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

Date	31 January 3035
Team ID	LTVIP2026TMIDS38689
Project Name	Electric Motor Temperature Prediction using Machine Learning
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

Architectural process flow:

Input Phase: Industrial maintenance engineers enter data (Voltage, Speed, Torque).

Processing Phase:

* **Sub-process A:** Data is converted to a NumPy array.

Sub-process B: The pre-trained scaler is applied to the input.

Inference Phase: The Random Forest model processes the scaled vector.

Output Phase: The Flask app pushes the result (Motor Temp in °C) to the dashboard.

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	A responsive dashboard for entering motor parameters (Ambient, Torque, Speed, etc.).	HTML5, CSS3, JavaScript (Inter font)
2.	Application Logic-1	Backend routing and API management using Flask.	Python 3.x (Flask)
3.	Application Logic-2	Automated feature scaling and data normalization.	Scikit-learn (StandardScaler)
4.	Application Logic-3	Real-time predictive engine for motor health analytics.	Joblib / Pickle
5.	Database	Historical sensor measurements used for training.	CSV (measures_v2.csv)
6.	Cloud Deployment	Hosting environment for remote monitoring.	IBM Watson / Cloud Foundry
7.	Machine Learning Model	Regression algorithm for temperature forecasting.	Random Forest Regressor
8.	Infrastructure	Environment for development and production hosting.	Local Server / IBM Cloud

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source	Core libraries used for ML and Web serving.	Flask, Pandas, Scikit-learn
2.	Security	Access controls and sanitization of numeric sensor inputs.	Flask-CORS, Python Error Handling
3.	Scalability	3-Tier architecture allows updating the ML model without downtime.	Micro-services ready architecture
4.	Performance	Optimized inference returning results in <200ms.	Pre-serialized .pkl weights

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