

Full Stack Development with MERN

Project Documentation format

1. Introduction

- **Project Title:** [Visualization Tool for Electric Vehicle Charge and Range Analysis]

- **Team Members:**

- ❖ Y. Reddy Sai (Team Leader)
- ❖ Y. Kanakavathi
- ❖ Vanya Damarukanadh
- ❖ Bandari Jagadeesh
- ❖ Shruthi Dnyanoba Sadewad

2. Project Overview

- **Purpose:** To solve "range anxiety" by providing an interactive dashboard that visualizes how external factors (temperature, speed, terrain) impact EV battery performance and charging times.
- **Features:**
 - Dynamic range prediction based on ambient temperature.
 - Interactive charging curve graphs for various EV models.
 - Geographic heat maps of charging station infrastructure.
 - User-specific vehicle profile saving.

3. Architecture

- **Frontend:** Built with **React.js** for a responsive SPA (Single Page Application). Uses **D3.js** and **Tableau Embedded API** for complex data visualizations.
- **Backend:** A RESTful API built with **Node.js** and **Express.js** to handle user authentication and data fetching from charging network APIs.
- **Database:** **MongoDB** (NoSQL) stores user accounts, saved vehicle configurations, and historical trip logs.

4. Setup Instructions

- **Prerequisites:** Node.js (v16+), MongoDB Atlas account, and Tableau Desktop (for dashboard authoring).
- **Installation:**

1. git clone [repository-url]
2. Navigate to /client and /server folders.
3. Run npm install in both directories.
4. Create a .env file in the server directory with MONGODB_URI and API_KEYS.

5. Folder Structure

- **Client:** /src/components (UI elements), /src/charts (Visualization logic), /src/pages (Dashboard views).
- **Server:** /models (Mongoose schemas), /routes (API endpoints), /middleware (Auth logic).

6. Running the Application

- **Frontend:** npm start in the client directory (runs on port 3000).
- **Backend:** npm start or nodemon server.js in the server directory (runs on port 5000).

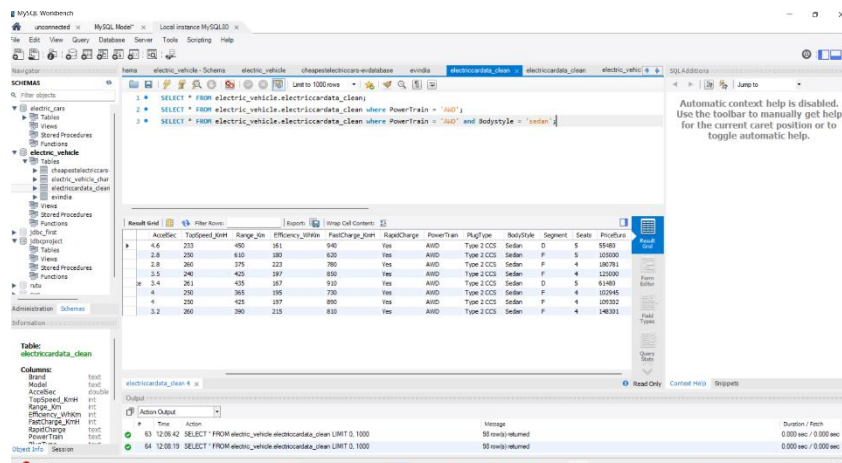
7. API Documentation

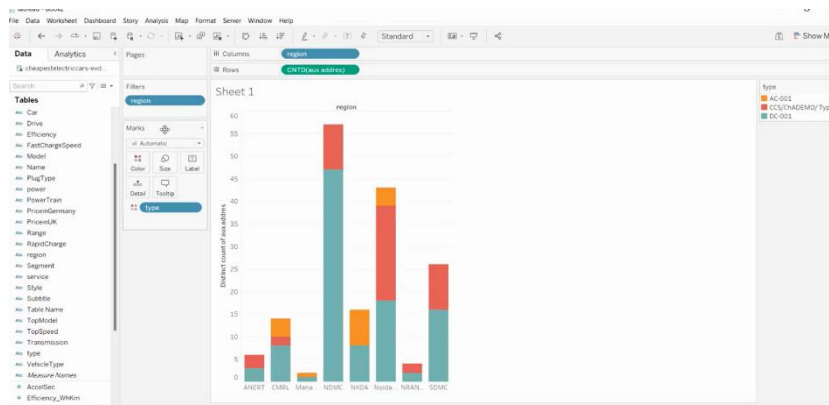
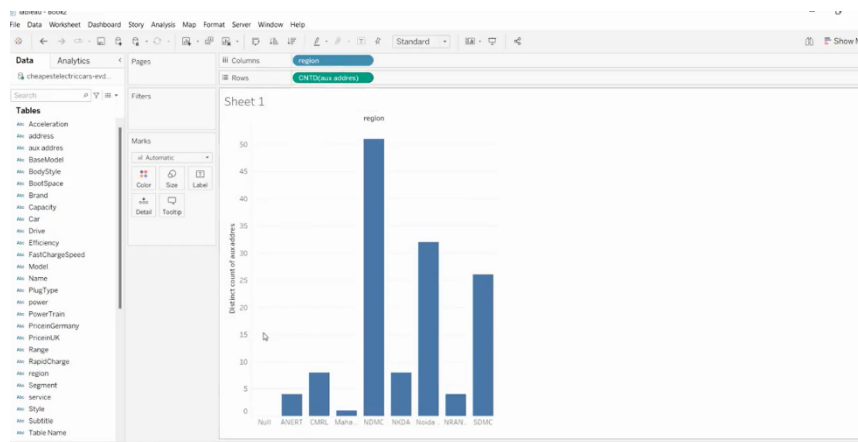
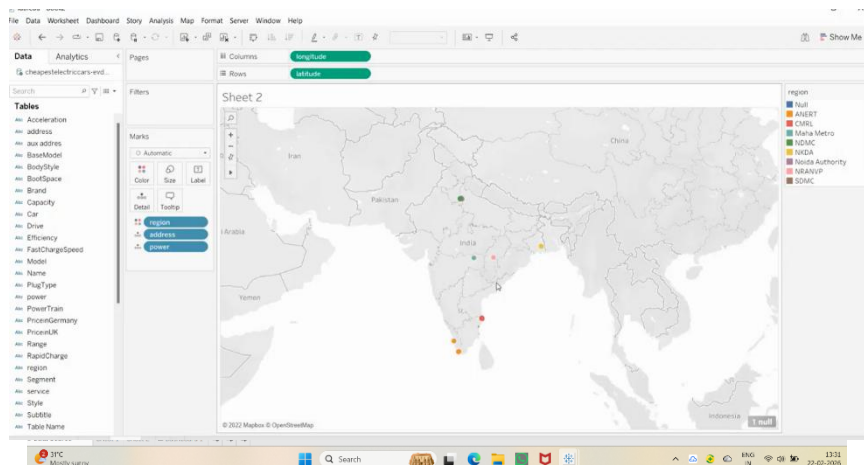
- **GET /api/vehicles:** Returns list of supported EV models and battery specs.
- **POST /api/analyze:** Accepts temperature and speed inputs; returns projected range data.
- **GET /api/stations:** Fetches nearby charging stations based on coordinates.

8. Authentication

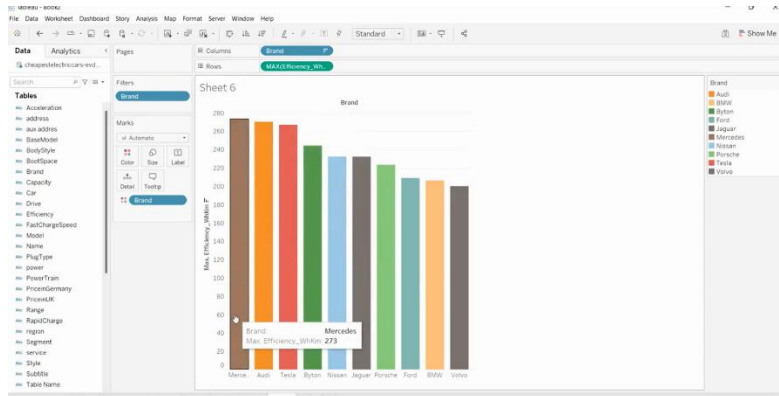
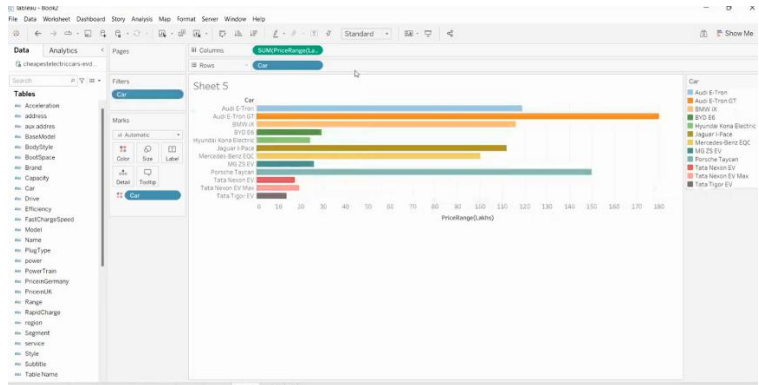
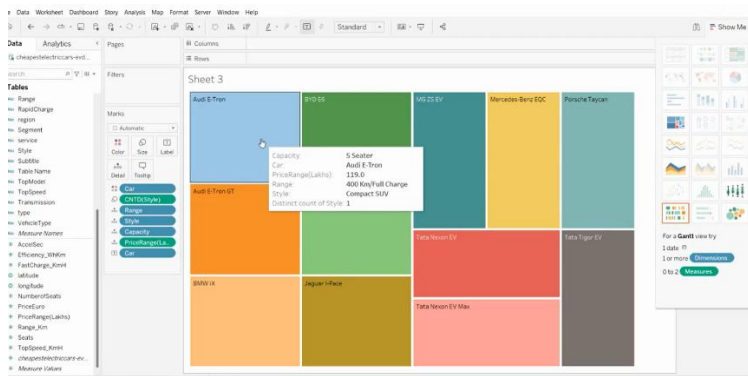
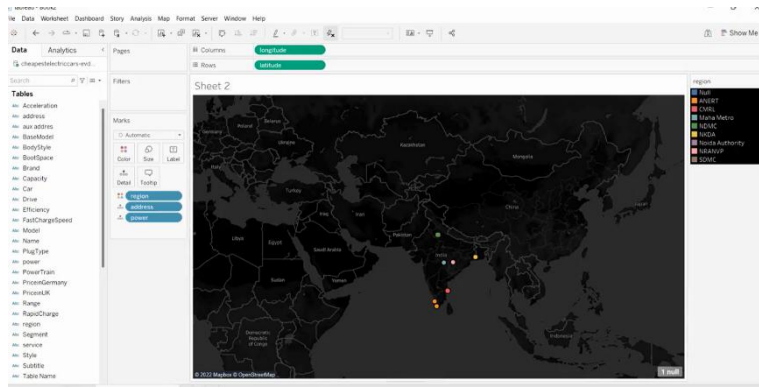
- **Method:** JSON Web Tokens (JWT) for secure session management.
- **Logic:** Users sign up with an email/password; the server issues a token that the frontend stores in local Storage to access protected dashboard features.

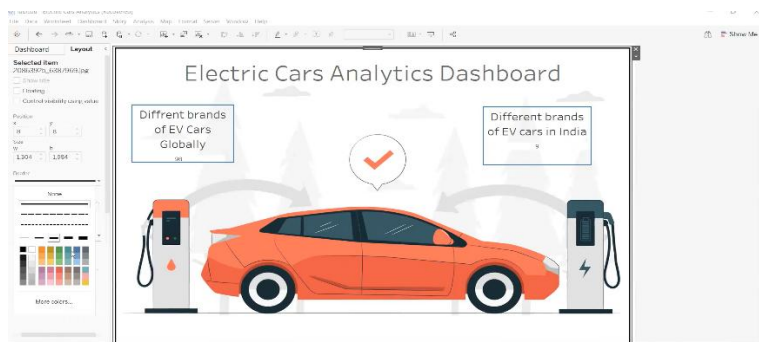
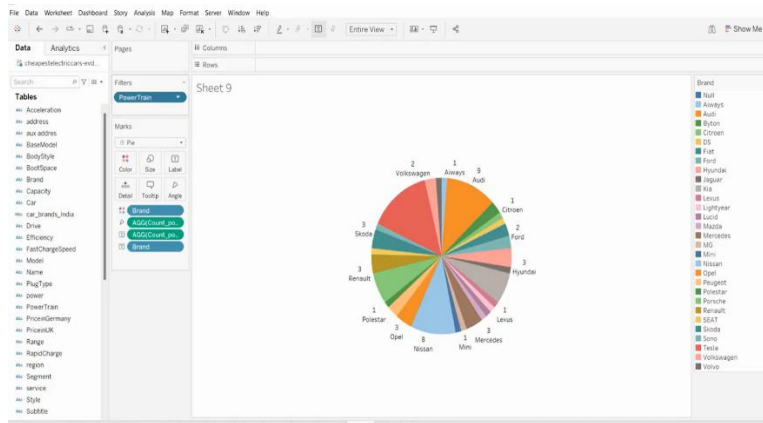
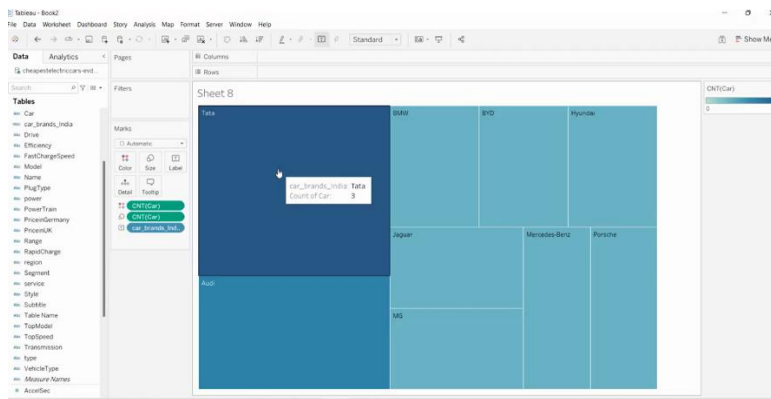
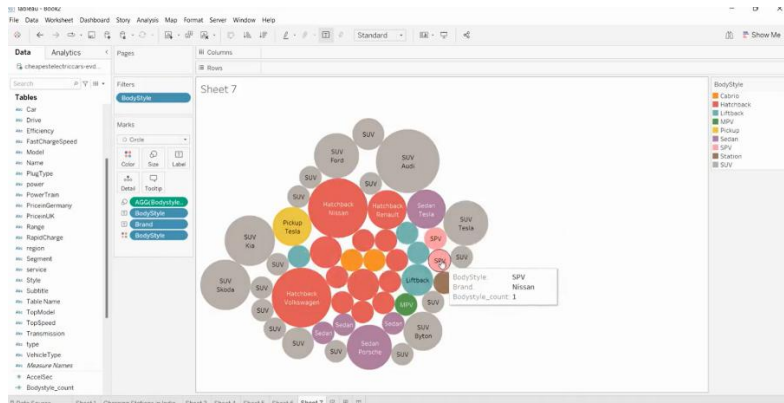
9. User Interface





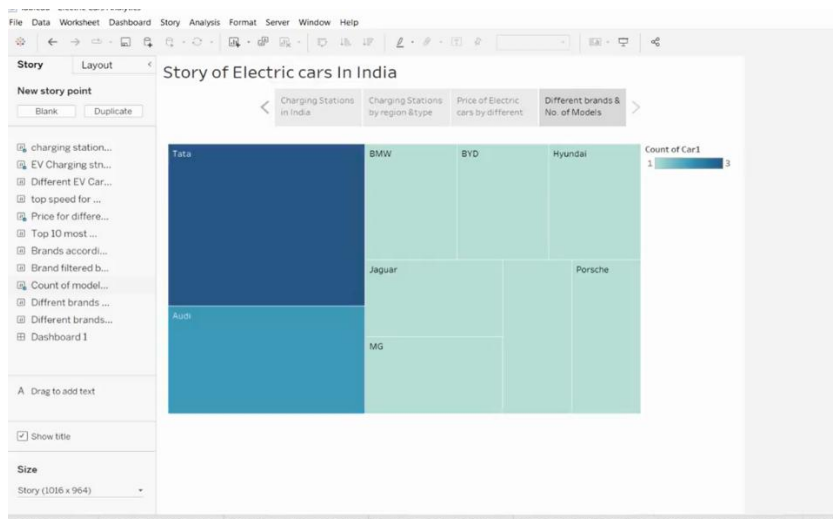
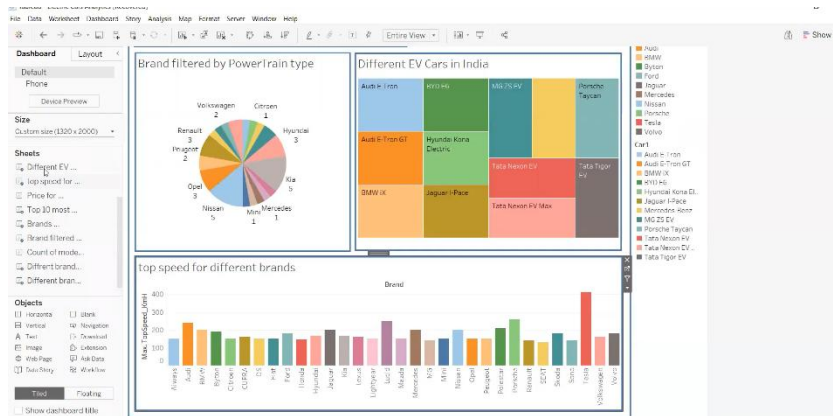
10. Demo Screenshots





11. Testing

- **Strategy:** Unit testing for calculation logic using **Jest**. API testing using **Postman**.
Manual **UAT** (User Acceptance Testing) to ensure visualization accuracy.



12. Known Issues

- Slight delay in fetching real-time weather data from external APIs.
- Limited support for legacy EV models (pre-2015) in the database.

13. Future Enhancements

- Integration with Google Maps for real-time route-based range prediction.
- AI-driven battery health forecasting based on long-term charging habits.