# The Weather APP

IOT GEOSPATIAL HACKATHON 2024

*Submitted by*

Dr. Minu R I

Adithya Krishna [RA2211003011212]

Vansh Bharadwaj [RA2211003011219]

P Sri Sathwik [RA2211003011253]

Om Bandyopadhyay[RA2211003011263]

*from*



*A report on IOT Geospatial Hackathon by*

**IIT BOMBAY**

**DEPARTMENT OF COMPUTING TECHNOLOGIES   
COLLEGE OF ENGINEERING AND TECHNOLOGY   
SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

# KATTANKULATHUR – 603203 September 2024

**TEAM DETAILS**

**TEAM NAME:** TEAMZEUS

**TEAM MEMBERS**

|  |  |
| --- | --- |
| Dr. Minu R I | Team Lead |
| Adithya Krishna | Developer (IOT) |
| P Sri Sathwik | Developer(Machine Learning) |
| Vansh Bharadwaj | Developer(Backend & IOT) |
| Om Bandyopadhyay | UI/UX & Developer(Frontend) |

**TABLE OF CONTENT**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Content** | **Page Number** |
| 1 | Problem statement | 1 |
| 2 | Objective | 2 |
| 3 | Description of the Project | 3 |
| 4 | Hardware/Software used | 4 |
| 5 | Key feature | 6 |
| 6 | Circuit diagram | 7 |
| 7 | Design prototype | 8 |
| 8 | Results | 9 |
| 9 | Challenges Encounter | 13 |
| 10 | Conclusion | 15 |

**Problem statement**

Design and develop an IoT device using open hardware components such as Arduino or Raspberry Pi, equipped with sensors to monitor environmental parameters like temperature, humidity, air quality, and soil moisture. Integrate GPS module for geolocation data. Create a GIS application to visualize and analyze the collected data, allowing users to identify patterns and trends in environmental conditions over time and space.

**Objective**

The objective of this problem statement is to use Arduino or Raspberry Pi to monitor environmental parameters like temperature, humidity, light intensity, soil moisture and Air Quality. Also, to integrate a GPS Module to share the geolocation. And finally to provide a Graphic User Interface which includes data recorded by the components in graphical manner for users to analyse the pattern in the change of parameters in their environment or a distant environment.

**Description of the Project**

This project consists of components like Raspberry , MQ135, Soil Moisture Module, LDR, Resistor, DHT11 Module, NEO 6M GPS Module, Jumper Wires and Bread Board to control the components, read Air Quality Index, Soil Moisture, Light Intensity, temperature, Humidity, Location, to connect to integrate everything respectively. We then store all data in an SQL database from where the Machine Learning model and the website gets its data from. We integrated a Machine learning model to predict the upcoming parameters. To visualize the data we made a fully fledged web application on local Network to present the location, current readings, the graph pattern that represents the before readings and the prediction from the machine learning.

The machine learning is programmed to take data entries of from the location it is installed at and then train it and predict parameters and then save it into a new database . Finally the server in which Machine learning is hosted sends data to the web application.

**Hardware/Software Used**

**Hardware:**

* Raspberry pi 3:

Has its own Linux OS so it's easy to control all component and save data in and host server.

* MQ135:

Used to sense ammonia gas, Sulfide, benzene series steam, also can monitor smoke and other toxic gases well.

* DHT11:

Used to read temperature and humidity from environment.

* LDR

When combined with a resistor its used to detect sunlight, to differ from cloudy ,partial cloudy, and sunny.

* Soil Moisture:

Used to detect soil moisture

* Neo 6M GPS:

Used in various applications that require accurate positioning and navigation capabilities.

* Breadboard:

To integrate all components.

* Jumper Wires:

To connect component.

* USB cable :

To charge the raspberry pi.

**Software:**

* Raspbian OS
* PUTTY:

To gain access of raspberry pi then to program and run the code.

* SSH

Another way to gain access of raspbian OS.

* Python

We used python to code the whole program for the IOT part and data storing and collection part.

* SQLite

To save data that was read by the component . Used due to its lightweight. And versatility and flexibility in saving database and copying the database to other OS.

* Circuit Designer(online)

To draw the circuit online for a clarity of reproducing the connection once disconnected.

* SCP

To get the database from raspberry for machine learning.

* Prophet Model

To run the Machine Learning Model we use Prohpet model by meta as it’s a time series model which is perfect for predictions. This model is also good in covering up missing data.

* Flask

To run and make the backend.

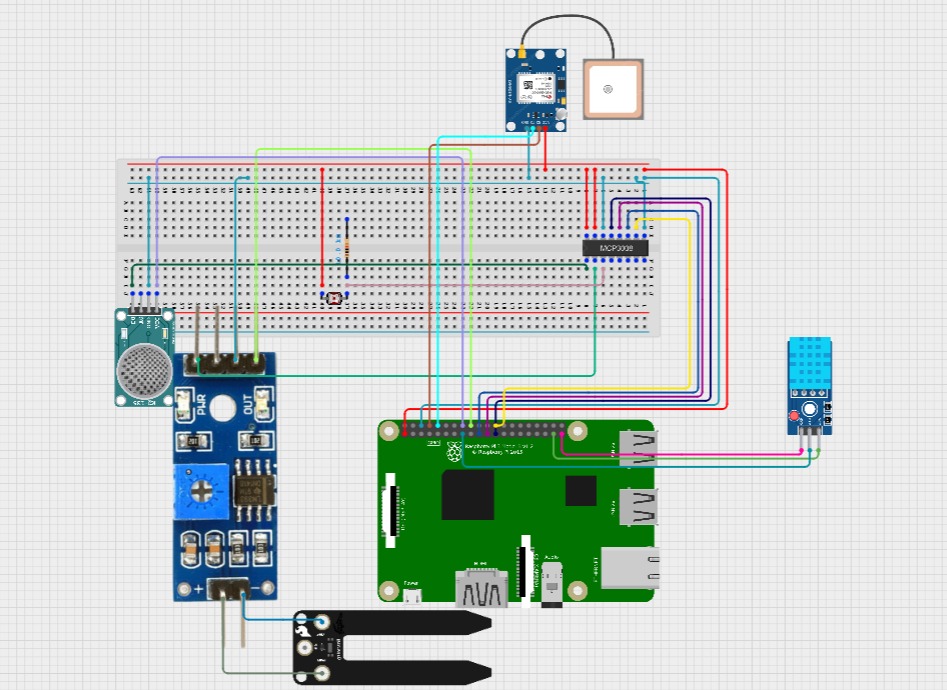
* Chart.js

To design the charts in frontend.

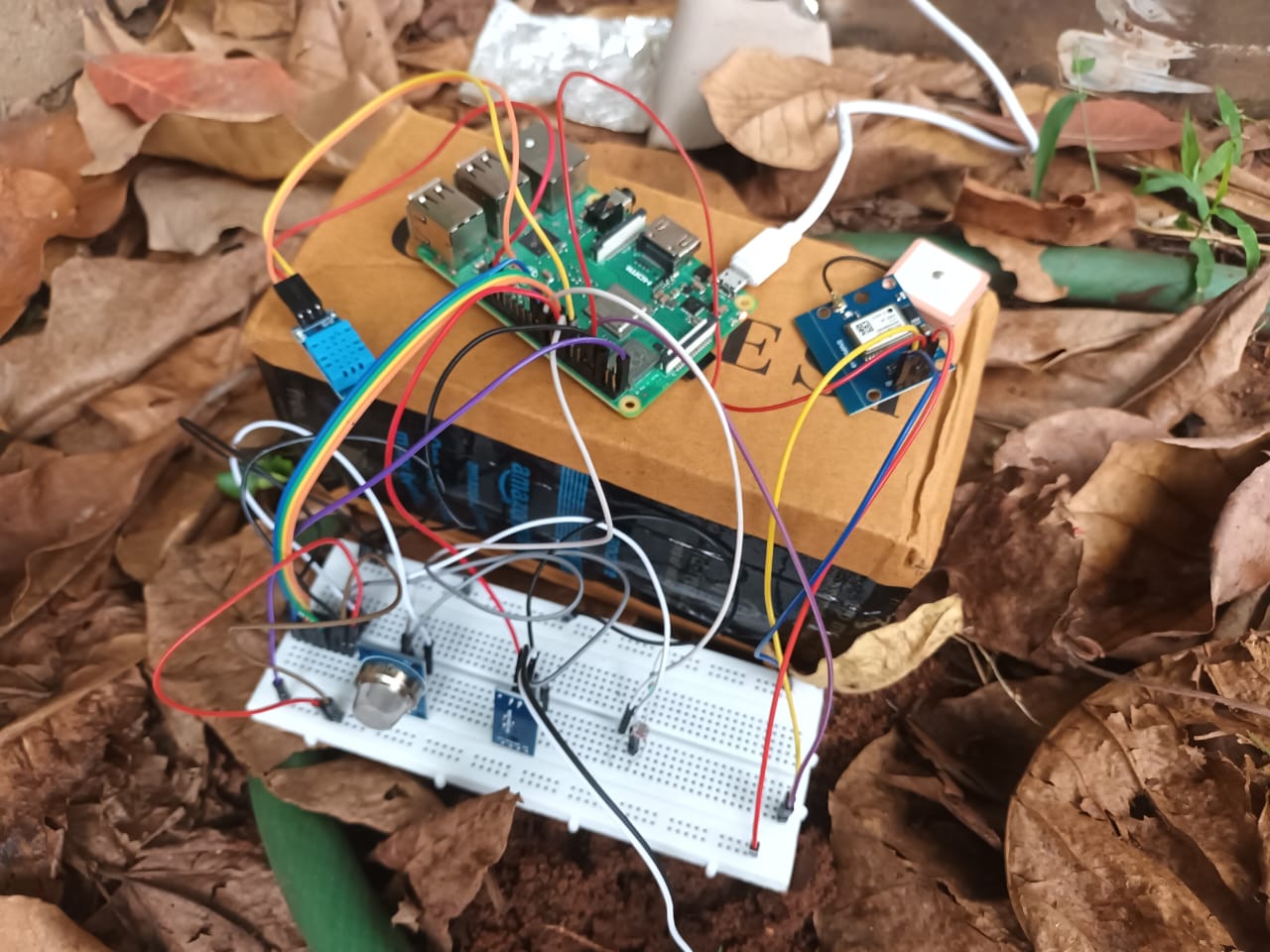
**Key Features**

* **Multi-Sensor Integration**:  
  Monitors environmental parameters including temperature, humidity(via DHT11), air quality (via MQ135), soil moisture, and light intensity (using an LDR).
* **GPS Geolocation**:  
  Integrates a NEO 6M GPS module to capture and store accurate geolocation data for each environmental reading.
* **Real-Time Data Collection**:  
  Continuously collects and updates data from various sensors, ensuring users have access to the latest environmental information.
* **Data Storage and Management**:  
  Utilizes SQLite for efficient and lightweight data storage, enabling easy data retrieval and management for further analysis.
* **Machine Learning Integration**:  
  Incorporates a machine learning model to predict future parameter trends based on historical data, providing valuable insights for users.
* **WebApp-based Visualization**:  
  Develops a user-friendly web application that displays collected data in graphical formats, allowing users to visualize patterns and trends over time and space.
* **Energy Efficiency**:  
  Designed to be energy-efficient by putting components in deep sleep, making it suitable for long-term deployment in various environmental settings.

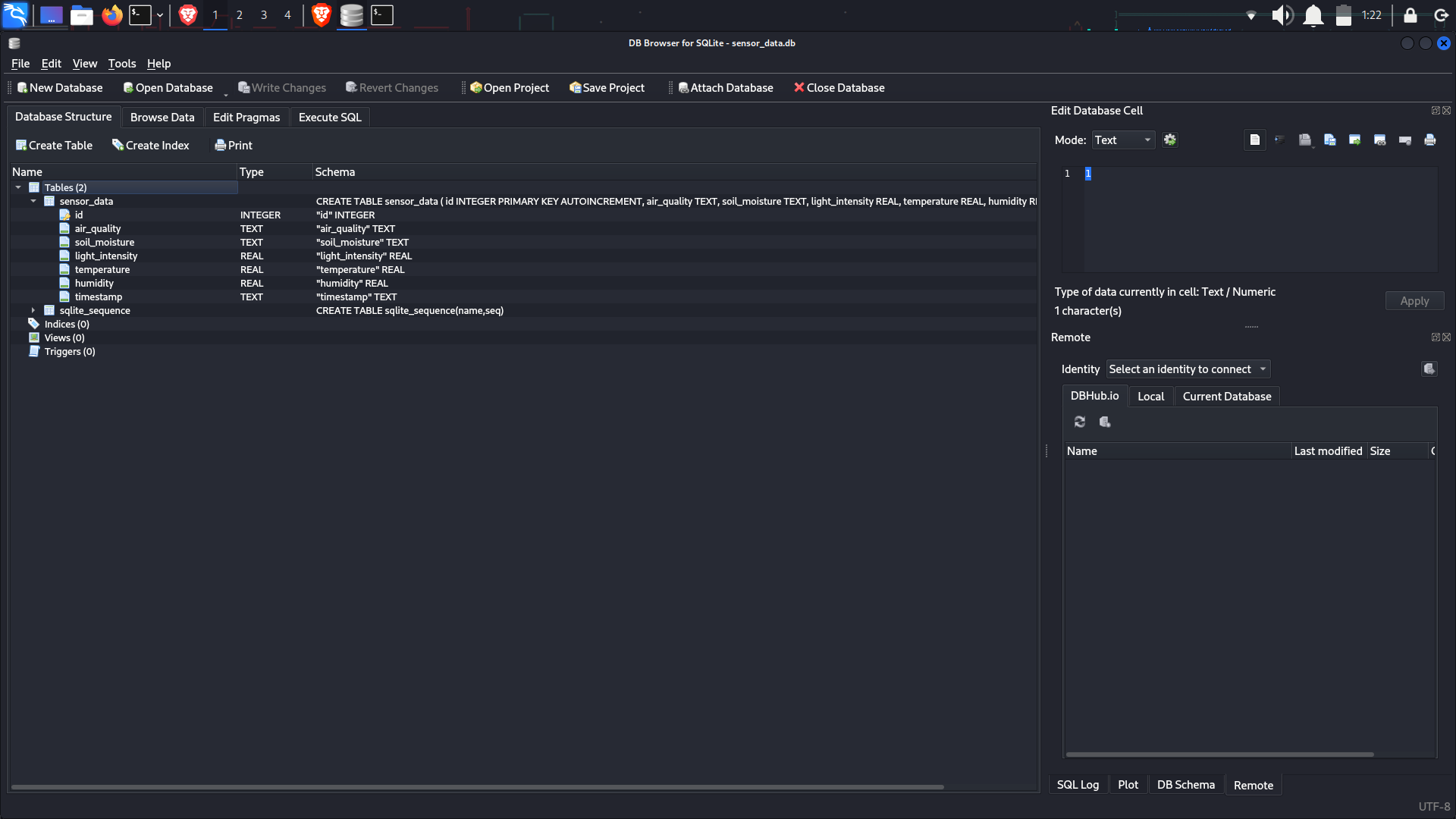
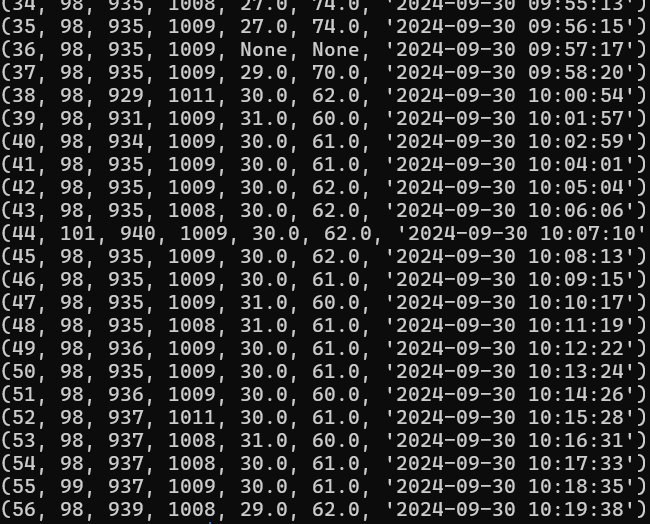
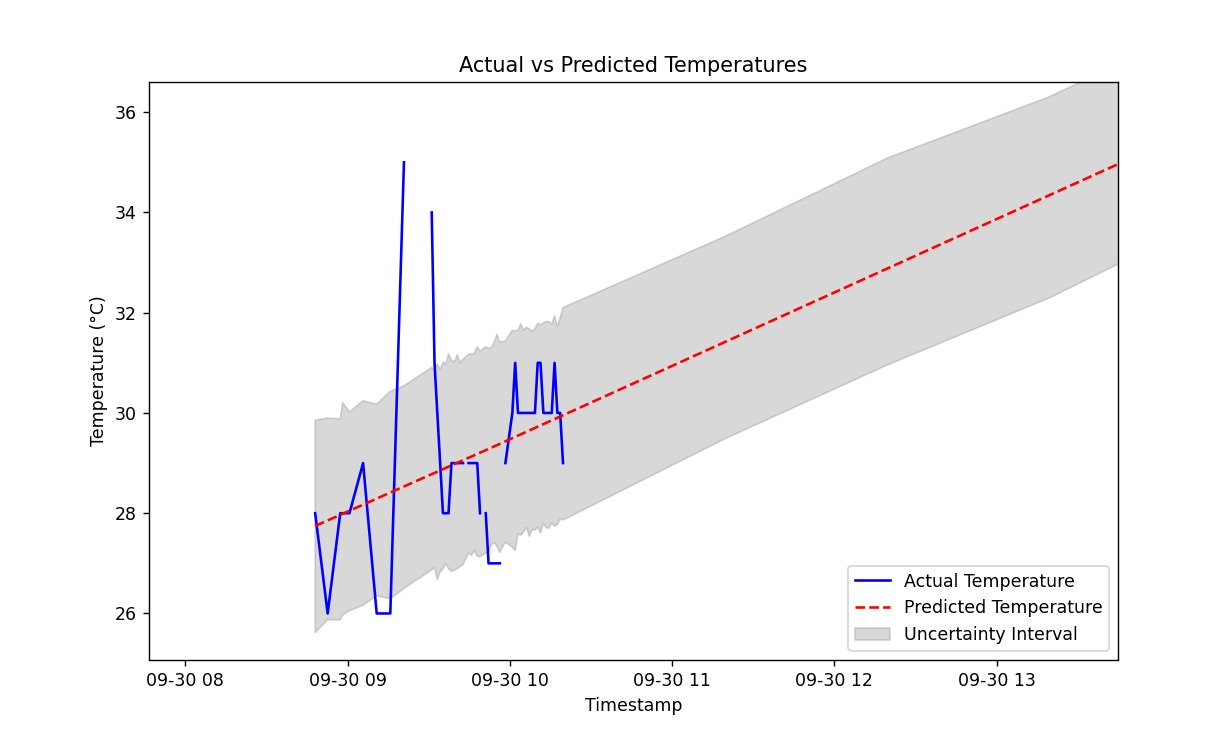
**BLOCK DIAGRAM**

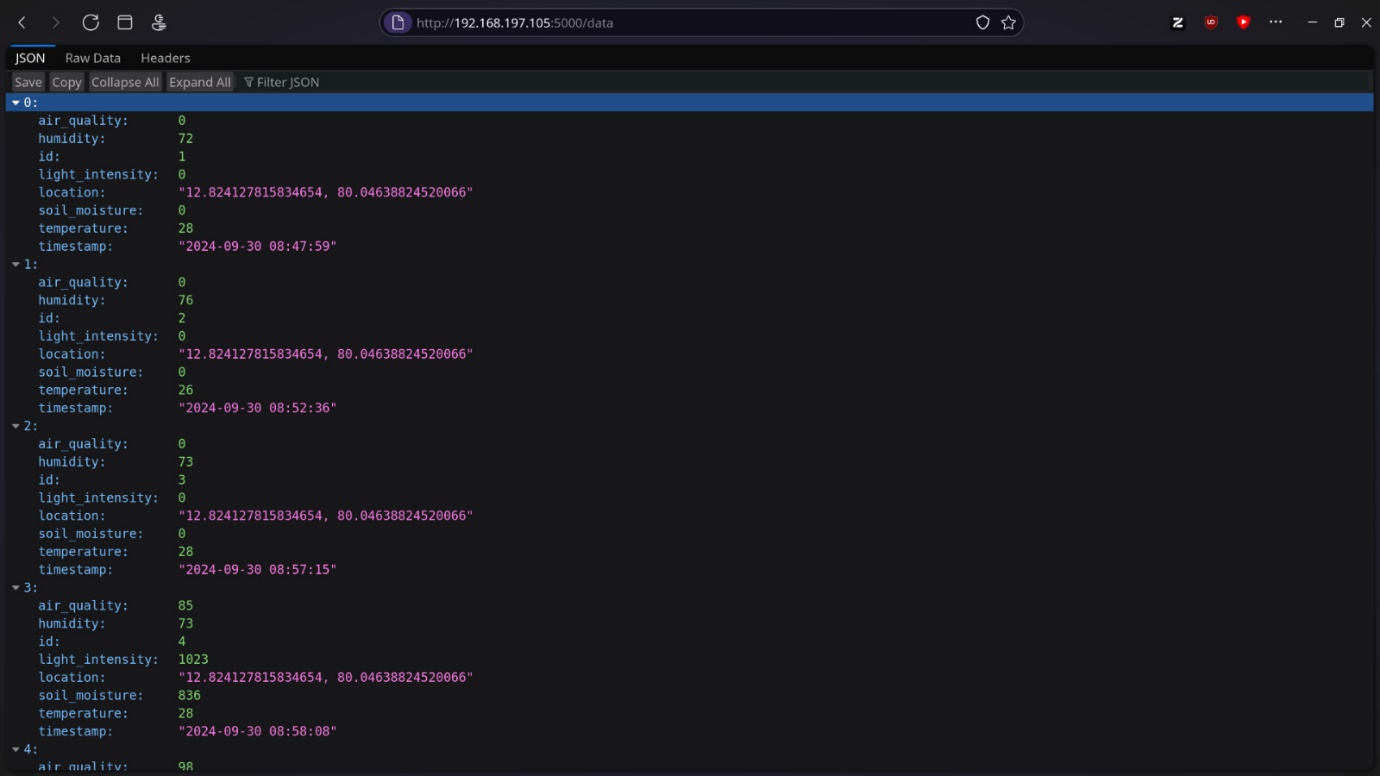


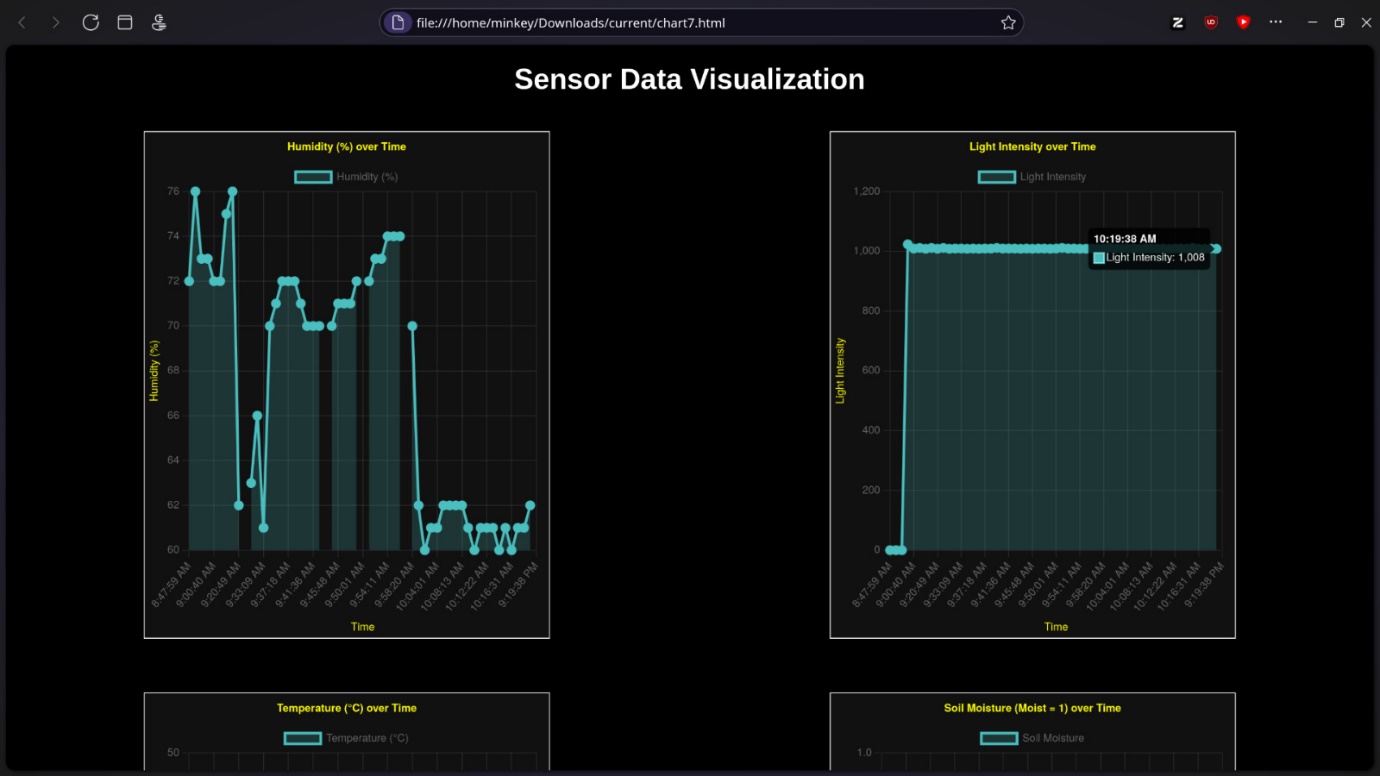
**Design prototype**

 **Results**

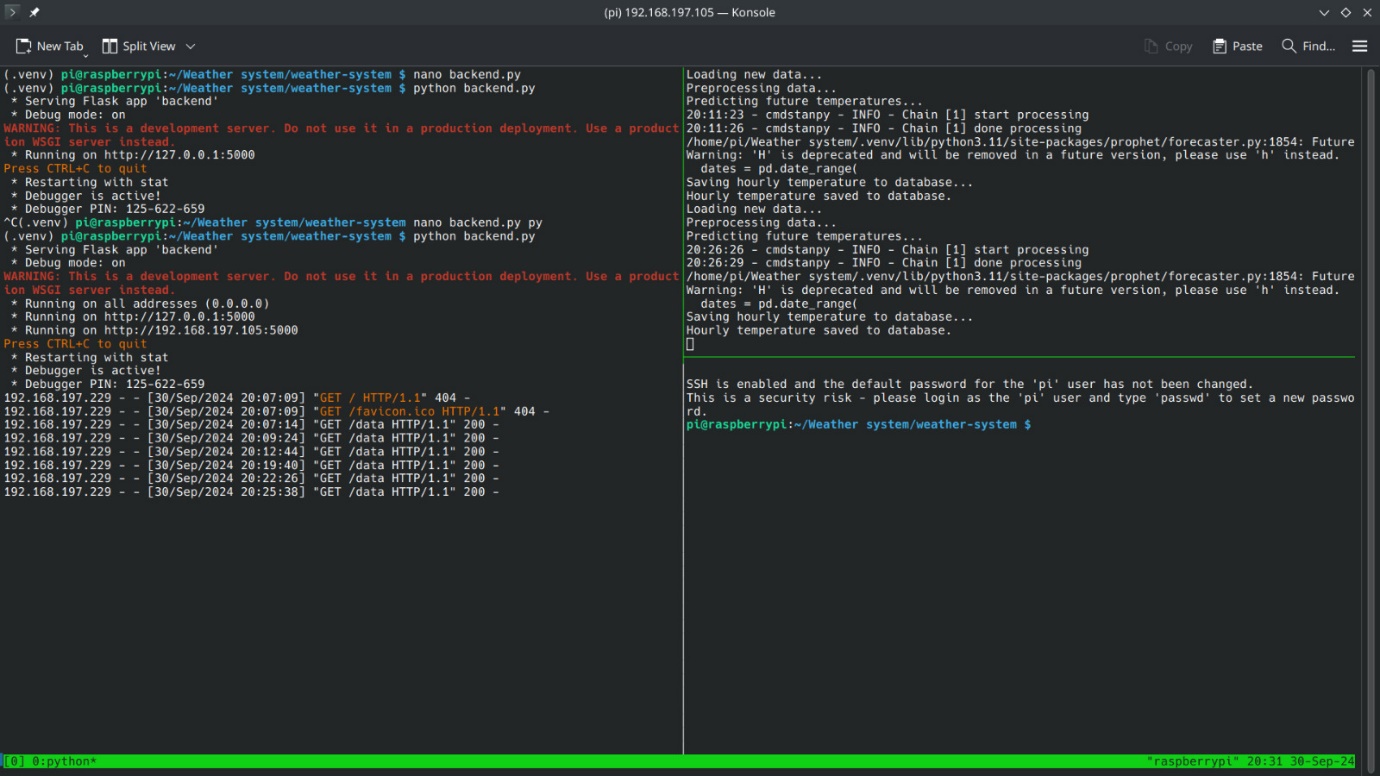
We successfully managed to integrate all the components required plus an extra component,  
managed to integrate a machine learning model for prediction, and was successfully able to show the output on the website, with live data reading being entered in the SQL database   
of every 5 minutes also being updated in the website .









**Challenges Encountered**

* **Analog Signal Issues with Raspberry Pi:**

The Raspberry Pi does not natively support analog inputs, which led to difficulties when trying to read data from sensors like the MQ135 and Soil Moisture Sensor. To overcome this, we integrated an MCP3008 Analog-to-Digital Converter (ADC), allowing us to convert analog signals from these sensors into digital values that the Raspberry Pi could process. This step was crucial for accurate data collection.

* **Database Management Transition:**

Initially, we opted to use CSV files for data storage, but we quickly encountered limitations regarding scalability and data retrieval efficiency. As the project evolved, we switched to SQLite, which proved to be a more suitable solution for our web application. SQLite offered better performance for managing larger datasets and simplified interactions with our data during visualization and machine learning model integration.

* **GPS Module Malfunction:**

Near the project's completion, we faced a critical issue with the NEO 6M GPS module, which unexpectedly stopped functioning. This setback posed challenges in collecting accurate geolocation data. To address this, we conducted thorough troubleshooting, including checking connections, updating firmware, and replacing the module. This experience underscored the importance of having backup components and contingency plans for critical hardware in future projects.

* **DHT11 Sensor Longevity:**

The DHT11 sensor is known to be vulnerable to degradation when powered continuously for extended periods. To mitigate this issue, we implemented a strategy to power the sensors only during data collection periods. After each reading, the sensors are put into a deep sleep mode. This approach not only increased the longevity of the sensors but also significantly improved energy efficiency, allowing for longer deployment times without battery replacement.

* **Sensor Calibration:**

Ensuring the sensors provided accurate readings required calibration and fine-tuning. This process involved comparing readings from our sensors with those from reliable external sources, which was time-consuming but necessary for data validity.

* **Insufficient Data:**

Time-series models like Prophet thrive on longer-term data that exhibit enough variation to capture trends, seasonality, and noise patterns. In your case, the limited amount of sensor data available (likely only 60 data points) constrained the model’s ability to detect these patterns. Without enough history, the model can’t generalize well or learn from past behaviour. With sparse or incomplete data, the model doesn't have enough training material to learn meaningful patterns. Short datasets also make it hard to detect recurring patterns like daily or weekly cycles, which are crucial in temperature forecasting.

* **Noise in Data:**

Sensor data often contains noise, such as abrupt fluctuations or faulty readings, which can distort the learning process. If not handled properly (e.g., through data smoothing or outlier detection), the model may end up focusing on this noise rather than the underlying trend. This could explain why your predictions are inaccurate, as the model may have been distracted by irrelevant data points.

**Conclusion**

In this Project, we successfully managed to integrate multiple sensors ,both digital or analog sensors and read data from the environment such as temperature, humidity, air quality, soil moisture and light intensity. We needed to use MCP3008 to change analog signals from MQ135, Soil Moisture and LDR to digital signals for raspberry. Then we saved all the values into SQLite so that it reduces weight load on raspberry.

Prophet Model is done in the pi itself as its light weight, and then the machine learning prediction is saved in a new database and both new and old database is hosted for the web application.

Overall, this project demonstrated the versatility and efficiency of Raspberry Pi in sensor-based data collection. The system can be further optimized by adjusting sensor sensitivity, reducing power consumption, and enhancing data visualization techniques. These improvements would make the system more robust for long-term monitoring in practical applications like smart agriculture or environmental tracking.