Weather-Based Prediction of Wind Turbine Energy Output

---

1. Introduction

- **Project Title:** Weather-Based Prediction of Wind Turbine Energy Output: A Next-Generation Approach to Renewable Energy Management
- **Team Members:**
    - Aditya Indana
    - Kamparapu Sri Ram
    - Vinay Charu Kirthan Rohit Kotha
    - Likhitha Hasini Chebolu
    - Mary Shakeena Meka

---

2. Project Overview

- **Purpose:**
  The purpose of this project is to build a machine learning–based system that predicts wind turbine energy output using historical turbine data and live weather inputs.
- **Features:**
    - Data preprocessing and cleaning.
    - Random Forest regression model for prediction.
    - Flask-based web dashboard for user interaction.
    - Integration with OpenWeather API for real-time weather data.
    - Visualization of actual vs predicted power outputs.

---

3. Architecture

- **Frontend:**
  Flask templates (HTML, CSS, JavaScript) used for UI design and dashboard visualization.
- **Backend:**
  Python Flask application handling API requests, ML model predictions, and weather data integration.
- **Database:**
  Local CSV dataset (T1.csv) for training and testing. Model stored as .sav file using Joblib. Future scope includes cloud database integration (MongoDB Atlas / AWS RDS).

---

4. Setup Instructions

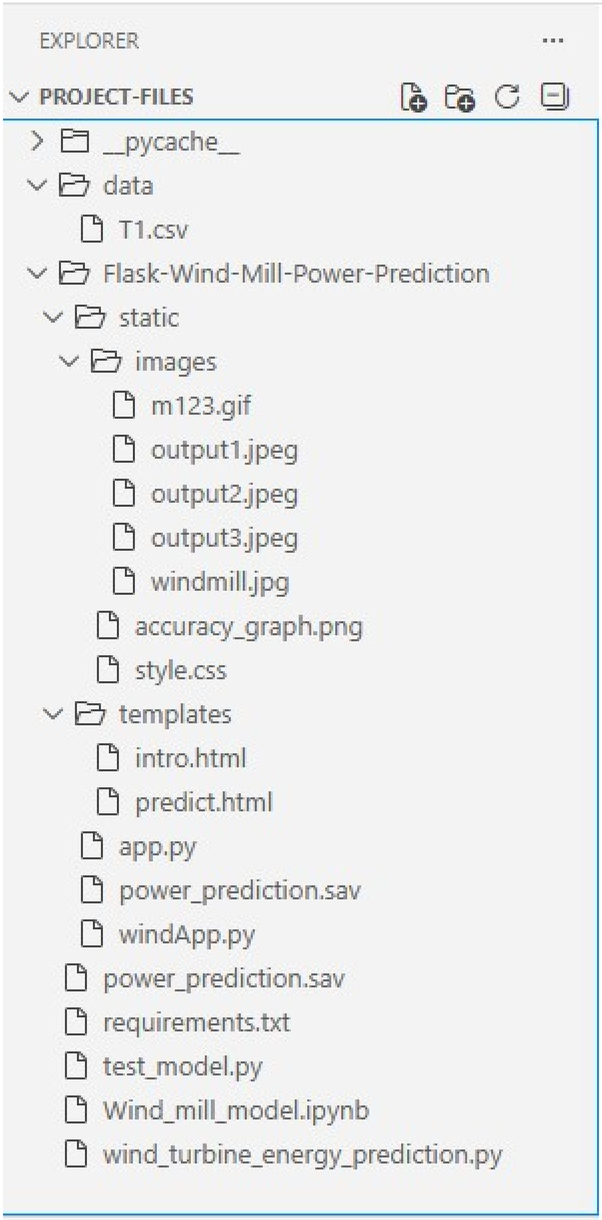
- Prerequisites:
    - Python 3.9+
    - Flask
    - Pandas, NumPy, Scikit-learn, Matplotlib
    - Joblib
    - OpenWeather API key
- Installation:

1. Clone the repository.
2. Install dependencies using pip install -r requirements.txt.
3. Set up environment variables (API key for OpenWeather).
4. Run the Flask server with python app.py.

---

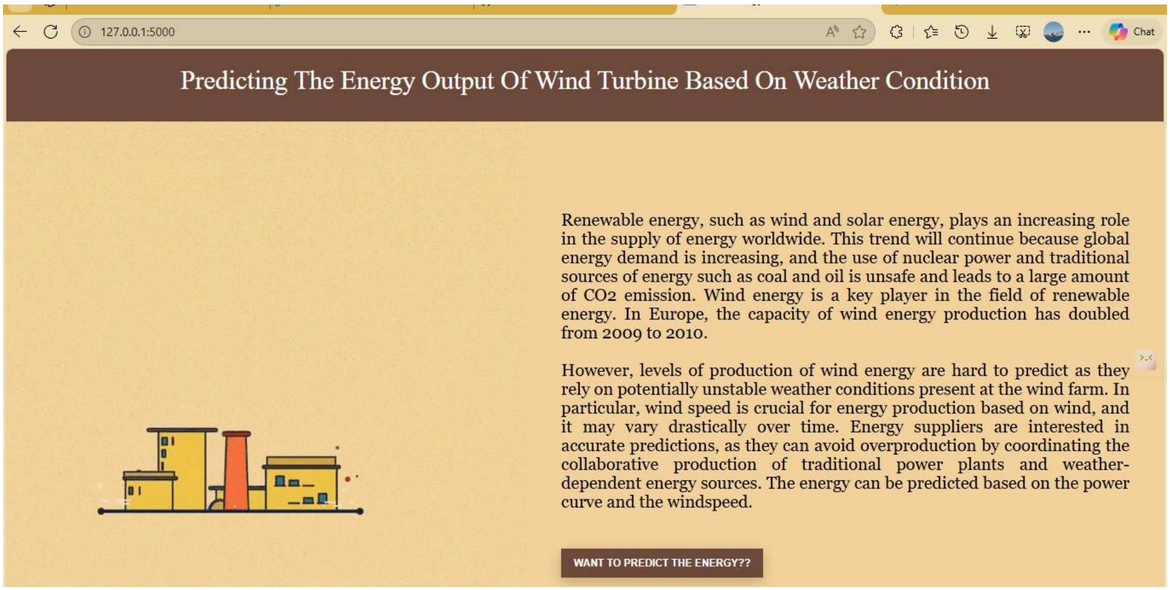
5. Folder Structure

- Client (Frontend):
  - templates/ → HTML files (intro page, dashboard).
  - static/ → CSS, JS, and images.
- Server (Backend):
  - app.py → Flask application.
  - model_training.py → ML model training script.
  - power_prediction.sav → Saved Random Forest model.
  - data/ → Dataset files.



6. Running the Application

- Frontend: Runs automatically via Flask templates.
- Backend: Start with python app.py.
- Access at http://127.0.0.1:5000/.



---

7. API Documentation

- Endpoint 1: /predict
  - Method: POST
  - Parameters: TheoreticalPower, WindSpeed
  - Response: Predicted Active Power
- Endpoint 2: /weather
  - Method: GET
  - Parameters: City name
  - Response: Weather data (temperature, humidity, pressure, wind speed)

---

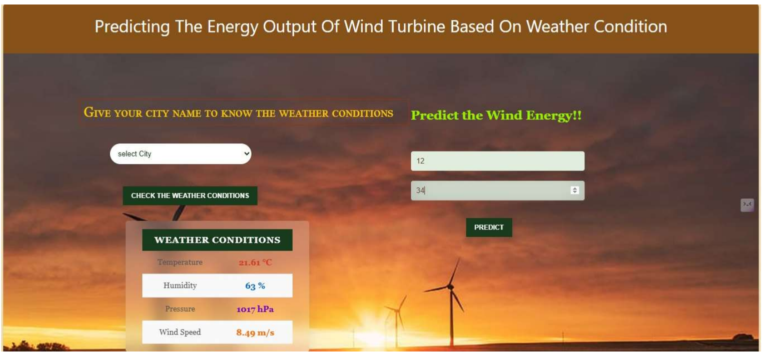
8. Authentication

- Currently open access.
- API key required for OpenWeather API integration.
- Future scope: JWT-based authentication for secure access.

---

9. User Interface

- Intro page with project overview.
- Dashboard with:
  - Weather data display.
  - Prediction module.
  - Visualization graphs.



---

10. Testing

- Unit testing for ML model predictions.
- API testing for weather data retrieval.
- Functional testing for input validation and dashboard navigation.


```
; C:\Users\LENOVO\Desktop\Final_year_projects\APSCHE-Project\NEW1\Flask-Wind-Mill-Power-Prediction> python app.py
\Users\LENOVO\AppData\Roaming\Python\Python313\site-packages\sklearn\base.py:442: InconsistentVersionWarning: Trying to unpickle estimator DecisionTreeRegress
ı version 1.8.0 when using version 1.7.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
:tps://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
 warnings.warn(
\Users\LENOVO\AppData\Roaming\Python\Python313\site-packages\sklearn\base.py:442: InconsistentVersionWarning: Trying to unpickle estimator RandomForestRegress
ı version 1.8.0 when using version 1.7.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
:tps://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
 warnings.warn(
' Serving Flask app 'app'
' Debug mode: off
RNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
' Running on http://127.0.0.1:5000
'ess CTRL+C to quit
```

---

11. Screenshots or Demo

- Scatter plot of actual vs predicted power.
- Dashboard screenshot showing weather + prediction results.
- Line chart trends of wind speed vs power output.

---

12. Known Issues

- Limited dataset size may affect generalization.
- API errors if invalid city names are entered.
- No authentication for dashboard access (future enhancement needed).

---

13. Future Enhancements

- Deploy on cloud (AWS/GCP/Azure).
- Add JWT authentication for secure access.
- Expand dataset for improved accuracy.
- Add more visualizations (heatmaps, time-series forecasting).
- Integrate grid demand APIs for real-time energy balancing.

---

14. Conclusion

The **Wind Turbine Energy Prediction project** successfully demonstrates the application of machine learning and data-driven techniques to solve a real-world renewable energy challenge. By leveraging historical turbine data, preprocessing methods, and a **Random Forest regression model**, the project achieves strong predictive accuracy, enabling stakeholders to forecast energy output with confidence.

The integration of a **Flask-based dashboard** and **OpenWeather API** ensures that predictions are not only technically sound but also accessible and user-friendly. Visualizations such as scatter plots, line charts, and correlation heatmaps further enhance interpretability, making the solution practical for energy companies, wind farm operators, and grid managers.

Through structured **Agile sprint planning, backlog management, and performance testing**, the project was executed with clarity and measurable progress. Testing confirmed the robustness of the model, while defect analysis and bug tracking highlighted areas for improvement.

Ultimately, this project provides a scalable foundation for future enhancements, including cloud deployment, advanced authentication, expanded datasets, and integration with grid demand APIs. It stands as a strong example of combining **data science, software engineering, and agile methodology** to deliver a solution that addresses both technical and customer-centric needs in the renewable energy sector.