# ⚡ Real-Time Weather Forecaster: Bridging the Prediction Gap

## 🎯 Project Focus: *From Global Reports to Hyper-Local Action*

### Overview

This project solves a crucial problem in weather prediction: the lack of fast, accurate, and localized short-term forecasts (known as **Nowcasting**). We move beyond slow, traditional models to implement a robust **Machine Learning system** that consumes live API data and predicts future conditions—such as **Rainfall, Temperature, and Humidity**—for any city, providing a 5-hour predictive window.

Our system is an agile, dual-model data pipeline designed for **high-impact, low-latency applications** in fields like transport, energy management, and smart agriculture.

## 💡 Why This System is Unique

Traditional forecasting relies on deterministic physics; we use **Data Science**.

### 1. Dual-Model Architecture: Specialized Intelligence

The core of our intelligence lies in two specialized Random Forest models trained on historical data (weather.csv):

| **Model** | **Task** | **Value Proposition** | **Output** |
| --- | --- | --- | --- |
| **RandomForestClassifier** | **Classification** | Predicts the most critical factor: **Rainfall**. | **YES** (1) or **NO** (0) |
| **RandomForestRegressor** | **Regression** | Predicts the **iterative time-series** of continuous values (Temp/Humidity) for the next 5 hours. | 5-hour time-series forecast. |

### 2. The Power of the Iterative Loop

Our Regressor doesn't just predict the next hour; it uses that **predicted value as the input for the following hour's forecast**, creating a reliable, cascading 5-hour predictive window essential for operational planning.

## ⚙️ Setup & Execution: Get Forecasting in Minutes

### Prerequisites

Ensure you have Python 3.x and the necessary scientific and networking libraries installed:

pip install pandas numpy scikit-learn requests pytz

### Configuration (The Two Essentials)

1. API Key (The Live Data Source):  
   The project is currently configured with a placeholder key. You must replace this placeholder with your valid OpenWeatherMap API Key in the API\_Key variable within the notebook.
2. Historical Data (The Brain):  
   Ensure the weather.csv file (your historical training data) is in the same directory as the Jupyter Notebook.

### How to Run

1. Open the Weather\_forecasting.ipynb notebook.
2. Execute all cells sequentially.
3. The final cell will execute the weather\_view() function, prompting you for input:

# The system asks for a city name:  
weather\_view()

1. The output provides the current weather snapshot and the structured **5-Hour Forecast tables**.

## 📈 Performance and Validation

We validate the model's reliability through a proxy metric:

* **Classification Accuracy:** **~82.19%**
  + *This high score validates the model's reliability in predicting rainfall based on the current feature set, demonstrating effective feature engineering.*
* **Demonstrated Forecast (Example):**
  + The iterative regression model successfully demonstrated predictive capacity, showing logical trends (e.g., predicted temperature increase over morning hours in the Delhi test run).