

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

The screenshot shows the MySQL Workbench interface. On the left, the 'Schemas' tree view is expanded to show the 'electric_vehicle' schema, which contains tables like 'electric_cars', 'electric_car_clean', and 'electricdata_clean'. In the center, a query editor window displays three SQL statements:

```

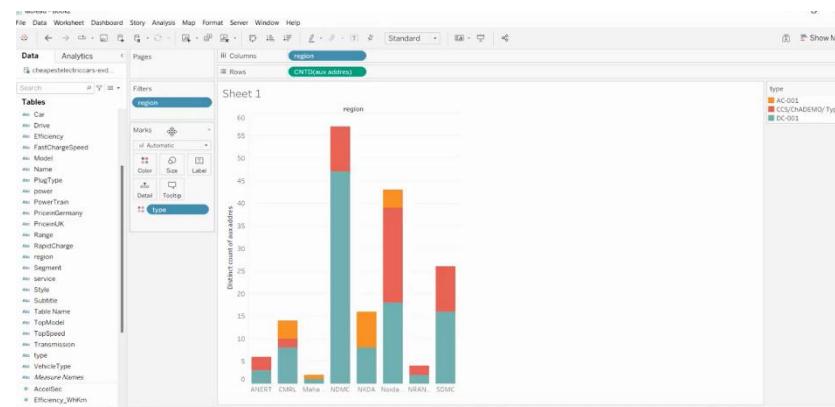
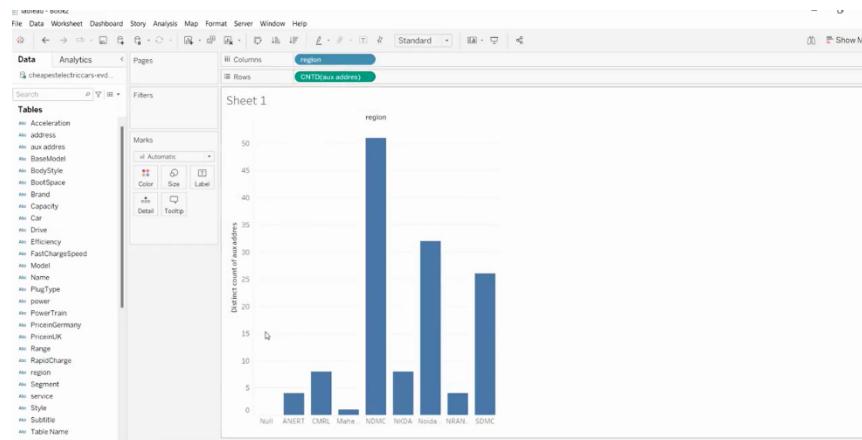
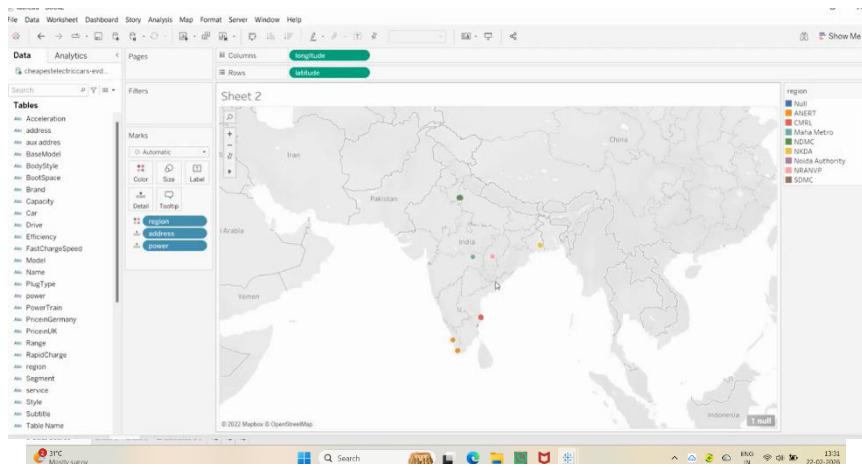
1. SELECT * FROM electric_vehicle.electricdata_clean;
2. SELECT * FROM electric_vehicle.electricdata_clean WHERE PowerTrain = 'AWD';
3. SELECT * FROM electric_vehicle.electricdata_clean WHERE PowerTrain = 'AWD' AND BodyStyle = 'sedan';

```

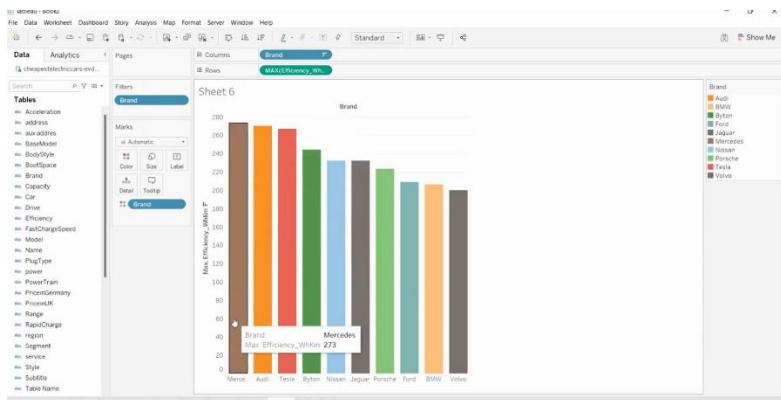
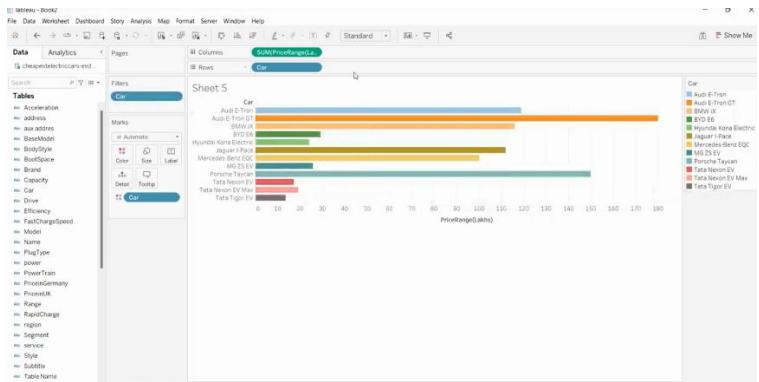
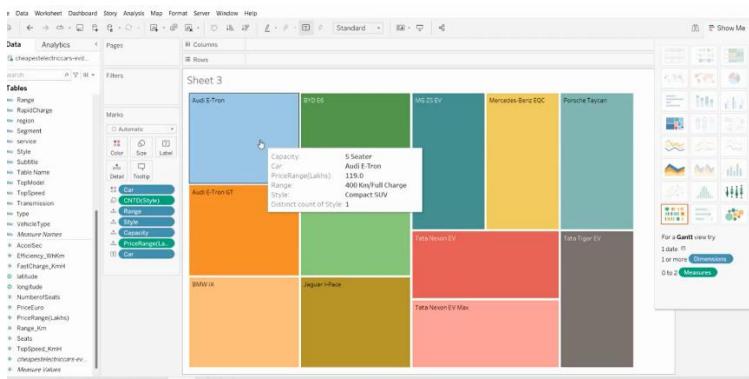
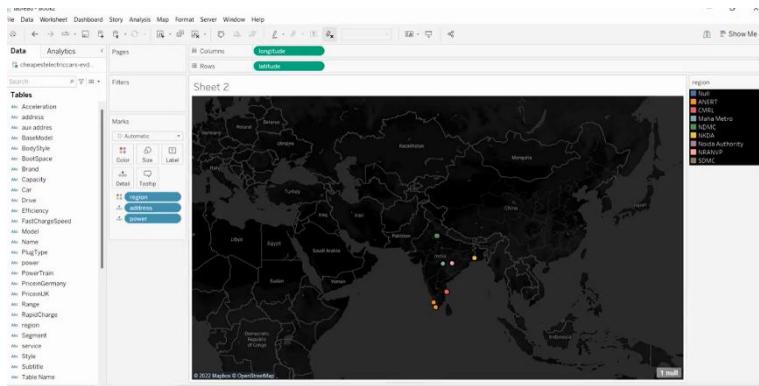
The third query's results are shown in a grid below:

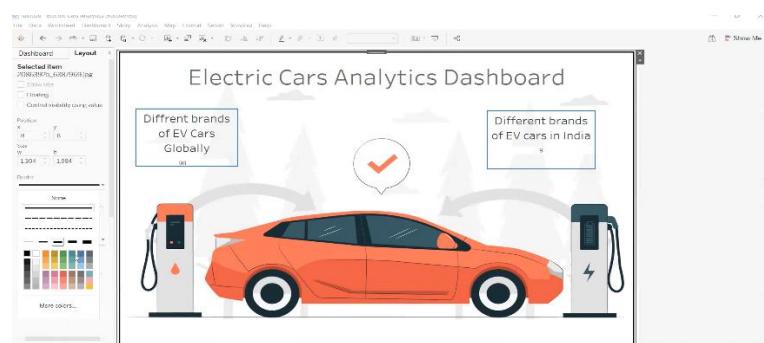
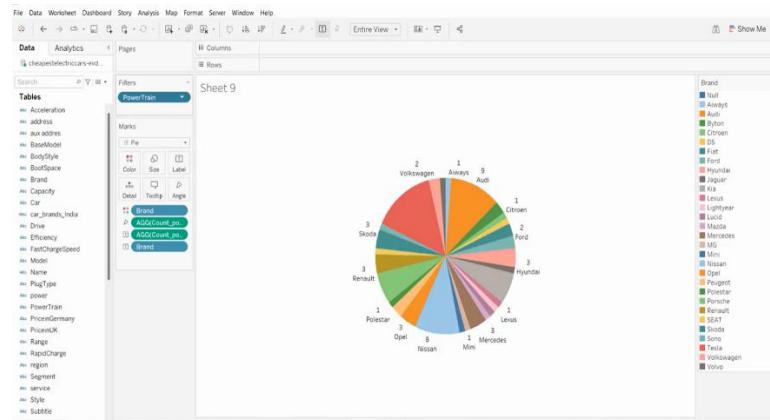
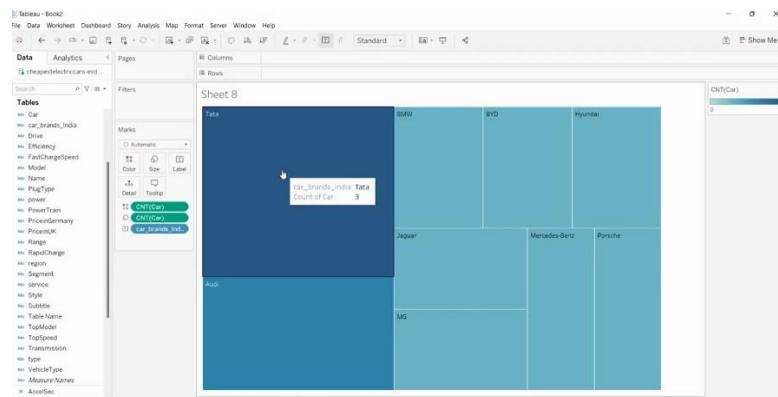
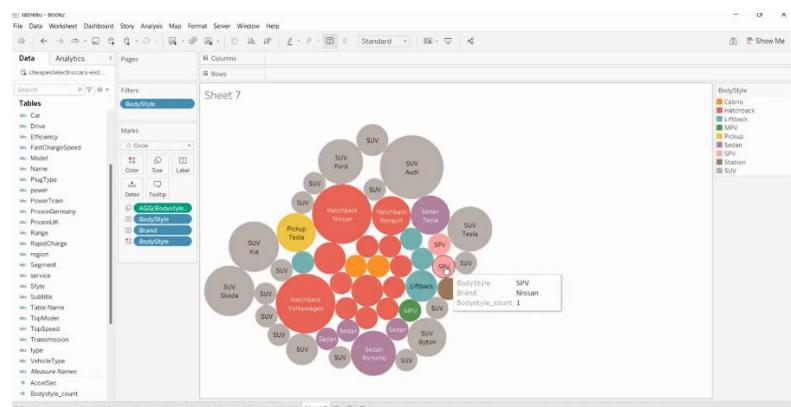
Accelerate_0to60_Km	TopSpeed_KmH	Range_Km	Efficiency_Wlmp	FastCharge_0to80%	PowerTrain	Type	BodyStyle	Segment	Seats	PriceEUR	
4.6	233	450	161	940	AWD	Type 2 CCS	Sedan	D	5	\$5480	
2.8	260	380	180	840	Yes	Type 2 CCS	Sedan	D	5	\$4990	
2.8	275	225	760	Yes	AWD	Type 2 CCS	Sedan	F	4	\$6775	
3.5	240	425	287	850	Yes	Type 2 CCS	Sedan	F	4	\$12000	
3.4	260	430	187	930	Yes	Type 2 CCS	Sedan	D	5	\$12490	
4	270	365	195	730	Yes	Type 2 CCS	Sedan	F	4	\$12945	
4	250	425	197	890	Yes	Type 2 CCS	Sedan	F	4	\$10332	
3.2	260	390	215	830	Yes	AWD	Type 2 CCS	Sedan	F	4	\$48301

At the bottom of the interface, the status bar shows 'Duration: 1.7 sec' and '0.000 sec / 0.000 sec'.



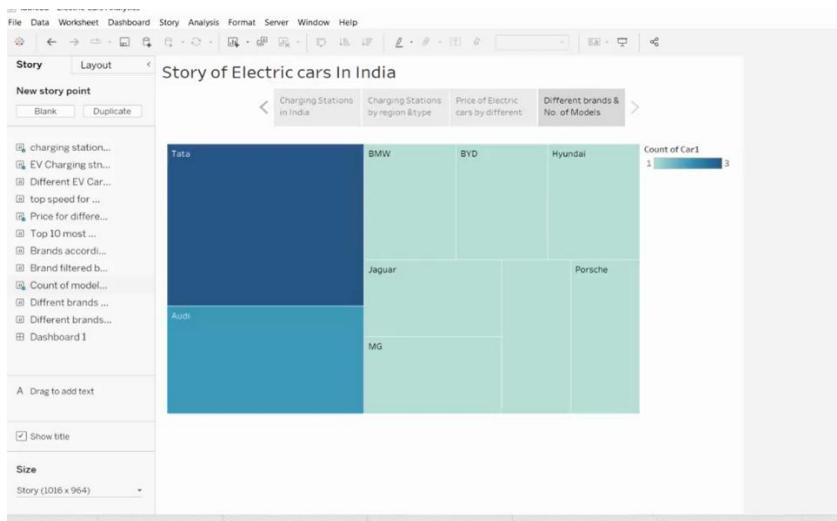
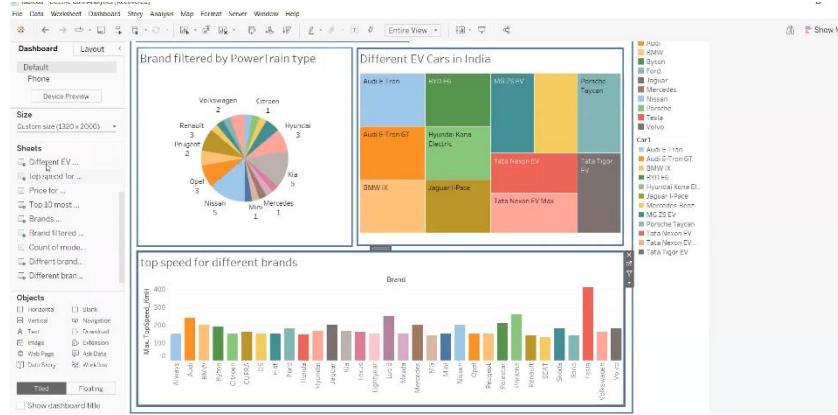
10. Demo Screenshots





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.