even through our database is huge and the response can increase as the database grows.

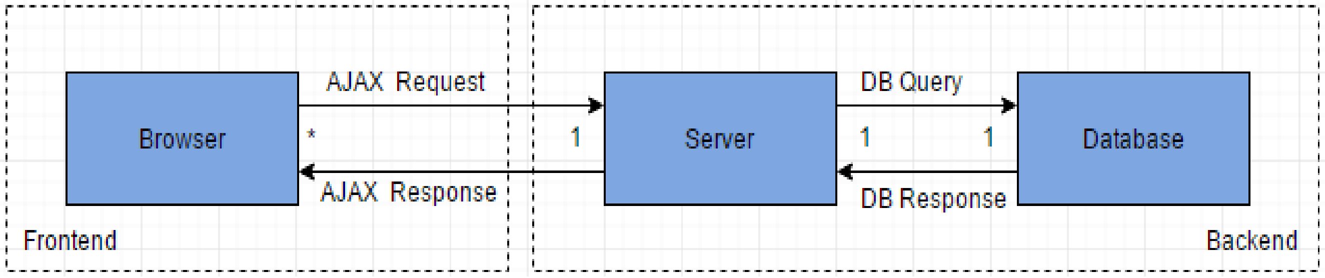
- Security

  We really handles all information carefully since all of the data are very critical and sensitive. Therefore, to use the site, we cooperate with the Chicago Police Department and instead of getting their credentials, we will add our test credentials into the their database to ensure the site is running security without exposing Police Department's credentials.

Design Outline

High Level Overview

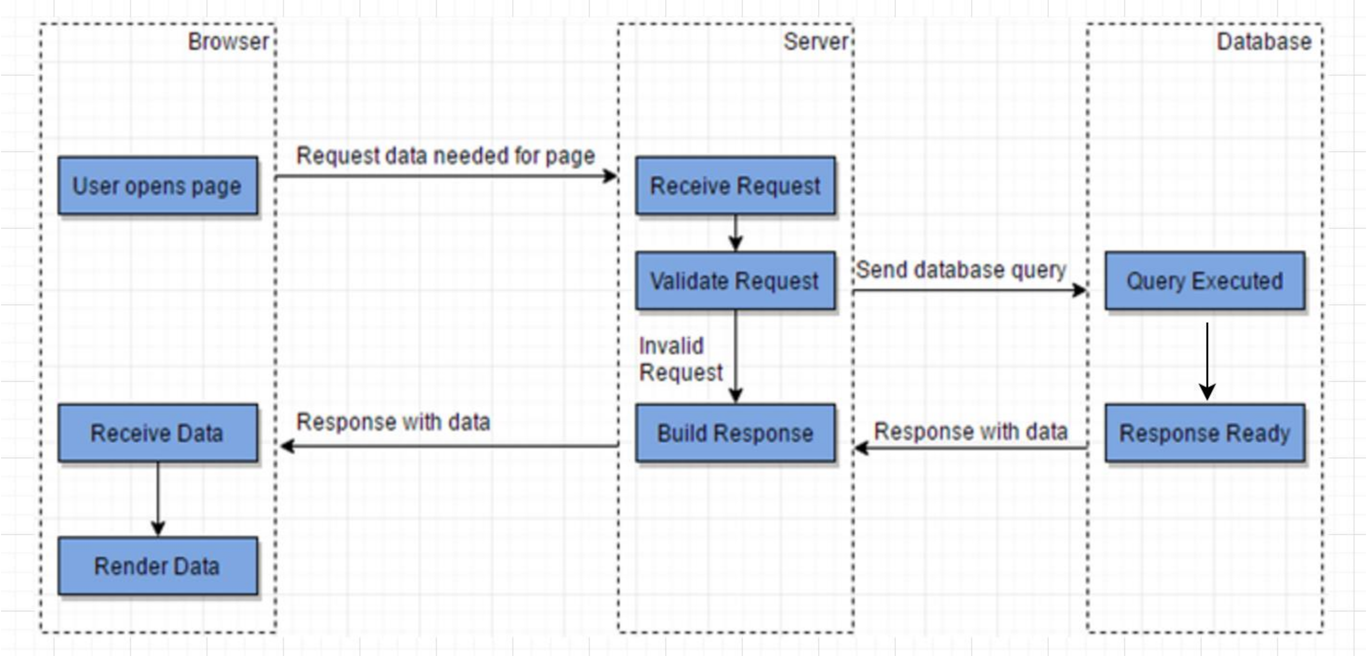
Our project will use the browser-server model. The server will respond to the browser requests, and the browser will parse and render the response data using JavaScript and Map Engine. The figure below demonstrates a high-level overview of the system.



1. Browser
   a. Browser sends AJAX requests to the server.
   b. Browser receives an AJAX response from the server.
   c. Response data is interpreted and rendered to user using JavaScript and Map Engine
2. Server
   a. Receives and handles all client requests
   b. Validates requests before processing
   c. Queries the database and generates appropriate response to send to client
3. Database
   a. Database stores all different type of addresses(such as Zip Code, County, Community, etc..), offenders and users' credentials.

Flow of Events

The diagram below shows a typical flow of events. It starts with a user opening the webpage through the browser. When some events happening on the webpage need to require data from the server, the browser will make a AJAX request to the server. The server will firstly validate the request to make sure the authorization of the requesting user. If the request is valid, the server will query the database and then package the data into a response object to return to the browser. If the request is invalid, without querying the database, an error response package will be sent instead. At last, the browser will parse and render the data onto the webpage for the frontend to render the map.



Design Issues

Functional Issues

1. Does user need to login to use Civil Map?
   - No need to login
   - **Login is required**

   Decision: Civil map is a visualization toolkit designed particularly for Chicago Police Department (CPD) with advanced data aggregation and visualization functionalities. The data deployed to this project is real crime data collected from CPD, and it should be used privately and securely. Login feature guarantees that only entitled person or entity is allowed to access the crime data. To ensure data security, we do not provide account registration, and login will be redirected to CPD website login page. For development purpose, CPD and Clarity will set up temporary accounts for us and remove once the project is done.

2. Shall we provide predefined map when on data filter is deployed?
   - **Yes**
   - No

   Decision: Predefined map is highly preferred when crime data is not filtered. Predefined map provides a big picture of crime occurrence pattern and density over a default time range. The default time range is subject to change based on the feedback from CPD and Clarity. The current default time range is set to past one month. Once user selects one or more data filters, the predefined map is cleared and refined map will be shown on the same window.

3. What is the options for Date Filter?
   - Customized periods
   - Predefined Periods as default
   - **Both**

   Decision: When the date filter is not deployed, it's default option is a predefined period. For now, the predefined time range is past one month, and it is subject to change. When crime data is not filtered by date, date filter's default option, predefined period, is applied. When data is filtered by date, the selected period will be applied. Periods include yesterday, past one week, past one month, past three months, past six months, and past one year. More time ranges may be added in the future based on feedback.

4. Shall we allow user to export aggregated maps as PDF?
    - **Yes**
    - No

    Decision: Civil Map should be developed and used with respect to data privacy and data security. Login feature guarantees that the crime data can only be accessed by entitled person or entity. Along with that, we allow entitled users to export aggregated maps with filtered data as PDF for future reference.

5. How does user select a certain area for crime filtering?
    - Drag a rectangle on map as selected area
    - **Select center and radius**

    Decision: Police Department has a separate system used to subdivide a city into a series of police blocks. And police blocks are predefined in the data collected from CPD. Police blocks include beats, communities, districts and neighborhoods. Each of these can be used as a center and user is free to select an appropriate radius. A randomly dragged rectangle on map is not easy to capture and might result in problems for area filtering.

6. Shall we allow users to apply multiple data filtering on one map?
    - **Yes**
    - No

    Decision: Multi-filtering would be much more useful than a single one. Multi-filtering will also contribute to refining search criteria and focusing on concerned criminal activities. Data filters include race, gender, age, crime type, time range and area. All these filters can be applied together.

7. Shall we allow a zoom in/out on a map view?
    - **Yes**
    - No

    Decision: A zoom in / out feature is prompted on a map view, which allows users to see data features on one map instead of switching between multiple maps. The predefined map scale is on city level. On city level, user will be able to see various crime density and any possible patterns throughout the city. User is also free to zoom in the map to a district level or street level, for anyone would like to focus on crime happened in a particular district or street. User can go back to city level by zooming out.

Non-Functional Issues

1. What should our backend services be implemented with?
    - **Asp.Net**
    - Java
    - Node.js
    - PHP
    - Python

    Decision: All languages listed are languages that are easily deployed within Microsoft Azure. We chose ASP.net due to previous experience within our team, as well as lack of experience with other languages. The company we are working with, Clarity, is also using ASP.net so in the case that we need help or our project needs modifications or maintenance after the semester the company will be able familiar with the technology.

2. How are we going to host our backend services?
    - Amazon Web Services
    - **Microsoft Azure**

    Decision: Both services are easily developed on and can host databases. We chose Microsoft Azure due to the company's previous experience as well as existing developments within the service. They already have an account that we can easily be added to included previously existing databases to assist in development and testing.

3. What database should we use to host our data?
    - **SQL Server**
    - MangoDB (No SQL)

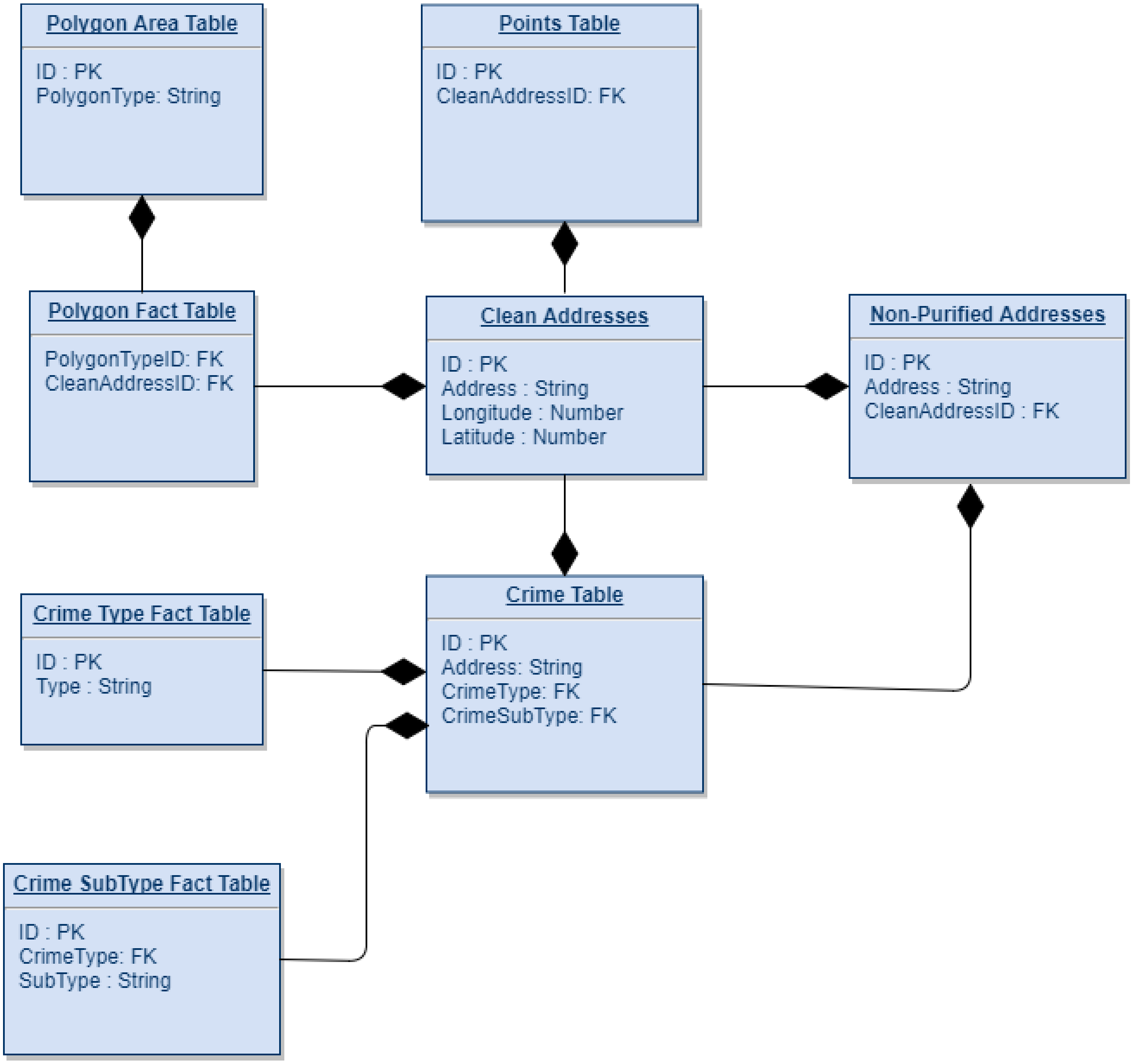
    Decision: Due to working with Microsoft Azure, needing a relational database across a large schema of databases we chose SQL Server. SQL Server is already hosted by Clarity in Microsoft Azure and has been populated with test data, so no major decision making needs to be made.

4. How we use a Geocoding technology to correct the addresses input into the database?
    - Use an open-source Geocoding API
    - Use US Postal Service API
    - Use Google's Geocoding API
    - **Use a caching database**

   Decision: We chose to use a caching database that we would create, initially with Google's Geocoding API. Other API's would work, but Google would be the most uniform model to use, and provides a free number of API hits per month. Initially Google's Geocoding would be hit frequently with "dirty" address that would populate our caching database, but the "clean" or correct addresses would be populated within it's own correct database that the cached database would reference if a user were to input the same "dirty" address.

Design Details

Data Class Level-Design



Description of Data Classes and their Interactions

Our data class design is based on what data and information will be needed on visualization and how they will be represented in relational database. Each box in the data class design will represent a model in database. And there will be one table for each model. There are several tables already existed in CPD system, we are supposed to deploy them and set up interactions between them by adding new tables.

Crime Type Fact Table

- Indicate the type of a point
- A point is a case handled by CPD
- The type of a point can be crime or arrest

Crime Subtype Fact Table

- Indicate the subtype of a point type (crime, arrest)
- The subtype of crime includes burglary, robbery, shooting .etc
- The subtype of arrest includes sex offender, drug abuse .etc

Crime Table

- Represent detailed information regarding a point - default crime data we will work on
- The crime is identified by ID
- Address indicates where the crime is occurred. Address is obtained from clean address table
- Crime type indicates the type of the case, including crime and arrest. Crime type is collected by joining crime table with crime type fact table
- Crime subtype indicates the subtype of a case type. Crime subtype is collected by joining crime table with crime subtype fact table

Polygon Area Table

- Represents all types of polygons defined by CPD
- Each polygon type is identified by polygon ID
- Polygon type includes police beats, districts, and neighborhoods

Polygon Fact Table

- This table is used to map each clean address with police defined polygon
- Record in this table is identified by clean address ID
- Each clean address can belong to one of police beat, district and neighborhood

Clean Address Table

- Stores purified address returned from geocoding API
- Each clean address is identified by address ID
- The longitude and latitude of the clean address are also stored in this table. Longitude and latitude are used for area selection

Points Table

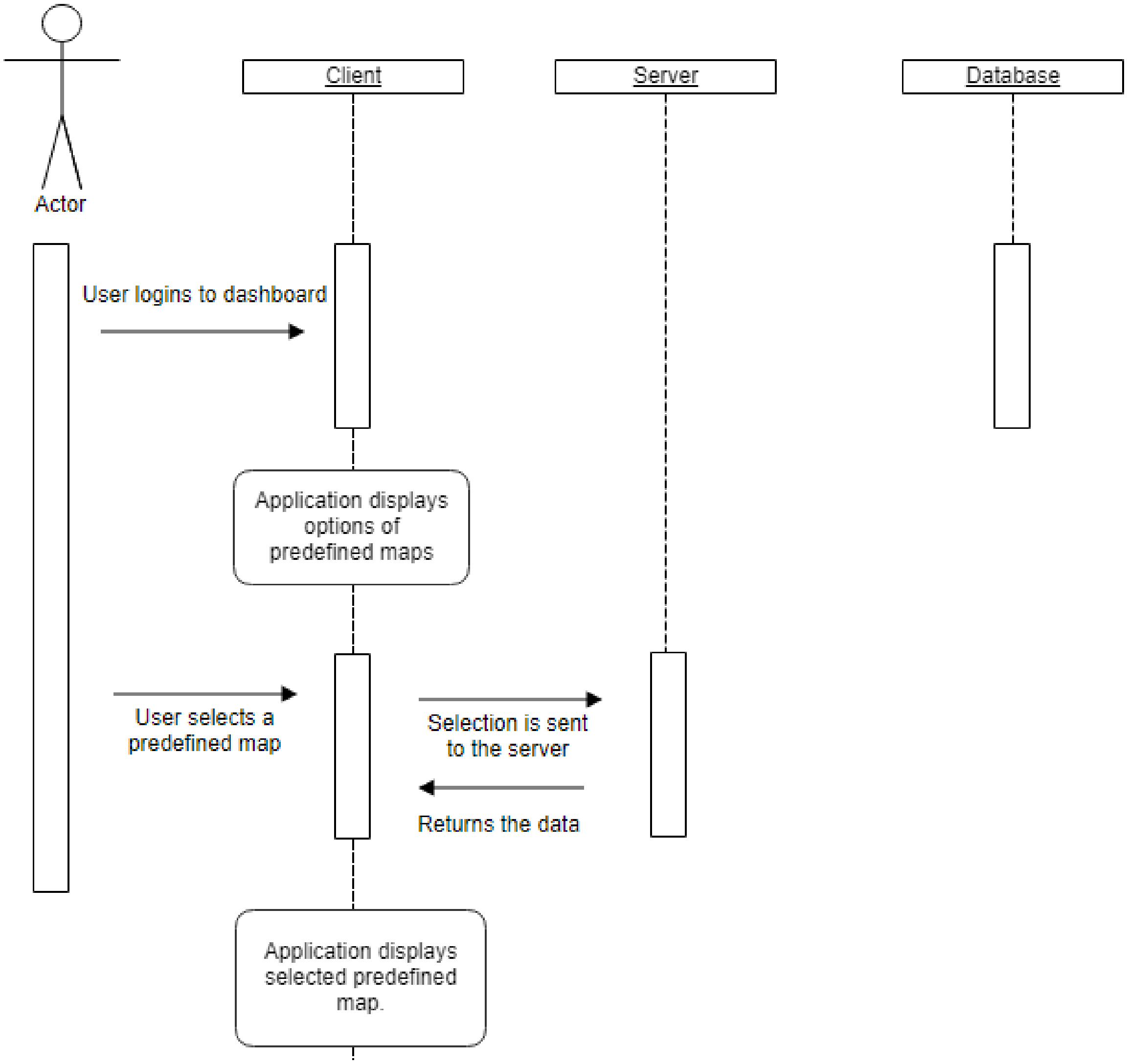
- Represents all points reported to and handled by CPD
- This table is designed for data aggregation APIs for third party company
- Each point is identified by point ID
- The table stores the location of the point. Multiple points can happen in the same address
- Point address is collected from clean address table

Non-Purified Address

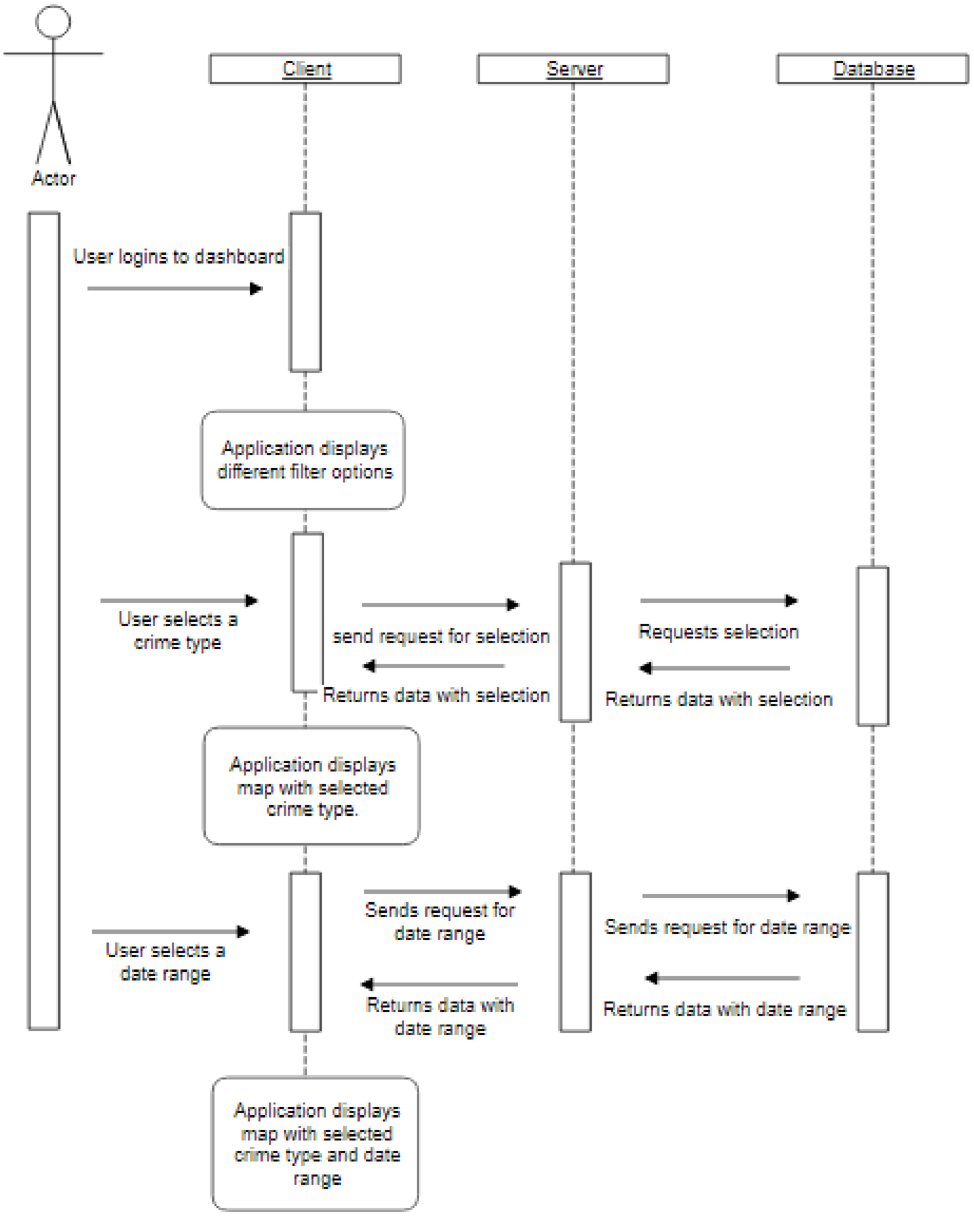
- Stores non-purified address and its corresponding clean address return by geocoding API
- Each record is identified by non-purified address ID
- Corresponding clean address can be collected by joining non-purified address table with clean address table

Sequence Diagrams

Sequence of events when a user select a predefined map with authorization

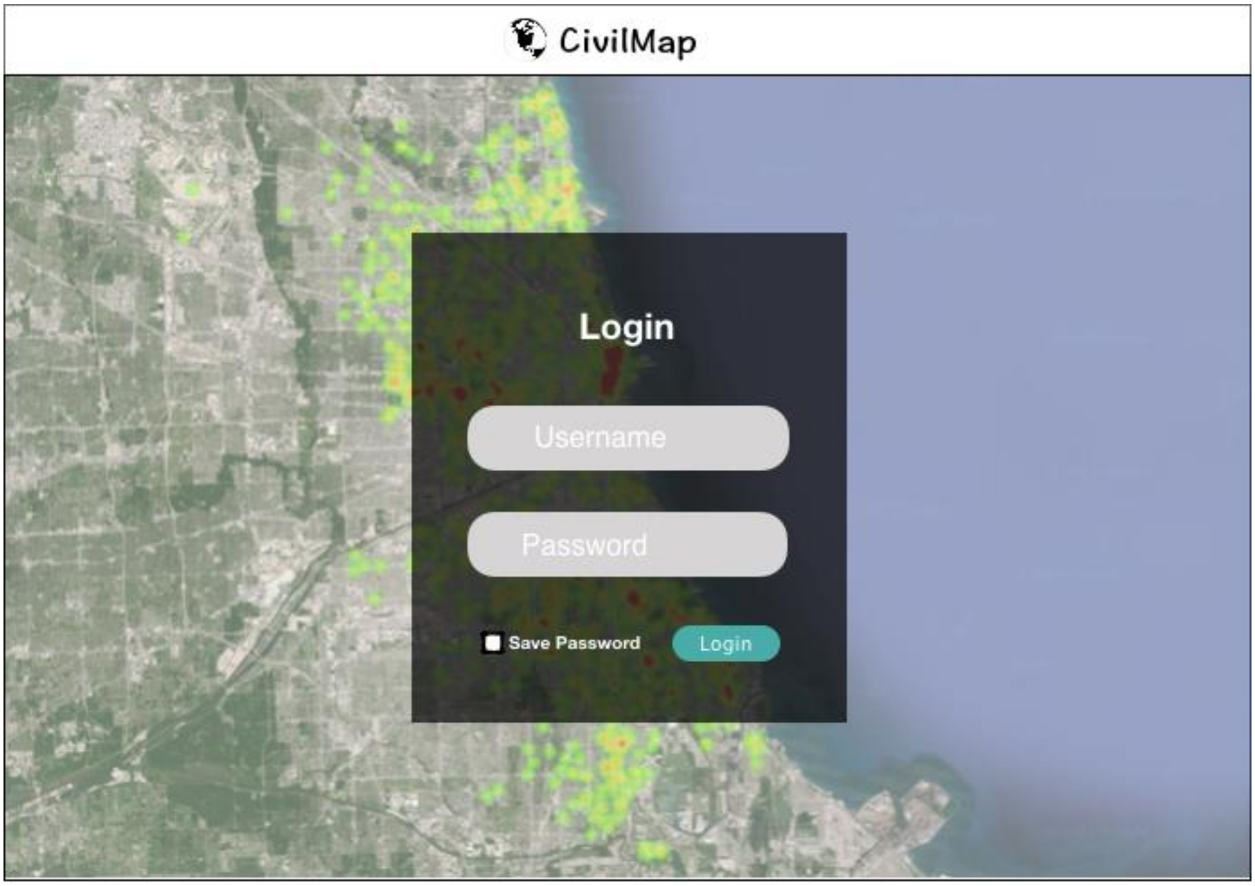

Actor
Client
Server
Database
User logins to dashboard
Application displays options of predefined maps
User selects a predefined map
Selection is sent to the server
Returns the data
Application displays selected predefined map.

Sequence of events when a user select crime type and a data range (advanced filtering)


Actor
Client
Server
Database
User logins to dashboard
Application displays different filter options
User selects a crime type
send request for selection
Requests selection
Returns data with selection
Returns data with selection
Application displays map with selected crime type.
User selects a date range
Sends request for date range
Sends request for date range
Returns data with date range
Returns data with date range
Application displays map with selected crime type and date range

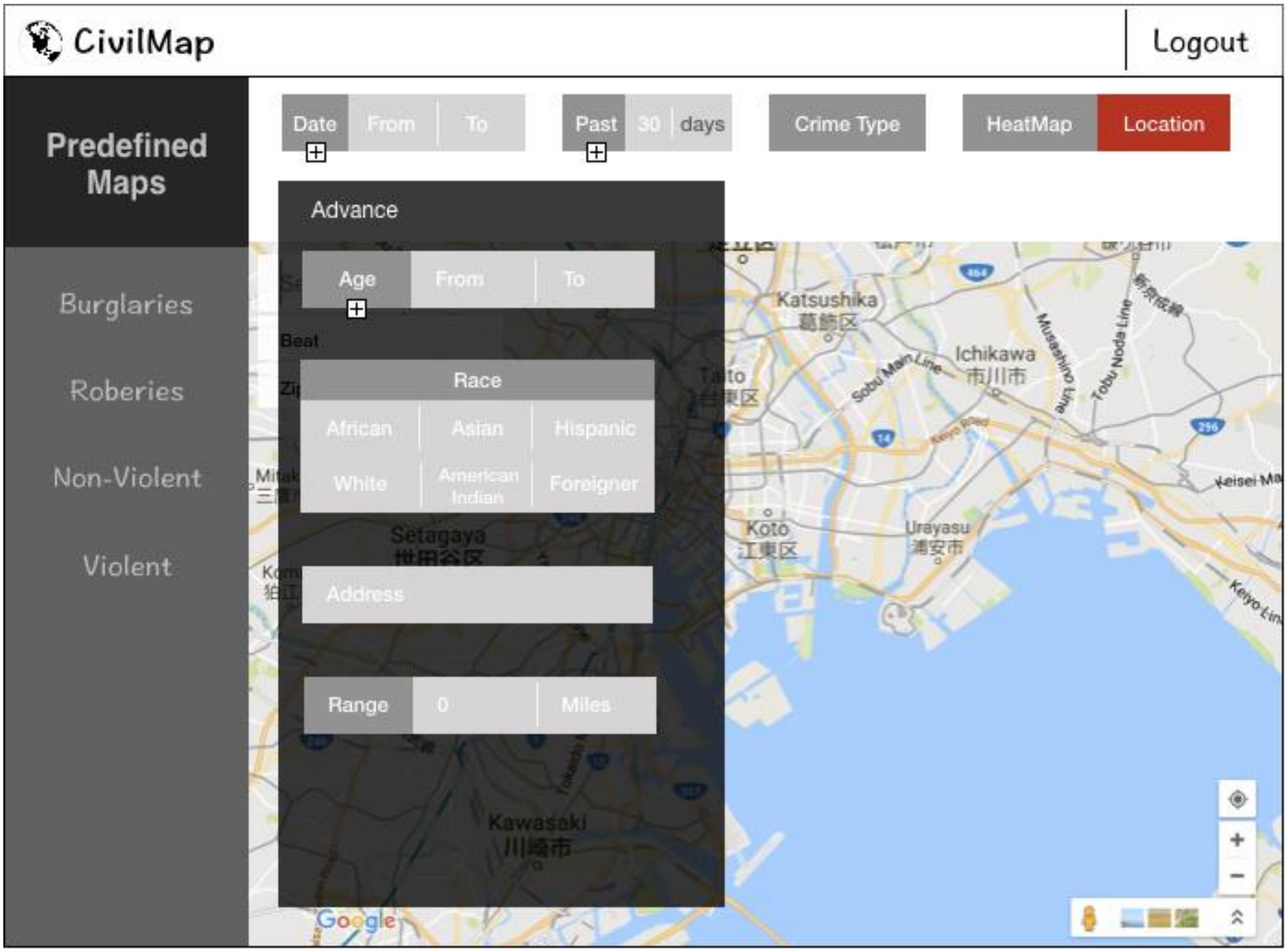
UI Mockups

Login



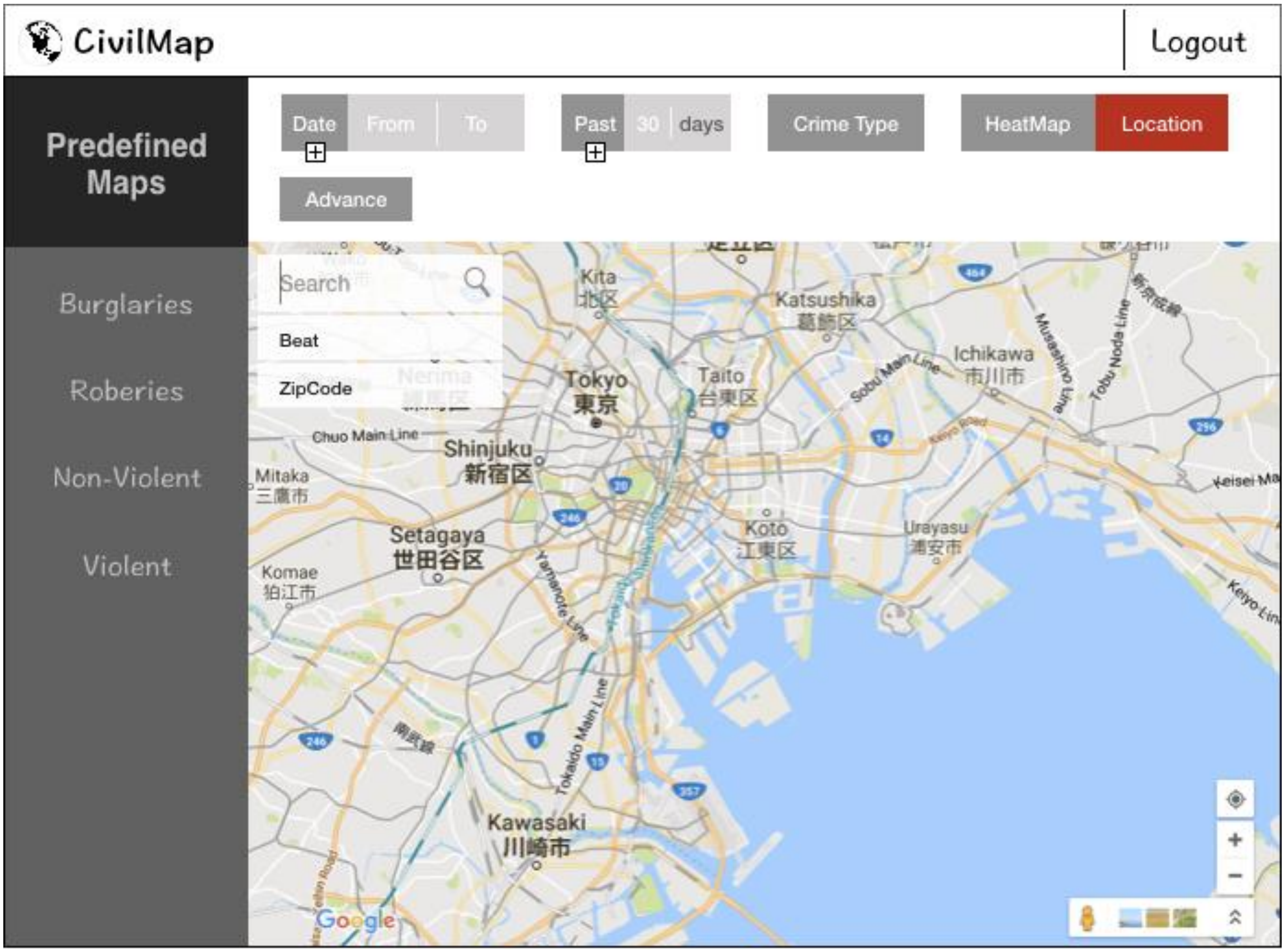
- No Access of any data will be given before authentication

Main Dashboard



- with predefined map options and advanced filtering features

Quick Search Bar



- with auto completed and quick sort functions to better assist the user with their search experience