**PROG8850 – Assignment 3 Report – NYC 311 ETL + Flask Web App**

**Zafar Ahmed Shaik**

# Project Overview

This project demonstrates an end-to-end data pipeline using Docker containers. Data from the NYC 311 Service Requests (January 2023) dataset is extracted, cleaned, and loaded into a MySQL database through a Python ETL script. A Flask web application provides a search and summary interface, and Selenium tests ensure that the pages render correctly.

# ETL Summary

* ETL Script: etl/etl.py
* Dataset: NYC 311 Service Requests – January 2023
* Rows Loaded: ~242,173
* Transformations: Cleaned missing values, converted date and numeric columns
* Indexes Created:
* idx\_borough on borough
* idx\_created\_date on created\_date

These indexes optimize queries filtering by borough or date.

# Database Verification

Verified schema and table creation using SHOW TABLES.

Confirmed data load with SELECT COUNT(\*) FROM service\_requests.

Verified indexes with SHOW INDEXES FROM service\_requests.

Confirmed query optimization using EXPLAIN SELECT ... showing 'Using index condition'.

# Flask Web Application

The Flask app provides two main routes:

* /search — lets users filter complaints by borough and date.
* /summary — shows complaint counts per borough using a bar chart.Both pages run successfully in Docker and respond to queries.

# Testing and CI/CD

Selenium tests (tests/test\_app.py) verify that the search and summary pages load correctly.

All tests passed (pytest -v shows 2 passed).

GitHub Actions CI workflow (.github/workflows/ci.yml) automatically runs ETL and tests on each push.

# Scaling Notes

To scale for larger datasets:

* Use chunked ETL loading (already implemented).
* Add pagination in Flask search results.
* Consider table partitioning by month for faster queries.
* Use cloud-hosted MySQL for larger data volumes.

# Reflection

Through this project, I learned how to:

* Build a containerized ETL pipeline with Docker and MySQL.
* Integrate Flask for web data visualization.
* Use SQL indexes to improve query performance.
* Automate testing using Selenium and GitHub Actions.

# Screenshots

1. Docker containers running (docker compose ps)
2. Row count (SELECT COUNT(\*))
3. SHOW INDEXES output
4. EXPLAIN SELECT query result
5. Flask homepage
6. Search page
7. Summary page

A black screen with white text

AI-generated content may be incorrect.

A computer screen with white text

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

A black screen with white text

AI-generated content may be incorrect.

A white background with black border

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.