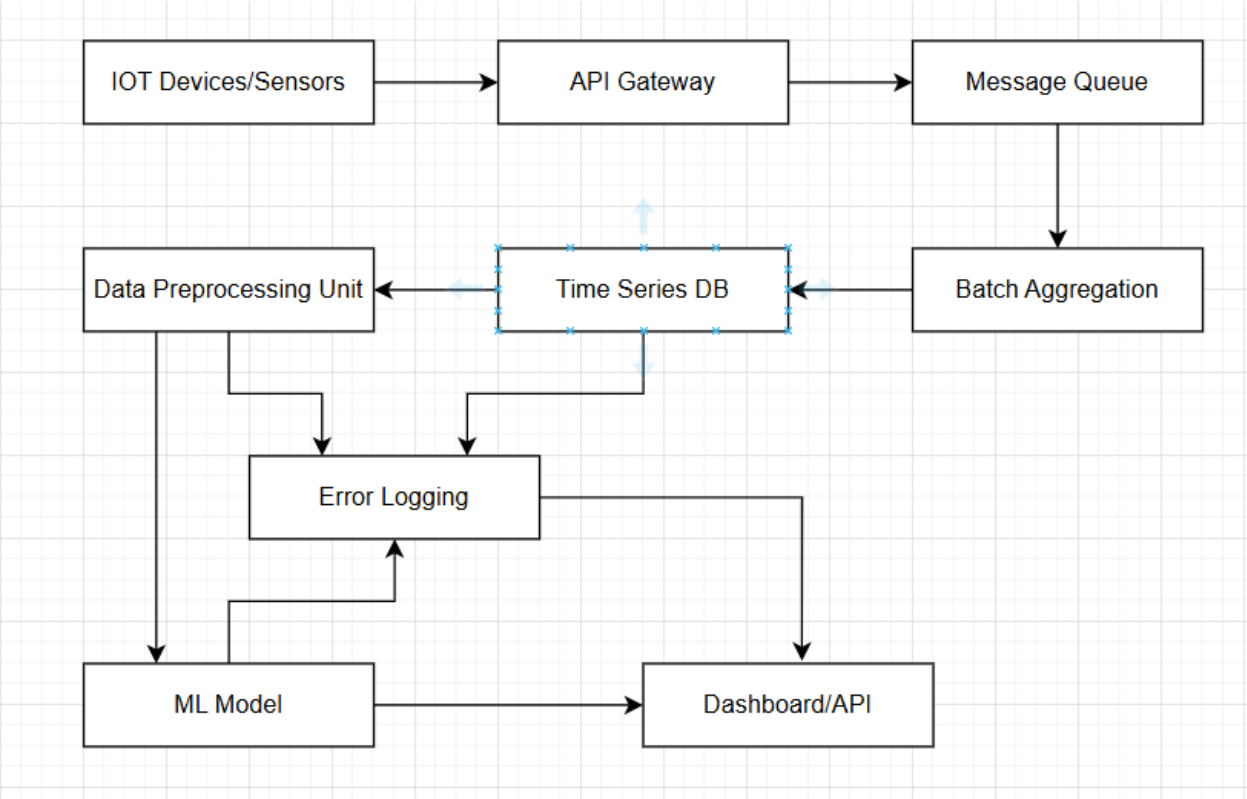
Machine Learning-Based Rain Prediction System

Team – Hiccups

Submitting date – 10/03/2025

System Diagram



System Component Descriptions

1. IoT Devices/Sensors

These devices collect real-time weather data every minute, measuring parameters like temperature, humidity, wind speed, and cloud cover. They form the foundation of the system, but since they can malfunction, redundancy (multiple sensors) and fallback mechanisms are implemented to handle missing or faulty data. Without accurate data from these devices, the system would lack the foundation needed for reliable predictions.

2. API Gateway

The API Gateway acts as the entry point for all incoming data from IoT devices. It validates the data to ensure it meets expected formats and ranges (e.g., temperature between -50°C and 50°C) before forwarding it to the Message Queue. By filtering out invalid or malformed data at this stage, the API Gateway prevents downstream components from being overwhelmed by errors or inconsistencies. If the API Gateway detects missing or faulty data, it logs the issue and routes valid data to the Message Queue, triggering alerts if a sensor stops sending data or sends unrealistic values.

3. Message Queue

The Message Queue temporarily stores incoming data streams from the API Gateway, acting as a buffer to handle high-frequency data and prevent data loss during spikes or system failures. Real-time data ingestion can be unpredictable, with sudden bursts of data or temporary downtimes. The Message Queue decouples data producers (IoT devices) from consumers (processing systems), ensuring smooth and reliable data flow. If downstream systems (e.g., Batch Aggregation) fail, the Message Queue retains the data until they recover, ensuring no data is lost during system interruptions.

4. Batch Aggregation

This component retrieves minute-level data from the Message Queue and aggregates it into daily features required for prediction, such as mean temperature, maximum humidity, and average wind speed. Raw minute-level data is too granular for machine learning models, so aggregation simplifies the dataset while retaining meaningful patterns. If aggregation fails due to corrupted or missing data, the system retries the process and uses fallback mechanisms (e.g., historical averages) to fill gaps. Errors are logged, and alerts are sent to the maintenance team.

5. Time-Series Database

The Time-Series Database stores both raw and aggregated data for efficient querying. It is optimized for time-stamped data, making it easy to retrieve historical trends or analyze specific time periods. Storing data in a structured and queryable format is essential for training the ML model and generating predictions. The database also archives older data for long-term analysis. If the database goes offline, data is temporarily stored in a backup location (e.g., cloud storage). Once the database recovers, the data is re-ingested, with alerts notifying stakeholders of any downtime.

6. Data Preprocessing Unit

This unit cleans and transforms the aggregated data into a format suitable for the ML model. It handles tasks like imputing missing values, detecting anomalies, and engineering new features (e.g., cyclic encoding, interaction terms). Clean and well-structured data is critical for the accuracy of the ML model, as poor-quality data can lead to unreliable predictions. If preprocessing fails, the system uses fallback values (e.g., historical averages) and flags the issue for review. Errors are logged, and alerts are sent to the relevant teams.

7. ML Model

The ML Model is the heart of the system, responsible for predicting the probability of rain for the next 21 days. It uses the preprocessed data to generate predictions based on patterns learned during training. The model translates raw data into actionable insights, enabling the project manager to make informed decisions about irrigation, planting, and harvesting. If the model fails to generate predictions (e.g., due to corrupted input data), the system retries the process with cleaned or imputed data. In extreme cases, default predictions (e.g., historical averages) are used until the issue is resolved.

8. Dashboard/API

The Dashboard/API presents predictions in a user-friendly format for the project manager. It displays rain probabilities for the next 21 days using visualizations like line charts and bar charts. It also exposes predictions via an API for integration with other systems, enabling seamless connectivity with mobile apps or other platforms. A clear and intuitive interface ensures that stakeholders can easily interpret the predictions and take action. If the dashboard or API crashes, predictions are temporarily displayed in a backup format (e.g., raw JSON or CSV), with alerts notifying stakeholders of the issue.

9. Error Logging

Error Logging is a centralized system that records issues occurring at any stage of the pipeline. It captures details like timestamps, error types, and affected data points for debugging and monitoring. Comprehensive error logging helps identify and resolve issues quickly, minimizing downtime and ensuring the system remains operational. Error Logging is connected to key components like the Data Preprocessing Unit, ML Model, and Dashboard/API to capture errors and trigger alerts when necessary.