DTCA

WIND FARM DIGITAL TWIN WITH ML INTEGRATION

GROUP - 3

PREPARED BY:

Khader Zaahid Umar S20220020287
Murugan S S20220020306
Charvi Palem S20220010160
Sharan Karthick BL S20220020311

INTRODUCTION

Wind energy is a key contributor to global sustainable power generation.

Accurate prediction of wind turbine output is essential for monitoring, planning, and digital-twin applications.

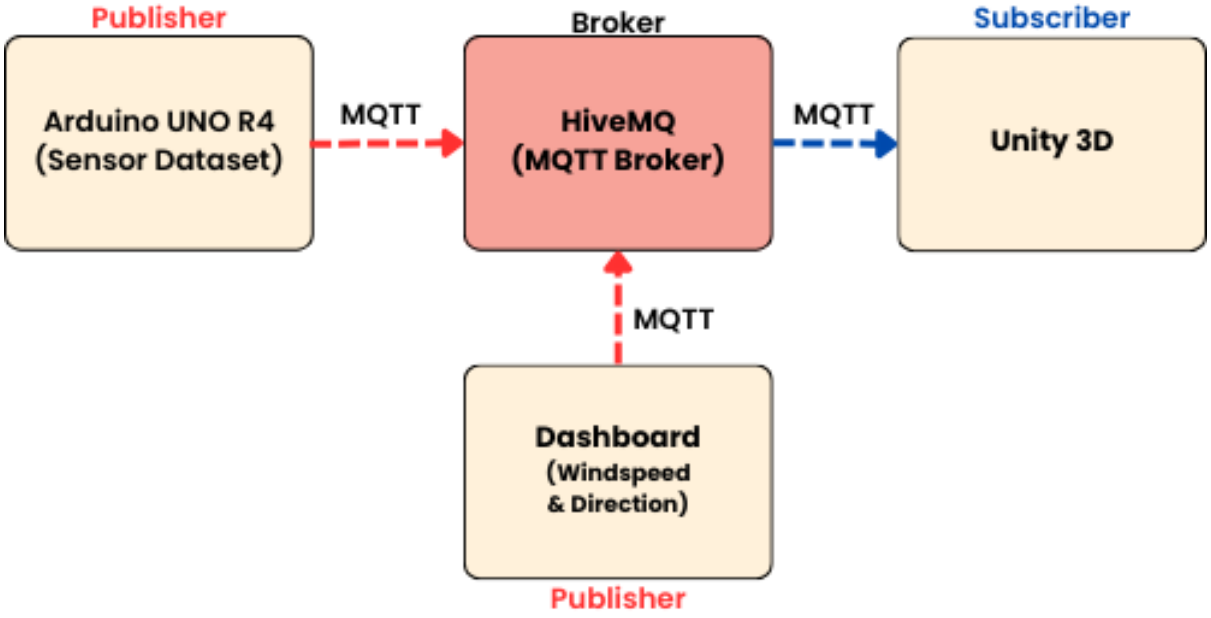
This project develops a machine learning model that predicts turbine Power Generated (in kW) using Wind Speed (m/s) from a real SCADA data on Kaggle.

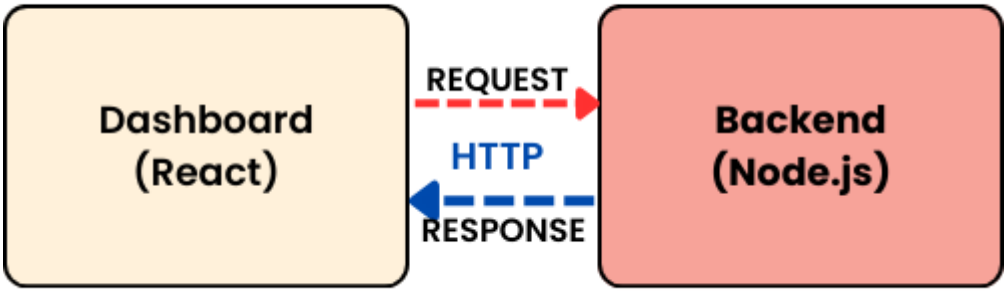
The model forms the core predictive engine of a Unity-based digital twin, which combines:

- Live IoT sensor data (Arduino → MQTT → Unity)
- A predictive simulator (ML model in ONNX format)
- A 3D visual farm environment (Unity)

The final ML model is exported as ONNX, enabling efficient real-time inference inside Unity.

BLOCK DIAGRAM





ML MODEL

1. **Random Forest** Regression Model used in our project uses an ensemble of multiple decision trees, combining their outputs to produce accurate and stable predictions even in noisy real-world datasets.
2. It captures **nonlinear relationships** between wind speed and power output, making it ideal for SCADA-based wind turbine modeling.
3. The model is **highly robust**, **reduces overfitting** through averaging, and **delivers reliable performance** suitable for deployment in real-time digital twin systems.

Our model configuration:

- n_estimators=100 - Create 100 different trees
- max_depth=20 - Each tree can ask up to 20 questions
- min_samples_split=5 - Need at least 5 examples to create a split
- min_samples_leaf=2 - Each final answer must have at least 2 examples
- random_state=42 - Ensures same results every time
- n_jobs=-1 - Use all CPU cores for faster training

Training process:

The model looks at 40,424 training examples and learns patterns:

- "When wind is 7.5 m/s, power is usually around 1,150 kW"
- "When wind is 12 m/s, power is usually around 2,400 kW"
- And so on...

ACCURACY SCORES:

- Training Set Accuracy: 94.75% ($R^2 = 0.9475$)
- Testing Set Accuracy: 89.20% ($R^2 = 0.8920$)
- Model Generalization: Excellent

ERROR METRICS:

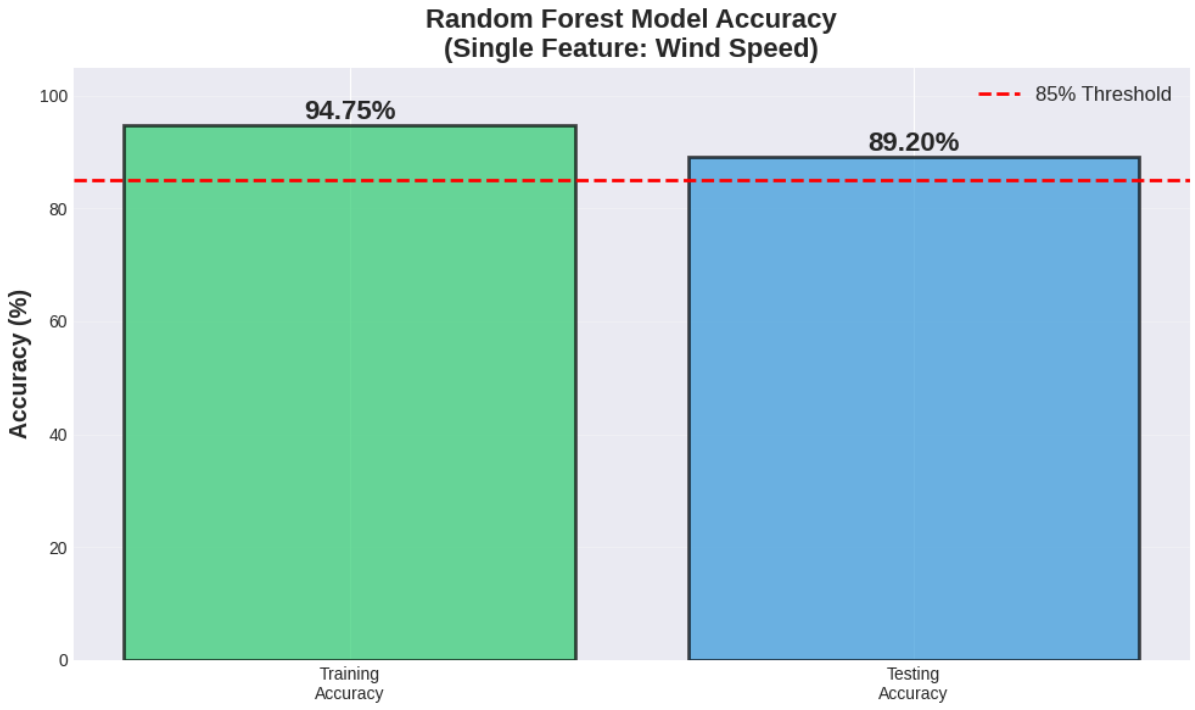
1) TRAINING SET:

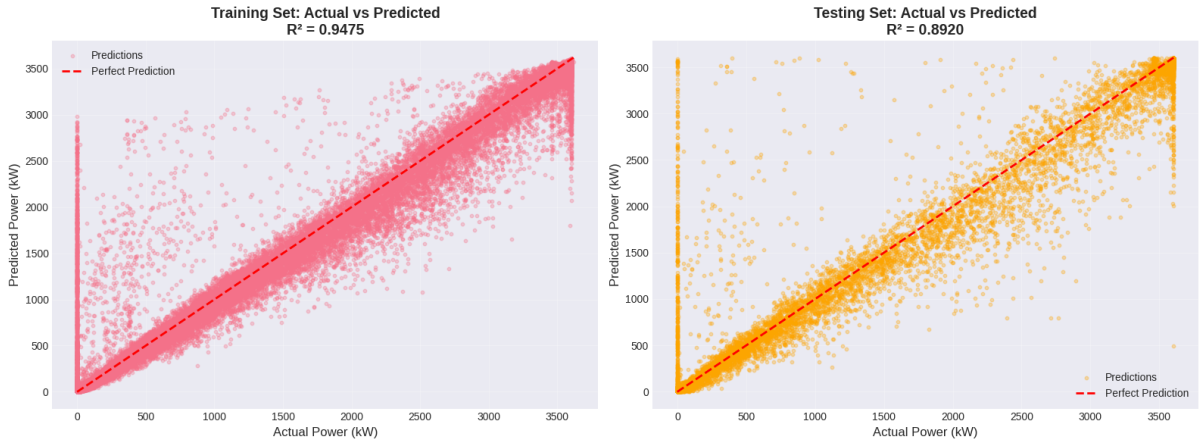
- Root Mean Square Error (RMSE): 301.12 kW
- Mean Absolute Error (MAE): 124.85 kW
- R^2 Score:0.9475

2) TESTING SET:

- Root Mean Square Error (RMSE): 429.23 kW
- Mean Absolute Error (MAE): 181.72 kW
- R^2 Score: 0.8920

MODEL EXPLAINS 89.20% OF VARIANCE IN POWER GENERATION





Visualization In Unity

Digital Twin – Unity Implementation

1). Wind Turbine System

- TurbineController.cs rotates blades based on wind speed
- TurbineController.cs handles yaw → smoothly faces WindZone direction
- Works with all sources: CSV (CsvPlaybackManager.cs), MQTT (MqttWindClient.cs), manual mode (SimulationManager.cs)

2). Wind Environment

- WindManager.cs updates WindZone speed + direction
- WindManager.cs drives particle direction & color
- WindParticles VFX uses elongated particles for wind visualization

3). Data Input Modes (Controlled by SimulationManager.cs)

- CSV playback (CsvPlaybackManager.cs)
- MQTT–Arduino (MqttWindClient.cs)
- MQTT–Website (MqttWindClient.cs)
- Manual mode using inspector booleans in SimulationManager.cs

4). CSV Playback System (CsvPlaybackManager.cs)

- Reads dataset (timestamp, wind speed, direction, active power)
- 144 samples per day
- Interpolates between samples
- Adjustable simulation time scaling (SecondsPerDay)

5). ONNX Power Prediction (OnnxPowerPredictor.cs)

- Loads model.onnx from StreamingAssets
- Feeds wind speed to ONNX model
- Outputs predictedPower
- Accessible globally for graph + logging

6). Power Comparison Logging (OnnxPowerPredictor.cs)

- Creates power_comparison.csv
- Logs: DateTime, WindSpeed, WindDirection, ActivePower, PredictedPower
- Syncs logging with index from CsvPlaybackManager.cs or MQTT timestamps

7). MQTT Integration (MqttWindClient.cs)

- Uses MQTTnet + MQTTnet.Extensions.ManagedClient DLLs
- Secure TLS connection (port 8883)
- Subscribes to JSON payload { windSpeed, windDirection, activePower, timestamp }
- Sends received values to SimulationManager.cs

8). Simulation Manager (SimulationManager.cs)

- Master selection of data sources: CSV / MQTT / Manual
- Feeds wind speed & direction to WindManager.cs
- Feeds actual power to graph via PowerGraphUI.cs
- Coordinates ONNX prediction per frame

9). Real-Time Graph Visualization (PowerGraphUI.cs)

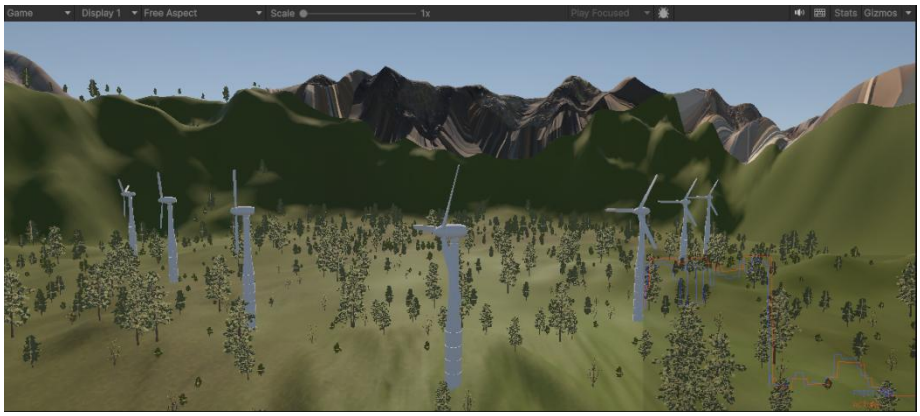
- Draws predicted vs actual power
- Uses runtime texture for graph
- Interpolates plotted points
- Adds legend text: “Predicted”, “Actual”

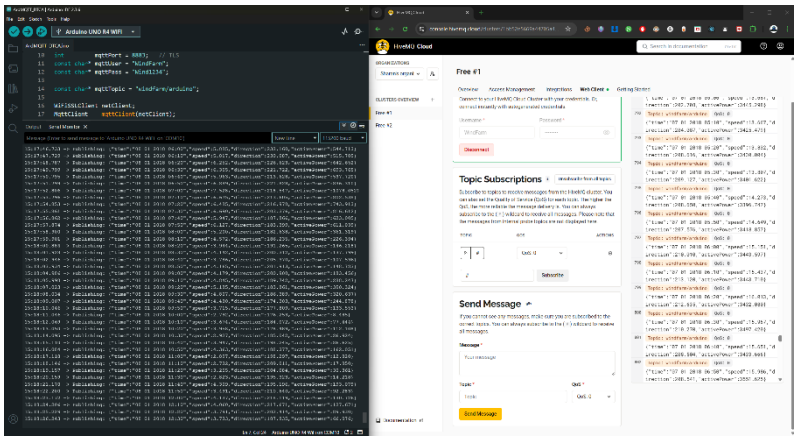
10). Project Output

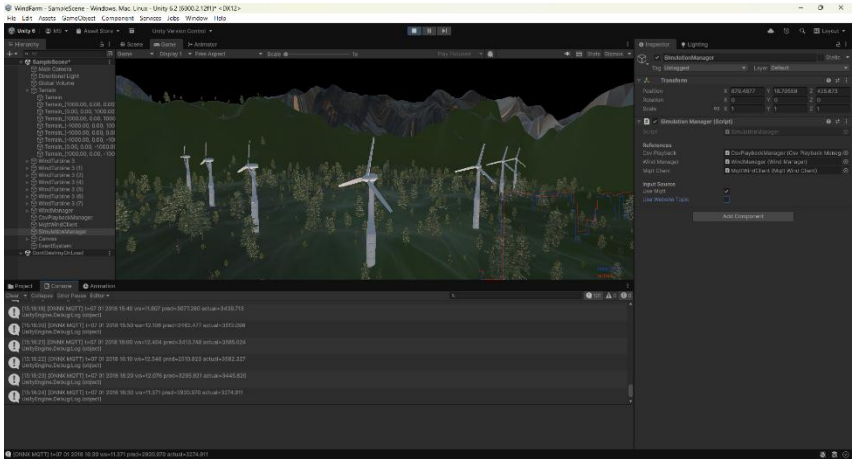
- Animated wind turbines reacting to real wind data
- Real-time ML prediction (via OnnxPowerPredictor.cs)
- Real-time telemetry via MQTT (MqttWindClient.cs)
- Real-time energy graph (PowerGraphUI.cs)
- CSV logging (OnnxPowerPredictor.cs) for MATLAB/Python analysis

RESULTS

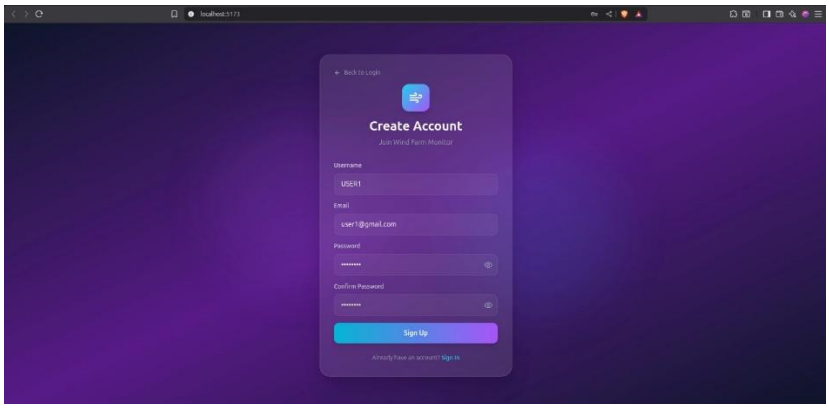
1). Unity 3D



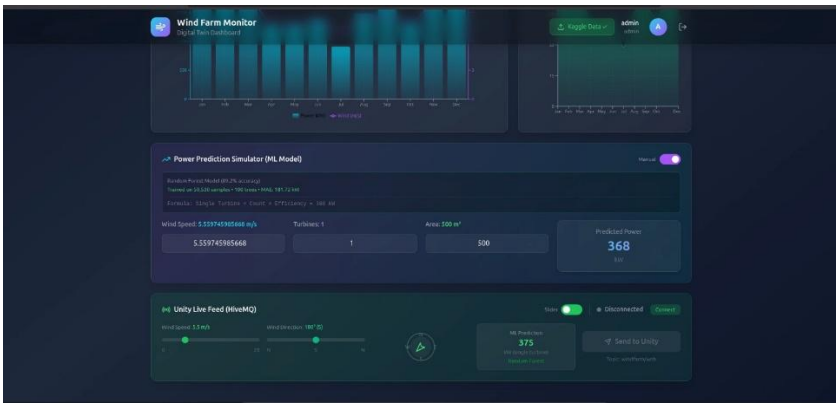




2). Dashboard


Create Account
Join Wind Farm Monitor
Username
USER1
Email
user1@gmail.com
Password
Confirm Password
Sign Up


Wind Farm Monitor
Digital Twin Dashboard
admin
Data Filters
All Months
All Weeks
All Days
Monthly
Filtered Records
39,692
Avg Power
1,665
Avg Wind
8.8
Peak Power
3,619
Power & Wind — Monthly View
Monthly Efficiency


Wind Farm Monitor
Digital Twin Dashboard
admin
Power Prediction Simulator (ML Model)
5.559745985668
1
500
Predicted Power
368
Unity Live Feed (HiveMQ)
Disconnected
375
Send to Unity

THANK YOU