Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	31 January 3035
Team ID	LTVIP2025TMID20775
Project Name	Visualization Tool for Electric Vehicle Charge and Range Analysis
Maximum Marks	4 Marks

Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

Example: Order processing during pandemics for offline mode

Reference: https://developer.ibm.com/patterns/ai-powered-backend-system-for-order-processing-during-pandemics/

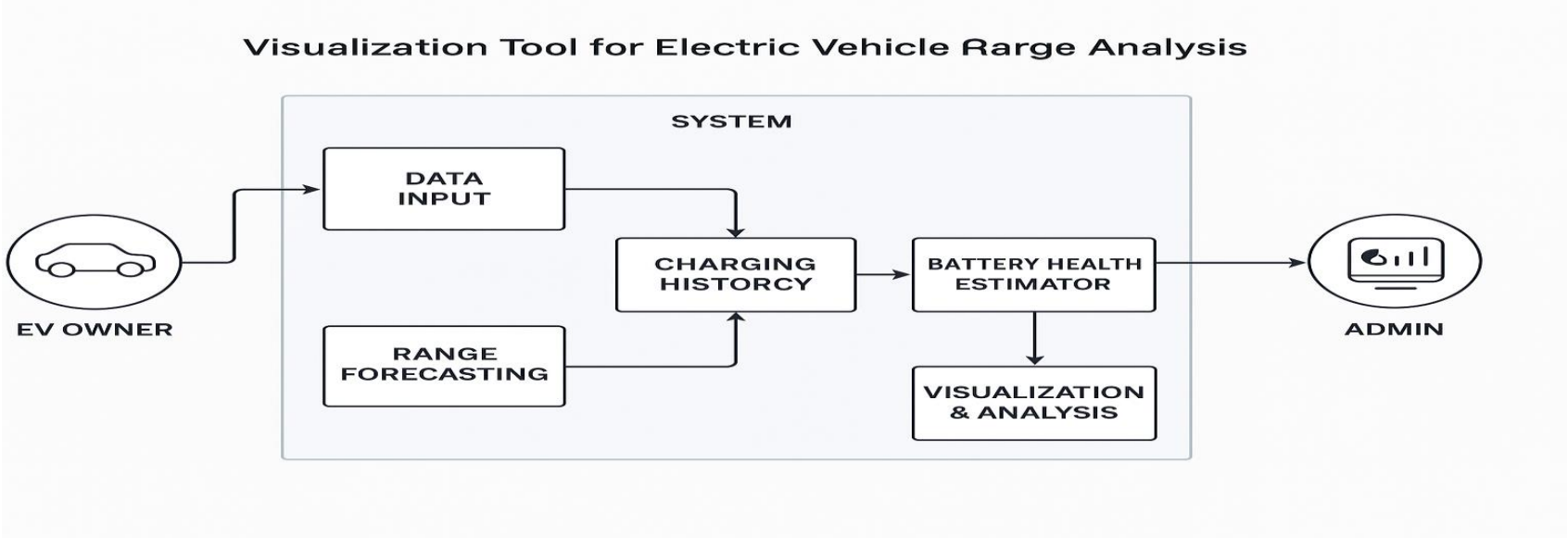

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	Provides interactive views for EV Owners, Admins, and Analysts	HTML, CSS, JavaScript, React or Vue.js
2.	Authentication Module	Handles secure login, signup, and role-based access	Firebase Auth, OAuth 2.0, JWT
3.	Vehicle Input System	Allows users to enter battery %, vehicle model, and trip preferences	JavaScript Form Logic, Python Flask AP
4.	Range Estimation Engine	Calculates real-time range based on input and terrain data	Python, Pandas, NumPy
5.	Charging Station Mapper	Displays nearby stations and highlights reachable zones.	Leaflet.js, Google Maps API
6.	Charging History Module	Visualizes past charging sessions with analytics	Chart.js, D3.js, MongoDB.
7.	Recommendation System	Suggests ideal routes and charging stops	Machine Learning Model, Scikit-learn
8.	Admin Dashboard	Admin access to station management and user analytics	React Admin, Node.js, MongoDB
9.	Database	Stores user data, station info, vehicle profiles	MongoDB, Firebase Firestore
10.	Hosting & Deployment	Runs backend and frontend on scalable infrastructure	Vercel, Netlify, AWS EC2 or Azure App Service.
11.	API Integration Layer	Connects external services like Maps and EV data APIs	REST APIs, GraphQL, Axios.

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Performance	Rapid data response for range calculations, map updates, and analytics	Redis Cache, CDN, Async Processing
2.	Maintainability	Easy to update components like maps, APIs, or authentication without affecting the entire system.	Modular Design, Git-based CI/CD.

S.No	Characteristics	Description	Technology
3.	Scalable Architecture	Ensures the system handles increasing users, stations, and data; follows modular principles for flexibility	Microservices, Docker, Kubernetes
4.	Availability	Stable performance across user scenarios with accurate real-time outputs	Load Balancer (NGINX), Multi-region Hosting (Azure/AWS)
5.	Reliability	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	Automated Testing, Monitoring (Prometheus, Grafana)

References:

https://c4model.com/

https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/

https://www.ibm.com/cloud/architecture

https://aws.amazon.com/architecture

https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d