Mini Project Report on

IOT based Weather Reporting

Submitted in partial fulfillment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE & ENGINEERING

Submitted by:

Student Name:	University Roll No. :
ABHINAV BANSAL	2118072

Under the Mentorship of

Mr. Akash Chauhan Sir

Assistant Professor

Department of Computer Science and Engineering

Graphic Era Hill University

Dehradun, Uttarakhand

Graphic Era Hill University

DEHRADUN • BHIMTAL • HALDWANI

CANDIDATE’S DECLARATION

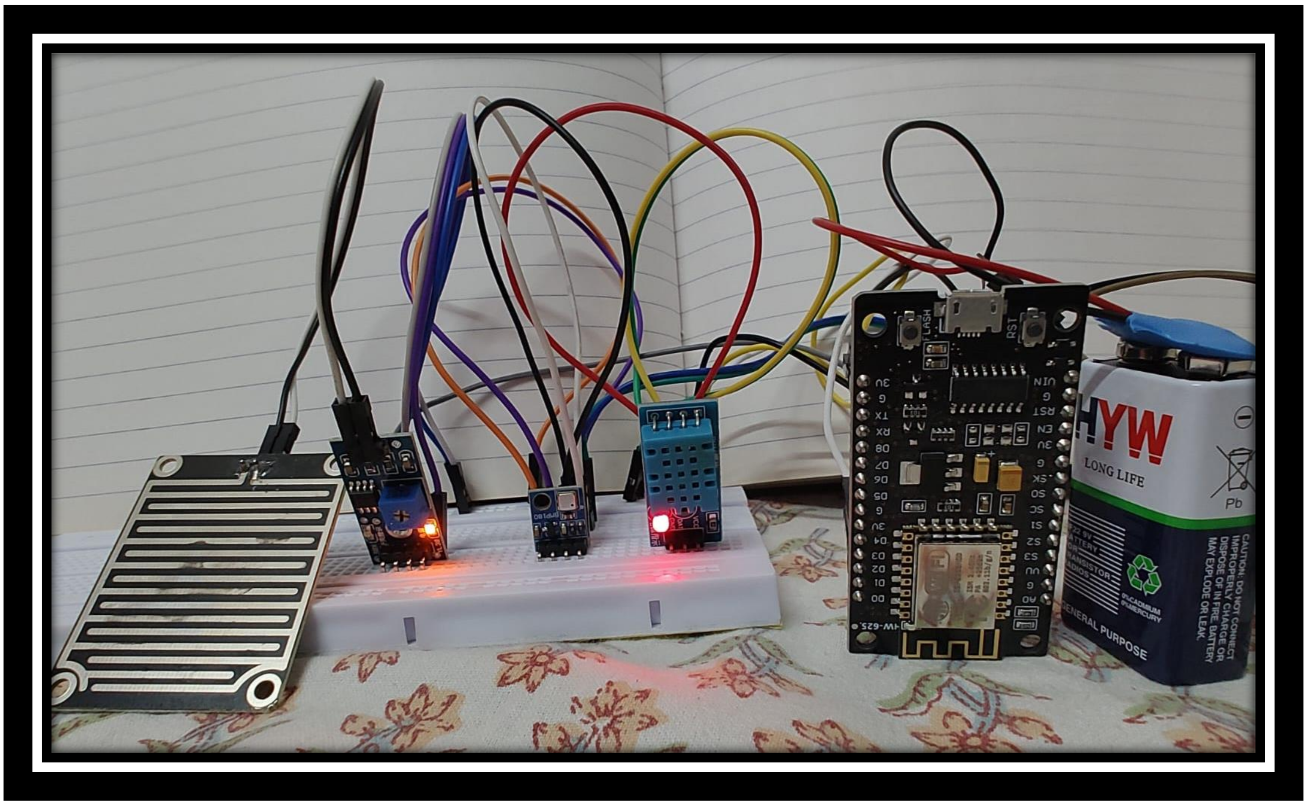
I hereby certify that the work which is being presented in the project report entitled **IOT based Weather Reporting** in partial fulfillment of therequirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineering of the Graphic Era Hill University, Dehradun shall be carried out by myself under the mentorship of **Mr. Akash Chauhan, Assistant Professor**, Department of Computer Science and Engineering, Graphic Era Hill University, Dehradun.

Name:	**University Roll no. :**
ABHINAV BANSAL	**2118072**

Table of Contents

CHAPTER 1

INTRODUCTION



1.1 Introduction

IoT technology plays a crucial role in integrating sensors, data collection devices, and communication systems to gather valuable weather-related information from diverse locations. These IoT devices, ranging from temperature sensors and humidity sensors to barometric pressure sensors and wind speed sensors, collect precise data that is transmitted to a central system for processing, analysis, and dissemination to users.

- Real-time Data: IoT-based weather reporting systems enable the collection and transmission of real-time weather data from multiple locations. This means that users can access accurate and up-to-date information about temperature, humidity, precipitation, wind speed, and other critical weather parameters.

- Early Warning Systems: With access to real-time data and advanced analytics, IoT-based weather reporting systems contribute to the development of early warning systems for severe weather events. This capability enables timely evacuations, effective disaster management, and the reduction of potential risks associated with extreme weather conditions.

- Accessibility and User-Friendly Interfaces: IoT-based weather reporting systems provide weather information through user-friendly interfaces such as mobile apps or web portals. This accessibility ensures that a wide range of users, including individuals, businesses, and government agencies, can easily access and utilize weather data for informed decision-making.

1.2 Problem Statement

The specific problems to be addressed in the development of an IoT-based weather reporting system include:

1. Limited Coverage: Traditional weather reporting systems are often limited to specific weather stations or regions, resulting in incomplete and localized data. There is a need to establish a network of IoT devices that can collect weather data from multiple locations, providing a wider coverage area for accurate and comprehensive weather monitoring.

2. Data Accuracy and Precision: Existing weather reporting systems may suffer from inaccuracies and inconsistencies in data collection due to limitations in sensor quality and calibration. Developing IoT-based weather reporting systems that utilize advanced sensors and calibration techniques is crucial to ensure accurate and precise measurements of weather parameters.

3. Data Integration and Analytics: Traditional weather reporting systems may lack effective integration of data from various sources, such as weather stations, satellites, and ground-based sensors. There is a need for an IoT-based system that can seamlessly integrate data from multiple sources, enabling comprehensive data analysis and modeling for improved weather forecasting and prediction capabilities.

1.3 Objective of the project

The objectives of the project report on IoT-based weather reporting are as follows:

1. To provide a comprehensive overview of IoT technology and its application in weather reporting it will explain the fundamental concepts of IoT, its components, and

its role in transforming weather reporting systems. It will highlight the key features and advantages of IoT-based weather reporting.

2. The project report will analyze the shortcomings of traditional weather reporting systems, such as limited coverage, data accuracy issues, and delayed information dissemination. It will examine the challenges faced by current systems and how IoT can address these challenges.

3. To propose a system architecture for an IoT-based weather reporting system: The project report will outline a system architecture that integrates sensors, data collection devices, communication modules, and data processing infrastructure. It will describe the components and their functionalities in collecting, transmitting, and processing weather data.

CHAPTER 2

Literature Survey

2.1 Current state of weather reporting systems

The current state of weather reporting systems is limited in coverage, reliant on manual data collection methods, and faces challenges in data transmission, processing, integration, and analysis. These limitations affect the accuracy, timeliness, and accessibility of weather data. However, integrating IoT technology into weather reporting systems offers promising solutions.

By using IoT devices, these systems can achieve broader coverage, continuous data collection, faster data transmission and processing, seamless data integration, and user-friendly interfaces.

IoT-based weather reporting systems have the potential to revolutionize the field by providing more accurate and timely weather data, improving forecasting and prediction, and enhancing decision-making in various sectors.

2.2 IoT applications in weather monitoring

The integration of IoT technology in weather monitoring has revolutionized the collection, analysis, and distribution of weather data. IoT devices with sensors are deployed in different locations to gather real-time data on temperature, humidity, pressure, and wind speed. This data is wirelessly transmitted to a central system for processing and analysis. By integrating data from multiple sources, including weather stations, satellites, sensors, and user devices, IoT-based weather monitoring systems provide a comprehensive view of weather conditions for more accurate forecasting.

IoT enables the development of early warning systems by continuously monitoring weather parameters and issuing timely alerts. Cloud computing and data analytics enhance weather monitoring by storing, processing, and analyzing large volumes of data using advanced techniques like machine learning and artificial intelligence.

2.3 Review of existing IoT-based weather reporting systems

IoT-based weather reporting systems have revolutionized the collection, analysis, and dissemination of weather data. These systems integrate sensors, data collection devices, and communication infrastructure to provide real-time and accurate weather information. Several successful examples highlight the effectiveness of IoT technology in weather monitoring and forecasting.

One such example is the WeatherBug IoT Weather Station Network, which deploys IoT weather stations in various locations. These stations collect data on temperature, humidity, wind speed, and other weather parameters. The data is transmitted to a central server for analysis and is made available to users through a mobile app, offering real-time updates and personalized forecasts.

These IoT-based weather reporting systems exemplify comprehensive data collection, real-time updates, and advanced analytics capabilities. They significantly enhance the accuracy, coverage, and accessibility of weather information, thereby improving weather forecasting, enabling early warning systems, and facilitating informed decision-making across various sectors.

CHAPTER 3

Methodology

1.3 Detailed Hardware Specifications

1. NodeMCU ESP8266

   The NodeMCU ESP8266 Wi-Fi module is specifically designed for IoT projects, providing built-in Wi-Fi connectivity for easy communication with other devices over the internet.

Key features of the NodeMCU:

- Flash Memory: The module includes 4MB of flash memory for storing firmware, program code, and data.
- RAM: It is equipped with 80KB of RAM for data storage and program execution.
- Wi-Fi Connectivity: The NodeMCU supports 802.11 b/g/n Wi-Fi standards, allowing it to connect to Wi-Fi networks as both a client and an access point.
- GPIO Pins: It provides 11 digital input/output pins (GPIO) that can be configured as inputs or outputs. These pins support PWM and interrupt functionality, making it easy to interface with external sensors, actuators, and devices.
- Analog Input: The NodeMCU has one analog input pin for reading analog voltage levels ranging from 0 to 3.3V.
- Interfaces: It supports various interfaces such as UART, SPI, and I2C, enabling communication with external devices like sensors, displays, and modules.
- Operating Voltage: The NodeMCU operates at a voltage of 3.3V, which is different from the standard 5V used by many Arduino boards. Careful attention is required when interfacing with external components that operate at a different voltage level.

2. DHT11:

- The DHT11 is a cost-effective digital temperature and humidity sensor commonly used for monitoring these parameters in various projects. Here are the key features of the DHT11 sensor:
- Sensor Type: The DHT11 utilizes a capacitive humidity sensing element and a thermistor for temperature measurement.
- Measurement Range: It can measure temperatures from 0 to 50 degrees Celsius with an accuracy of ±2 degrees Celsius. For humidity, it has a measurement range of 20% to 90% with an accuracy of ±5%.
- Operating Voltage: The DHT11 operates on a voltage range of 3V to 5.5V, which makes it compatible with a wide range of microcontrollers and development boards.

3Rain Sensor Module:

- Detection Principle: The rain sensor module typically consists of a conductive surface that detects water and changes its conductivity. When raindrops are present, the resistance between the conductive traces on the sensor module decreases.
- Digital Output: The rain sensor module provides a digital output signal, which indicates the presence or absence of rain. The signal can be connected to a microcontroller or development board for further processing and control.
- Sensitivity Adjustment: Some rain sensor modules offer sensitivity adjustment to fine-tune the detection threshold based on the application requirements.
-

4. BMP180:

- Sensor Type: The BMP180 employs a piezo-resistive pressure sensor and an integrated temperature sensor.
- Measurement Range: The BMP180 can measure pressure in the range of 300 hPa to 1100 hPa (hectopascals) with an accuracy of ±1 hPa. It can measure temperature in the range of -40°C to +85°C with an accuracy of ±1°C.
- Low Power Consumption: The BMP180 sensor is designed to operate at low power, making it suitable for battery-powered applications.
- Altitude Calculation: By measuring atmospheric pressure, the BMP180 can also calculate the altitude above sea level.

5. Blynk

- Blynk is a popular platform for IoT projects, offering real-time data access and control.
- It provides a user-friendly interface for web and mobile applications, enabling remote monitoring and control of connected devices.
- Blynk operates on a cloud-based infrastructure, ensuring seamless connectivity between IoT devices and the Blynk app.
- Real-time data monitoring allows users to display sensor readings on the app's interface, enabling tracking and analysis of device status.
- The platform supports remote control and actuation of devices, allowing users to send commands and trigger actions.
- Blynk offers cross-platform support, enabling access and control from web browsers and mobile applications.
- It integrates with various hardware platforms, including Arduino, Raspberry Pi, and ESP8266.
- Blynk provides extensive libraries and APIs for easy integration with different programming languages.
- Data logging capabilities allow storage and analysis of historical data, enabling visualization and informed decision-making.

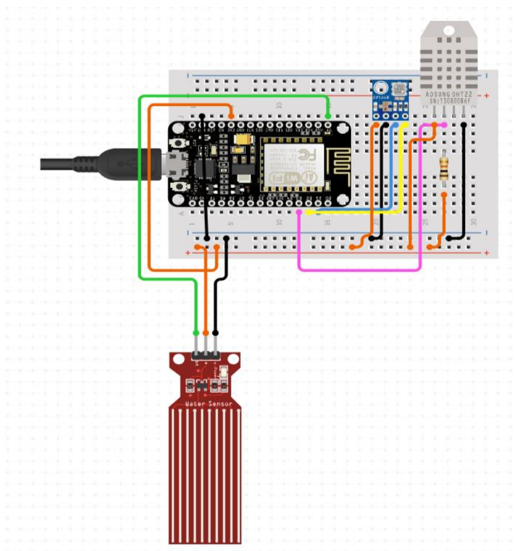
CHAPTER 4

Design and Implementation

4.1 Design

- Circuit

  To set up the circuit, you need to connect the NodeMCU ESP8266 board to a 3.3V power source and ground. Then, connect the DHT11 sensor to the NodeMCU's data pin D4. If you're using a BMP180 sensor with I2C communication, connect its SDA and SCL pins to the corresponding pins on the NodeMCU. For the moisture/rain sensor, connect its VCC pin to 5V power, GND pin to ground, and the analog output pin to A0 on the NodeMCU. Make sure you have a stable power supply and connect the NodeMCU's ground to the power supply's ground. Once the circuit is set up, upload the code and monitor the sensor readings on the serial monitor and Blynk web server or mobile app.

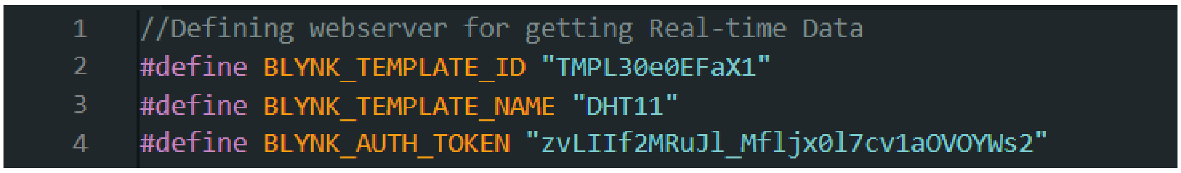


To set up the circuit for the provided code:

- Connect the NodeMCU's VCC to 3.3V, GND to ground.
- Connect DHT11's VCC to 3.3V, GND to ground, and data pin (DHTPIN) to D4 on NodeMCU.
- For BMP180 (optional), if using I2C, connect VCC to 3.3V, GND to ground, SDA to NodeMCU's SDA, and SCL to NodeMCU's SCL.
- For the moisture/rain sensor, connect VCC to 5V, GND to ground, and analog output to A0 on NodeMCU.
- Ensure a stable power supply for the NodeMCU and connect its GND to the power supply's ground.

4.2 Code snippets

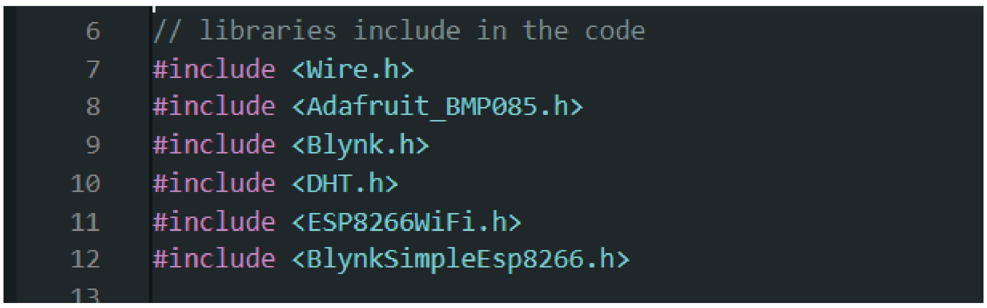
- ➢ Webserver Configuration:



```
//Defining webserver for getting Real-time Data
#define BLYNK_TEMPLATE_ID "TMPL30e0EFaX1"
#define BLYNK_TEMPLATE_NAME "DHT11"
#define BLYNK_AUTH_TOKEN "zvLIIf2MRuJl_Mfljx0l7cv1aOVOYWs2"
```

The code begins by defining the Blynk template ID, template name, and authentication token. These values are necessary for connecting the NodeMCU board with the Blynk cloud server.

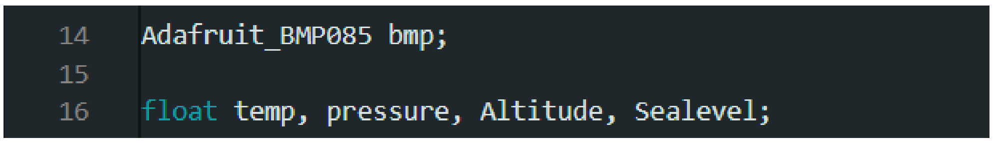
- ➢ Library Inclusions:



```
// libraries include in the code
#include <Wire.h>
#include <Adafruit_BMP085.h>
#include <Blynk.h>
#include <DHT.h>
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
```

The code includes several libraries that are required for the project: Wire, Adafruit_BMP085, Blynk, DHT, ESP8266WiFi, and BlynkSimpleEsp8266. These libraries provide the necessary functions and utilities to interact with the sensors, connect to Wi-Fi, and communicate with the Blynk server.

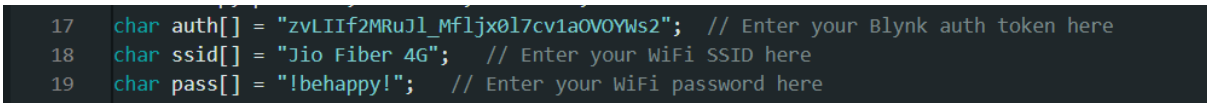
- ➢ Global Variables:



```
Adafruit_BMP085 bmp;

float temp, pressure, Altitude, Sealevel;
```

The code declares global variables to store the temperature, pressure, altitude, and sea level pressure values. These variables will be used to store sensor readings and send them to the Blynk app.

- ➢ Wi-Fi and Blynk Initialization:



```
char auth[] = "zvLIIf2MRuJl_Mfljx0l7cv1aOVOYWs2";  // Enter your Blynk auth token here
char ssid[] = "Jio Fiber 4G";   // Enter your WiFi SSID here
char pass[] = "!behappy!";   // Enter your WiFi password here
```

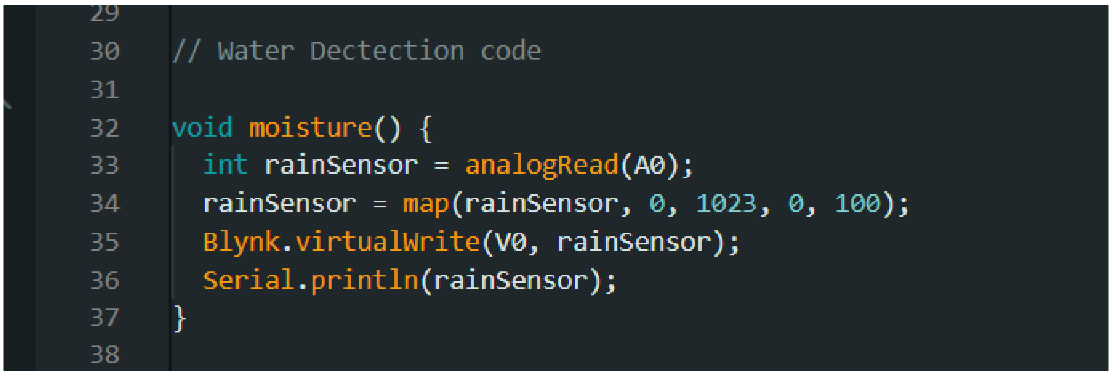
In the setup() function, the code initializes the serial communication for debugging, connects to the specified Wi-Fi network using the provided SSID and password, and establishes a connection with the Blynk server using the authentication token.

It also initializes the DHT11 sensor and checks for the presence of the BMP085 sensor. If the BMP085 sensor is not found, an error message is printed to the serial monitor.

- Timer Setup:

The code sets up two timers using the BlynkTimer library: one for sending sensor data (temperature and humidity) at a 1-second interval, and the other for reading the moisture/rain sensor at a 100-millisecond interval.

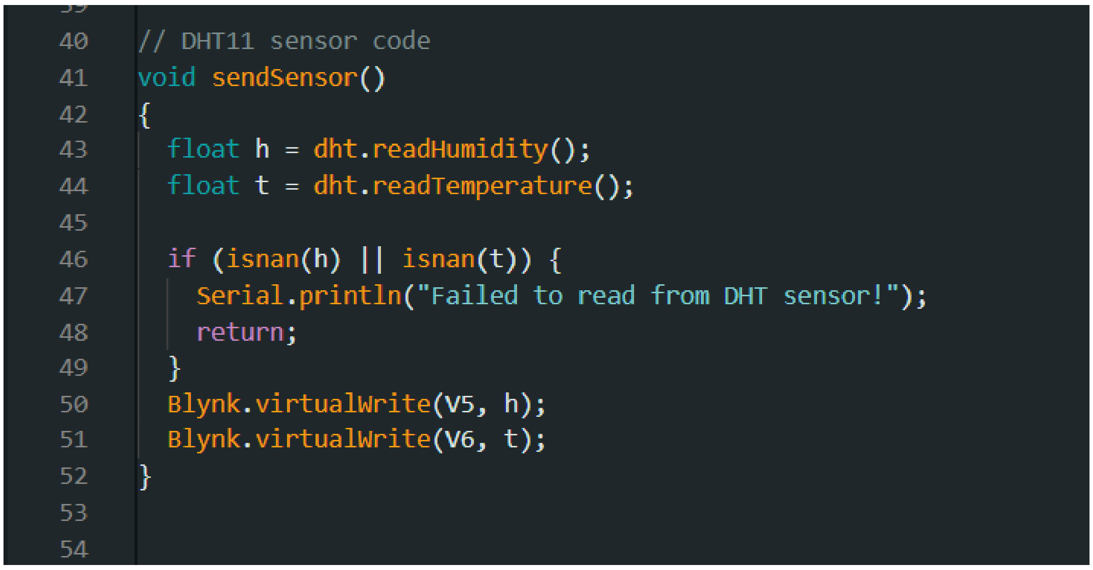
- Moisture/Rain Sensor:



```

// Water Dectection code

void moisture() {
  int rainSensor = analogRead(A0);
  rainSensor = map(rainSensor, 0, 1023, 0, 100);
  Blynk.virtualWrite(V0, rainSensor);
  Serial.println(rainSensor);
}

```

The moisture() function reads the analog value from the moisture/rain sensor connected to pin A0. It then maps the sensor reading from the range 0-1023 to the range 0-100 and sends the value to the virtual pin V0 in the Blynk app. The moisture value is also printed to the serial monitor.

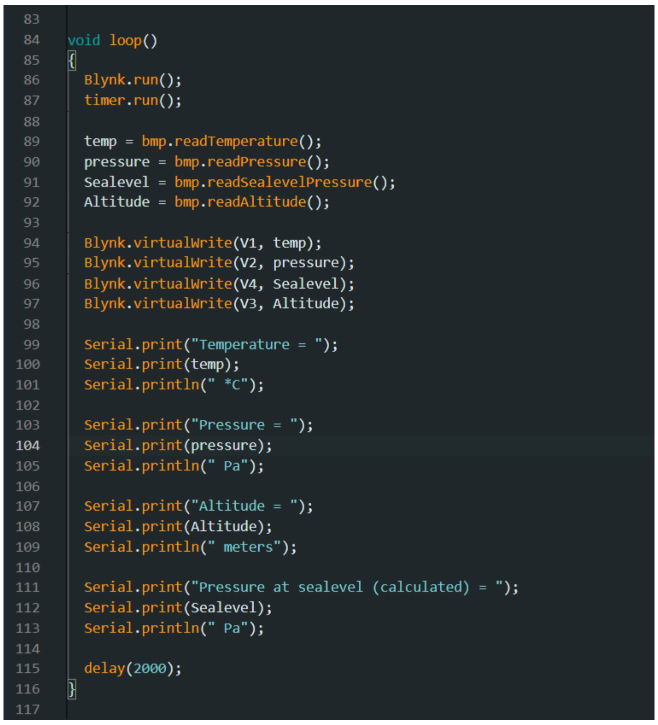
➢ DHT11 Sensor:



```
// DHT11 sensor code
void sendSensor()
{
  float h = dht.readHumidity();
  float t = dht.readTemperature();

  if (isnan(h) || isnan(t)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }
  Blynk.virtualWrite(V5, h);
  Blynk.virtualWrite(V6, t);
}
```

The sendSensor() function reads the temperature and humidity values from the DHT11 sensor using the readTemperature() and readHumidity() functions. It checks if the readings are valid and then sends them to the virtual pins V5 and V6 in the Blynk app.

➢ Main Loop:



```
void loop()
{
  Blynk.run();
  timer.run();

  temp = bmp.readTemperature();
  pressure = bmp.readPressure();
  Sealevel = bmp.readSealevelPressure();
  Altitude = bmp.readAltitude();

  Blynk.virtualWrite(V1, temp);
  Blynk.virtualWrite(V2, pressure);
  Blynk.virtualWrite(V4, Sealevel);
  Blynk.virtualWrite(V3, Altitude);

  Serial.print("Temperature = ");
  Serial.print(temp);
  Serial.println(" *C");

  Serial.print("Pressure = ");
  Serial.print(pressure);
  Serial.println(" Pa");

  Serial.print("Altitude = ");
  Serial.print(Altitude);
  Serial.println(" meters");

  Serial.print("Pressure at sealevel (calculated) = ");
  Serial.print(Sealevel);
  Serial.println(" Pa");

  delay(2000);
}
```

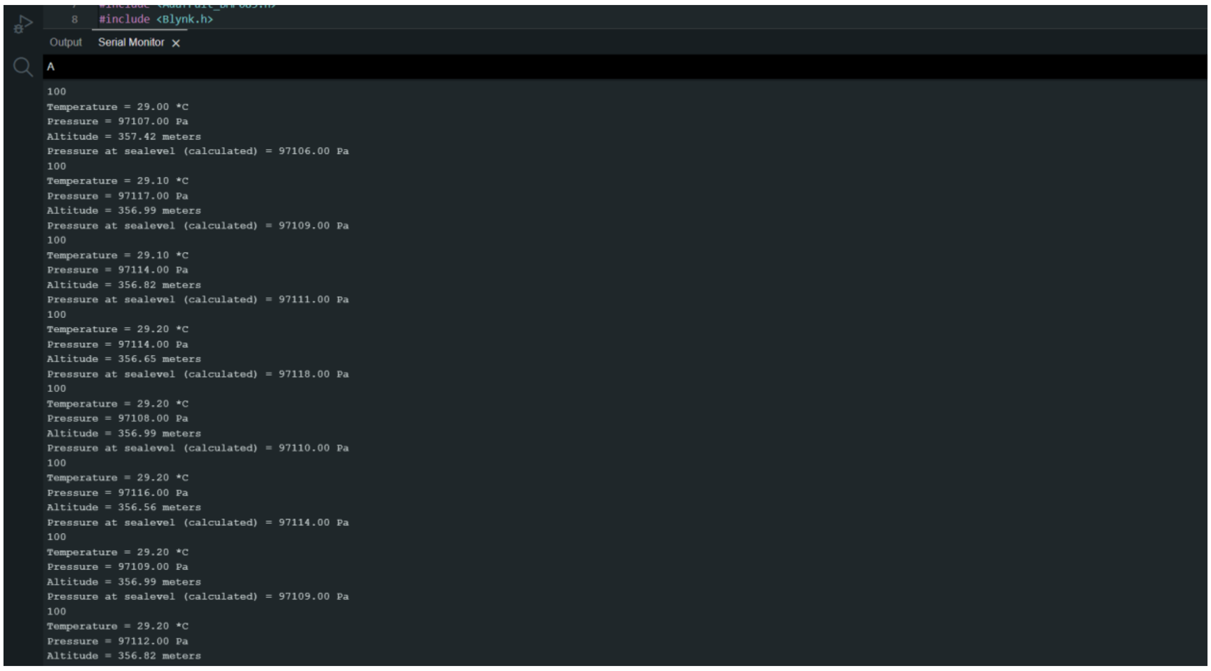
The loop() function is where the main code execution happens. It continuously runs the Blynk.run() and timer.run() functions, which are responsible for handling Blynk-related tasks and executing the defined timer functions.

Inside the loop, it reads the temperature, pressure, sea level pressure, and altitude values from the BMP085 sensor and sends them to the respective virtual pins in the Blynk app.

The sensor values are also printed to the serial monitor for debugging purposes.

A delay of 2 seconds is added at the end of each iteration to avoid flooding the system with excessive data.

- ➢ Serial Monitor:

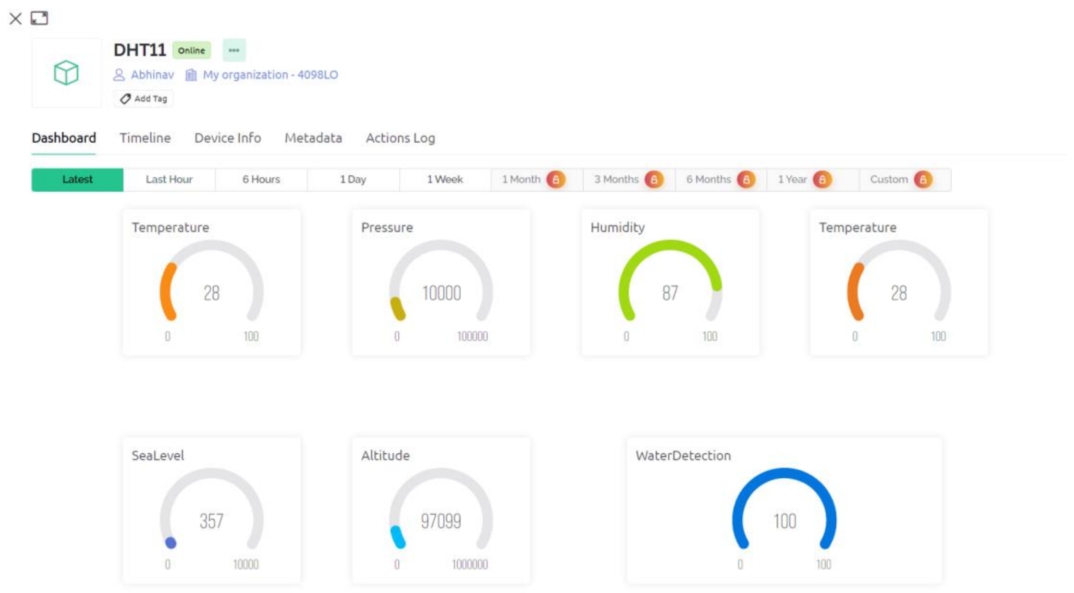


```
#include <Blynk.h>
```

```
A
100
Temperature = 29.00 *C
Pressure = 97107.00 Pa
Altitude = 357.42 meters
Pressure at sealevel (calculated) = 97106.00 Pa
100
Temperature = 29.10 *C
Pressure = 97117.00 Pa
Altitude = 356.99 meters
Pressure at sealevel (calculated) = 97109.00 Pa
100
Temperature = 29.10 *C
Pressure = 97114.00 Pa
Altitude = 356.82 meters
Pressure at sealevel (calculated) = 97111.00 Pa
100
Temperature = 29.20 *C
Pressure = 97114.00 Pa
Altitude = 356.65 meters
Pressure at sealevel (calculated) = 97118.00 Pa
100
Temperature = 29.20 *C
Pressure = 97108.00 Pa
Altitude = 356.99 meters
Pressure at sealevel (calculated) = 97110.00 Pa
100
Temperature = 29.20 *C
Pressure = 97116.00 Pa
Altitude = 356.56 meters
Pressure at sealevel (calculated) = 97114.00 Pa
100
Temperature = 29.20 *C
Pressure = 97109.00 Pa
Altitude = 356.99 meters
Pressure at sealevel (calculated) = 97109.00 Pa
100
Temperature = 29.20 *C
Pressure = 97112.00 Pa
Altitude = 356.82 meters
```

The code includes various Serial.print() statements that print sensor readings and other information to the serial monitor. For example, temperature, pressure, altitude, and sea level pressure values are printed with appropriate labels, such as "Temperature =", "Pressure =", etc. You can open the serial monitor in the Arduino IDE (set at a baud rate of 9600) to view this output.

➢ Blynk Web Server and Mobile Application:

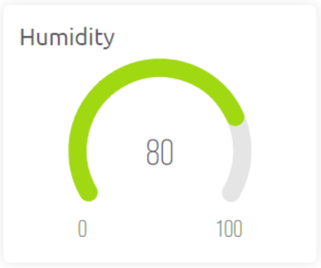


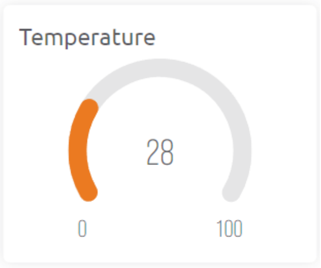
The Blynk platform allows you to create a user interface to visualize and control connected devices. In the code, the Blynk.virtualWrite() functions are used to send sensor data to virtual pins in the Blynk app. For example, temperature and humidity readings are sent to virtual pins V5 and V6 using Blynk.virtualWrite(V5, h) and Blynk.virtualWrite(V6, t). Widgets, such as value displays, graphs, or gauges, can be created in the Blynk app and assigned to the corresponding virtual pins. The received sensor data will be displayed in real-time on these widgets. By using the Blynk app on your mobile device or accessing the Blynk web server, can remotely monitor the sensor readings and control the connected devices if applicable.

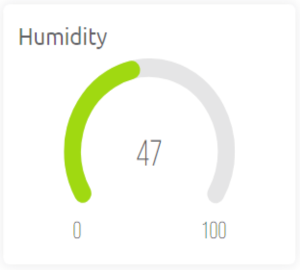
4.3. <u>User interface</u>

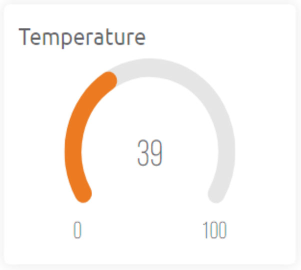
1. DHT 11 SENSOR TEST BEFORE

BEFORE AFTER





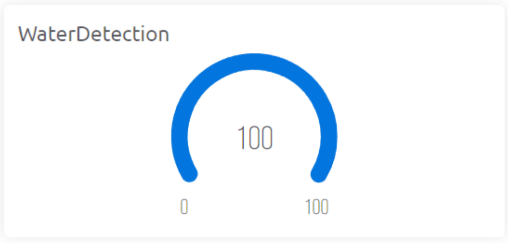


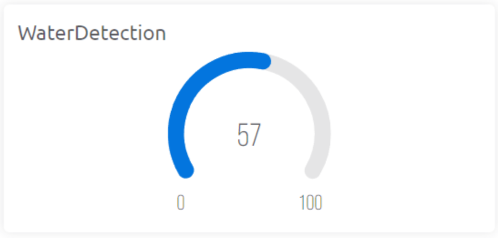


2. RAIN SENSOR TEST

BEFORE

AFTER

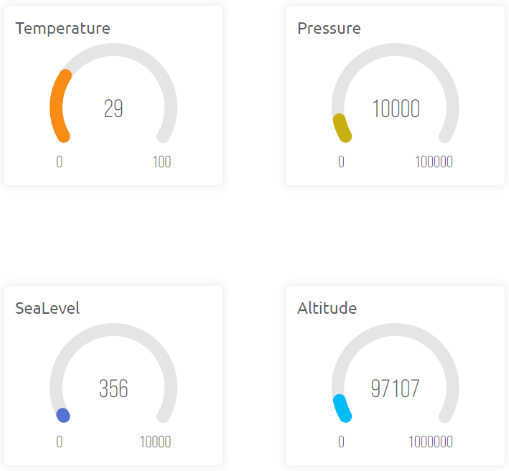


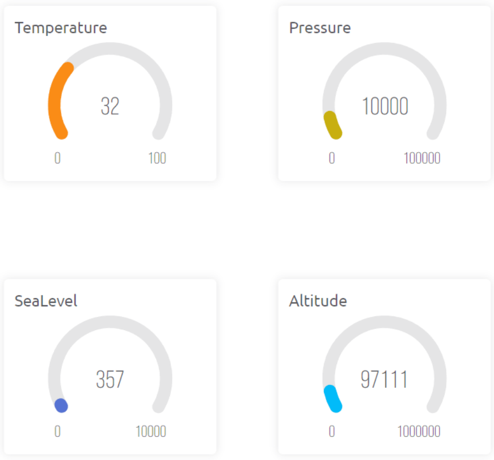


3. BMP180

BEFORE

AFTER





CHAPTER 5

Results and Discussion

5.1 Data Collection and Transmission Performance:

IoT-based weather reporting systems utilize sensors like DHT11, Rain Sensor Module, and BMP180 for real-time data collection on temperature, humidity, rainfall, and barometric pressure. These systems offer broad coverage and up-to-date weather information.

There are challenges regarding sensor accuracy, reliability, and data transmission in IoT weather reporting systems. Regular calibration and maintenance are necessary, and ensuring stable data transmission can be complex due to network connectivity issues.

5.2 Data Processing and Analytics Insights:

IoT-based weather reporting systems offer advantages such as improved forecasting through data processing and analytics algorithms, leading to accurate predictions. They enable pattern recognition by analyzing historical and real-time data, enhancing our understanding of climate changes.

Processing and analyzing large volumes of data can be computationally intensive, requiring substantial processing power and storage capacity. The accuracy and reliability of the algorithms used are crucial for generating meaningful insights.

CHAPTER 6

CONCLUSION

The project aimed to develop an IoT-based weather reporting system using NodeMCU ESP8266 and Blynk. It incorporated sensors like DHT11, Rain Sensor Module, and BMP180 for temperature, humidity, precipitation, and barometric pressure measurements. The methodology involved deploying these sensors in various locations and wirelessly transmitting the data to a central system. Blynk was used for data processing and visualization, providing a user-friendly interface.

Key findings included real-time data collection, accurate sensor measurements, and reliable data transmission. The system demonstrated seamless connectivity and convenient weather information interpretation. Future scope includes integrating additional sensors, incorporating external data sources, and enhancing data analytics and prediction algorithms. Artificial intelligence and machine learning techniques can be implemented for more precise weather forecasting.

Overall, the IoT-based weather reporting system using NodeMCU ESP8266, Blynk, and sensors offers potential for advanced and accurate weather monitoring systems, with scope for further improvements and expansions.

CHAPTER 7

REFERENCES

- https://espressif.com/sites/default/files/documentation/esp8266-technical_reference_en.pdf

- Analytics for the Internet of Things
  BY- **Andrew Minteer**

- https://arduino-esp8266.readthedocs.io/en/latest/esp8266wifi/readme.html

- Building the internet of things : Implement New Business Models , Disrupt Competitors ,Transform Your Industry 1st Edition
  BY – **Maciej Kranz**