Problem Statement

1. Congestion and Delays:
- Long waiting times at toll booths causing traffic jams and increased emissions.

2. Revenue Loss:
- Toll evasion leading to significant revenue loss and challenges in monitoring.

3. Manual Collection Issues:
- Inefficiencies and errors in manual toll collection, increasing operational costs.

Goal:

- Develop an efficient, automated toll collection system.
- Simulate and visualize vehicle movements and toll payments.
- Enhance overall toll system efficiency and effectiveness.

Unique Idea Brief (Solution)

1. Automated Toll Collection:
- Utilize GPS technology to automate toll payments, reducing manual intervention and human errors.

2. Real-Time Simulation:
- Visualize vehicle movements and toll transactions in real-time using an interactive map interface.

3. Data-Driven Efficiency:
- Implement data analytics to monitor traffic patterns, optimize toll charges, and enhance system efficiency.

Features Offered

1. Real-Time Vehicle Tracking:
- Continuous monitoring and visualization of vehicle positions on the highway.

2. Automated Toll Payments:
- Seamless toll transactions based on vehicle type and route.

3. Dynamic Data Updates:
- Real-time updates on vehicle movements, distances traveled, and toll charges paid.

Process flow

1. Vehicle Generation:
- Randomized creation of vehicles with different types and routes.

2. Simulation of Vehicle Movement:
- Vehicles travel along predefined routes, with positions updated in real-time.

3. Toll Calculation and Payment:
- Automated toll fee calculations at designated toll gates, with real-time charge updates and notifications.

Technologies used

1. Python:
- Core programming language for simulation and logic implementation.

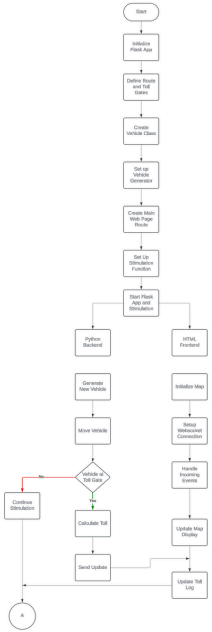
2. Flask:
- Web framework for building the application and handling web requests.

3. SimPy:
- Simulation library used to model and manage vehicle movements and toll collection.

4. Socket.IO:
- Real-time communication between server and client for dynamic updates.

5. Leaflet:
- JavaScript library for interactive maps and route visualization.

Flowchart


Start
Initialize Flask App
Define Route and Toll Gates
Create Vehicle Class
Set up Vehicle Generator
Create Main Web Page Route
Set Up Simulation Function
Start Flask App and Simulation
Python Backend
HTML Frontend
Generate New Vehicle
Initialize Map
Move Vehicle
Setup Websocket Connection
Vehicle at Toll Gate
Handle Incoming Events
No
Continue Simulation
Yes
Calculate Toll
Update Map Display
Send Update
Update Toll Log
A

Conclusion

- Efficiency: The system significantly improves the efficiency of toll collection by automating the process and reducing human error.

- Visualization: Real-time visualization of vehicle movements and toll payments enhances the transparency and manageability of toll systems.

- Future Potential: The project lays the groundwork for further advancements in automated toll collection systems, potentially incorporating more complex features and wider scalability.