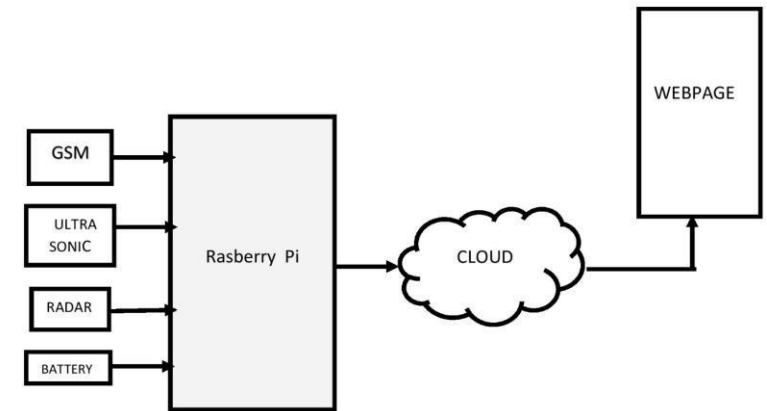


# PUBLIC TRANSPORTATION OPTIMIZATION

The project will provide a valuable service to both public transportation administrators and riders. It can lead to more efficient and user-friendly transportation systems while also offering opportunities for data analysis and optimization.



# COMPONENTS

## HARDWARE

- Raspberry Pi board (e.g., Raspberry Pi 3 or 4).
- GPS module (e.g., NEO-6M GPS module).
- -Mobile data modem (optional for internet connectivity).
- Power source for Raspberry Pi.
- Vehicle with Raspberry Pi mounted.

## • SOFTWARE

- Python
- Database system
- Location tracking libraries
- Web frameworks and libraries
- Data analysis libraries
- Machine learning
- Security and privacy tools

# PYTHON PROGRAM

```
import gpsd

# Connect to the SQLite database
db_connection = sqlite3.connect('riders.db')
db_cursor = db_connection.cursor()

# Create a table for rider data
db_cursor.execute("""
    CREATE TABLE IF NOT EXISTS riders (
        id INTEGER PRIMARY KEY,
        name TEXT,
        location TEXT,
        timestamp DATETIME DEFAULT CURRENT_TIMESTAMP
    )
""")
db_connection.commit()

while True:
    try:
```

```
        packet = gpsd.get_current()
        if packet.mode >= 2: # Check if GPS has a fix
            latitude = packet.lat
            longitude = packet.lon
            rider_name = input("Enter rider's name: ")

            # Insert rider's data into the database
            db_cursor.execute("INSERT INTO riders (name, location) VALUES (?, ?)", (rider_name, f"{latitude}, {longitude}"))
            db_connection.commit()

            print(f"Location: {latitude}, {longitude} saved for {rider_name}")

        except Exception as e:
            print(f"Error: {e}")

# Close the database connection when done
db_connection.close()
```

# WORKING MODULE

## 1. Location Tracking:

- Utilize GPS or other location tracking technologies to collect real-time data on the buses or vehicles in your public transportation system.
- Consider using libraries like `geopy` or `GPSD` in Python to gather location data.

## 2. Rider Database:

- Create a database to store information about riders, including their boarding and alighting locations, timestamps, and unique identifiers.
- You can use a relational database like SQLite or MySQL, or a NoSQL database like MongoDB.

## 3. Data Collection:

- Develop a data collection system to record when and where riders board and alight from vehicles. You can use sensors or mobile applications for this purpose.
- Integrate the data collected with the rider database.

## 4. Route Optimization:

- Use the location data to optimize bus or vehicle routes. Consider implementing algorithms like Dijkstra's or A\* for route planning.
- Take into account traffic conditions and real-time location updates to make dynamic adjustments.

## 5. Real-time Updates:

- Create a system to provide real-time updates to riders about the estimated time of arrival (ETA) and any delays.

## 6. Web Interface:

- Build a web application that allows riders to access information about bus routes, ETAs, and delays.
- Implement a dashboard for administrators to manage and visualize the collected data.

## 7. Data Analysis:

- Analyze the collected data to identify patterns and make data-driven decisions for route optimizations.
- Python libraries like Pandas and Matplotlib can help with data analysis and visualization.

## 8. Machine Learning (Optional):

- Implement machine learning models to predict future ridership and optimize routes based on historical data.

## 9. Security and Privacy:

- Implement security measures to protect rider data and ensure privacy compliance.

## 10. Documentation and User Manuals:

- Provide clear documentation and user manuals for both administrators and riders on how to use the system.

NAME: NAYAZ.S. REG NO:

610821106064