|  |  |  |
| --- | --- | --- |
| **S.no** | **Content** | **Page No** |
| **1** | **Introduction** | 1 |
| **2** | **Problem Definition** | 1 |
| **3** | **Objectives** | 1 |
| **4** | **Scope of the Project** | 1 |
| **5** | **System Requirements** | 1 |
| **6** | **Literature Review / Existing System** | 1 |
| **7** | **Proposed System** | 2 |
| **8** | **System Architecture / Block Diagram** | 2 |
| **9** | **ER Diagram & Schema** | 2 |
| **10** | **Database Design (Normalization)** | 3 |
| **11** | **MySQL Implementation & Screenshots** | 3 |
| **12** | **Sample Queries and Outputs** | 3 |
| **13** | **Data Analysis & Reports Generated** | 3 |
| **14** | **Results & Discussion** | 4 |
| **15** | **Conclusion** | 4 |
| **16** | **Future Enhancements** | 4 |
| **17** | **References** | 4 |
| **18** | **Appendix / Source Code** | 5 |

**1.Introduction**

In the modern logistics and transportation industry, the efficient management of vehicle fleets is critical for profitability and customer satisfaction. Traditional methods of fleet management, which rely on manual phone calls, paper logs, and static spreadsheets, are no longer sufficient to meet the demands of real-time supply chains. **FleetPro** is a web-based, full-stack application designed to bridge this gap. It provides a centralized platform for fleet owners to track vehicles in real-time, analyze driver performance, and manage fleet operations seamlessly, while offering drivers a mobile-friendly interface to manage their duties.

**2. Problem Definition**

The current logistics landscape faces several critical challenges that hinder efficiency:

* **Manual & Inefficient Processes:** Relying on paper-based workflows for dispatching and reporting leads to data entry errors and significant productivity losses.
* **Fragmented Visibility:** Owners often lack real-time visibility into the exact location of their vehicles, leading to uncertainty and inability to provide accurate ETAs to customers.
* **Poor Performance Data:** Without standardized data collection, it is difficult to track driver efficiency, idle time, or fuel wastage, making optimization impossible.
* **Driver Resistance:** Drivers often feel micromanaged due to intrusive monitoring systems that offer them no personal benefit or transparency.

**3.** **Objectives**

The primary objectives of the FleetPro system are:

* **To develop a Role-Based System:** Create distinct interfaces for "Owners" (Management) and "Drivers" (Operations) with secure authentication.
* **To Enable Real-Time Tracking:** Implement WebSocket technology (Socket.io) to track and visualize vehicle movement on a live map.
* **To Digitize Operations:** Replace manual logs with digital status updates (On-Duty, In-Transit, Idle).
* **To Provide Actionable Analytics:** Generate financial and performance reports (Revenue, Speed, Safety Ratings) to help owners make data-driven decisions.
* **To Enhance Driver Accountability:** Implement a rating and review system to incentivize safe driving habits.

**4.** **Scope of the Project**

The scope of FleetPro includes:

* **Web Application:** A responsive web dashboard for Fleet Owners to view maps, charts, and tables.
* **Driver Module:** A mobile-optimized view for drivers to toggle status and view their assignments.
* **Geographical Focus:** The system is currently optimized for operations within the Coimbatore region, with simulation capabilities for testing.
* **Data Management:** Storing and retrieving user profiles, vehicle details, and historical performance data using a cloud database.

**5.** **System Requirements**

**Hardware Requirements:**

* **Server:** Standard Cloud Server (e.g., AWS EC2 or Heroku dyno) or Local Machine with at least 4GB RAM.
* **Client:** Any device with a modern web browser (Laptop for Owners, Smartphone for Drivers).

**Software Requirements:**

* **Operating System:** Windows/Linux/MacOS.
* **Frontend:** React.js, Tailwind CSS, Leaflet (Maps).
* **Backend:** Node.js, Express.js.
* **Database:** MongoDB (Atlas Cloud).
* **Real-time Communication:** Socket.io.
* **IDE:** Visual Studio Code.

**6.**  **Existing System**

Existing systems often fall into two categories: high-end enterprise solutions (too expensive for small fleets) or basic GPS trackers (lack operational context).

* **Limitation of Existing Systems:** Most basic systems only show dots on a map without context. They do not calculate revenue per trip, nor do they allow drivers to interact with the system to update their status.
* **Gap Analysis:** There is a lack of affordable, user-friendly solutions that combine *tracking* with *business analytics* (Revenue & Ratings). FleetPro addresses this by merging these two domains.

**7.** **Proposed System**

FleetPro proposes a unified platform where data flows seamlessly between the road and the office.

* **Dynamic Route & Status Updates:** Instead of calling a driver, the owner sees the status ("In-Transit") instantly on the dashboard.
* **Automated Data Entry:** When a driver starts a trip, the system automatically logs the start time and tracks speed, eliminating manual logbooks.
* **Performance Analytics:** The system automatically calculates revenue based on distance/time and aggregates safety ratings, providing a "Driver Scorecard."

**8.** **System Architecture / Block Diagram**

The system follows the MERN architecture:

1. **Client Layer (React):** Handles the UI. Drivers send location updates; Owners receive live feeds.
2. **API Layer (Express/Node):** Processes authentication (JWT), manages API endpoints, and handles WebSocket connections.
3. **Data Layer (MongoDB):** Stores User profiles, Vehicle schemas, and persistent logs.
4. **Real-Time Layer (Socket.io):** Opens a bi-directional channel for sub-second location updates between Client and Server.

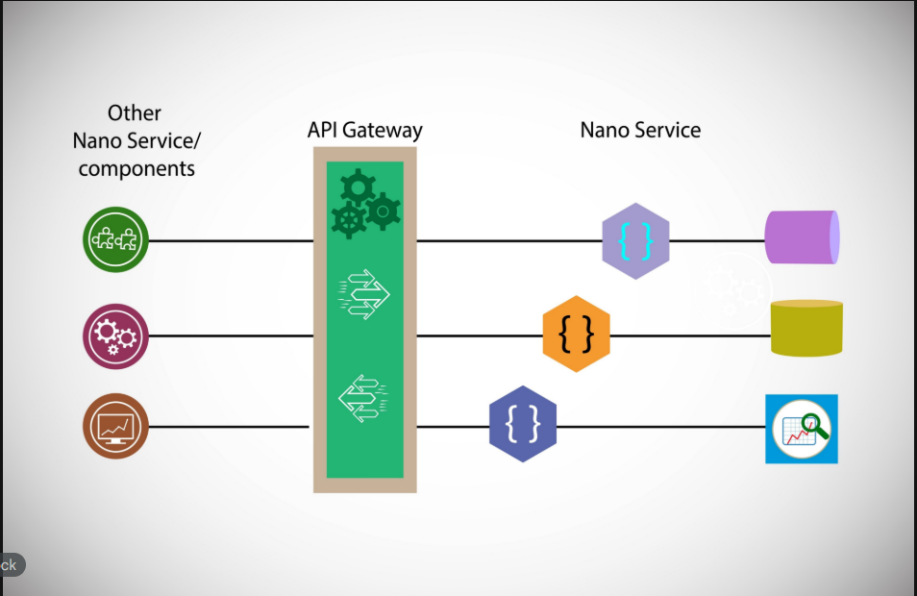


Figure 8.1 System Architecture / Block Diagram

**9.** **ER Diagram & Schema**

The database is designed around two primary entities: User and Vehicle (often merged for simplicity in this MVP).

{

"\_id": "ObjectId",

"name": "String",

"email": "String (Unique)",

"password": "String (Hashed)",

"role": "Enum ['owner', 'driver']",

"vehicleId": "String (e.g., TN-37-A1)",

"licenseNumber": "String",

"rating": "Number",

"revenue": "Number",

"status": "Enum ['Active', 'Idle', 'In-Transit']"

}

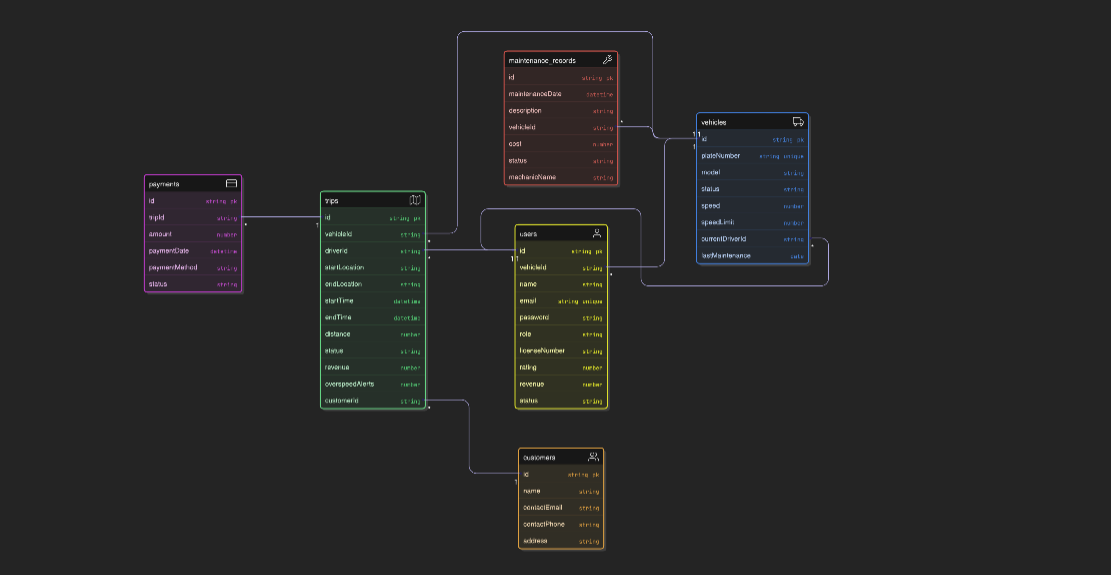


Figure 9.1 ER diagram & Schema

**10.** **Database Design (Normalization)**

While MongoDB is a NoSQL database, the design follows principles to reduce redundancy:

* **Authentication Separation:** User credentials and roles are handled in a centralized collection to ensure security.
* **Data Integrity:** Unique constraints are applied to email and licenseNumber to prevent duplicate drivers.
* **Scalability:** The schema is flexible; adding new fields (like "Fuel Level" or "Maintenance Date") does not break existing records.

**11.** **Database Implementation (NoSQL)**

* **Backend Security:** Passwords are hashed using bcryptjs before storage. JSON Web Tokens (JWT) are used to secure API routes.
* **Frontend Visualization:** React-Leaflet is used to render the interactive map. Custom markers (Truck icons) distinguish vehicles from standard waypoints.
* **Live Simulation:** A custom Node.js script (simulate.js) was written to mathematically simulate 15 trucks moving around Coimbatore coordinates, providing realistic test data for the owner dashboard.

**12.** **Sample Queries and Outputs**

**Find all active drivers in Coimbatore.**

db.users.find({ role: "driver", status: "In-Transit" })

**Calculate total fleet revenue**.

db.users.aggregate([

{ $match: { role: "driver" } },

{ $group: { \_id: null, totalRevenue: { $sum: "$revenue" } } }

])

**13.** **Data Analysis & Reports Generated**

The Owner Dashboard provides instant analysis:

* **Revenue Analysis:** A visual card displays total earnings (e.g., "Rs. 1,25,000"), allowing owners to track profitability.
* **Fleet Utilization:** The "Active Fleet" counter (e.g., "8 / 15") shows how much of the fleet is currently utilized vs. sitting idle.
* **Safety Monitoring:** The system flags drivers exceeding speed limits (e.g., >80 km/h) with visual red alerts and an "OVERSPEEDING" tag.

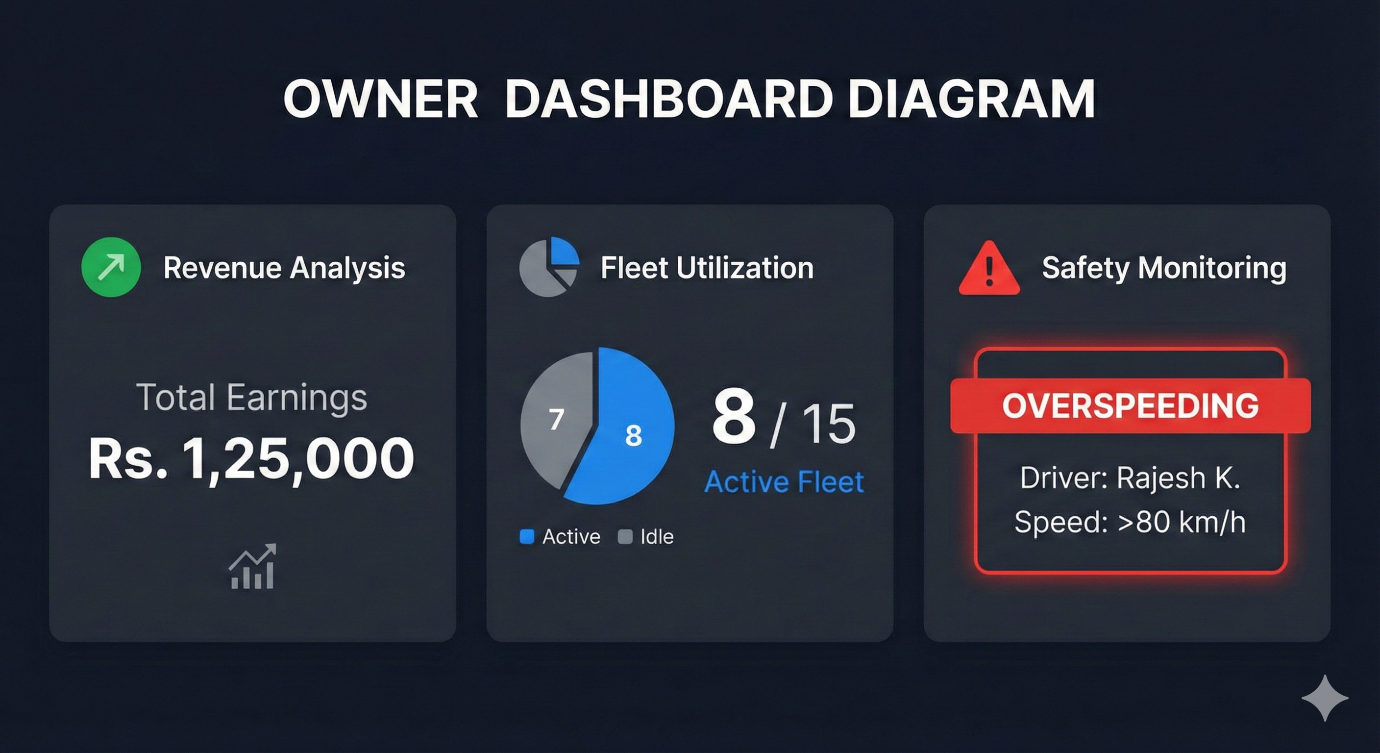


Figure 9.1 Data Analysis & Reports Generated

**14.** **Results & Discussion**

The system was successfully tested with 15 simulated vehicles in the Coimbatore region.

* **Latency:** Location updates on the map occurred with <500ms latency via Socket.io.
* **Usability:** The Role-Based Access Control (RBAC) successfully prevented drivers from accessing sensitive Owner settings.
* **Visualization:** The "Drivers Grid" view provided a much clearer operational picture than traditional list views, allowing for quicker decision-making.

**15.** **Conclusion**

FleetPro successfully demonstrates that complex fleet management operations can be digitized using modern web technologies. By integrating real-time tracking with business intelligence (revenue/ratings), the system solves the core problem of "Fragmented Visibility." It provides a scalable foundation for any logistics company looking to modernize their operations.

**16.** **Future Enhancements**

* **AI Route Optimization:** Implementing algorithms to suggest the fastest route based on traffic.
* **Maintenance Prediction:** Using mileage data to automatically alert owners when a truck needs an oil change.
* **Geofencing:** Triggering automatic alerts when a truck leaves a designated city zone.

**17.** **References**

* MongoDB Documentation: <https://www.mongodb.com/docs/>
* React.js Official Docs: <https://react.dev/>
* Socket.IO Documentation: <https://socket.io/docs/v4/>
* Leaflet Maps API: <https://leafletjs.com/>

**18.** **Appendix / Source Code**

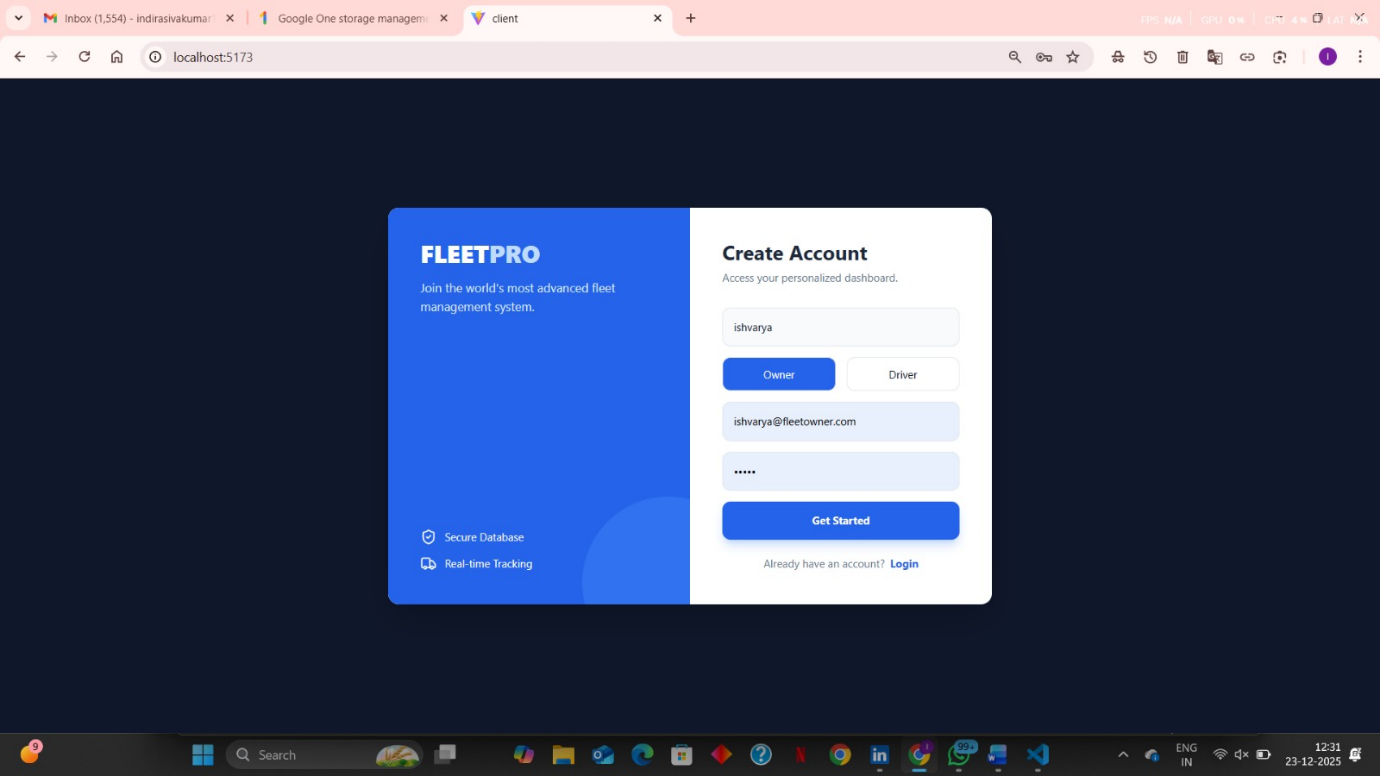
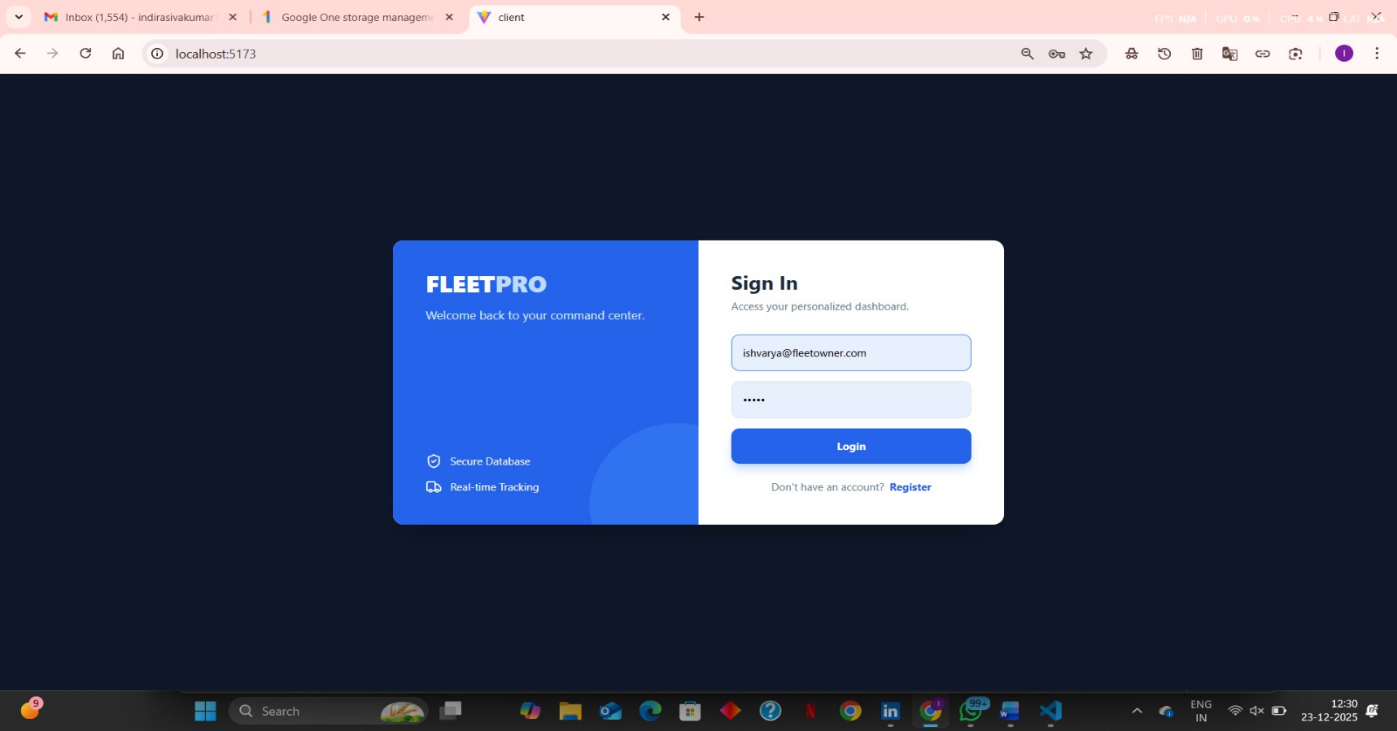
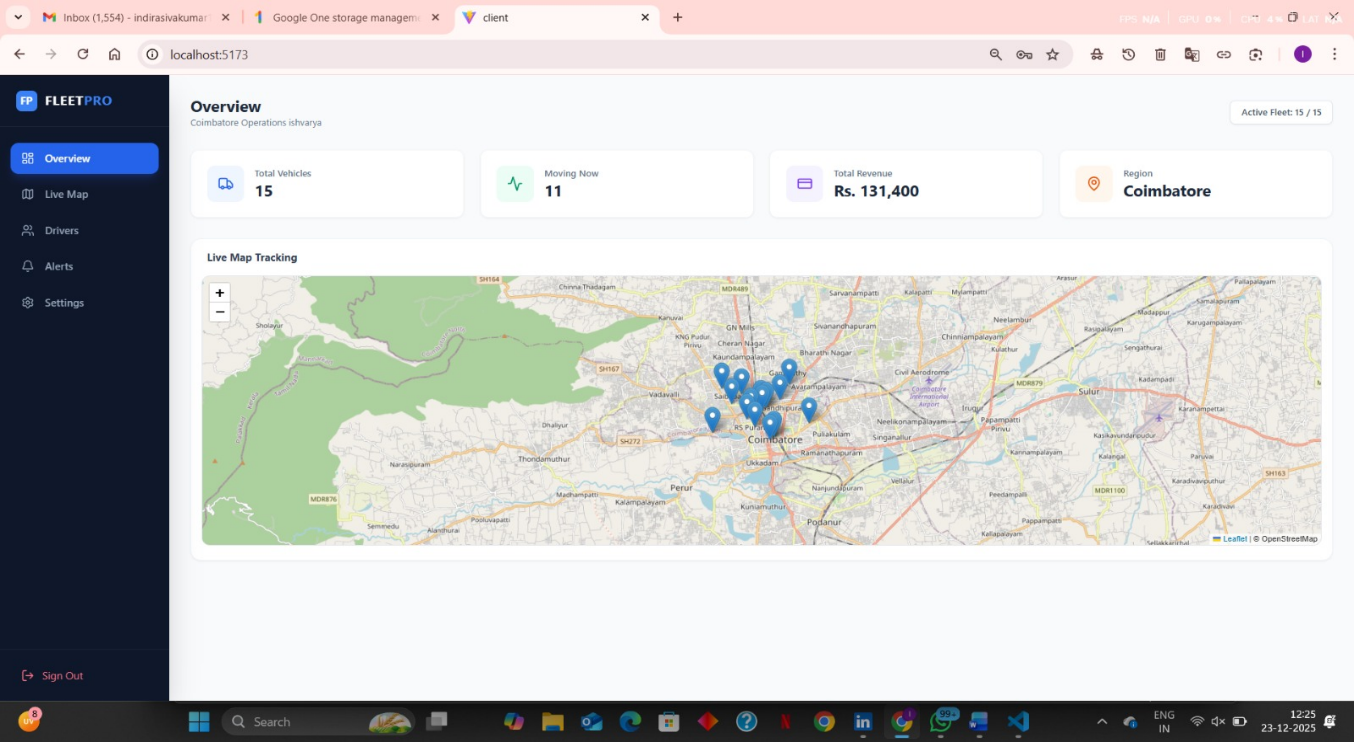
****

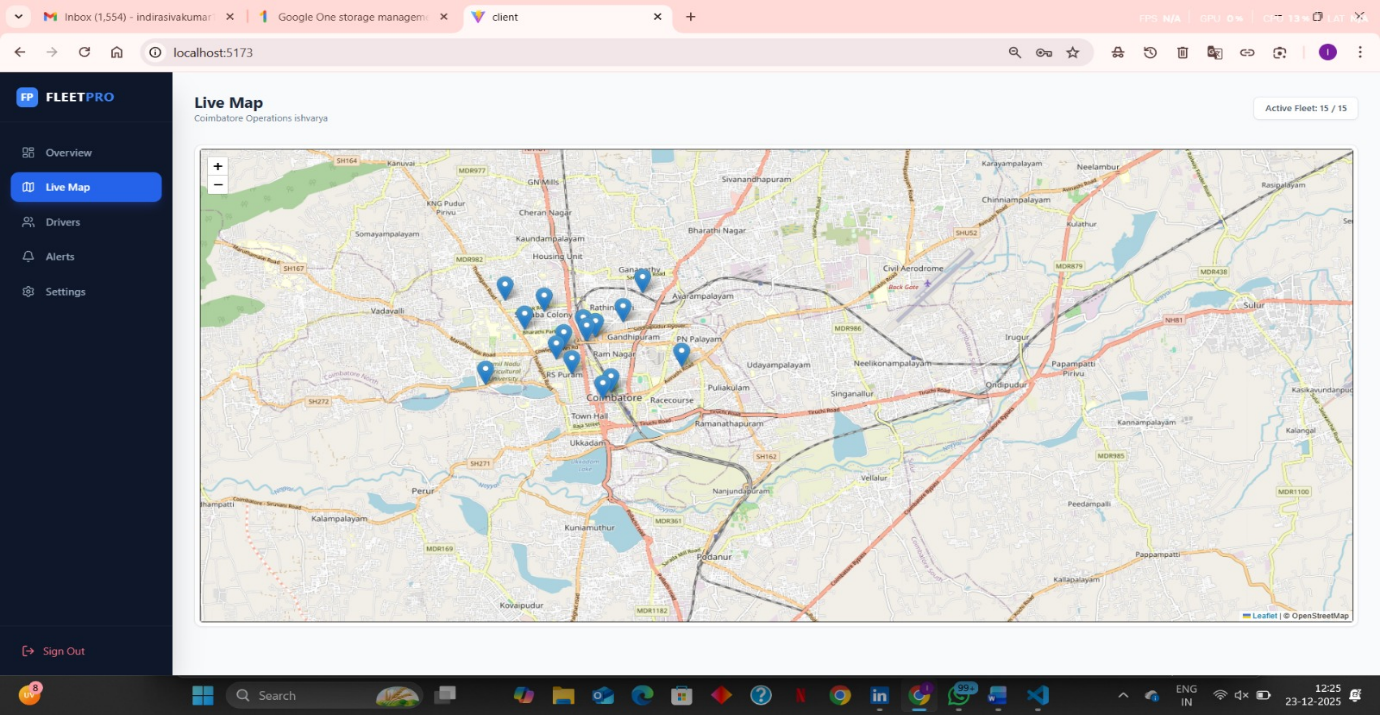
Fig.1 creating login page for registering owner of fleet

****

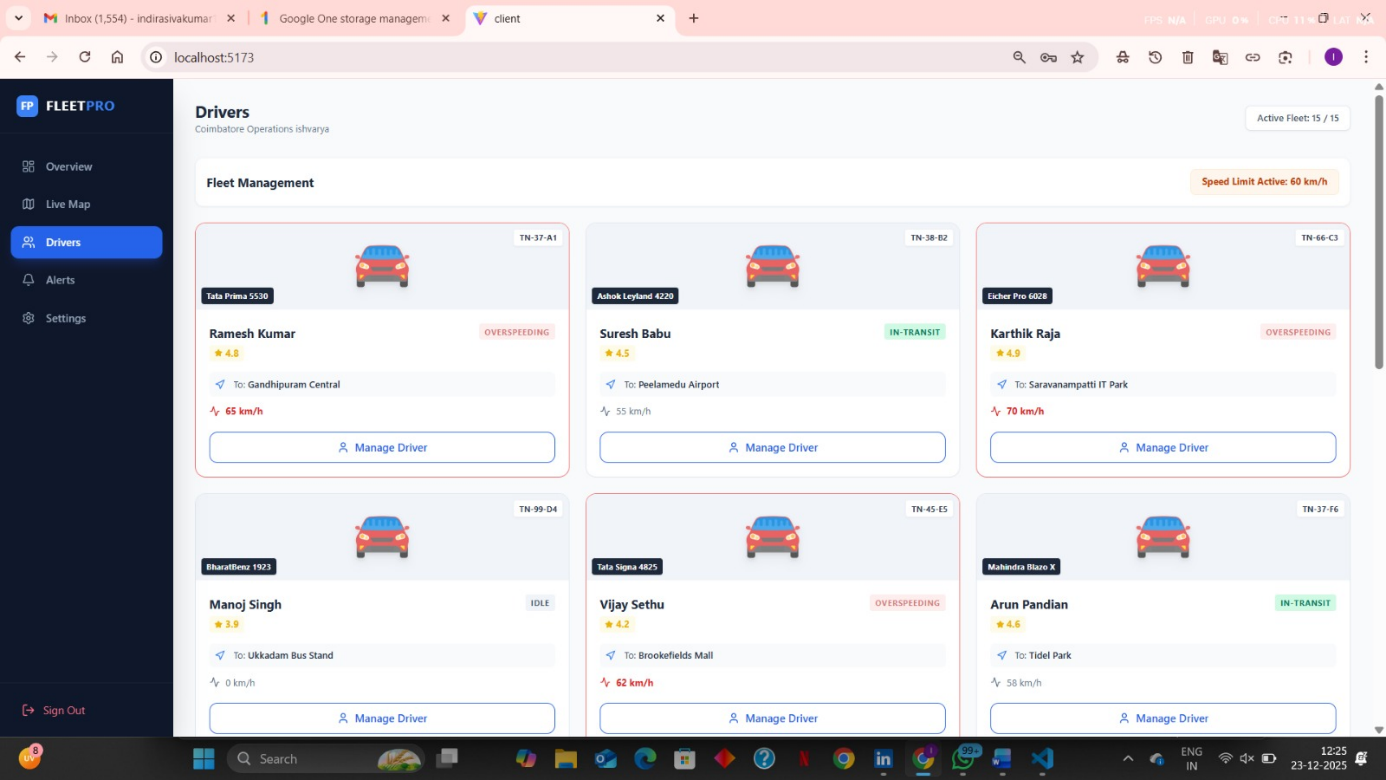
Fig, 2 login page of fleet tracking for owner

****

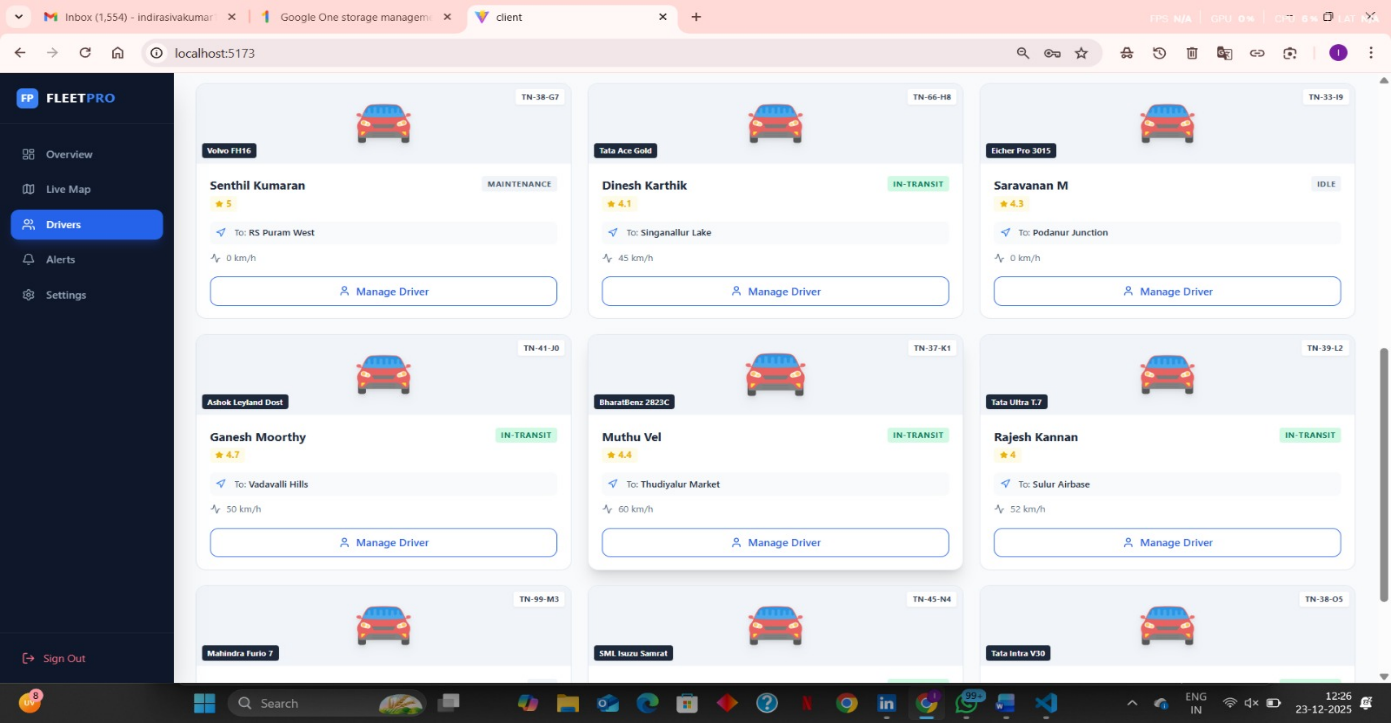
Fig, 3 opening the fleet tracking page with vehicles details

****

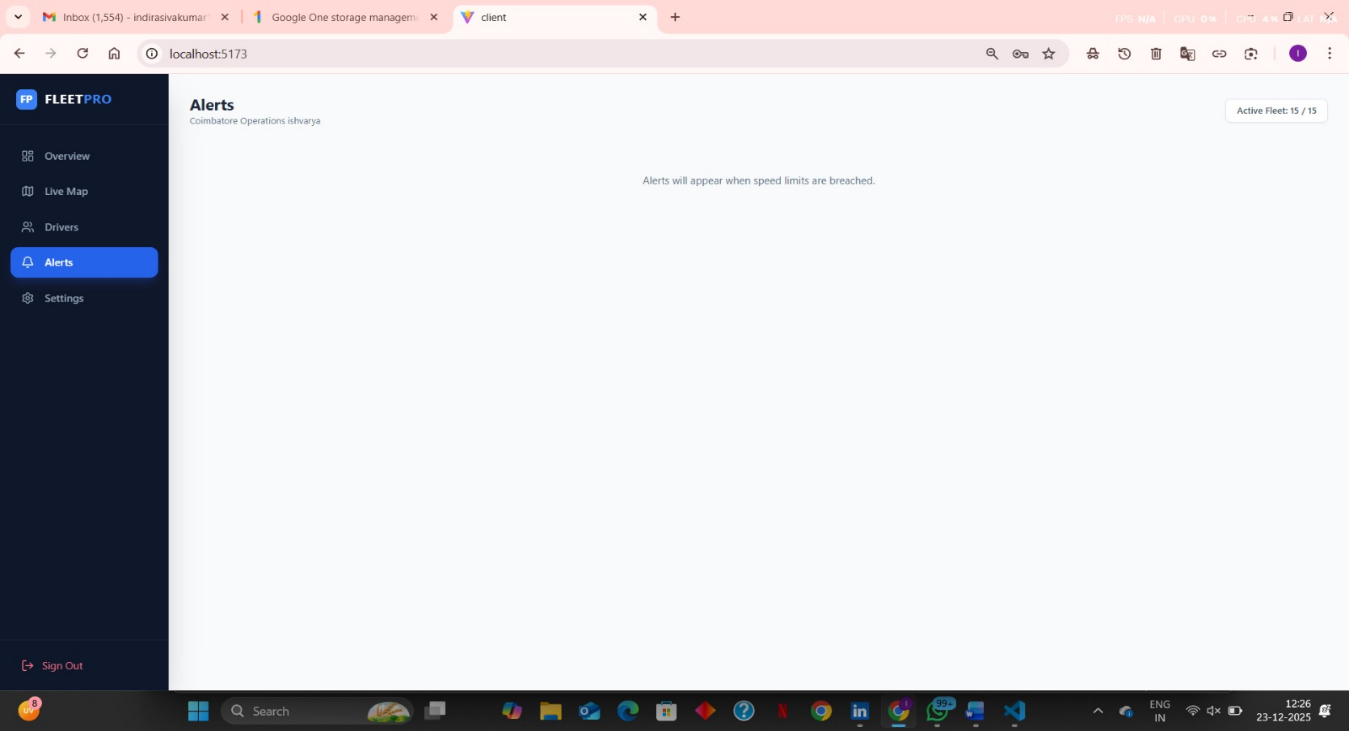
Fig,4 vehicles going live location in map

****

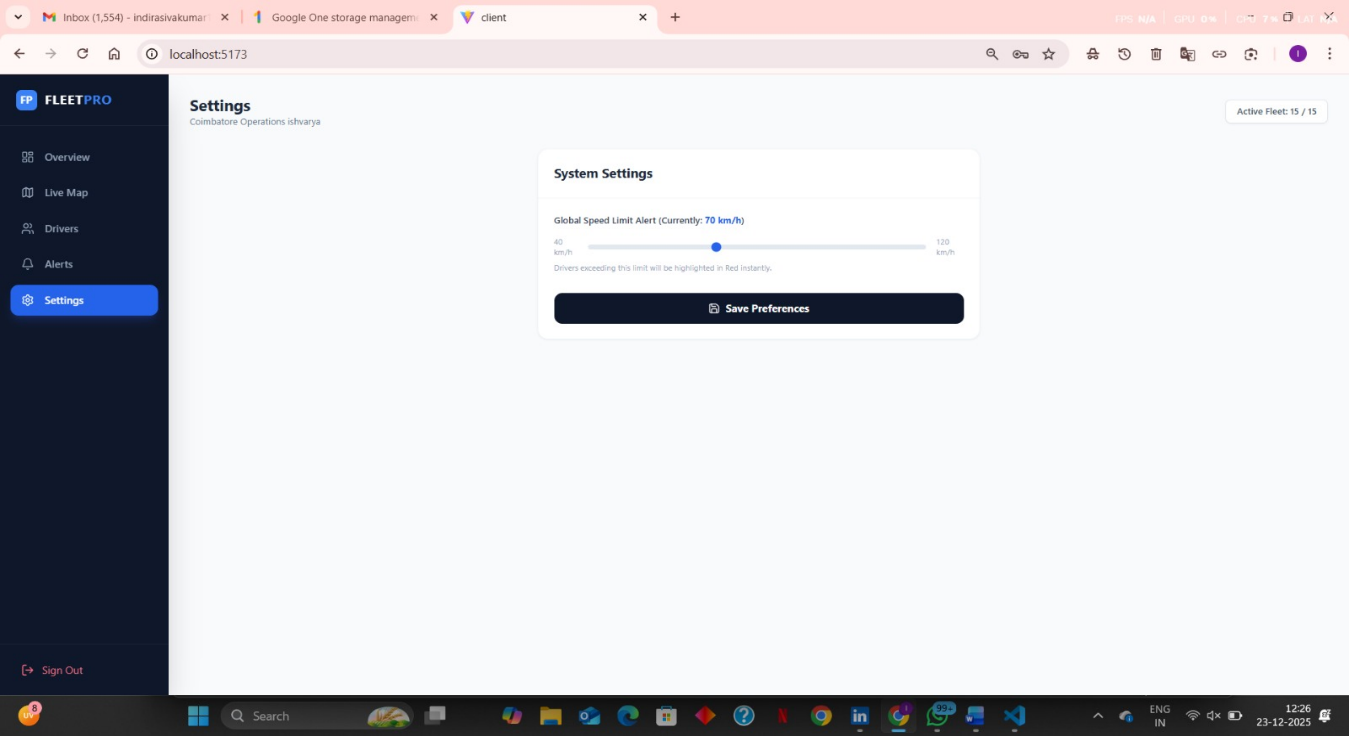
Fig, 5 fleet driver list and details

****

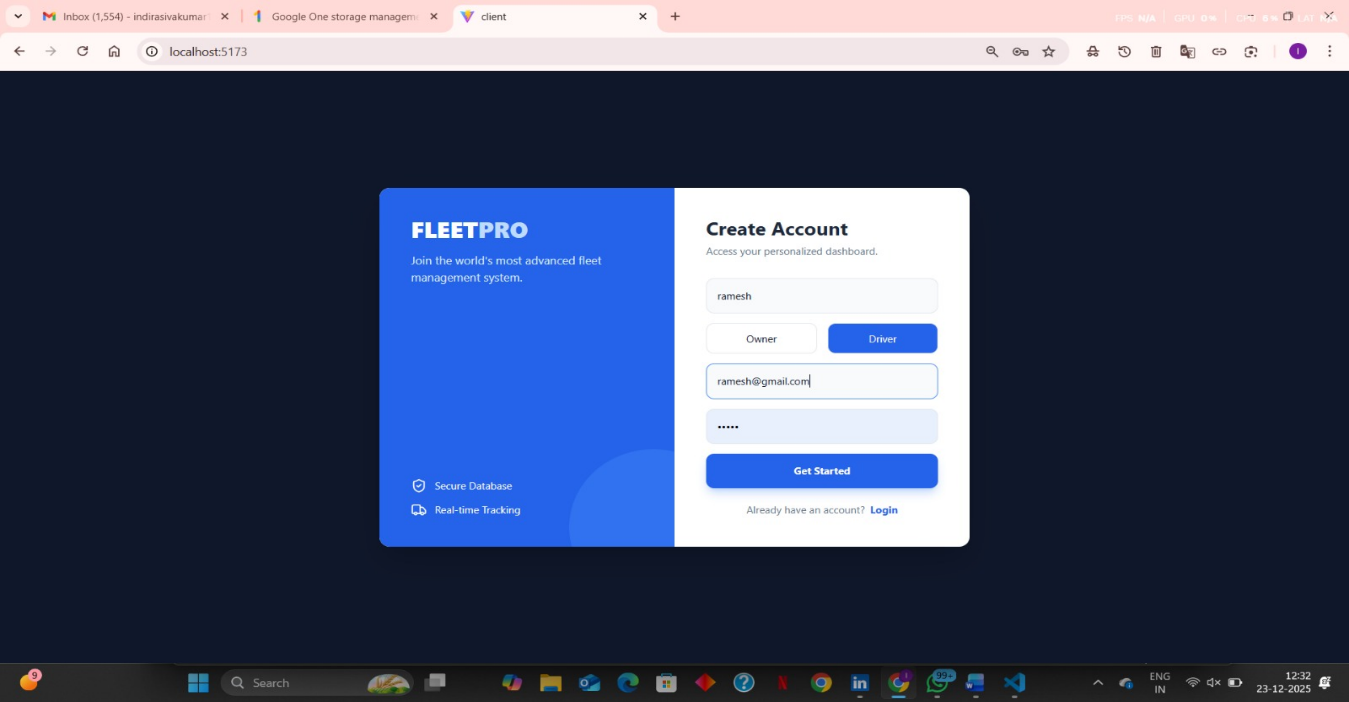
Fig, 6 drivers list and vehicle details

****

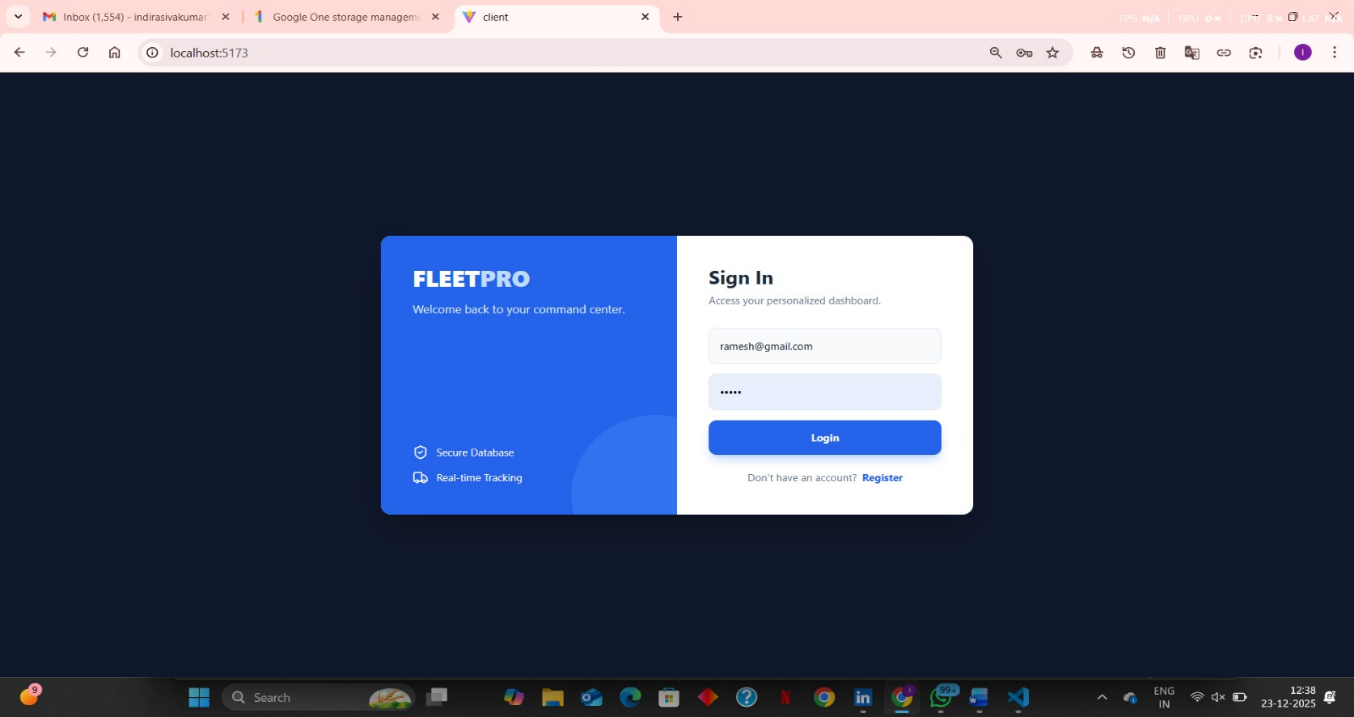
Fig, 7 In fleet tracking the speed existed and other illegal activites its alerts from tracking page

****

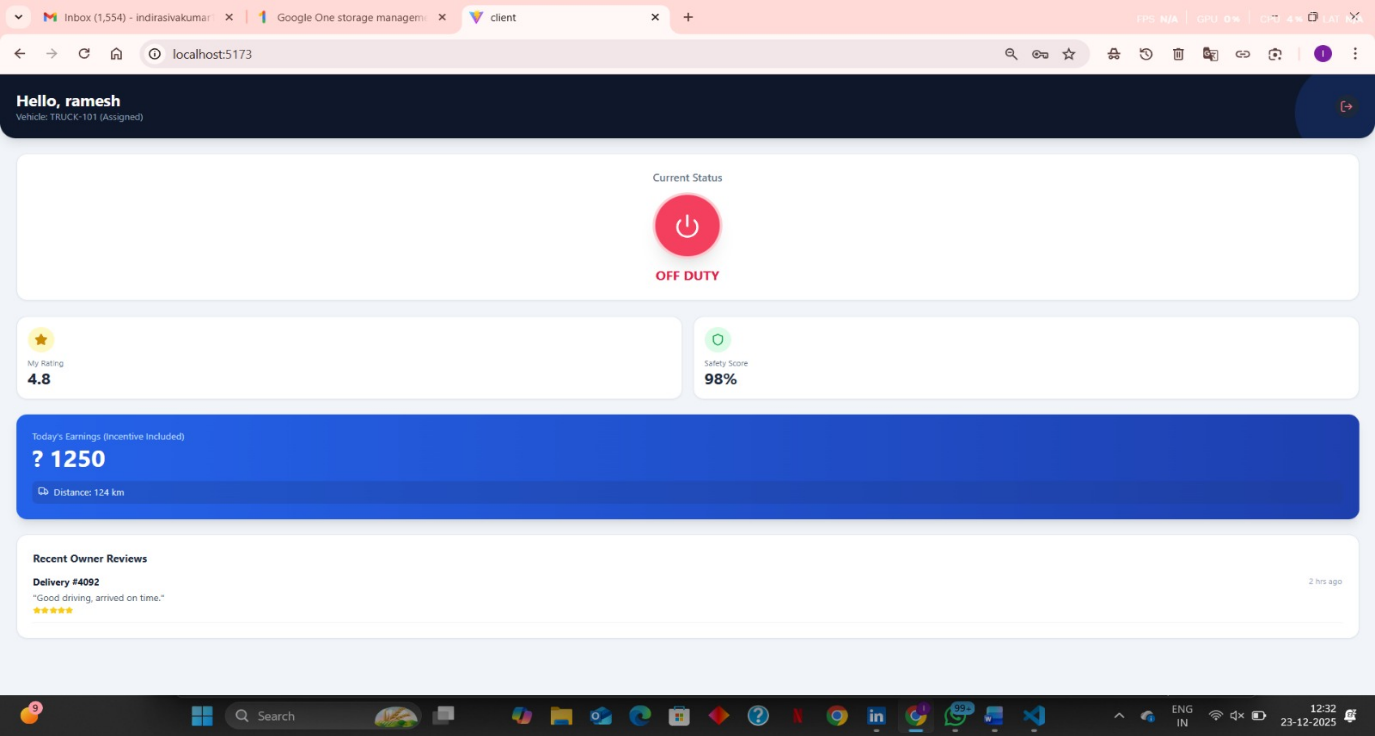
Fig, 8 speed customize from the owner to driver



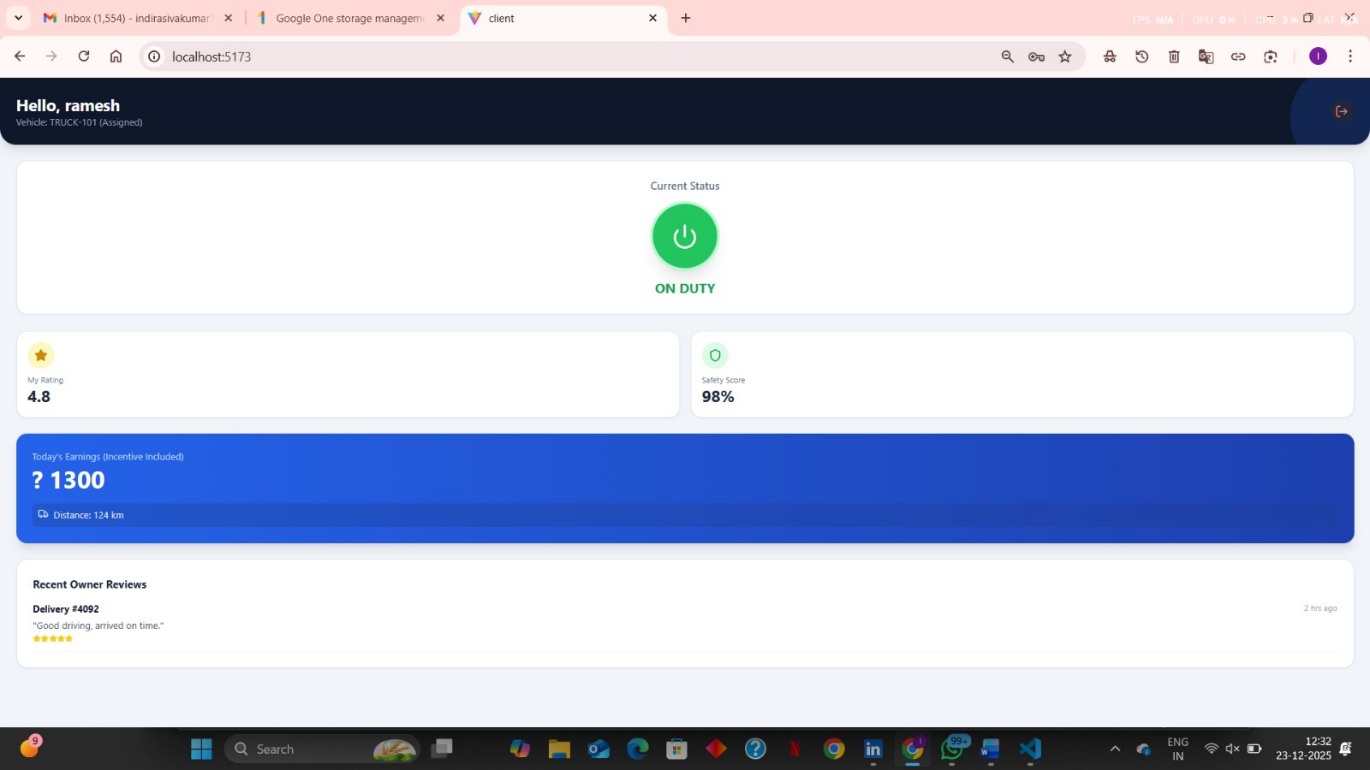
Fig, 9 login 2 for the driver login registering driver of fleet



Fig, 10 login 2 for driver



Fig, 11 drivers dashboard



**Fig, 12 drivers dashboard part 2**

**The complete source code for the Fleet Tracking project, including the frontend (React, tailwind.js), backend (MangoDB), and database schemas, is hosted on GitHub.**

* **Repository Name: Fleet Tracking & Delivery Performance Analysis**
* **Version: 1.0.0 (Platinum Release)**
* **Project Link:**