A Report on

IOT BASED WEATHER MONITORING SYSTEM

for

Mini Project 1B (REV- 2019 C Scheme)
of Second Year, (SE Sem- IV)

in

Electronics & Telecommunication Engineering

by

3021105 Ariba Afroz
3021113 Rahul Deoghare
3021116 Aditya Gawande
3021119 Dipraj Hindole

under the guidance of

Mr. Ankur Ganorkar

Department of Electronics & Telecommunication Engineering

Fr. C. Rodrigues Institute of Technology

Sector 9A, Vashi, Navi Mumbai-400705

University of Mumbai

Certificate

This is to certify that the half-year project entitled “IOT Based Weather Monitoring System” is a bonafide work of

Roll no	Name
3021105	Ariba Afroz
3021113	Rahul Deoghare
3021116	Aditya Gawande
3021119	Dipraj Hindole

submitted to the University of Mumbai in partial fulfilment of the requirement for the award confirming completion of Mini Project 1B (REV- 2019 C Scheme) of Second Year, (SE Sem-IV) in Electronics & Telecommunication Engineering as laid down by University of Mumbai during the First Half of academic year 2023.

Certified by

Examiner/Reviewer-1
Name & Signature
with Date

Examiner/Reviewer-2
Name & Signature
with Date

Name & Signature of Guide

Name & Signature of Head of Department

Declaration

We declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, we have adequately cited and refer- enced the original sources. we also declare that we have adhered to all principles of aca-demic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

3021105	Ariba Afroz	Signature:
3021113	Rahul Deoghare	Signature:
3021116	Aditya Gawande	Signature:
3021119	Dipraj Hindole	Signature:

Date:

Abstract

Innovations in weather monitoring have focused on controlling and monitoring different weather conditions. IoT devices help measure the physical parameters of a certain location and upload them in real-time to cloud storage, where the data can be analyzed immediately.These systems make climate monitoring in difficult geographical terrains possible. The manpower required for accurate microclimate forecasts is significantly reduced.Sensor devices placed in particular locations can do all the work to detect current climate details, such as rainfall, wind speed, humidity, soil moisture, CO2 levels, and other data needed for forecasting.Going a step further, when one connects the weather station to the internet, IoT can be used much more extensively in predicting and disseminating accurate weather data in a particular location. This information can then be made available anywhere in the world.The output of the arduino is stored in the local database using the influxdb. A Web Application dashboard is developed using HTML, CSS and JavaScript. The web application collects data from the Node-MCU and the dashboard visualizes all the sensor data like temperature, humidity, barometric pressure, light intensity, air quality and rainfall. The paper gives a description of using IoT to help the user to identify and monitor the temperature, humidity, barometric pressure, air quality, light intensity and rainfall in the environment. This dashboard will update real time values of the above parameters to help monitor, control and maintain a particular environment.

Contents

Chapter 1

Introduction

Recent years have seen a lot of interest in environmental monitoring and climatic change. Man wants to be aware of the most recent weather conditions in any location, such as a college campus or any other specific facility. There should be weather stations because the climate is changing so quickly. Here, we offer a weather station that is beneficial for all locations. Based on IOT, this weather station (internet of things). It has environmental sensors that can be utilised to report measurements from a particular location. The Weather Monitoring Systems in global Market is projected to grow from 20 thousand crores in 2021 to 26 thousand crores by 2026, at an annual growth of 5.1% in terms of value during the forecasted period. The motivation of this project is to create an ideal environment and keep a track of the internal and external surrounding weather of the aquaponics unit. The term "Internet of Things" (IoT) refers to networks of physical objects (things) that are equipped with sensors, software, and other technologies to connect to other devices and systems online and exchange data with them. It is a broad field that offers a place to build a variety of prototypes, from simple domestic devices to sophisticated industrial devices. The paper gives a description of using IoT to help the admin to identify and monitor the temperature, humidity, barometric pressure, air quality, light intensity and rainfall in the environment. This dashboard will update real time values of the above parameters to help monitor, control and maintain a particular environment.

1.1 Problem Overview

Human activities are influenced by weather conditions, monitoring of weather conditions can help in controlling of the activities. It is important to monitor and study the pattern of weather at surrounding. Limited way for user to know about weather such as temperature, humidity and wind speed. Without weather station, user can't be alerted of the extreme heat, heat waves or any other weather-related emergency.Furthermore, difficulty in making weather forecasts without data. When user use weather station, user can view the history of information as well. User can figure out the trends in the measurements. This will allow user to analyze the trends in a more effective way.

1.2 Objective

In this project, we have proposed an IOT and cloud based Weather Monitoring System. The aim of weather monitoring system is to detect , record and display various weather parameters such as temperature , humidity. This system makes use of sensors for detecting and monitoring weather parameters and then this collected information is sent to the cloud which can be accessed using the internet . The data displayed as an output can be observed and forecasted . The system engages an Arduino UNO board , sensors , WIFI Module which sends data to cloud computing services. A web page is also created which exhibits the data and displays it to users.
The purpose of this paper is to understand how IoT can be used to build a personalized weather monitoring system which can help users to visualize the real time data of various weather conditions in a dashboard.

1.3 Report Outline

I. Introduction
- Definition of IoT-based weather monitoring system
- Importance of weather monitoring

II. Components of IoT-based weather monitoring system
- Sensors
- Gateway
- Cloud platform
- User interface

III. Sensor technologies for weather monitoring
- Temperature sensors
- Humidity sensors
- Barometric pressure sensors
- Rainfall sensors
- Wind sensors

IV. Data transmission and communication
- Wireless communication protocols
- Data encryption and security

V. Cloud platform and analytics
- Data storage and management
- Real-time data visualization and analysis
- Machine learning algorithms for predictive weather analysis

VI. User interface and mobile application
- Dashboard and user interface design
- Alerts and notifications
- Data sharing and access control

VII. Case study: Implementation of IoT-based weather monitoring system
- Overview of the project
- Installation and configuration of components
- Data analysis and results

VIII. Conclusion and future directions
- Benefits and challenges of IoT-based weather monitoring system
- Future directions for research and development
- Conclusion and recommendations.

Chapter 2

2.1 History of IoT

The history of IoT (Internet of Things) can be traced back to the late 1990s when the concept of "embedded internet" was first introduced. The term "Internet of Things" was coined by Kevin Ashton in 1999, who envisioned a future where everyday objects could be connected to the internet and communicate with each other.Over the years, the development of IoT technology has been driven by advances in wireless communication, miniaturization of sensors, and the increasing availability of cloud computing. The emergence of IPv6, which provides a virtually unlimited number of IP addresses, has also played a significant role in the growth of IoT.In the early 2000s, the first IoT applications were developed, including RFID (Radio Frequency Identification) tags for inventory management and smart meters for utility companies. In 2008, the term "Industry 4.0" was introduced, referring to the integration of IoT and other technologies in manufacturing processes.In recent years, the growth of IoT has been exponential, with the number of connected devices expected to reach 25 billion by 2025. IoT technology is being applied in various fields, including healthcare, agriculture, transportation, and smart cities.The development of IoT-based weather monitoring systems is a recent application of IoT technology, which aims to improve the accuracy of weather forecasting and support a range of industries that rely on weather information.

Overall, the history of IoT reflects a gradual evolution of technology, driven by advances in communication, sensing, and computing, and a growing recognition of the potential benefits of connecting everyday objects to the internet.

2.1.1 Timeline

1. Research and Planning
2. Identify the specific needs and requirements for the weather monitoring system
3. Conduct market research to identify available IoT technologies and vendors
4. Develop a project plan and budget
5. Hardware and Software Selection
6. Evaluate different hardware and software options, including sensors, cameras, and communication protocols
7. Select the most appropriate components and vendors for the project
8. System Design and Development

2.1.2 Architecture

The architecture of an IoT-based weather monitoring system project would typically involve the Fig1 describes the development of an IoT-Based Weather Monitoring System in Aquaponics is one of a monitoring system with main focus on climate and weather monitoring. the sensors had to work without any connection to Raspberry Pi, since it will be mounted on the bread board and should be at rest at all times. The raspberry pi is then accessed from a desktop. Later the arduino code is run and the sensor output is retrieved. Finally the user can login to the web application and view the real time data from sensors. This is a general description of weather monitoring system architecture and it's working a detailed one involves a more in-depth knowledge.

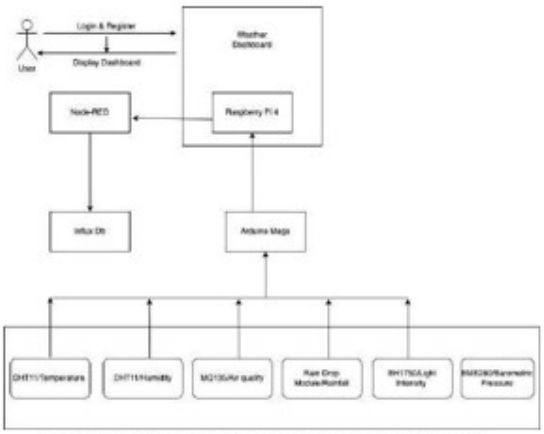


Fig. 1: Architecture of Weather Monitoring System

2.2 Throughput

In general, the throughput of an IoT-based project can be defined as A real-time weather monitoring system for a smart home and data is sent to cloud and data using telemetry transport protocol.There are many local weather stations around the globe, which can be collected from the authorities and this data can be visualized for the farmers to view real-time weather data. Cloud services can be used to store sensor data and later the information can be shown on website. Modules like NodeMCU and MQTT client can used. Microcontroller like Arduino and low-cost computers like Raspberry Pi can be used to build weather stations. This helps developers to build their own gadgets. Front-end technologies can be used to visualize sensor data and information in various formats and dashboard.

Chapter 3

3.1 Methodology and Set up

The methodology and set up for an IoT-based weather monitoring system typically involves the following steps:

1. Define the requirements: Determine the specific weather parameters that need to be monitored, the desired level of accuracy and precision, the frequency of data collection, and the response time for alerts and notifications.

2. Sensor selection: Select the appropriate sensors based on the defined requirements. Common weather sensors include temperature, humidity, pressure, wind speed and direction, and precipitation sensors.

3. Network architecture: Determine the network architecture for the weather monitoring system, including the type of wireless communication protocol and the network topology. This may involve selecting a low-power, long-range wireless protocol such as LoRaWAN or Sigfox, and determining the number and placement of gateways and repeaters.

4. Cloud-based data processing: Select a cloud-based data processing platform, such as AWS or Azure, for storing and processing the weather data. This may involve setting up data pipelines and defining the data processing workflows, including data cleaning, normalization, and analysis.

5. Visualization and reporting: Develop user interfaces and dashboards for visualizing and reporting the weather data. This may involve using tools such as Tableau or Power BI to create interactive visualizations and alerts.

6. Deployment and maintenance: Install the sensors, gateways, and repeaters in the field and ensure that they are properly calibrated and functioning. Set up a maintenance plan to ensure that the sensors are regularly serviced and updated.

Overall, the methodology and set up for an IoT-based weather monitoring system require careful planning and consideration of the specific application requirements. The use of low-power wireless communication protocols, cloud-based data processing platforms, and data visualization tools can help to enable real-time monitoring and analysis of weather data for a range of applications.

3.2 Description of working principle:-

3.2.1 – Block Diagram

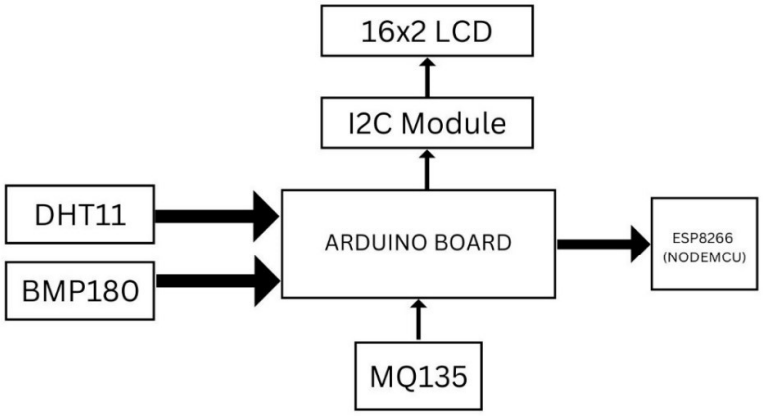


Figure 1.1: Block Diagram of Weather Monitoring System

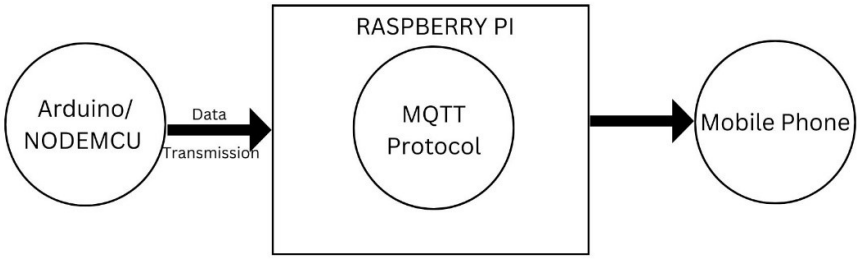
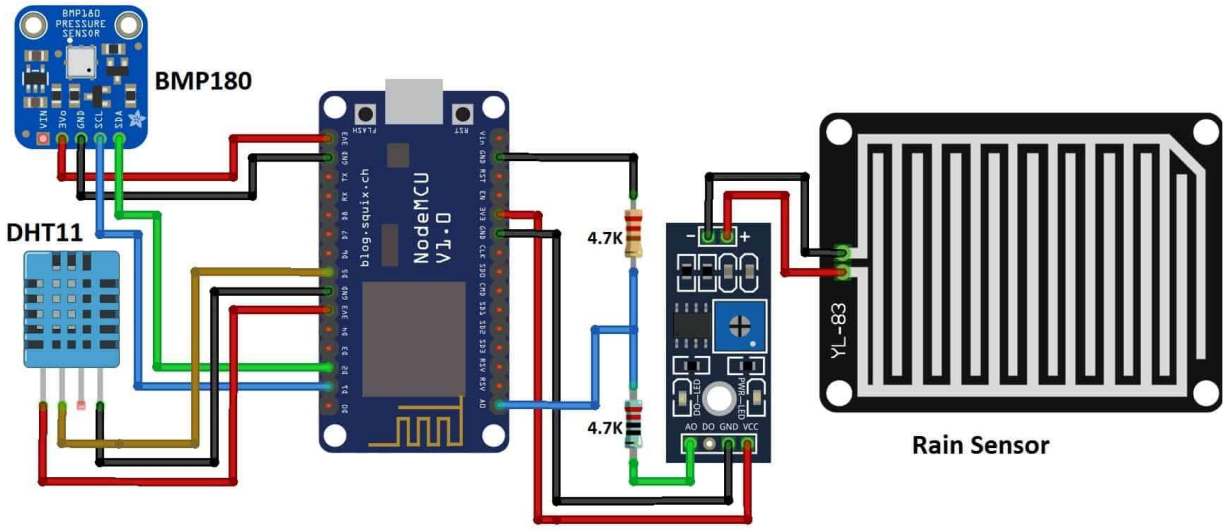


Figure 1.2: Block Diagram of Raspberry pi

3.2.2 Circuit diagram



NODEMCU (ESP8266)

3.3 Working principle

The working principle of an IoT-based weather monitoring system involves the following steps:

1. Data collection: Weather sensors are placed at various locations to measure different weather parameters such as temperature, humidity, pressure, wind speed and direction, and precipitation. These sensors are designed to collect data at regular intervals, typically ranging from a few seconds to a few minutes.

2. Wireless communication: The collected data is transmitted wirelessly using low-power, long-range wireless communication protocols such as LoRaWAN or Sigfox. This enables the sensors to communicate with gateways and repeaters, which act as intermediaries between the sensors and the cloud-based data processing platform.

3. Cloud-based data processing: The transmitted data is received by a cloud-based data processing platform such as AWS or Azure. This platform is responsible for processing the data in real-time, including data cleaning, normalization, and analysis. The processed data is stored in a database for later retrieval and analysis.

4. Data visualization: The processed data can be visualized using interactive dashboards and charts. This allows users to monitor weather conditions in real-time and identify patterns and trends over time.

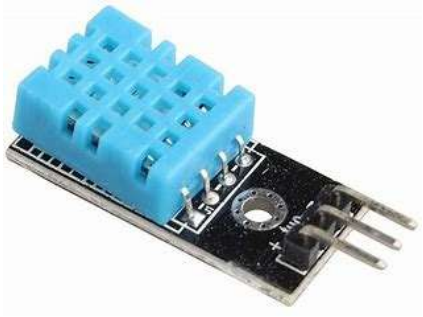
5. Alerts and notifications: The system can be configured to send alerts and notifications when specific weather conditions are met. For example, if the temperature exceeds a certain threshold, an alert can be sent to users via email or SMS.

3.4 Components

Hardware

1. Arduino uno- Arduino Uno is a microcontroller board based on the ATmega328P (datasheet). It has 14 digital input/output pins. Arduino UNO is a low-cost, flexible, and easy-to-use programmable open-source microcontroller board that can be integrated into a variety of electronic projects. This board can be interfaced with other Arduino boards, Arduino shields, Raspberry Pi boards and can control relays, LEDs, servos, and motors as an output

2. DHT11 Sensor - DHT11 is a low-cost digital sensor for sensing temperature and humidity. The temperature range of DHT11 is from 0 to 50 degree Celsius with a 2-degree accuracy. Humidity range of this sensor is from 20 to 80% with 5% accuracy



3. ESP8266 - ESP8266 is a low-cost, Wi-Fi-enabled microcontroller developed by Espressif Systems, a Chinese company. It is designed for embedded systems and IoT (Internet of Things) applications, and it has become popular among hobbyists and professionals alike due to its low cost and flexibility.

4. LCD Display(16x2) - LCD stands for Liquid Crystal Display. An LCD display is a flat-panel display that uses the light-modulating properties of liquid crystals to display images, videos, and other content.LCDs consist of several layers, including two polarized glass panels with liquid crystals in between them. The liquid crystals align themselves in response to the electrical current passing through them, allowing or blocking the light passing through the panels. This modulation of light creates the image that is displayed on the screen.

5. BMP180 - BMP180 is one of sensor of BMP XXX series. They are all designed to measure Barometric Pressure or Atmospheric pressure. BMP180 is a high precision sensor designed for consumer applications. Barometric Pressure is nothing but weight of air applied on everything. The air has weight and wherever there is air its pressure is felt.

6. MQ135 - The MQ 135 sensor can be implemented to detect smoke, benzene, vapors, and other hazardous gases. It can detect various harmful gases. It can be used for air quality monitoring, noxious gas detection, home air pollution detection, industrial pollution detection, portable air pollution detection, etc

CODE:

Arduino Code

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <SFE_BMP180.h>
#include <Wire.h>

#include "index.h" //Our HTML webpage contents with javascripts
#include "DHTesp.h" //DHT11 Library for ESP

#define LED 2 //On board LED
#define DHTpin 14 //D5 of NodeMCU is GPIO14

SFE_BMP180 pressure;

#define ALTITUDE 1655.0 // Altitude in meters

DHTesp dht;

//SSID and Password of your WiFi router
const char* ssid = "Alexahome";
const char* password = "hngzhowxiantan";

ESP8266WebServer server(80); //Server on port 80

void handleRoot() {
String s = MAIN_page; //Read HTML contents
server.send(200, "text/html", s); //Send web page
}

float humidity, temperature;

void handleADC() {
char status;
double T,P,p0,a;
double Tdeg, Tfar, phg, pmb;

status = pressure.startTemperature();
if (status != 0)
{
// Wait for the measurement to complete:
delay(status);
status = pressure.getTemperature(T);
if (status != 0)
{
// Print out the measurement:
Serial.print("temperature: ");
Serial.print(T,2);
Tdeg = T;
```

```
Serial.print(" deg C, ");
Tfar = (9.0/5.0)*T+32.0;
Serial.print((9.0/5.0)*T+32.0,2);
Serial.println(" deg F");

status = pressure.startPressure(3);
if (status != 0)
{
// Wait for the measurement to complete:
delay(status);
status = pressure.getPressure(P,T);
if (status != 0)
{
// Print out the measurement:
Serial.print("absolute pressure: ");
Serial.print(P,2);
pmb = P;
Serial.print(" mb, ");
phg = P*0.0295333727;
Serial.print(P*0.0295333727,2);
Serial.println(" inHg");

p0 = pressure.sealevel(P,ALTITUDE); // we're at 1655 meters (Boulder, CO)
Serial.print("relative (sea-level) pressure: ");
Serial.print(p0,2);
Serial.print(" mb, ");
Serial.print(p0*0.0295333727,2);
Serial.println(" inHg");

a = pressure.altitude(P,p0);
Serial.print("computed altitude: ");
Serial.print(a,0);
Serial.print(" meters, ");
Serial.print(a*3.28084,0);
Serial.println(" feet");
}
else Serial.println("error retrieving pressure measurement\n");
}
else Serial.println("error starting pressure measurement\n");
}
else Serial.println("error retrieving temperature measurement\n");
}
else Serial.println("error starting temperature measurement\n");


int rain = analogRead(A0);

//Create JSON data
String data =
"{\"Rain\":\""+String(rain)+"\",\"Pressuremb\":\""+String(pmb)+"\",\"Pressurehg\":\"
```

```
"+String(phg)+"\", \"Temperature\":\""+ String(temperature) +"\", \"Humidity\":\""+
String(humidity) +"\"}";

digitalWrite(LED,!digitalRead(LED)); //Toggle LED on data request ajax
server.send(200, "text/plane", data); //Send ADC value, temperature and humidity
JSON to client ajax request

delay(dht.getMinimumSamplingPeriod());

humidity = dht.getHumidity();
temperature = dht.getTemperature();

Serial.print("H:");
Serial.println(humidity);
Serial.print("T:");
Serial.println(temperature); //dht.toFahrenheit(temperature));
Serial.print("R:");
Serial.println(rain);
}

void setup()
{
Serial.begin(115200);
Serial.println();

// dht11 Sensor

dht.setup(DHTpin, DHTesp::DHT11); //for DHT11 Connect DHT sensor to GPIO 17
pinMode(LED,OUTPUT);

//BMP180 Sensor
if (pressure.begin())
Serial.println("BMP180 init success");
else
{
Serial.println("BMP180 init fail\n\n");
while(1); // Pause forever.
}

WiFi.begin(ssid, password); //Connect to your WiFi router
Serial.println("");

// Wait for connection
while (WiFi.status() != WL_CONNECTED) {
delay(500);
Serial.print(".");
}

//If connection successful show IP address in serial monitor
Serial.println("");
```

```
Serial.print("Connected to ");
Serial.println(ssid);
Serial.print("IP address: ");
Serial.println(WiFi.localIP()); //IP address assigned to your ESP

server.on("/", handleRoot); //Which routine to handle at root location. This is
display page
server.on("/readADC", handleADC); //This page is called by java Script AJAX

server.begin(); //Start server
Serial.println("HTTP server started");
}

void loop()
{
server.handleClient(); //Handle client requests
}
```

HTML File:

```
const char MAIN_page[] PROGMEM = R"=====(
<!DOCTYPE html>
<html>
<head>
<title>Mini Project OUTPUT</title>
</head>
<style>
@import url(https://fonts.googleapis.com/css?family=Montserrat);
@import url(https://fonts.googleapis.com/css?family=Advent+Pro:400,200);
*{margin: 0;padding: 0;}

body{
  background:#544947;
  font-family:Montserrat,Arial,sans-serif;
}
h2{
  font-size:14px;
}
.widget{
  box-shadow:0 40px 10px 5px rgba(0,0,0,0.4);
  margin:100px auto;
  height: 330px;
  position: relative;
  width: 500px;
}

.upper{
  border-radius:5px 5px 0 0;
  background:#f5f5f5;
  height:200px;
  padding:20px;
```

```
}

.date{
  font-size:40px;
}
.year{
  font-size:30px;
  color:#c1c1c1;
}
.place{
  color:#222;
  font-size:40px;
}
.lower{
  background:#00A8A9;
  border-radius:0 0 5px 5px;
  font-family:'Advent Pro';
  font-weight:200;
  height:130px;
  width:100%;
}
.clock{
  background:#00A8A9;
  border-radius:100%;
  box-shadow:0 0 0 15px #f5f5f5,0 10px 10px 5px rgba(0,0,0,0.3);
  height:150px;
  position:absolute;
  right:25px;
  top:-35px;
  width:150px;
}

.hour{
  background:#f5f5f5;
  height:50px;
  left:50%;
  position: absolute;
  top:25px;
  width:4px;
}

.min{
  background:#f5f5f5;
  height:65px;
  left:50%;
  position: absolute;
  top:10px;
  transform:rotate(100deg);
  width:4px;
}
```

```
.min,.hour{
  border-radius:5px;
  transform-origin:bottom center;
  transition:all .5s linear;
}

.infos{
  list-style:none;
}
.info{
  color:#fff;
  float:left;
  height:100%;
  padding-top:10px;
  text-align:center;
  width:25%;
}
.info span{
  display: inline-block;
  font-size:40px;
  margin-top:20px;
}
.weather p {
  font-size:20px;padding:10px 0;
}
.anim{animation:fade .8s linear;}

@keyframes fade{
  0%{opacity:0;}
  100%{opacity:1;}
}

a{
 text-align: center;
 text-decoration: none;
 color: white;
 font-size: 15px;
 font-weight: 500;
}
</style>
<body>


<div class="widget">
  <div class="clock">
    <div class="min" id="min"></div>
    <div class="hour" id="hour"></div>
  </div>
  <div class="upper">
```

```
    <div class="date" id="date">21 March</div>
    <div class="year">Temperature</div>
    <div class="place update" id="temperature">23 &deg;C</div>
  </div>
  <div style="text-align: center;"><a style="align:center">Mini Project
OUTPUT</a></div>
  <div class="lower">
    <ul class="infos">
      <li class="info temp">
        <h2 class="title">TEMPERATURE</h2>
        <span class='update' id="temp">21 &deg;C</span>
      </li>
      <li class="info weather">
        <h2 class="title">PRESSURE</h2>
        <span class="update" id="pressure">0 mb</span>
      </li>
      <li class="info wind">
        <h2 class="title">RAIN</h2>
        <span class='update' id="rain">0%</span>
      </li>
      <li class="info humidity">
        <h2 class="title">HUMIDITY</h2>
        <span class='update' id="humidity">23%</span>
      </li>
    </ul>
  </div>
</div>

<script>
setInterval(drawClock, 2000);

function drawClock(){
    var now = new Date();
    var hour = now.getHours();
    var minute = now.getMinutes();
    var second = now.getSeconds();

    //Date
    var options = {year: 'numeric', month: 'long', day: 'numeric' };
    var today  = new Date();
    document.getElementById("date").innerHTML = today.toLocaleDateString("en-US",
options);

    //hour
    var hourAngle = (360*(hour/12))+((360/12)*(minute/60));
    var minAngle = 360*(minute/60);
    document.getElementById("hour").style.transform = "rotate("+(hourAngle)+"deg)";
    //minute
    document.getElementById("min").style.transform = "rotate("+(minAngle)+"deg)";
```

```
    //Get Humidity Temperature and Rain Data
    var xhttp = new XMLHttpRequest();
    xhttp.onreadystatechange = function() {
      if (this.readyState == 4 && this.status == 200) {
      var txt = this.responseText;
      var obj = JSON.parse(txt); //Ref:
https://www.w3schools.com/js/js_json_parse.asp
        document.getElementById("rain").innerHTML = obj.Rain + "%";
        document.getElementById("temperature").innerHTML =
Math.round(obj.Temperature) + "&deg;C";
        document.getElementById("temp").innerHTML = Math.round(obj.Temperature) +
"&deg;C";
        document.getElementById("humidity").innerHTML = Math.round(obj.Humidity) +
"%";
        document.getElementById("pressure").innerHTML = Math.round(obj.Pressuremb) +
" mb";
      }
    };
   xhttp.open("GET", "readADC", true); //Handle readADC server on ESP8266
   xhttp.send();
}
</script>
</body>
</html>
)=====";
```

RASPBERRY PI CODE:

Cloud Code:

```
import paho.mqtt.client as mqtt
import urllib3

myAPI = 'apikey'
baseURL = 'https://api.thingspeak.com/update?api_key=%s' % myAPI
http = urllib3.PoolManager()
val=''
val2=''
def on_connect(client, userdata, flags, rc):
    print("Connected with result code "+str(rc))

    client.subscribe("/esp8266/temperature")
    client.subscribe("/esp8266/humidity")

def on_message(client, userdata, message):
    print("Received message '" + str(message.payload) + "' on topic '" +
message.topic)

    humi = 0
    temp = 0
    if message.topic == "/esp8266/temperature":
        print("temperature update")
```

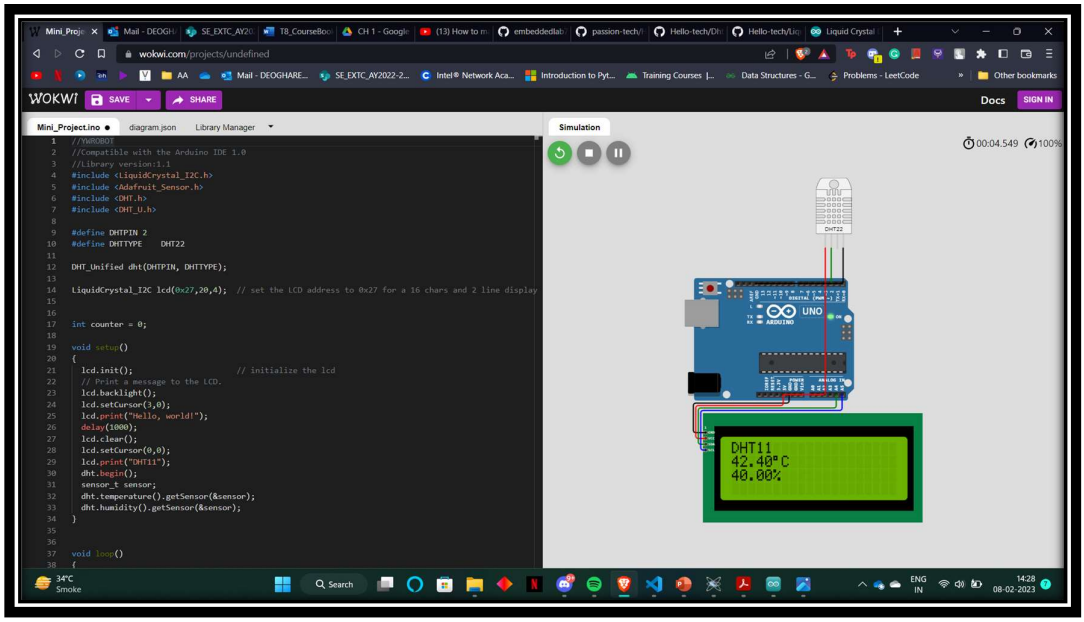
```
        temp = str(message.payload, 'UTF-8')
        temp = temp.strip()
        print(temp)
        global val
        val = temp

    if message.topic == "/esp8266/humidity":
        print("humidity update")
        humi = str(message.payload, 'UTF-8')
        humi = humi.strip()
        print(humi)
        global val2
        val2 = humi


    if val != '' and val2 != '':
        print(val, val2)
        conn = http.request('GET', baseURL + '&field1=%s&field2=%s'%(val,val2))
        print(conn.status)
        conn.read()
        conn.close()
        val = ''
        val2 = ''

def main():
    mqtt_client = mqtt.Client()
    mqtt_client.on_connect = on_connect
    mqtt_client.on_message = on_message

    mqtt_client.connect('localhost', 1883, 60)
    # Connect to the MQTT server and process messages in a background thread.
    mqtt_client.loop_start()

if __name__ == '__main__':
    print('MQTT to InfluxDB bridge')
    main()
```

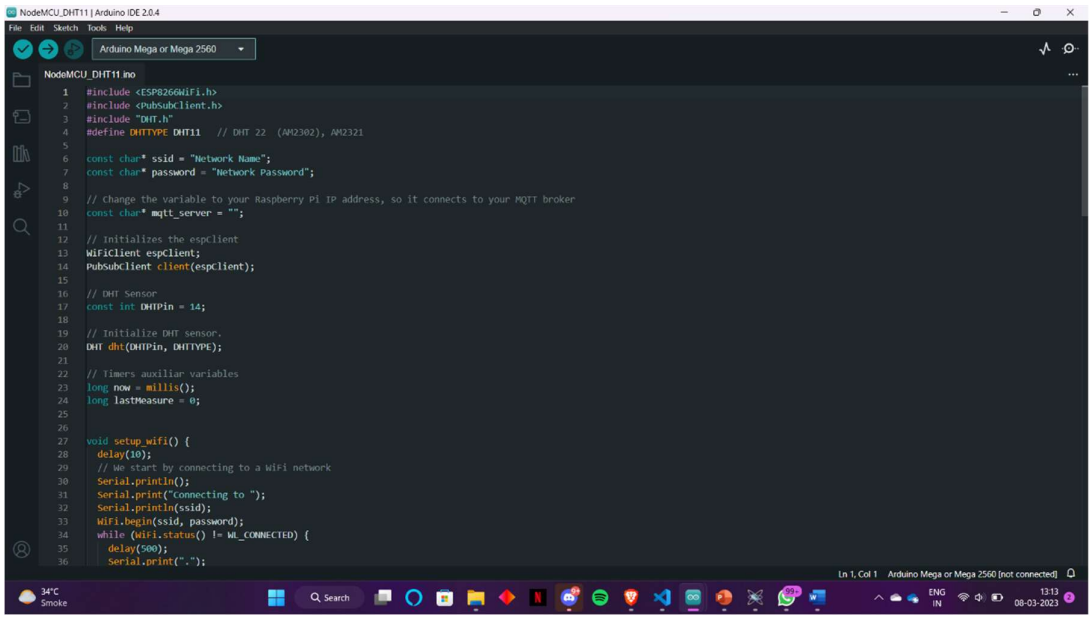
Server Code:

```
import paho.mqtt.client as mqtt

def on_connect(client, userdata, flags, rc):
    print("Connected with result code "+str(rc))

    # Subscribing in on_connect() means that if we lose the connection and
    client.subscribe("/esp8266/temperature")
    client.subscribe("/esp8266/humidity")

def on_message(client, userdata, message):
```

```
    print("Received message '" + str(message.payload) + "' on topic '" +
message.topic)


def main():
    mqtt_client = mqtt.Client()
    mqtt_client.on_connect = on_connect
    mqtt_client.on_message = on_message

    mqtt_client.connect('localhost', 1883, 60)
    # Connect to the MQTT server and process messages in a background thread.
    mqtt_client.loop_start()

if __name__ == '__main__':
    print('MQTT to InfluxDB bridge')
    main()
```

SOFTWARE SIMULATION:





Implementation (NODEMCU ESP8266)

3.6 Applications

An IoT-based weather monitoring system has various applications across different industries and domains, including:

1. Agriculture: Farmers can use weather data to optimize crop production and minimize the impact of adverse weather conditions such as drought, frost, or flooding.2. Transportation: Weather data can help to improve transportation safety by providing real-time information about road conditions, visibility, and wind speed.3. Energy: Energy companies can use weather data to optimize energy production and reduce downtime by anticipating changes in weather conditions.4. Aviation: Weather data is essential for safe and efficient aviation operations. Airlines use weather data to make informed decisions about flight routes, schedules, and delays.5. Disaster management: Weather data can help to mitigate the impact of natural disasters such as hurricanes, floods, or wildfires. Real-time weather data can provide early warning systems and help emergency services to respond quickly and effectively.6. Smart cities: Weather data can be used to optimize city operations, including traffic management, public safety, and environmental monitoring.

3.7 Future scope-

The technology of IoT has expanded in all sectors, and with the future scope and advantages of IoT-based weather monitoring systems, numerous industries can leverage them. The IoT weather reporting system has an application for farmers where they can ensure higher productivity of crops and lower the risk of weather hazards via the IoT weather. The IoT-based weather station proves helpful for monitoring the weather in areas like places with volcanoes or rain forests. This is especially important with drastic changes in the weather conditions we are experiencing. The IoT weather monitoring system using IoT supporting controllers is fully automated and efficient. It does not require any manual labor or attention. You can plan and visit the places anytime you like with prior notification of the weather conditions. You can simply get the status of the weather condition and the air quality, etc. Therefore, with the help of embedded devices and sensors, any environment can be converted to a smart environment for accumulating the data and analyzing the environment with real-time monitoring. Hence, with such advances on the Internet of Things (IoT), organizations are focusing on understanding the impact of weather on their operations and finding cutting-edge analytics on how to control the impact of their business

Chapter 4

4.1 Conclusion

There is a lot of variations continuous changes happening suddenly in the environment. So there is necessity of a system which constantly monitors the various parameters like temperature, humidity, rainfall, air quality, light intensity, barometric pressure. These are the basic weather parameters which needs to be monitored in any environment. These parameters will help judge users to take decisions in their sector if there are sudden changes in weather conditions. This dashboard developed using front end technologies help the users to monitor the weather conditions easily.When objects like an environment furnished with sensor devices, microcontrollers, and different software applications become a self-monitoring and self-protecting environment, it is called a smart environment.Similarly, here, the system uses sensors to monitor and adjust environmental parameters such as temperature, CO levels, and relative humidity. Then, it sends the data to a web page to plot the sensor data, shown as graphical statistics. The data updated from this system can be accessed on the internet from anywhere in the world. The embedded system enables the user to access the various criteria and store the data in the cloud.Hence, the Internet of Things (IoT) is the core root of linking all the sensors to the internet and monitoring the weather in real-time.

Bibliography

1. M. Lekić and G. Gardašević, "IoT sensor integration to Node-RED platform," 2018 17th International Symposium INFOTEH-JAHORINA (INFOTEH), East Sarajevo, Bosnia and Herzegovina, 2018, pp. 1-5, doi: 10.1109/INFOTEH.2018.8345544.
2. N. Naik, "Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP," 2017 IEEE International Systems Engineering Symposium (ISSE), Vienna, Austria, 2017, pp. 1-7, doi: 10.1109/SysEng.2017.8088251.
3. Internet of Things (IOT) Based Weather Monitoring system . AUTHOR : Bulipe Srinivas Rao, Prof. Dr. K. Srinivasa Rao, Mr. N. Ome.
4. Srivastava, D., Kesarwani, A., & Dubey, S. (2018). Measurement of Temperature and Humidity by using Arduino Tool and DHT11. International Research Journal of Engineering and Technology (IRJET), 5(12), 876-878.
5. Rahut, Y., Afreen, R., Kamini, D., & Gnanamalar, S. S. (2018). Smart weather monitoring and real time alert system using IoT. International Research Journal of Engineering and Technology.
6. https://www.mobileappdaily.com/iot-in-weather-technologies
7. http://www.pollutionequipmentnews.com/the-importance-of-weather-monitoring
8. http://www.ijser.in/archives/v6i7/IJSER172702

Acknowledgment

It is our pleasure to acknowledge the support and gratitude towards our guide Prof. Ankur Ganorkar for helping and accompanying the team in all the possible situations. The completion of this project could not have been achieved without the immense support of our guides and would like to thank them all for allowing us to express and put forth an idea worth making human life easier.

Ariba Afroz(3021105)

Rahul Deoghare(3021113)

Aditya Gawande(3021116)

Dipraj Hindole(3021119)