

Project Design Phase-II

Data Flow Diagram & User Stories

Date	
Team ID	LTVIP2026TMIDS45779
Project Name	Electric Motor Temperature Prediction Using Machine Learning
Maximum Marks	4 Marks

Data Flow Diagrams:

Electric Motor Temperature Prediction System

A **Data Flow Diagram (DFD)** is a graphical representation of how data moves within the Electric Motor Temperature Prediction System. It clearly illustrates how input data enters the system, how it is processed by different components, and how the final prediction result is generated and displayed.

In this project, the DFD represents the flow of motor sensor parameters (such as ambient temperature, coolant temperature, voltage components, and current components) through preprocessing and machine learning stages to produce the predicted Permanent Magnet (PM) surface temperature.

The DFD helps in understanding:

- How user input is captured
- How data is transformed and scaled
- How the machine learning model processes the data
- Where the trained model and scaler are stored
- How the predicted temperature is returned to the user

Data Flow Diagram (DFD)

Electric Motor Temperature Prediction System

Level 0: Context Diagram

The Level 0 Context Diagram shows a central process box labeled "Electric Motor Temperature Prediction System". It has two external entities, both represented by a "User (Maintenance Engineer)" icon. The first user provides "Input Motor Parameters" to the system. The system outputs the "Predicted Temperature" to the second user, who is shown a display reading "75.3°C".

Level 1: Detailed Data Flow Diagram

The Level 1 Detailed Data Flow Diagram breaks down the system into five numbered processes:

- 1 Input Interface (Web Form)**: Receives "Motor Parameters" from the User and exchanges "HTML/CSS" data with the Flask Backend.
- 2 Flask Backend Processing**: Receives data from the Input Interface and sends it to the Data Preprocessing process.
- 3 Data Preprocessing**: Uses "Joblib" and "MinMaxScaler" to create "Scaled Data". It receives data from the Flask Backend and sends it to the Machine Learning Model.
- 4 Machine Learning Model**: Uses a "Random Forest Regressor". It receives "Scaled Data" and sends the "Predicted Temperature" to the User Interface Display and the Model & Scaler File Storage.
- 5 User Interface Display**: Shows the "Predicted Temperature" (75.3°C) to the User.

Additional components and data flows include:

- Model & Scaler File Storage**: A storage area containing "Random Forest Model model.save" and "MinMaxScaler transform.save". It receives files from the Machine Learning Model and provides them to the Data Preprocessing process.
- Joblib**: Used for saving and loading model and scaler files.
- MinMaxScaler**: A scaler used for data normalization.
- Random Forest Regressor**: The machine learning model used for temperature prediction.
- Scikit-learn**: The library used for the Random Forest model.

