1. Project Title: Traffic Management System
Subtitle : Leveraging Real-Time Data to Optimize Traffic Flow and Avoid Accidents

2. Contact Information

Maryam Kurbanova
Role: Project Lead (solo)
Email: kurbanovamaryam1@gmail.com **Phone:** 347-930-7130
Supervisor: Lola Valente, Supervisor, lola.valente@girlswhocode.com

3. Abstract

The Traffic Management System aims to analyze and indicate potential traffic patterns, congestion times, and accident hotspots using real-time data from various sources, such as city cameras and GPS. By making machine learning models, the system will try to optimize traffic light timings and suggest alternative routes for commuters to enhance traffic flow. Creating the system includes collecting real-time data and training machine learning models in order to improve NYC'S transportation system and help prevent accidents.

4. Tools List

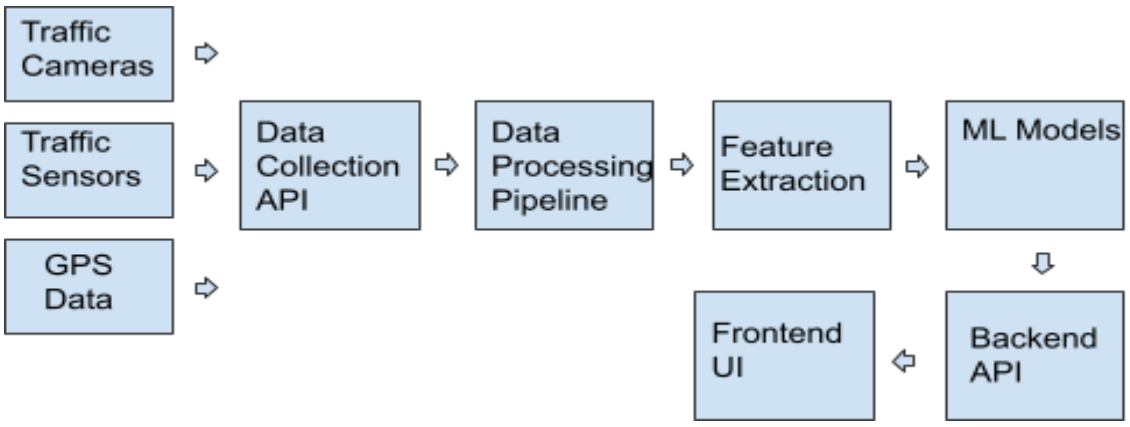
Software:

- Python (for machine learning and data processing)
- Pandas (data manipulation)
- NumPy (data manipulation)
- Scikit-learn (traffic pattern predictions)
- TensorFlow/PyTorch (deep learning)
- OpenCV (image/video data processing)
- Google Maps API (real-time mapping and routing)

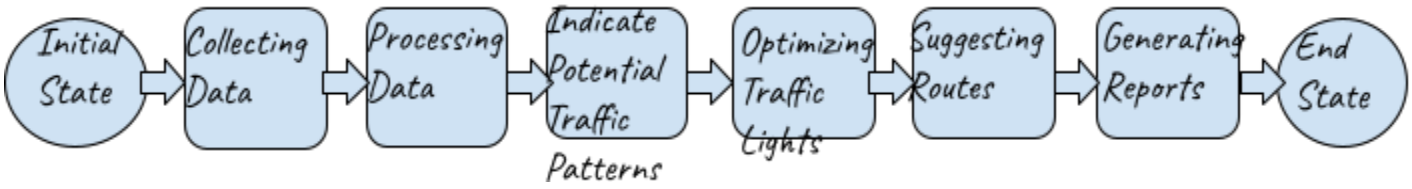
Hardware: VS Code/GitHub (IDE)

5. Diagrams/Designs

Architecture Design



State Diagram



6. Tentative Schedule

Week 1-2: Project Research (10 hours)

- Define project requirements, research data sources, and create diagrams.

Week 3-4: Data Collection (20 hours)

- Collect and preprocess traffic data from city cameras and GPS.

Week 5-6: Model Development and Training (20 hours)

- Train machine learning models to indicate potential traffic patterns and congestion analysis.

Week 7-8: Backend Development (20 hours)

- Implement backend services for data processing and real-time analysis.

Week 9-10: Frontend Development (20 hours)

- Develop a user interface for visualizing traffic data, potential traffic patterns and suggestions.

Week 11-12: Integration and Testing (10 hours)

- Integrate the frontend and backend components, and perform comprehensive testing of the system.

Week 13: Improvements and Finalization (5 hours)

- Improve the system based on feedback.

7. Data Sources

- **NYC Open Data:** Traffic related data. Reports on car accidents.
- **New York City Area Traffic** https://511ny.org/region/New%20York%20City%20Area
- **Google Maps API:** Traffic conditions and route information.

8. Use Cases

Use Case 1: Traffic Light Optimization

- The system analyzes real time traffic data to optimize traffic light timings. As a result it adjusts traffic light timings to reduce traffic.

Use Case 2: Indicates Potential Congestion Areas and Gives a Different Route

- The system comes up with potential congestion patterns and suggests different routes based on real-time and historical traffic data. And gives suggested routes in order to reduce the commuters travel and avoid traffic.

Use Case 3: Accident Hotspot Identification

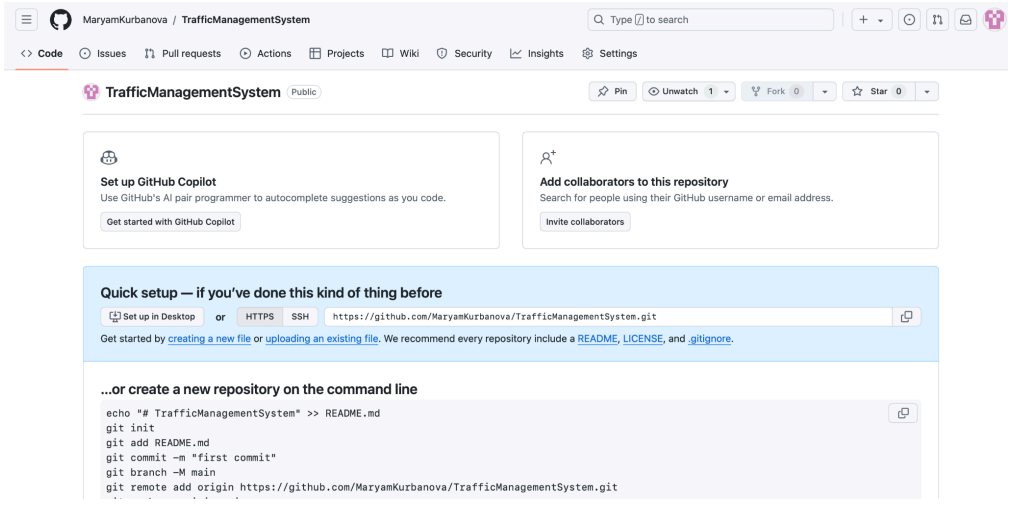
- The system identifies potential accident hotspots based on traffic data and accident reports. And Identifies hotspots of where accidents are likely to occur with recommendations for a different route. There's an accident hotspot next to my apartment. A couple cars have crashed in the same area in the past few years. If commuters knew about this they would drive with caution when approaching this area.

UPDATES

Github link: https://github.com/MaryamKurbanova/TrafficManagementSystem

-The reason that there are no files so far is because the files that I'm trying to push are too big. I'm currently trying to congest the data to make it smaller.

Trello link: https://trello.com/invite/b/66f04c31046be12b12233feb/ATTI46b6d70596752b6c1a103d0e964ed415C9D6861D/my-trello-board



~/TrafficManagementSystem/traffic.py

traffic.py 1

Users > maryamkurbanova > TrafficManagementSystem > traffic.py > ...

```python
import pandas as pd
import matplotlib.pyplot as plt
import gzip

# Load datasets from compressed files
with gzip.open('data/Automated_Traffic_Volume_Counts_20240922.csv.gz', 'rt') as f:
    traffic_volume = pd.read_csv(f)

with gzip.open('data/NYCOpenDataMotorVehicleCollisionsCrashes.csv.gz', 'rt') as f:
    collisions = pd.read_csv(f, low_memory=False)

with gzip.open('data/NYCOpenDataFHVBaseAggregateReport.csv.gz', 'rt') as f:
    fhv_data = pd.read_csv(f)

with gzip.open('data/NYCOpenDataVZVLeadingPedestrianIntervalSignals.csv.gz', 'rt') as f:
    pedestrian_signals = pd.read_csv(f)

# Read the compressed USAccidents file
with gzip.open('data/USAccidents.csv.gz', 'rt') as f:
    us_accidents = pd.read_csv(f)

print("Traffic Volume Columns:", traffic_volume.columns)

# Function to handle NaN values in the traffic volume data
def handle_nans(dataframe):
    # Prints the number of NaN values in each column
    print("NaN values in each column:\n", dataframe.isnull().sum())

    # Replaces NaN values in the 'Vol' column with 0, since this column represents traffic volume
    dataframe['Vol'] = dataframe['Vol'].fillna(0)

# Analysis function
def analyze_data():
    # Handles NaN values in traffic volume data
    handle_nans(traffic_volume)
```

```
(myenv) maryamkurbanova@Maryams-Air TrafficManagementSystem % ls
README.md       data            myenv           traffic.py
(myenv) maryamkurbanova@Maryams-Air TrafficManagementSystem % cd data
(myenv) maryamkurbanova@Maryams-Air data % ls
Automated_Traffic_Volume_Counts_20240922.csv.gz
NYCOpenDataFHVBaseAggregateReport.csv.gz
NYCOpenDataMotorVehicleCollisionsCrashes.csv.gz
NYCOpenDataVZVLeadingPedestrianIntervalSignals.csv.gz
USAccidents.csv.gz
(myenv) maryamkurbanova@Maryams-Air data %
```