**BIG DATA PROJECT**

**Overview**

This project involves extracting and uploading a large dataset of incident reports from the City of New York's open data portal to an Elasticsearch instance for further analysis and visualization using OpenSearch Dashboards. The goal of the project is to fetch around 2 million records from the dataset. By utilizing Elasticsearch and OpenSearch Dashboards, the project aims to provide insights into various metrics such as response times and incident distribution across different boroughs of NYC.

**Components**

1. **Data Extraction Script**: A Python script using the Socrata API and requests library to fetch data
2. **Docker Configuration**: Docker to run the Python script in a containerized environment
3. **Elasticsearch**: For storing and indexing the data
4. **OpenSearch Dashboards**: For data visualization

**Setup**

### **Environment Variables**

The script requires several environment variables for configuration. Ensure the following variables are set in your environment:

* DATASET\_ID: The dataset ID from the City of New York's open data portal (e.g., 8m42-w767 for this website).
* APP\_TOKEN: Your Socrata API token.
* ES\_HOST: URL of your Elasticsearch instance.
* ES\_USERNAME: Elasticsearch username.
* ES\_PASSWORD: Elasticsearch password.
* INDEX\_NAME: The name of the Elasticsearch index where the data will be stored.

**Docker Configuration**

A Dockerfile is provided to create a containerized environment for running the data extraction and upload script.

## 

## **Python Script**

The script data\_upload.py is designed to fetch incident report data from the NYC open data portal and upload it to an Elasticsearch index.

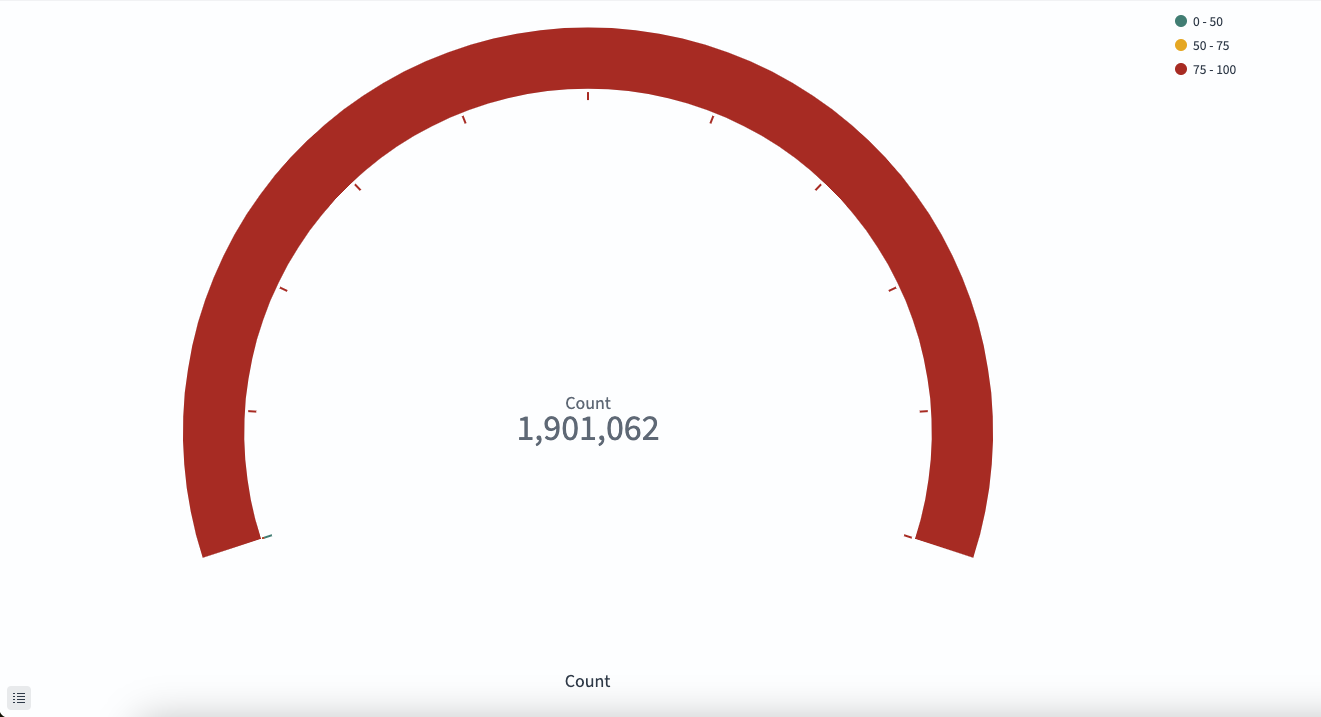
### **Arguments**

* --page\_size: (Required) Number of rows to fetch per page.
* --num\_pages: (Optional) Total number of pages to fetch. If not provided, the script fetches all available data.
* --start\_page: (Optional) The page number to start fetching data from. Useful for resuming data fetches.

**Docker Run Example**

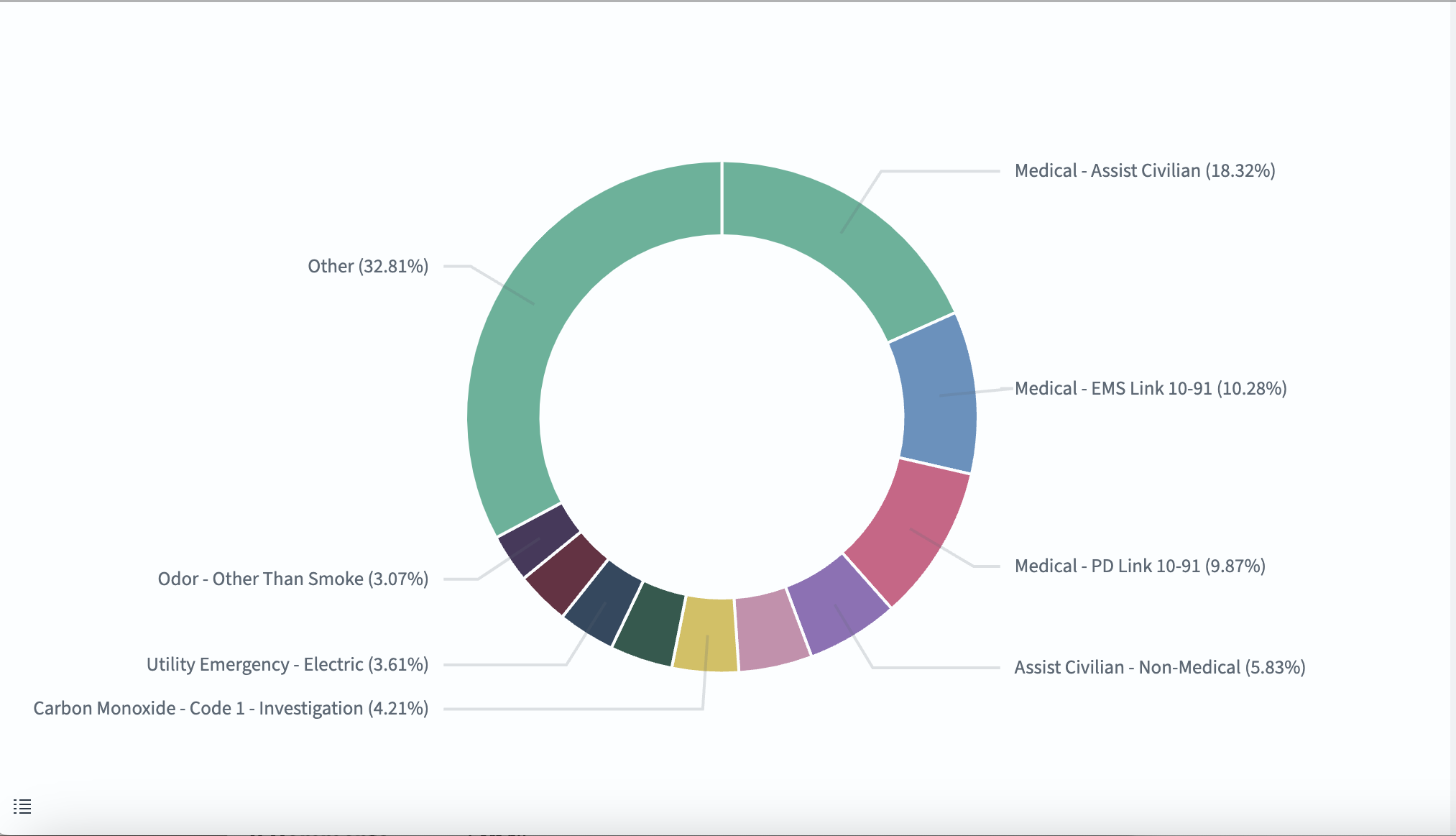
docker run -e DATASET\_ID="8m42-w767" -e APP\_TOKEN="krMesDIBHUwoHj7jDY0S9z197" -e ES\_HOST="https://search-project-fire-grixv2gudd2dlzuuhsnq5eoivq.us-east-2.es.amazonaws.com" -e ES\_USERNAME="projectfire" -e ES\_PASSWORD="290296Aws@" -e INDEX\_NAME="fire" bigdataproject1:1.0 --page\_size=100 --num\_pages 1 --start\_page 1

**Data Visualization**



Total 1,901,062 data fetch from the website [**NYC Open Data**](https://opendata.cityofnewyork.us/).

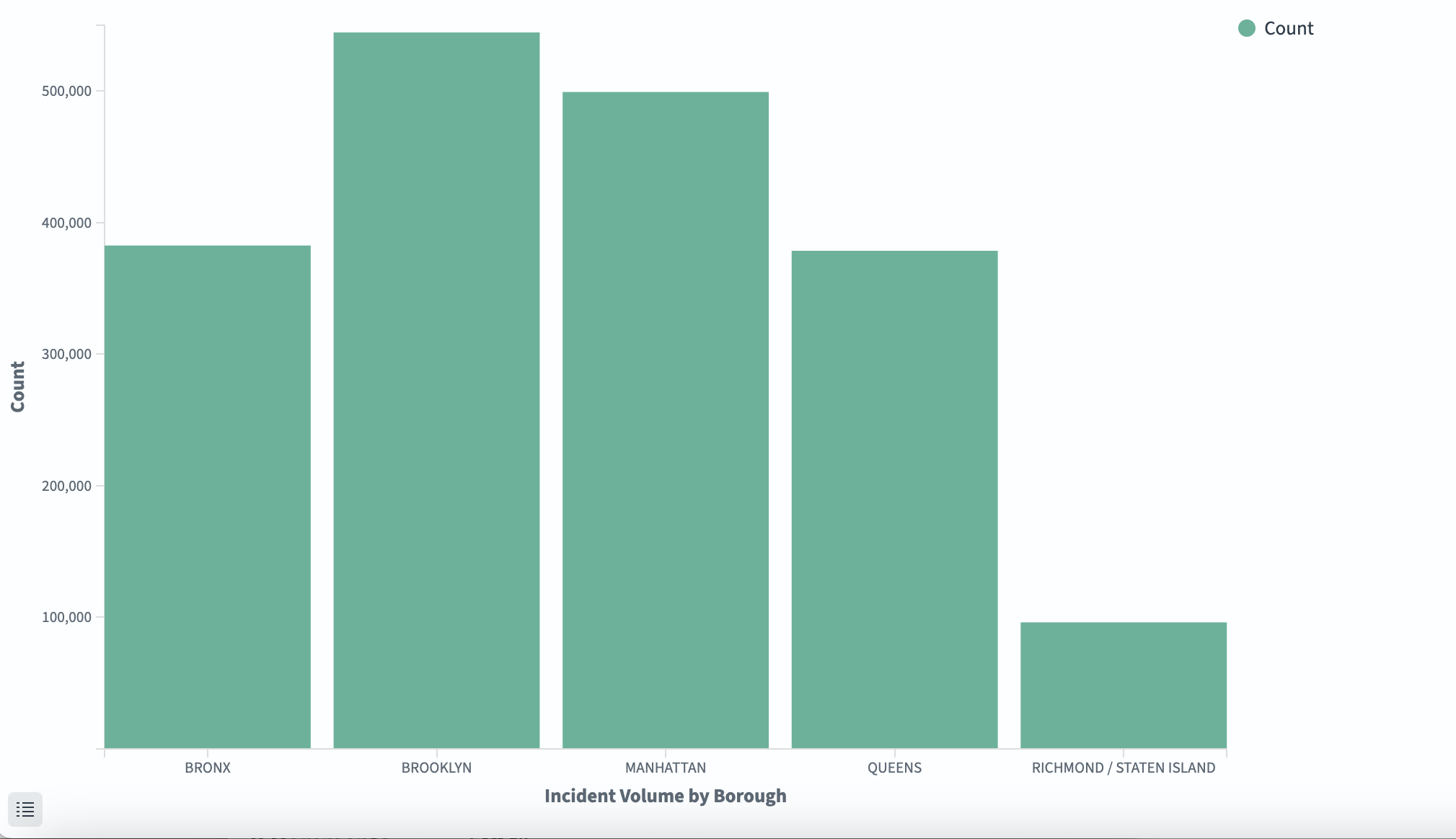
**Visualization 1: Incident Distribution by Classification**



This visualization represents the distribution of various incident classifications within the dataset. It is a donut chart that highlights the percentage share of each type of incident, allowing us to easily understand the most common types of incidents reported.

The largest category involves assisting civilians in medical emergencies, accounting for 18.32% of the incidents. This is followed by Emergency Medical Services (EMS), which makes up 10.28% of the total incidents, and the Police Department (PD) link, accounting for 9.87% of the total incidents. Observing the top incident categories, medical services were the most frequently requested.

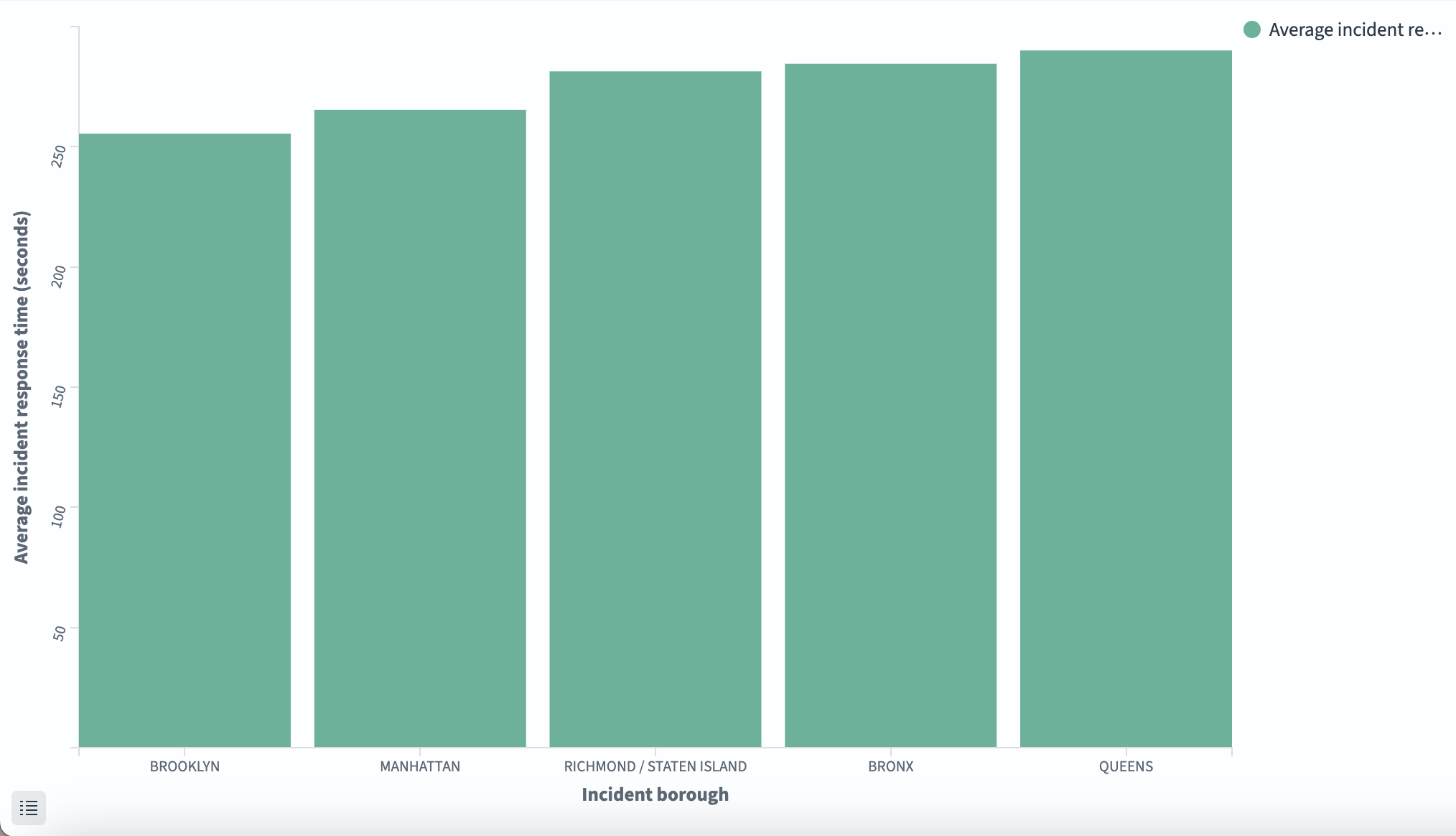
**Visualization 2: Incident Volume by Borough Visualization**

****

The bar chart displayed above illustrates the volume of incidents reported in each borough of New York City. Here's an analysis based on the visualization:

1. Brooklyn: This borough shows the highest number of incidents, exceeding 500,000. It suggests that Brooklyn experiences a higher frequency of reported incidents compared to other boroughs.
2. Manhattan: Close to Brooklyn, Manhattan also exhibits a high number of incidents, slightly below Brooklyn. This indicates a high demand for incident response services in this borough as well.
3. Queens: Queens ranks third in terms of incident volume, with counts significantly higher than the Bronx and Staten Island but lower than Brooklyn and Manhattan.
4. Bronx: The Bronx has a moderate number of incidents, with counts falling between those of Queens and Staten Island.
5. Richmond/Staten Island: This borough reports the lowest number of incidents, significantly less than the other four boroughs.

**Visualization 3: Average Incident Response Time per Borough**



The bar chart above represents the average incident response time in seconds for each borough of New York City. Here are the key insights:

1. Brooklyn: Brooklyn has the shortest average response time, around 250 seconds. This suggests that emergency services in Brooklyn are relatively faster compared to other boroughs.
2. Manhattan: The response time in Manhattan is slightly higher than in Brooklyn, indicating a slight delay in response times.
3. Richmond/Staten Island: The response time in Richmond/Staten Island is similar to Manhattan, showing that the emergency response is consistent across these two boroughs.
4. Bronx: The Bronx shows a higher average response time, indicating that it takes longer for emergency services to respond to incidents compared to Brooklyn and Manhattan.
5. Queens: Queens has the highest average response time, nearing 280 seconds. This suggests that there might be logistical or infrastructural challenges affecting the speed of emergency responses in Queens.

**Visualization 4: Average Incident Response Time per Incident Classification**



This bar chart presents the average incident response time in seconds for various incident classifications. Here are the key observations:

1. Utility Emergencies: Incidents related to utility emergencies, encompassing water, undefined, steam, gas, and electric issues, exhibit the longest average response times. This indicates potential challenges in addressing these types of incidents.
2. Undefined Non-Structural Fires: Fires categorized as undefined non-structural incidents have the shortest average response times. This suggests efficient handling of this type of fire.
3. Vehicle Accidents: Incidents involving vehicle accidents, particularly those with extrication, fall somewhere between these two extremes in terms of average response time.

**Conclusion**

This project showcases the extraction, transformation, and loading of large datasets into Elasticsearch for analysis and visualization. By leveraging OpenSearch Dashboards, you can gain valuable insights and make data-driven decisions based on incident report data.