
# NYC Parking Violations Retrieval System

ANALYZE PARKING VIOLATIONS TO MAKE NYC BETTER!

APAN5400
Peiqi Tan
Zeyi Bai
Jiarong Guo
Yuxin Li

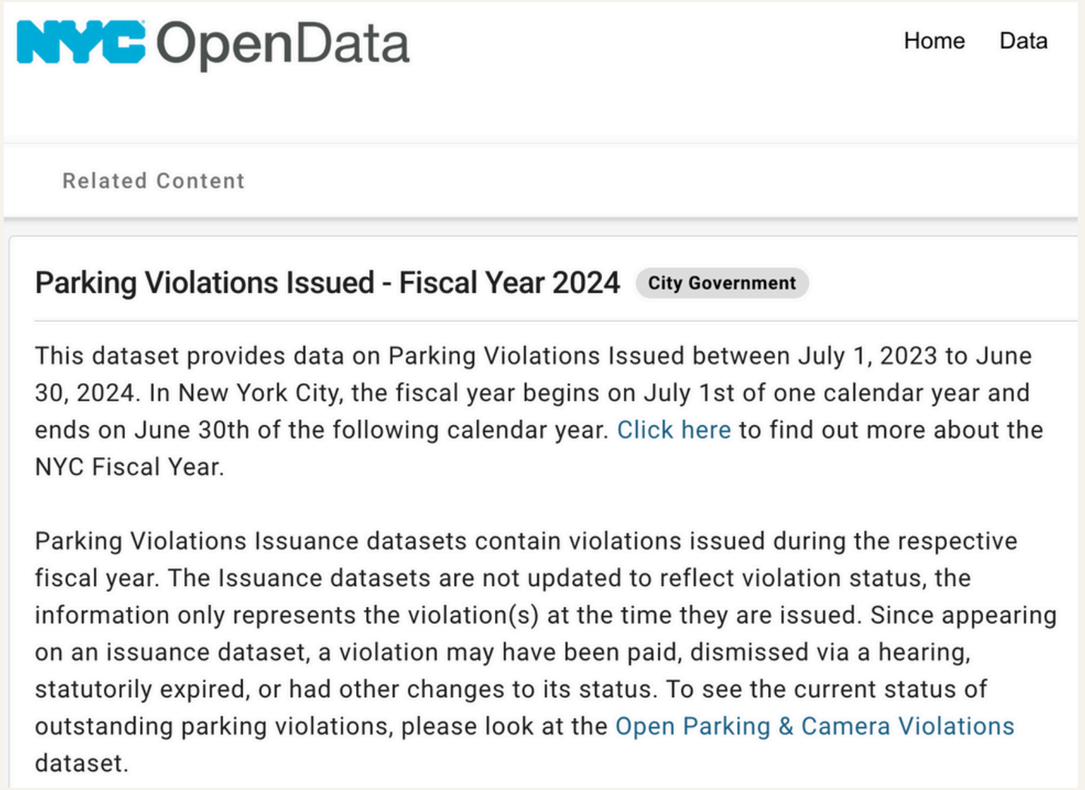

# Background & Definition of Use

Our project aims to build **a data-driven platform that enhances NYC's parking environment by providing insights from historical parking violations data.** The system will enable NYC government officials to search for violation records by input criteria (e.g., street code), presenting the data in user-friendly visualizations that reveal trends and patterns.

By using this information, municipal departments can **gain a better understanding of parking enforcement issues, pinpoint areas with high violation rates, and make more informed decisions to improve traffic flow and overall urban mobility.**


# Data Source Specification

Sourced from *NYC Open Data*[1], this dataset includes records of parking violations in New York City July 1, 2023 to June 30, 2024. It includes columns such as Issue Date, Violation Code, Street Code, Vehicle Type, and Fine Amount. The dataset captures each violation at the time it was issued, providing a snapshot of ticketing conditions and trends across the city.




1: https://data.cityofnewyork.us/City-Government/Parking-Violations-Issued-Fiscal-Year-2024/pvqr-7yc4/about_data


# Proposed Presentation Dashboard

**Object:** Build an information retrieval system with a front-end UI that allows users to input criteria (e.g., street name, issue precinct, violation type) and view visualizations (bar chart of violations by month, pie chart of violation types, pie chart of violations by county). The dashboard will initially show three visualizations for 2024.

**Technologies:**

- API: Use the NYC OpenData API to retrieve data.
- Spark: Use Apache Spark for distributed data processing and aggregation.
- MongoDB: Store the processed data in MongoDB (NoSQL).
- Additional Technologies: Use FastAPI for the back-end API, PostgreSQL (optional SQL storage), and a front-end framework like Streamlit or Dash for the UI and visualizations.


# System Archetecture

**Data Extraction (Extract):**
Fetch data from the NYC OpenData API, then use pagination to retrieve all records.

**Data Processing (Transform):**
Use Apache Spark to process and aggregate the data (e.g., group by month, violation type, county). And prepare the data for the default 2024 visualizations and user-driven queries.

**Data Storage (Load):**
Store the raw and aggregated data in MongoDB. Optionally, store aggregated data in PostgreSQL for faster SQL-based queries (if needed).

**Back-End API:**
Use FastAPI to create endpoints that query MongoDB (or PostgreSQL) based on user inputs and return data for visualizations.

**Front-End UI:**
Use Streamlit (a Python-based framework) to build the UI and display visualizations (bar charts and pie charts) using Plotly


# Scalability and Cost Implications

- **Data Governance & Policies:** The data is publicly available on NYC Open Data, so there will be no personally identifiable information or cost implications under the current usage.
- **Scalability Path:** As usage grows, we can scale to cloud to support larger data volume and faster processing speed. Key components such as Apache Spark, MongoDB, and FastAPI are well-suited for scalable deployment on platforms like AWS or GCP.
- **Preliminary Projected Cost:**
  - No licensing costs for developer-tier Spark, MongoDB, or FastAPI.
  - Estimated monthly cloud deployment cost:
    - MongoDB Atlas (managed NoSQL database): ~$65/month
    - Apache Spark cluster for data processing: ~$250–300/month
    - Front-end hosting and backend API (FastAPI + Streamlit): ~$30–50/month
    - Total estimated cost: ~$350–$415/month
- **Partnership:** Integrate our system with existing databases/systems of NYPD or NYC Department of Transportation (NYCDOT) for real-time analysis and result optimization, which leads to efficiency and cost reduction.


# Evaluation Criteria

## Quantitative Metrics

Data Processing Performance

- ETL latency: <5 minutes per 100k records from data extraction to storage completion.
- Query response time: Avg. <2 seconds on dashboard.
- API reliability: no major system failures during testing.

Data Completeness & Accuracy

- >98% of violations retrieved and processed v.s. total available.
- Aggregation accuracy validation via test subsets.

## Qualitative Metrics

User Experience & Usability

- Intuitive dashboard design evaluated via user surveys.
- Clear visualization of trends .

Scalability & Flexibility

- Ability to integrate additional datasets (e.g., 2025 data, other boroughs).
- Seamless performance on larger data loads using Apache Spark and MongoDB.

## Stakeholder Value

Policy Relevance

- Insights applicable to urban planning and traffic enforcement strategy.

Reusability

- Modular ETL design and API endpoints reusable for other city datasets or analytics tools.


# Thank You!

APAN5400
Peiqi Tan
Zeyi Bai
Jiarong Guo
Yuxin Li