Weather Reporting System Using IOT

Prepared by

Mohamed Abdul-El Hamid	**IT**
Mohamed Adel	**IT**
Mohamed Atef	**IT**
Mohamed Taher	**IT**
Hamdy Salah	**IT**
Mohamed Rushdy	**IT**
Mohamed Abdul-El Mohsen	**IT**

Supervisor

Dr. Fatma Abd-Alhaleem

2023-2024

Table of Contents

Abstract

Climatic change and environmental monitoring have received much attention recently. Anyone wants to stay updated about the latest weather conditions of any place. Given the rapid pace of global changes, it's essential to have a dynamically updated weather reporting system. Our system is dedicated to accurate and real-time weather reporting thus enabling us to directly check the weather state online. The most important factor that inspired us to design our system is the lack of a local meteorological unit at our university, which led to incorrect decisions being made in previous years regarding canceling classes on certain days due to bad weather. Therefore, we decided, to the extent of available capabilities, to create a miniature model of a local meteorological unit. Our miniature model system consists of three sensors to measure the common weather variables, and then display them on our website in a dynamic manner. Through the integration of various sensors and leveraging Firebase, we enable seamless data collection, analysis, and visualization to empower users to access up-to-date weather information for informed decision-making. We also use the sensor's outputs to make a close forecast of the weather for a day or several days in the future to assist our university in making any urgent decision based on weather conditions.

Introduction

Weather reporting is an essential practical implementation of the concept of Internet of Things, it involves sensing and recording various weather parameters and using them for long term analysis and for identify and display trends in parameters using graphical representation. The devices used for this purpose are used to collect, organize and display information. The Internet of Things made an evolutionary leap in monitoring and controlling environmental phenomena by using sensors/devices that can capture, process, and transmit weather parameters. The captured data is transmitted to the cloud storage and displayed on a website.

1.1 Overview:

The Weather Reporting System seamlessly integrates Internet of Things technology to collect real-time weather data, with sensors strategically placed to capture essential meteorological parameters. This data is securely stored in the cloud, ensuring accessibility and scalability. The user-friendly web interface then presents this information, offering interactive features for users to explore historical trends, forecasts, and specific weather metrics.

Understanding and following up on weather changes prove instrumental across various sectors. In agriculture, timely updates empower farmers to optimize crop cycles and pest control measures, while in transportation, real-time information aids in route planning, reducing accidents and delays. The system's role extends to disaster management, helping communities prepare for and respond to adverse weather conditions. It influences energy production and consumption, supports public health measures, and contributes to environmental conservation efforts. The Weather Reporting System is not just a tool for immediate decision-making but a comprehensive solution enhancing resilience, safety, and sustainability across diverse societal sectors.

1.2 Historical Weather Data

Historical weather data refers to the recorded information about the past weather conditions over a specific period. It includes temperature, precipitation, humidity, clouds, wind speed and direction, atmospheric pressure, and more. Historical weather data allows researchers,

meteorologists, and businesses to analyze long-term climate patterns, track changes, and make informed decisions.

1.2.1 Types of Historical Weather Data:

a. Observational Data

Observational data is generally considered to be the most trustworthy, as it contains a record of what actually happened in the past. However, there are two kinds of observational data:

• **in-situ data** which is measured at the location of the event, such as that from a ground-based station, a radiosonde or a weather drone; and

• **remotely sensed data**, such as that which is derived from satellites or radar.

b. Model Data

Although a small amount of modeling is involved in retrieving remotely sensed variables, generally the term "model" is used to denote that data has been produced either in a forecast or a reanalysis.

A reanalysis is a weather model that uses the same principles as a forecast but is not constrained by the need to produce predictions before the weather occurs. Reanalysis are not used for forecasting future weather but instead provide a retrospective view of historical weather patterns, making them useful for climate research, understanding extreme weather events, and other applications.

1.2.2 Importance of Historical Weather Data:

Historical weather data is a crucial foundation, serving as a valuable resource for climate research and long-term pattern analysis. Scientists utilize this data to identify climate change trends and model future scenarios. Beyond research, historical weather data plays a vital role in various industries. It aids agriculture, energy, and insurance sectors in managing risks based on past weather patterns, ensuring informed decision-making. Moreover, its impact extends to disaster preparedness and response efforts by providing insights into extreme weather events. This data also supports historical analysis for tourism planning, infrastructure development, and urban planning, contributing to the creation of resilient and weather-adaptive systems. In the medical field, understanding historical weather patterns proves beneficial for informed

healthcare decisions, aligning with the meticulous nature of medical practices. Overall, historical weather data stands as an essential tool, influencing industries and activities alike with its insights and impact.

1.3 Weather Forecasting

Forecasting is the process of estimation in unknown situations from the historical data. Weather forecasting is one of the most scientifically and technologically challenging problems around the world. To make an accurate prediction is indeed, one of the major challenges that meteorologists are facing all over the world. Since ancient times, weather prediction has been one of the most interesting and fascinating domains. Scientists have tried to forecast meteorological characteristics using a number of methods.

Weather forecasting plays a vital role in our daily lives, ensuring safety, guiding economic decisions, and aiding in planning across various sectors. Nowadays, the weather forecast is unpredictable to be exact because the climate changes drastically over the weather. Therefore, the Weather Reporting System is mostly used to monitor the continuously changing weather conditions over controlled areas like houses, industries, agriculture, etc. in real-time monitoring. This system will monitor the changes in weather conditions happening over the environment and then provide the users the fastest way to access the information from anywhere.

1.3.1 Weather Forecasting Importance:

Weather forecasting is paramount in safeguarding lives, influencing economic decisions, and shaping planning strategies within diverse industries. Accurate predictions are crucial for emergency services and airlines, ensuring effective disaster response and secure flight planning. Economically, forecasts guide pivotal decisions in agriculture, impacting planting, harvesting, and crop management. The energy sector optimizes production based on predictive insights, while the construction industry integrates forecasts into project planning. Urban planners leverage weather data to design resilient infrastructure, and the transportation industry refines routes and logistics accordingly. In the medical field, weather forecasting plays a nuanced role, contributing to informed healthcare decisions. Understanding how weather patterns may impact health allows for proactive measures in disease control and patient care, aligning with the meticulous nature of medical practices. Moreover, the influence extends to tourism and

hospitality, where forecasts are instrumental in managing bookings and planning activities, creating a ripple effect across various sectors.

1.3.2 Weather Forecasting Methods:

Here are some of the most common methods for weather forecasting in more detail:

- **Numerical weather prediction (NWP):** NWP models are used to simulate the atmosphere and predict how it will change over time. These models are based on the laws of physics and use data from weather stations, satellites, and other sources to represent the current state of the atmosphere. NWP models are very complex and require a lot of computing power, but they are the most accurate method for forecasting the weather days or weeks in advance.
- **Nowcasting:** Nowcasting uses real-time data, such as radar and satellite images, to forecast the weather for the next few hours. This method is often used to forecast the location and intensity of thunderstorms and other short-lived weather events.
- **Analogue forecasting:** Analogue forecasting compares the current weather conditions to past weather patterns to predict how the weather will change. This method is not as accurate as NWP, but it can be useful for forecasting the weather in areas where there is limited data.

1.4 Document Organization

The remaining contents of the document are organized as follows:

Chapter 2: explores how our Weather Reporting System is structured, showing how IoT sensors, cloud storage, and the web interface work together.

Chapter 3: goes into the practical side, breaking down the algorithm with flowcharts and providing the source code to understand how the system is implemented.

Chapter 4: presents the LSMT model, demonstrating how this model is used to forecast the temperature.

Chapter 5: presents the results real-world scenarios, demonstrating how the WRS performs in different weather conditions.

Chapter 6: presents the conclusion.

System Architecture

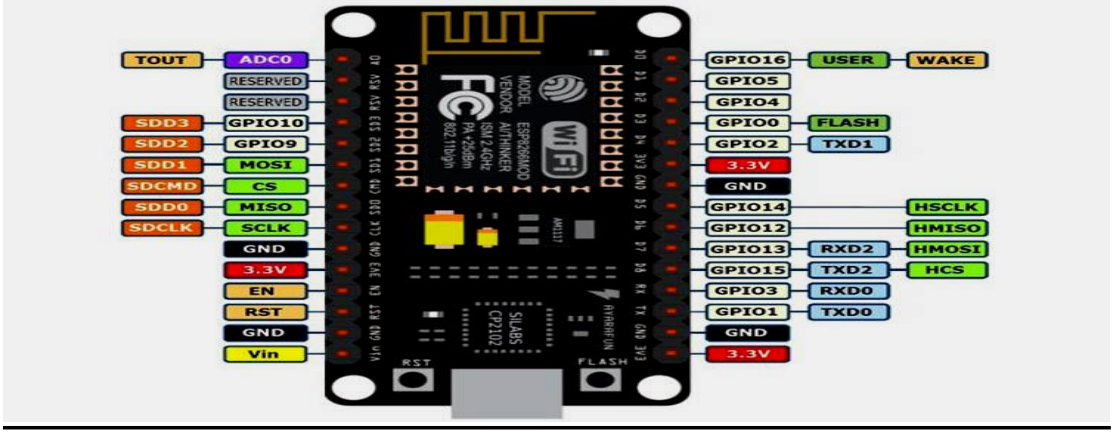
2.1 System hardware

In this concise document, we'll explore the key elements driving our IoT project, from micro-controllers to sensors, we'll uncover the essential components of our embedded system project.

Component	Description	Price
ESP8266	**Wi-Fi module for IoT connectivity.**	**$6.47**
Breadboard	**Prototyping tool for circuit assembly**	**$1.32**
Jumper wires	**Flexible connectors for circuit connections**	**$1.62**
Rain sensor	**Detects precipitation levels**	**$2.91**
DHT22 Sensor	**Measures temperature and humidity**	**$3.07**

The following is a detailed description of each component with images.

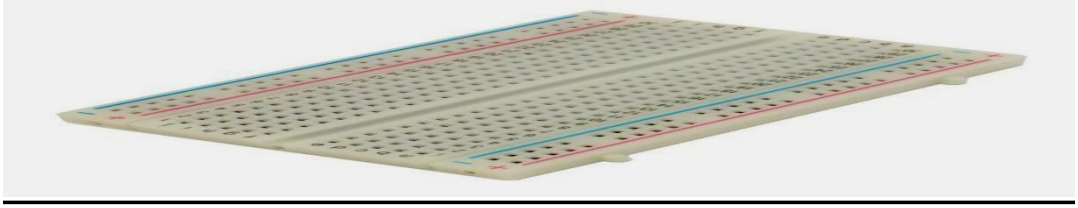
1. **ESP8266 Module** : is a Wi-Fi module used to provide wireless connectivity in electronics projects. It enables remote control of devices over the Internet, making it a common component in Internet of Things (IoT) applications. It is widely used in:
   - Remote Control: Allows remote control of devices such as lighting or home appliances via the Internet.
   - Environmental Monitoring: Measures temperature, humidity, and atmospheric pressure, transmitting this data to the cloud for remote monitoring.
   - IoT Applications: Utilized in IoT applications to connect devices and enable data exchange. For programming, the Arduino language is commonly used



2. Breadboard :

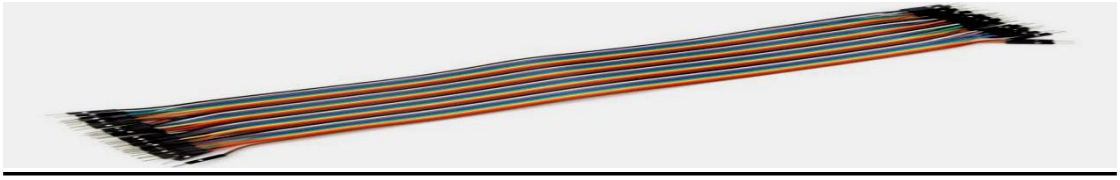
Breadboard is a rectangular plastic board with a bunch of tiny holes in it. These holes let you easily insert electronic components to prototype (meaning to build and test an early version of) an electronic circuit, like this one with a battery, switch, resistor, and an LED (light-emitting diode).

A breadboard is a fundamental tool in electronics prototyping and experimentation. It allows you to create temporary circuits without the need for soldering, making it easy to modify and test different configurations.



3. Jumper wires:

Jumper wires are essential components in electronics and prototyping. They are used to create electrical connections between various components on a breadboard or between different points in a circuit.



4. **Rain sensor:**

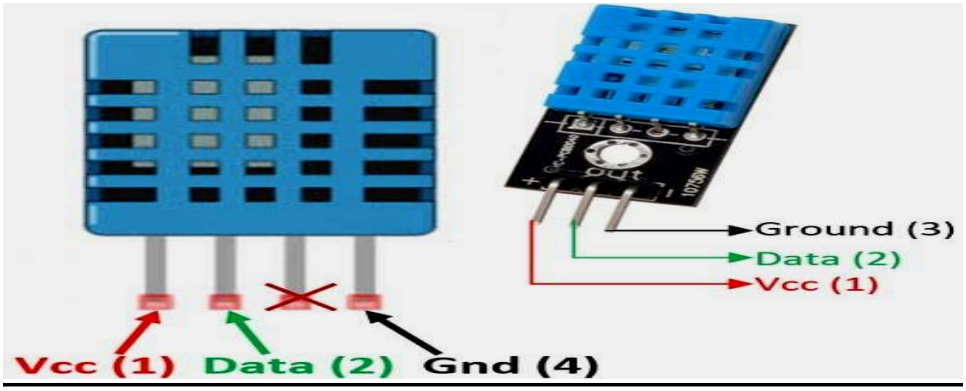
   Rain sensor is used to monitor rainfall and record data. This aids in improving the accuracy of weather forecasts and providing precise information about current weather conditions.

The functions of a rain sensor in a weather reporting system include:

- Measuring Rainfall Amount: It determines the quantity of rainfall during a specific period, contributing to the analysis of climatic data.
- Real-Time Updates: The rain sensor enables immediate updates to the system when rainfall begins, ensuring accurate reporting.
- Monitoring Weather Conditions: It helps in understanding the impact of rainfall on local conditions and weather predictions.
- Enhancing Predictive Accuracy: With the readings from the rain sensor, weather models can be improved, predicting future weather conditions more effectively.

5. **DHT22 Sensor:**

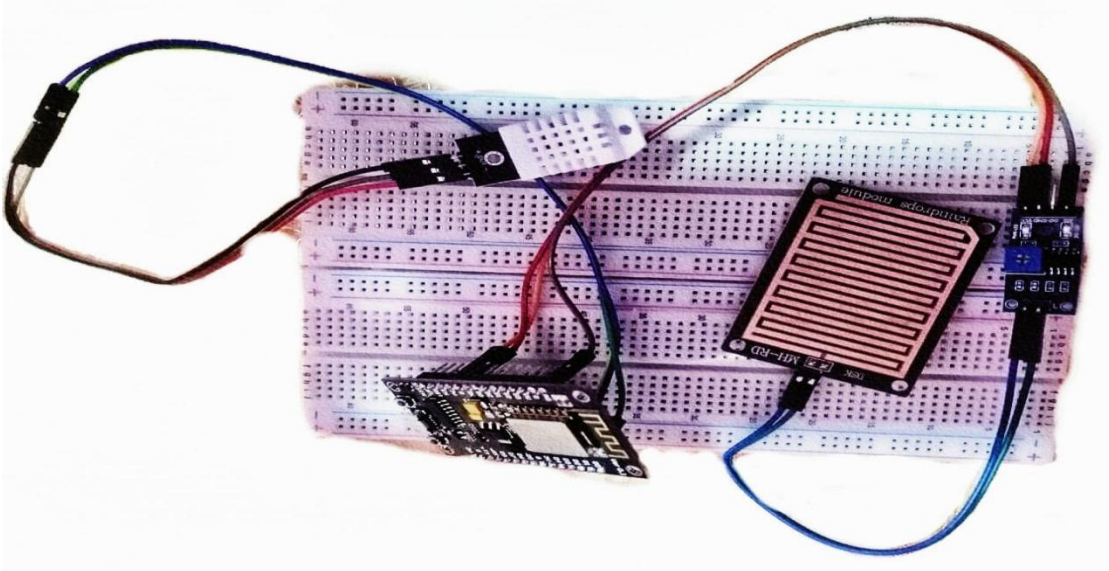
   DHT22 is a sensor used for measuring temperature and humidity. It connects to a micro-controller like Arduino, providing accurate readings in a temperature range of 0-50 degrees Celsius and humidity range of 20%-90%. It is utilized in projects and weather stations for efficient environmental monitoring.



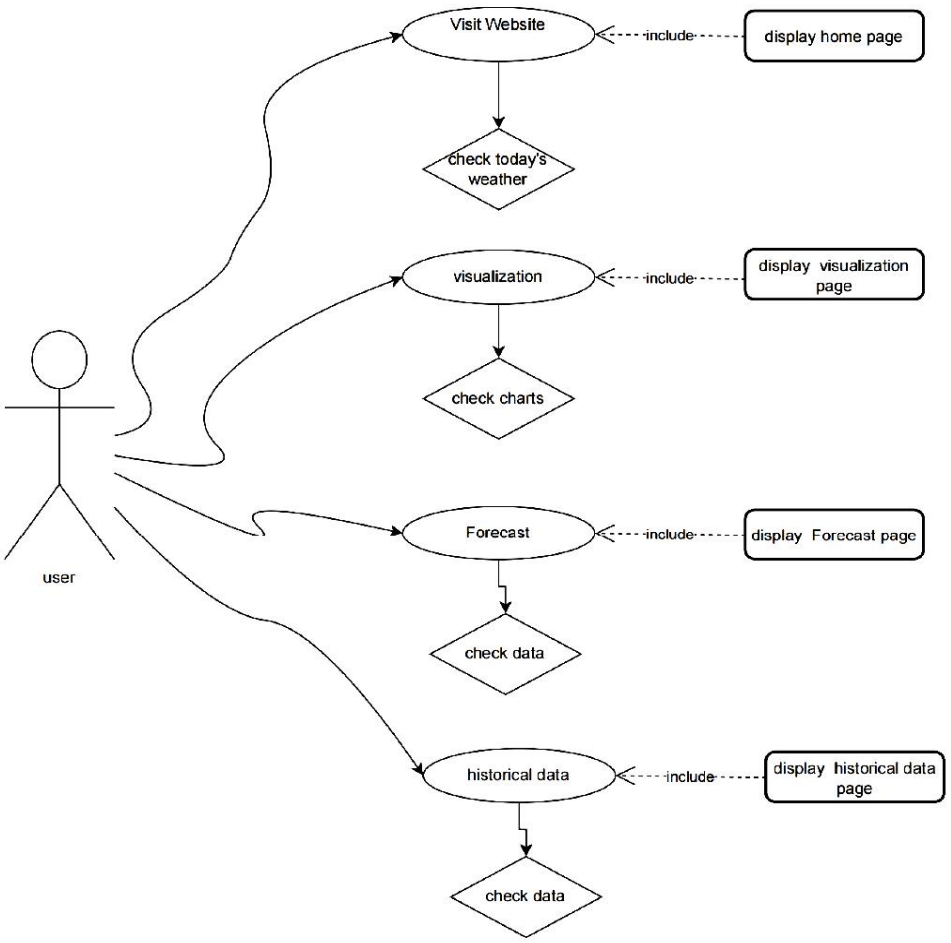
2.2 Circuit Diagram

The connection has been made as follows:

- The ground (GND) pin of the ESP module has been connected to the ground (GND) pin of both the DHT22 temperature sensor and the rain sensor.
- The power supply (3.3V) pin of the ESP module has been connected to the power supply (VCC) pin of both the DHT22 temperature sensor and the rain sensor.
- The signal (GPIO) pin of the ESP module has been connected to the signal (SIGNAL) pin of the DHT22 temperature sensor to read the temperature value.
- The signal (GPIO) pin of the ESP module has been connected to the signal (SIGNAL) pin of the rain sensor to read the rain value.



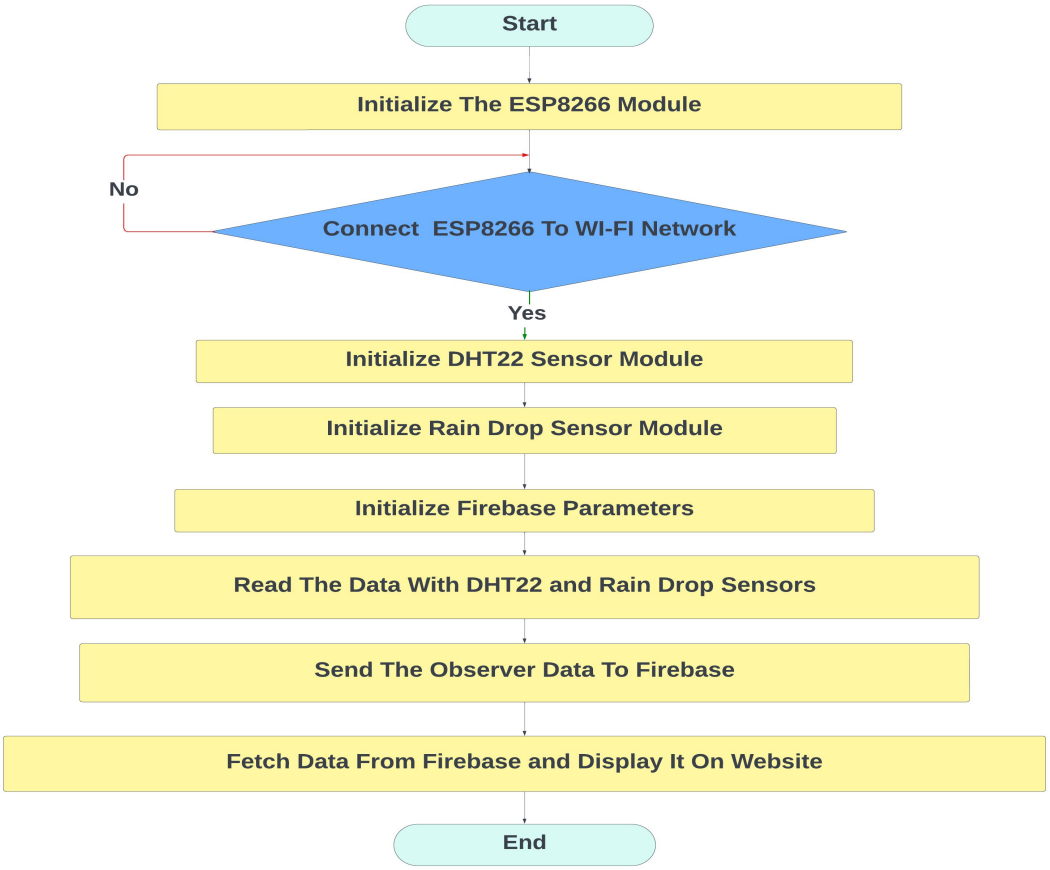
2.3 Use Case


Visit Website
include
display home page
check today's weather
visualization
include
display visualization page
check charts
Forecast
include
display Forecast page
check data
historical data
include
display historical data page
check data
user

Implementation

3.1 Algorithm

The weather monitoring system employs an ESP8266 microcontroller interfaced with DHT22 and Raindrop sensors. The DHT22 captures temperature and humidity, while the Raindrop sensor detects rain conditions. This data is continuously read, stored locally, and transmitted to Firebase for efficient real-time storage. On the website, data is fetched from Firebase, enabling users to monitor weather conditions remotely. The system leverages Firebase's Real-time Database, ensuring that any updates trigger events for seamless and dynamic information retrieval. This integrated solution offers a user-friendly and effective approach to weather monitoring, providing a comprehensive and accessible overview of environmental parameters.



3.2 Source Code

```
#include <ESP8266WiFi.h>
#include <NTPClient.h>
#include <WiFiUdp.h>
#include <FirebaseESP8266.h>
#include "DHT.h"
#define firebase_auth "AIzaSyBJ1ktYgQ5h-SHg0r5tvOfELQU8MnKcXps"
#define firebase_url "dhtdata-171c6-default-rtdb.firebaseio.com"
#define RAINPIN A0
#define DHTPIN 4     // Digital pin  #2 on ESP8266
#define DHTTYPE DHT22 // Sensor type
DHT dht(DHTPIN, DHTTYPE); // Create DHT object

// Variables to store sensor readings
float humidity,temperature_C,temperature_F ;
int RainDrop;

// WiFi credentials
const char* ssid = "Mohamido";
const char* pass = "1234567767";

WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "pool.ntp.org");

const long utcOffsetInSeconds = 7200; // 2 hours

FirebaseData firbasedata;
FirebaseAuth auth;
FirebaseConfig config;
```

```
void setup() {

  // Initialize serial communication
 Serial.begin(115200);

 // Connect to WiFi
 WiFi.begin(ssid, pass);
 while (WiFi.status() != WL_CONNECTED) {
   delay(1000);
   Serial.println("Connect to Wifi.......");
 }
 Serial.println("Connected to wifi");
 Serial.print("IP: ");
 Serial.println(WiFi.localIP());

  // Configure Firebase
 config.signer.test_mode = true;
 config.database_url = firebase_url;
 Firebase.begin(&config, &auth);
 Firebase.reconnectWiFi(true);

 // Initialize DHT sensor
  dht.begin();

  timeClient.setTimeOffset(utcOffsetInSeconds);
 // Initialize the NTPClient
 timeClient.begin();
}
```

```
void loop() {

// Read sensor values
temperature_C = dht.readTemperature();
temperature_F = dht.readTemperature(true);
humidity = dht.readHumidity();
RainDrop = analogRead(RAINPIN);

// Check for error reading
  if (isnan(humidity) || isnan(temperature_C) || isnan(temperature_F) || isnan(RainDrop)) {
    Serial.println(" Reading failed ");
    return;
  }

   // Print sensor readings
  Serial.print("Humidity: ");
  Serial.print(humidity);
  Serial.println("%");

  Serial.print("Temperature:");
  Serial.print(temperature_C);
  Serial.print(("°C ------- "));

  Serial.print(temperature_F);
  Serial.println("°F");

   Serial.print("RainDrop: ");
  Serial.println(RainDrop);

   // Update the timeClient
  timeClient.update();
  time_t now = timeClient.getEpochTime();
  // Convert the time to a readable format
  struct tm * timeinfo;
  timeinfo = localtime(&now);
  int hours = timeinfo->tm_hour;
  int minutes = timeinfo->tm_min;
  int seconds = timeinfo->tm_sec;
  int year = timeinfo->tm_year + 1900;
  int month = timeinfo->tm_mon + 1;
  int day = timeinfo->tm_mday;
  int hours12 = hours % 12;
  if (hours12 == 0) {
    hours12 = 12; // 0 should be displayed as 12 in 12-hour format
  }

  // Determine AM or PM
  const char* ampm = (hours < 12) ? "AM" : "PM";
  String current_time = String(hours12)+":"+(minutes < 10 ? "0" : "" )+String(minutes)+":"+(seconds < 10 ?
"0" : "" )+String(seconds)+ " " + ampm;
  String current_date = String(day)+"-"+String(month)+"-"+String(year);

 // Store sensor data in Firebase
 unsigned long timestamp = millis();
 String path = "/Weather Data/";
 path = path + String(timestamp);
 float value = temperature_C;
 float value1 = humidity;
 String HumidValue = String(humidity) + "%";
 String TempValue = String(temperature_C) + "°C";
 String RainValue = String(RainDrop);
  Firebase.setString(firbasedata, path+"/Humidity", HumidValue);
  Firebase.setString(firbasedata, path+"/Temperature", TempValue);
  Firebase.setString(firbasedata, path+"/RainDrop", RainValue);
  Firebase.setString(firbasedata, path+"/Date", current_date);
  Firebase.setString(firbasedata, path+"/Time", current_time);
  delay(600000);

}
```

Temperature Forecasting Using Multivariate LSTM

4.1 Brief overview

What is LSTM Model? LSTM (Long Short-Term Memory) is a type of recurrent neural network (RNN) architecture that is designed to model and predict sequences of data. It was introduced to address the limitations of traditional RNNs, particularly the issue of long-term dependencies and the vanishing gradient problem.

Why we use this model in our project? Using an LSTM model for temperature forecasting **Reasons for Using LSTM in Temperature Forecasting:**

1- Handling Sequential Data: Temperature data is inherently sequential, with each day's temperature influenced by previous days. LSTMs are specifically designed to work with sequence data and can capture the temporal dependencies in such datasets.

2- Long-Term Dependencies: Weather patterns can have long-term dependencies, where the temperature on a given day might be influenced by conditions from days or even weeks prior. LSTMs are well-suited for capturing these long-term relationships due to their architecture, which mitigates the vanishing gradient problem common in traditional RNNs.

3- Prediction Accuracy: LSTMs have been shown to provide higher accuracy in various time-series prediction tasks compared to simpler models like ARIMA or linear regression, especially when the data exhibits complex temporal patterns.

4- Flexibility: LSTM models can handle not only the temperature but also other related variables if they are included in the dataset (like humidity, wind speed, etc.), making the model versatile for weather forecasting.

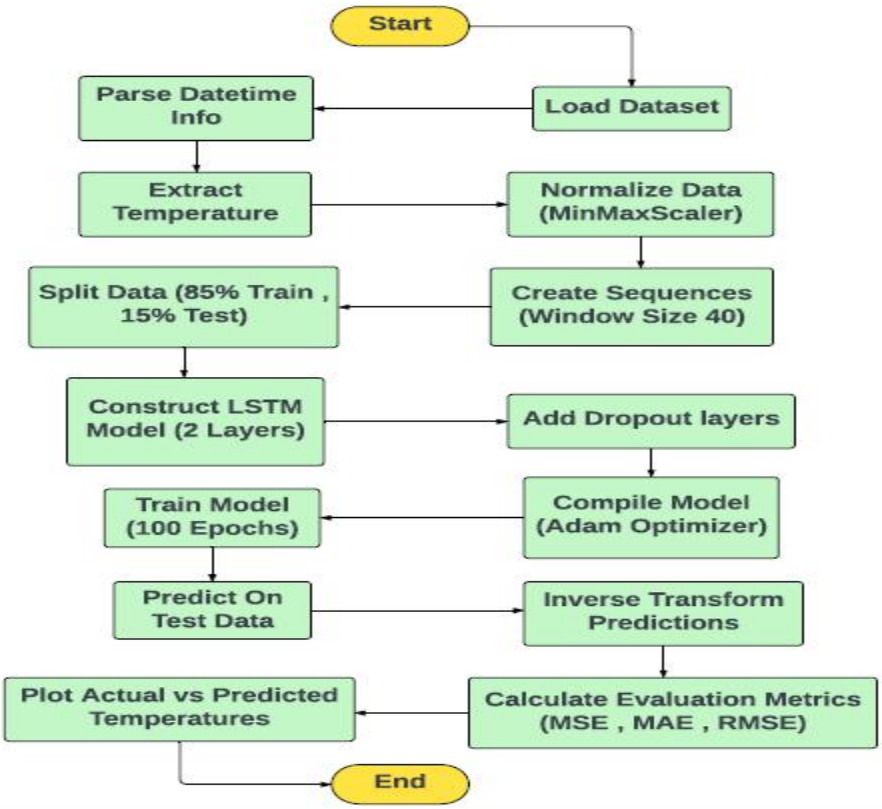
5- Stateful Processing: LSTMs maintain a state over time, which helps in learning and predicting more accurately by retaining important information over long sequences.

6- Iterative Forecasting: The code uses an iterative approach for forecasting, where each predicted value is fed back into the model to predict the next value. This

process helps in extending the forecast over multiple days, leveraging the LSTM's ability to maintain context over several steps.

Using an LSTM model in the temperature forecasting code is motivated by the need to accurately capture and predict complex temporal dependencies in sequential temperature data. LSTMs are specifically designed to handle such tasks, making them a suitable choice for this type of prediction. Their ability to manage long-term dependencies, flexibility in handling various input lengths, and stateful processing capabilities make them an excellent fit for weather forecasting applications.

In this chapter, we delve into the sophisticated methodology of employing a multivariate Long Short-Term Memory (LSTM) model for accurate temperature forecasting. Multivariate LSTM models stand out for their ability to handle and predict complex time series data, leveraging multiple variables to capture intricate patterns. Our focus was on predicting the temperature for Cairo, Egypt, utilizing historical data spanning from January 1, 2023, to June 25, 2024.



4.2 Data Collection and Preprocessing

The initial step involved meticulous data preprocessing. The data-set included date-time information and temperature values. We parsed the date-time information and set it as the index of the data-frame for time series analysis. Temperature values were normalized using the MinMaxScaler from the sklearn library, scaling them between 0 and 1. This normalization is crucial for enhancing the model's learning efficiency and prediction accuracy.

We then constructed sequences of 40 time steps, which served as input to our LSTM model, with the subsequent temperature value as the target. The data-set was strategically divided into training (85%) and testing (15%) subsets to validate the model's performance effectively.

4.3 Model Architecture

Our LSTM architecture comprised two robust layers:

First LSTM Layer: 100 units, returning sequences, with a dropout regularization of 0.2 to prevent overfitting.

Second LSTM Layer: 100 units, followed by dropout regularization of 0.2.

Dense Output Layer: A single neuron to predict the temperature.

The model was compiled using the Adam optimizer and the Mean Squared Error (MSE) loss function.

4.4 Training the Model

The model was trained for 100 epochs with a batch size of 32. The training process involved feeding the training data into the model and validating it on the test data. The history of training, including loss and validation loss, was recorded for performance analysis.

4.5 Forecasting and Results

Post-training, the model's efficacy was evaluated using key metrics:

Mean Squared Error (MSE): 4.7939

Mean Absolute Error (MAE): 1.6686

Root Mean Squared Error (RMSE): 2.1895

These metrics underscore the model's precision in forecasting temperatures. To provide a comprehensive comparison, the predicted temperatures were plotted against actual values, visually affirming the model's predictive accuracy.

4.6 Visualization

A plot was generated to visualize the actual and predicted temperatures over time. This visualization helps in understanding the model's performance in capturing the temperature trends and variations.

4.7 Summary

In this chapter, we demonstrated how to use an LSTM model for temperature forecasting. The model was trained on historical temperature data and successfully predicted future temperatures with reasonable accuracy. This methodology can be extended to more complex weather forecasting tasks, offering valuable insights for meteorological studies and applications.

Results

We show the weather reporting system observations in two ways, tables and graphs. All results are shown on our website (https://weather-system.netlify.app/)

HISTORICAL DATA

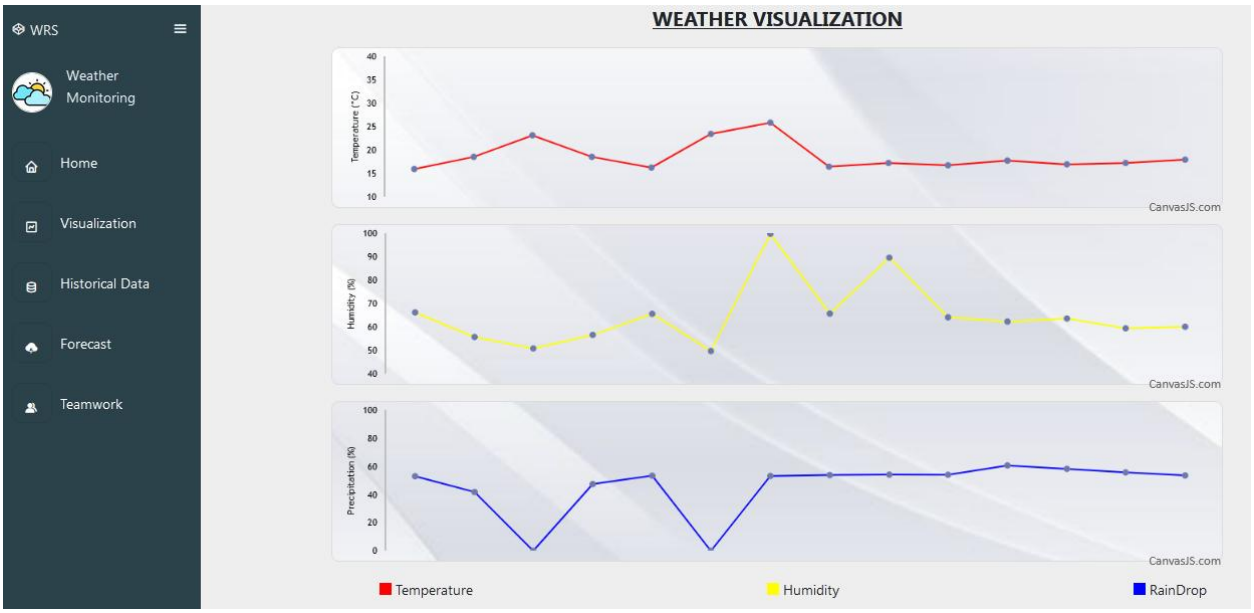
Date	Time	Temperature	Humidity	RainDrop	Precipitation
5-2-2024	11:52:24 AM	16.00°C	66.30%	480	53.13%
3-2-2024	8:36:49 PM	18.60°C	55.80%	595	41.89%
10-2-2024	11:22:54 PM	23.20°C	50.90%	1024	0.00%
3-2-2024	8:46:52 PM	18.60°C	56.70%	537	47.56%
5-2-2024	12:02:28 PM	16.30°C	65.70%	476	53.52%
10-2-2024	11:32:57 PM	23.50°C	49.80%	1024	0.00%
3-2-2024	8:56:54 PM	25.90°C	99.90%	478	53.32%

To understand the difference between the two columns (RainDrop and Precipitation), we must know what happens with the raindrop sensor when a raindrop falls on it. It includes a printed circuit board (control board) that "collects" the raindrops. As raindrops are collected on the circuit board, they create paths of parallel resistance that are measured via the op-amp. The lower the resistance (or the more water), the lower the voltage output. Conversely, the less water the greater the output voltage on the analog pin.

A completely dry board, for example, will cause the module to output 1024. Thus high raindrop intensity causes a low output value and low raindrop intensity causes a high output value.

Here we compute the percent of rainfall from the raindrop intensity by using this equation.

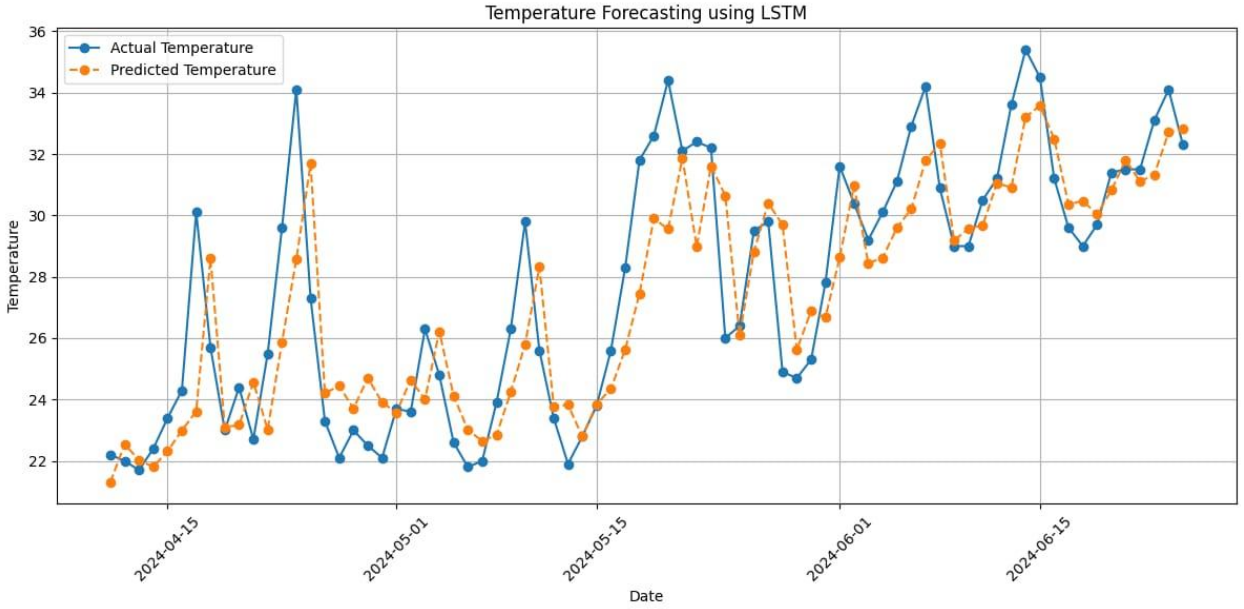
$$Precipitation = 100 - RainDrop * 100/1024$$

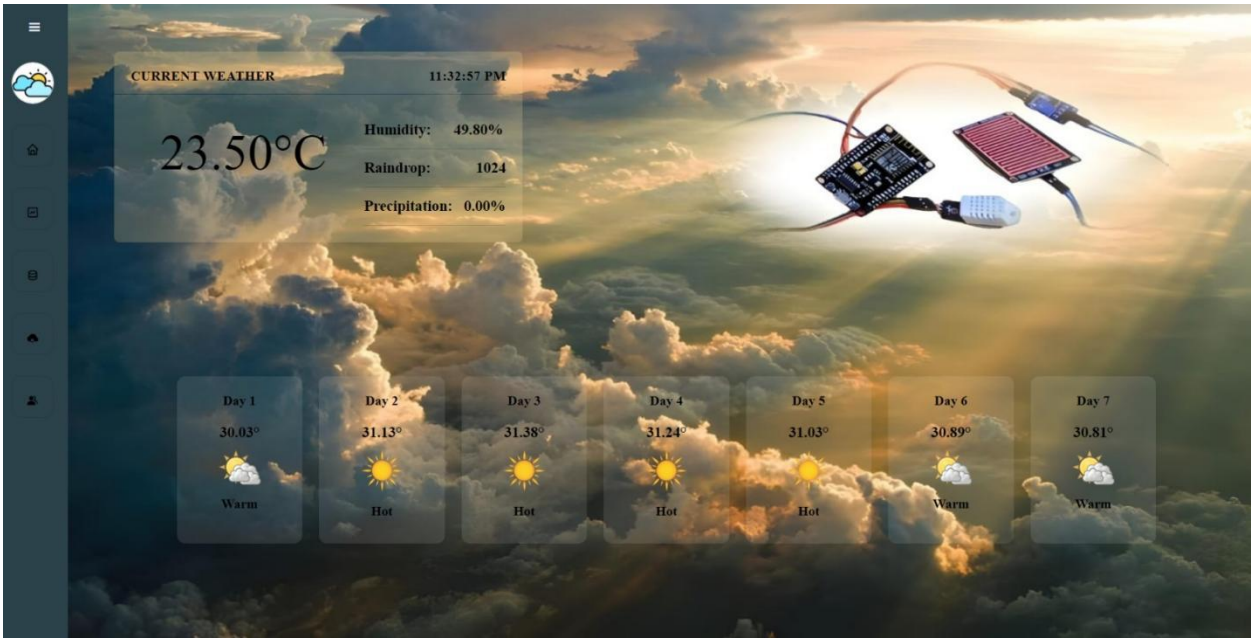


By using LSTM model we can predict the seventh future day's temperature as shown,

```
Forecasted Temperatures for the Next 7 Days:
Day 1: Forecasted Temp: 30.03 °C, Actual Temp: 29.70 °C
Day 2: Forecasted Temp: 31.13 °C, Actual Temp: 31.40 °C
Day 3: Forecasted Temp: 31.38 °C, Actual Temp: 31.50 °C
Day 4: Forecasted Temp: 31.24 °C, Actual Temp: 31.50 °C
Day 5: Forecasted Temp: 31.03 °C, Actual Temp: 33.10 °C
Day 6: Forecasted Temp: 30.89 °C, Actual Temp: 34.10 °C
Day 7: Forecasted Temp: 30.81 °C, Actual Temp: 32.30 °C
```

	Actual	Predicted	Difference
0	22.2	21.311110	0.888890
1	22.0	22.545580	0.545580
2	21.7	22.026737	0.326737
3	22.4	21.808201	0.591799
4	23.4	22.318560	1.081440
..	...	...	...
71	31.5	31.805210	0.305210
72	31.5	31.102421	0.397579
73	33.1	31.330059	1.769941
74	34.1	32.705002	1.394998
75	32.3	32.835464	0.535464


Temperature Forecasting using LSTM
Actual Temperature
Predicted Temperature
Temperature
Date
36
34
32
30
28
26
24
22
2024-04-15
2024-05-01
2024-05-15
2024-06-01
2024-06-15


CURRENT WEATHER
11:32:57 PM
23.50°C
Humidity: 49.80%
Raindrop: 1024
Precipitation: 0.00%
Day 1
30.03°
Warm
Day 2
31.13°
Hot
Day 3
31.38°
Hot
Day 4
31.24°
Hot
Day 5
31.03°
Hot
Day 6
30.89°
Warm
Day 7
30.81°
Warm

Conclusion

By using sensor devices in the environment, we could bring the environment into real life. Using the Internet of Things" (IoT) enabled us to provide users with real-time access to weather data from any location. The project not only offers current weather data but also allows users to review historical information. The system's ability to provide real-time alerts to users empowers them to plan their activities and respond to changing weather conditions promptly, potentially preventing emergencies and mitigating risks. This model can be expanded to monitor the developing cities and industrial zones for pollution monitoring, to protect public health from pollution.

We demonstrated how to use an LSTM model for temperature forecasting. The model was trained on historical temperature data and successfully predicted future temperatures with reasonable accuracy. This methodology can be extended to more complex weather forecasting tasks, offering valuable insights for meteorological studies and applications.

References

1. https://www.vedantu.com/geography/weather-forecasting
2. https://blog.weatherstack.com/blog/why-we-need-historical-weather-data/
3. https://goweatherforecast.com/news/importance-of-weather-forecasting-235
4. https://www.researchgate.net/publication/345327830_Development_of_IoT_Based_Weather_Reporting_System
5. https://www.google.com/url?q=https://www.rcciit.org/students_projects/projects/ee/2022/GR9.pdf&sa=U&ved=2ahUKEwirj9XJ5pSEAxX7U6QEHbhcABgQFnoECAoQAg&usg=AOvVaw3PWoY6FdeLReavQmg_m7Cc
6. https://www.meteomatics.com/en/weather-api/historical-data-weather-api/
7. https://makerselectronics.com/product/raindrop-sensor-module
8. https://www.visualcrossing.com/
9. https://www.geeksforgeeks.org/multivariate-time-series-forecasting-with-lstms-in-keras/