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

Date	1 February 2026
Team ID	LTVIP2026TMIDS61504
Project Name	visualization tool for electric vehicle charge and range analysis
Maximum Marks	4 Marks

Technical Architecture:

The deliverable shall include the architectural diagram of the proposed **Visualization Tool for Electric Vehicle Charge and Range Analysis**, along with the detailed information specified in **Table-1 (Components & Technologies)** and **Table-2 (Application Characteristics)**.

The architecture should clearly illustrate the system structure, application logic blocks, data storage components, external interfaces, infrastructure deployment (Local / Cloud), and optional machine learning integration.

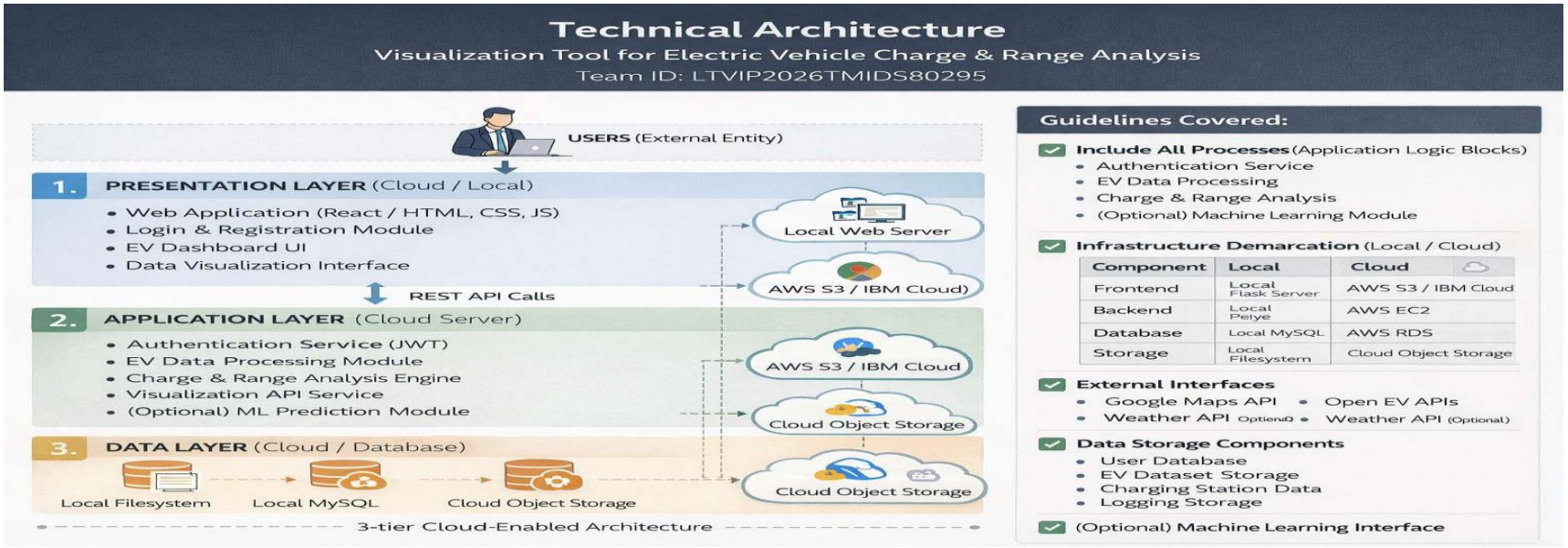

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	Interface through which users interact with the system (Web Dashboard, Login, Charts, Filters).	HTML, CSS, JavaScript, React JS
2.	Application Logic-1	User authentication, session management, and access control logic.	Python (Flask) / Node.js, JWT
3.	Application Logic-2	EV data processing, cleaning, transformation, and aggregation.	Python, Pandas, NumPy
4.	Application Logic-3	Charge & range analysis engine, real vs claimed comparison, charging time calculation.	Python (Analytics Engine), REST APIs
5.	Database	Stores user information, configurations, and metadata.	MySQL / PostgreSQL
6.	Cloud Database	Cloud-based storage for scalable EV datasets and user data.	AWS RDS / IBM Cloudant
7.	File Storage	Stores EV CSV datasets, logs, and backup files.	AWS S3 / Cloud Object Storage / Local Filesystem
8.	External API-1	Fetch charging station locations and mapping services.	Google Maps API
9.	External API-2	Fetch public EV datasets and optional weather impact data.	Open EV APIs / Weather API
10.	Machine Learning Model	Predict battery range and estimate charging duration based on usage patterns (optional).	Scikit-learn / TensorFlow
11.	Infrastructure (Server / Cloud)	Deployment of frontend, backend, and database services on local or cloud infrastructure.	AWS EC2 / IBM Cloud / Docker / Kubernetes

Table-2: Application Characteristics:

S. No	Characteristics	Description	Technology
1.	Open-Source Frameworks	The system uses open-source frontend and backend frameworks for efficient development, flexibility, and community support.	React JS, Flask / FastAPI, Pandas, NumPy
2.	Security Implementations	Secure authentication, encrypted data transmission, password hashing, role-based access control, and protection against common vulnerabilities.	JWT Authentication, HTTPS (SSL/TLS), SHA-256 Hashing, OWASP Guidelines.
3.	Scalable Architecture	The system follows a 3-tier architecture that separates presentation, application logic, and data layers. It supports horizontal scaling using cloud services and containerization.	3-Tier Architecture, Docker, Kubernetes, AWS EC2 / IBM Cloud
4.	Availability	The application ensures high availability using cloud infrastructure, load balancing, and distributed deployment to minimize downtime.	AWS Load Balancer, Cloud Hosting, Distributed Servers
5.	Performance	The system is optimized for fast response time using efficient database queries, caching mechanisms, and optimized API calls. It supports handling multiple concurrent users.	Redis Cache (optional), Optimized SQL Queries, CDN (CloudFront), REST APIs

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