📑 Project Documentation: Electric Motor AI Analysis

1. Project Overview

This project addresses the industrial challenge of monitoring internal motor temperatures without expensive physical sensors. By leveraging **Machine Learning**, we created a "Virtual Sensor" that predicts the **Rotor Temperature** of a Permanent Magnet Synchronous Motor (PMSM) using only external electrical and thermal data.

---

2. Technical Architecture

The system is built on a three-tier architecture:

A. Data Layer

- **Source:** Kaggle PMSM Temperature Dataset (measures_v2.csv).
- **Features:** 7 input parameters including Ambient Temperature, Coolant Temperature, Voltages (u_d, u_q), Currents (i_d, i_q), and Motor Speed.
- **Preprocessing:** Data was normalized using MinMaxScaler to ensure numerical stability during model training.

B. Modeling Layer

- **Algorithm: Decision Tree Regressor**.
- **Accuracy:** Successfully achieved an **R^2 score of 0.96 (96%)**, providing high-fidelity predictions.
- **Persistence:** The trained model and scaler are saved as model.save and transform.save using the pickle library for instant loading.

C. Presentation Layer (UI)

- **Design:** High-end **Glassmorphism** Dashboard.
- **Features:**
    - Dark-themed industrial aesthetic.
    - Real-time "System Analysis" output window.
    - Responsive grid layout for data input.

---

3. Implementation Details

Core Code Snippet (app.py)

The backend handles the routing and model inference:

Python

```
@app.route('/predict', methods=['POST'])
def predict():
    features = [float(x) for x in request.form.values()]
    scaled_features = scaler.transform([np.array(features)])
    prediction = model.predict(scaled_features)
    return render_template('Sensorpredict.html',
                           prediction_text=f'Predicted Rotor Temp: {prediction[0]:.2f}°C')
```

4. How to Execute

1. **Clone Repository:** git clone https://github.com/neelaveni-123/neelaveni_project.git
2. **Navigate to Flask Directory:** cd "archive (2)/flask".
3. **Run Server:** Execute python app.py.
4. **Access UI:** Open http://127.0.0.1:5000 in any web browser.

5. Conclusion

The **Neelaveni Project** successfully demonstrates the integration of complex Machine Learning models into a user-friendly web interface. With a **96% accuracy rate**, it provides a reliable and cost-effective solution for motor health monitoring in industrial environments.