IOT-BASED AIR QUALITY MONITORING SYSTEM

PHASE 5

Problem Definition:

Air quality is a critical environmental factor that directly impacts human health and well-being. Poor air quality can lead to various health issues, including respiratory problems, allergies, and even cardiovascular diseases. To address this issue, an IoT-based air quality monitoring project is proposed to provide real-time data and insights about air quality in a specific area.

The problem at hand is to design and implement an IoT-based air quality monitoring system that can measure various air pollutants and environmental parameters in real-time. The system should be capable of:

a. Monitoring air pollutants

b. Collecting environmental data

c. Real-time data transmission

d. Data analysis and reporting

e. User-friendly interface

f. Scalability and modularity

The project will focus on developing and deploying a functional prototype in a specific geographic area, with the potential for expansion to cover a wider region based on the project's success.

Design Thinking :

Objectives:

1. Introduction:

- Provide an overview of the IoT air quality monitoring project.

- Briefly explain the significance of monitoring air quality and the project's goals.

2. **Empathize**:

- Describe the research and activities undertaken to understand the needs and concerns of stakeholders:

  - Conducted interviews with local residents, environmental experts, and healthcare professionals.

  - Analyzed existing air quality monitoring systems and their limitations.

3. **Define**:

- Summarize the key problem areas and challenges identified during the empathize phase:
  - Inadequate public awareness about air quality.
  - Limited integration with local government initiatives.

4. **Ideate**:

- Present the brainstorming and idea generation process:
- Generated innovative concepts for IoT-based air quality monitoring solutions.
- Encouraged creative thinking and blue-sky ideas.

5. Prototype:

- Describe the development of a prototype for the IoT air quality monitoring system:
- Integration of data transmission and analysis components.
- Prototyping and testing iterations to refine the solution.

6. Test:

- Explain the testing phase and user feedback:
- Conducted usability testing with potential end-users.
- Iteratively improved the prototype based on user input.

7. Implement:

- Detail the implementation of the IoT air quality monitoring system:
- Deployment of sensors in the target area.
- Integration with cloud infrastructure for data storage and analysis.

AIR QUALITY MONITORING INNOVATION:

1. Sensor Integration:

Use advanced sensors that can detect various air pollutants such as PM2.5, PM10, VOCs (Volatile Organic Compounds), CO2, CO, NO2, and O3. These sensors should be highly accurate and capable of real-time data collection.

2.Data Fusion:

Combine data from multiple sensors to provide a comprehensive view of air quality. Advanced algorithms can be used to fuse and analyze data to generate meaningful insights.

3.Mobile App and Web Interface:

Create a user-friendly mobile app and web interface that allows users to access real-time air quality data from anywhere. Include features like historical data, location-based information, and customizable alerts.

4.Geolocation and Mapping:

Use GPS data to map air quality across different locations. This feature can help users identify pollution hotspots and make informed decisions about their activities.

5.Predictive Analytics:

Implement machine learning models to predict future air quality based onhistorical data, weather forecasts, and other relevant parameters. This can be valuable for proactive measures.

6.Air Quality Index (AQI) Calculation:

Calculate and display AQI values, which provide a simple and standardized way to communicate air quality to the public.

7.Integration with Smart Devices:

Integrate the system with smart home devices like air purifiers and HVAC systems to enable automatic adjustments based on air quality.

SENSORS:

Gas Sensors:

Gas sensors in the context of the Internet of Things (IoT) refer to sensors that detect and measure various gases present in the environment. These sensors play a critical role in monitoring air quality, ensuring safety in industrial environments, and enabling environmental monitoring systems.

Applications:

Air Quality Monitoring: Gas sensors are used to monitor air pollutants in urban areas, industrial zones, and indoor environments to assess air quality and provide alerts for potential health hazards.

Industrial Safety: Gas sensors are deployed in industrial settings to monitor for potentially dangerous gases and ensure the safety of workers.

Environmental Monitoring: Gas sensors help in monitoring and analyzing the presence of harmful gases in natural environments such as forests, bodies of water, and remote locations.

Automotive Applications: Gas sensors are utilized in vehicles to monitor exhaust emissions and maintain compliance with environmental regulations.

Sensors for Humidity and Temperature:

Integrated Sensors: Many modern sensors are capable of measuring both humidity and temperature simultaneously. Common examples include digital humidity and temperature sensors such as the DHT series (e.g., DHT11, DHT22), BME280, and SHT series sensors.

Humidity Sensors: These sensors measure the moisture content in the air and provide readings in percentage (% relative humidity).

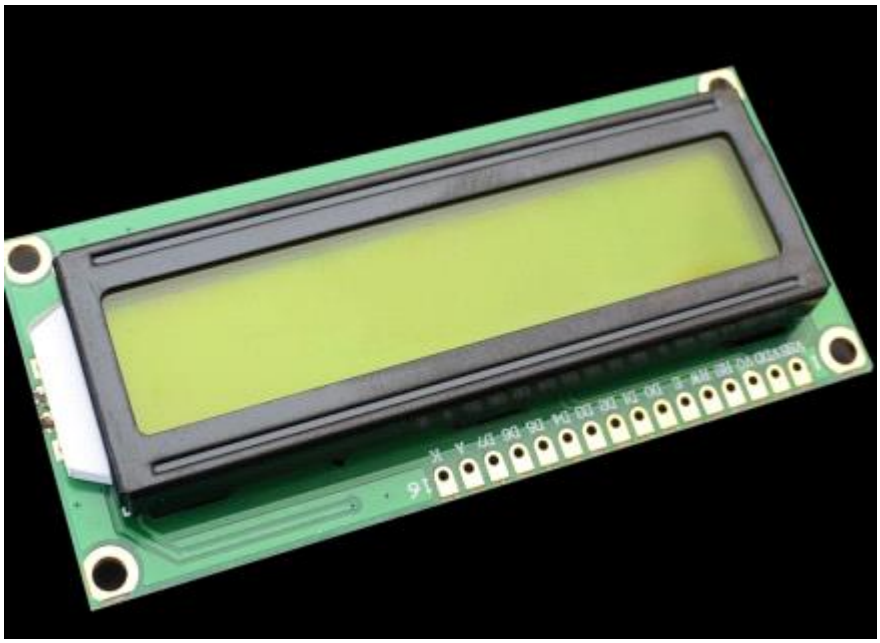
Temperature Sensors: These sensors measure the ambient temperature and provide readings in degrees Celsius (°C) or Fahrenheit (°F).

Arduino Uno:

Arduino Uno is a microcontroller board based on the ATmega328P microcontroller. It has digital and analog input/output pins that can be used to connect sensors, actuators, and other components.

Arduino Uno can be programmed using the Arduino Integrated Development Environment (IDE), which supports a simplified C/C++ programming language.

LCD DISPLAY:



1. Selecting an LCD Display:

Choose an appropriate LCD based on your requirements, such as size, types(character, graphical), and interface (e.g., I2C, SPI, parallel).

2. Connecting the LCD to Arduino Uno:

Wire the LCD display to the Arduino Uno using the required connections (e.g., data pins, power, ground). For example, if using an I2C-enabled LCD, connect it to the I2C pins (SDA and SCL) on the Arduino Uno.

3. Installing the LCD Library:

In the Arduino IDE, go to "Sketch" > "Include Library" > "ManageLibraries.&quot. Search for the appropriate LCD library for your display type (e.g.,LiquidCrystal, LiquidCrystal_I2C) and install it.

Creating an IoT-based air quality monitoring system using Arduino and Python involves several components and steps:

Components:

1. Arduino board (e.g., Arduino Uno or Arduino Nano)
2. Air quality sensor (e.g., MQ series sensor)
3. Wi-Fi module (e.g., ESP8266 or ESP32) for IoT connectivity
4. A computer or Raspberry Pi for running the Python program
5. Blynk or a similar IoT platform for data visualization (optional)

Arduino Code:

Write an Arduino sketch to read data from the air quality sensor and send it to the IoT platform (e.g., Blynk). Make sure to include the necessary libraries for your sensor and Wi-Fi module.

Here's a simplified example using the Adafruit CCS811 air quality sensor and an ESP8266 Wi-Fi module:

```
#include <Wire.h>
#include <Adafruit_CCS811.h>
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

char auth[] = "Your_Blynk_Auth_Token";
char ssid[] = "Your_WiFi_SSID";
char pass[] = "Your_WiFi_Password";

Adafruit_CCS811 ccs;

void setup() {
  Serial.begin(115200);
  Blynk.begin(auth, ssid, pass);
  ccs.begin();
}

void loop() {
  if (!ccs.available()) {
    return;
  }
  if (!ccs.readData()) {
    return;
  }
  float temperature = ccs.calculateTemperature();
  float humidity = ccs.calculateHumidity();
  int eCO2 = ccs.geteCO2();
  int TVOC = ccs.getTVOC();

  Serial.print("eCO2: ");
  Serial.println(eCO2);
  Serial.print("TVOC: ");
  Serial.println(TVOC);

  Blynk.virtualWrite(V1, eCO2); // Send eCO2 data to virtual pin V1
  Blynk.virtualWrite(V2, TVOC); // Send TVOC data to virtual pin V2
  Blynk.run();

  delay(1000); // Delay between readings
}
```

Python Code:

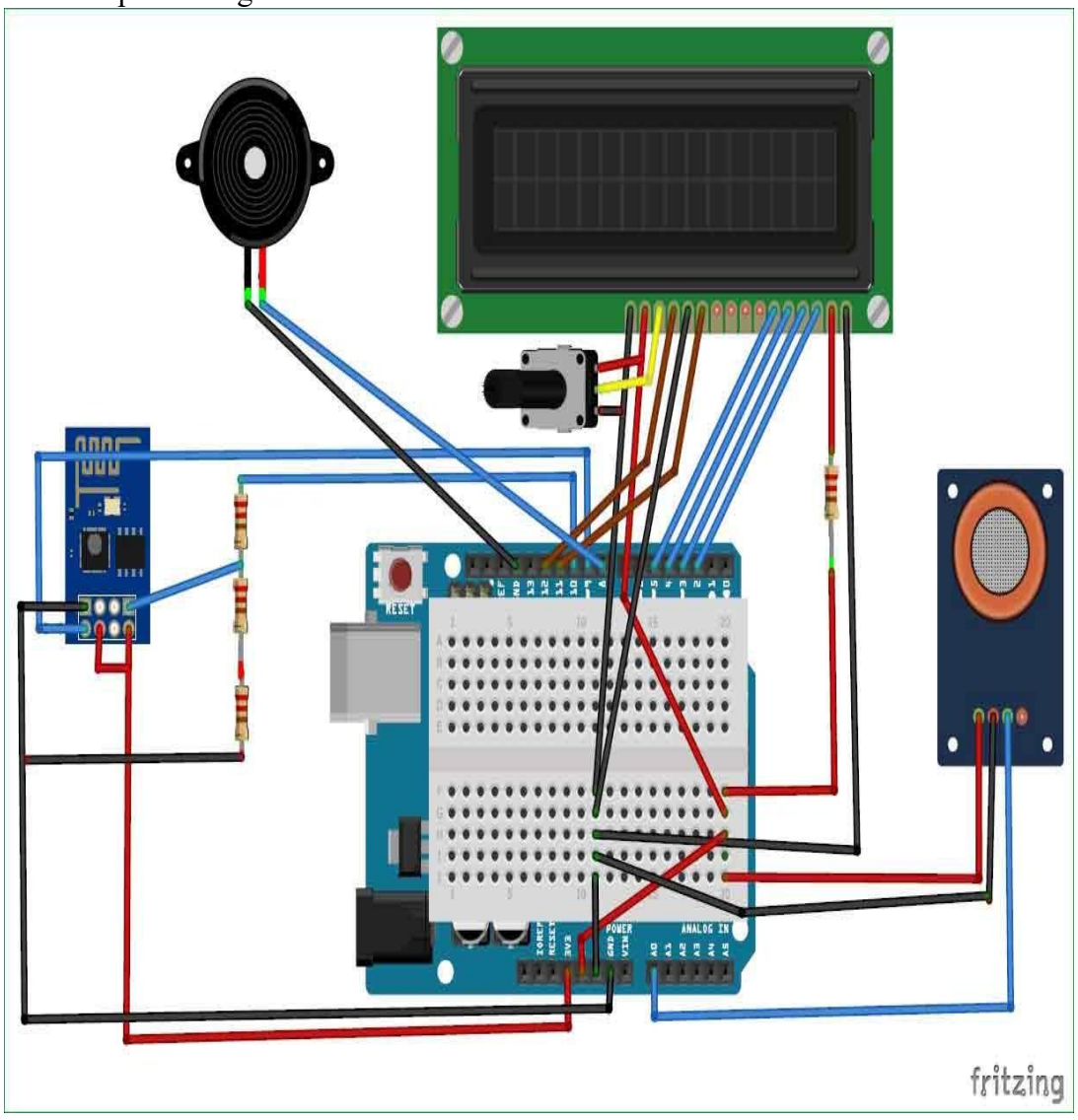
Create a Python program to receive the data sent by the Arduino via Wi-Fi. You can use the `pySerial` library to communicate with the Arduino over the serial port. Additionally, you may use an IoT platform's API for data retrieval if you're not using Blynk for visualization.

Here's a simple Python script to read data sent by the Arduino over the serial port:

```
import serial
ser = serial.Serial('COMX', 115200)  # Replace 'COMX' with the appropriate serial port
while True:
    data = ser.readline().decode('utf-8').strip()
    print(data)
```

CONNECTION:

In IoT-based air quality monitoring, the key is the seamless integration of sensors, IoT devices, connectivity, and a central platform that allows users to access and make informed decisions based on real-time and historical air quality data. The effectiveness of such a system depends on the reliability and accuracy of the sensors and the robustness of the communication and data processing infrastructure.



THING SPEAK CLOUD:

ThingSpeak is open-source software written in Ruby which allows users to communicate with internet-enabled devices. It facilitates data access, retrieval and logging of data by providing an API to both the devices and social network websites. ThingSpeak was originally launched by ioBridge in 2010 as a service in support of IoT applications. ThingSpeak has integrated support from the numerical computing software MATLAB from MathWorks, allowing ThingSpeak users to analyse and visualize uploaded data using MATLAB without requiring the purchase of a MATLAB license from MathWorks.

WORKING PROCEDURES:

NodeMCU plays the main controlling role in this project. It has been programmed in a manner, such that, it senses the sensory signals from the sensors and shows the quality level via led indicators. The DHT11 sensor module is used to measure the temperature and the humidity of the surroundings.With the help of the MQ-135 gas sensor module, air quality is measured in ppm. These data are fed to the ThinkSpeak cloud over the internet. We have also provided LED indicators to indicate the safety levels.

STEP 1 : Firstly, the calibration ofthe MQ-135 gas sensor module is done. The sensor is set to preheat for 24 minutes. Then the software code is uploaded to the NodeMCU followed by the hardware circuit to calibrate the sensor has been performed.

STEP 2 : Then, the DHT11 sensor is set to preheat for 10 minutes.

STEP 3 : The result of calibration found in STEP 1 is used to configure the final working code.

STEP 4 : The final working code is then uploaded to the NodeMCU.

STEP 5 : Finally, the complete hardware circuit is implemented.

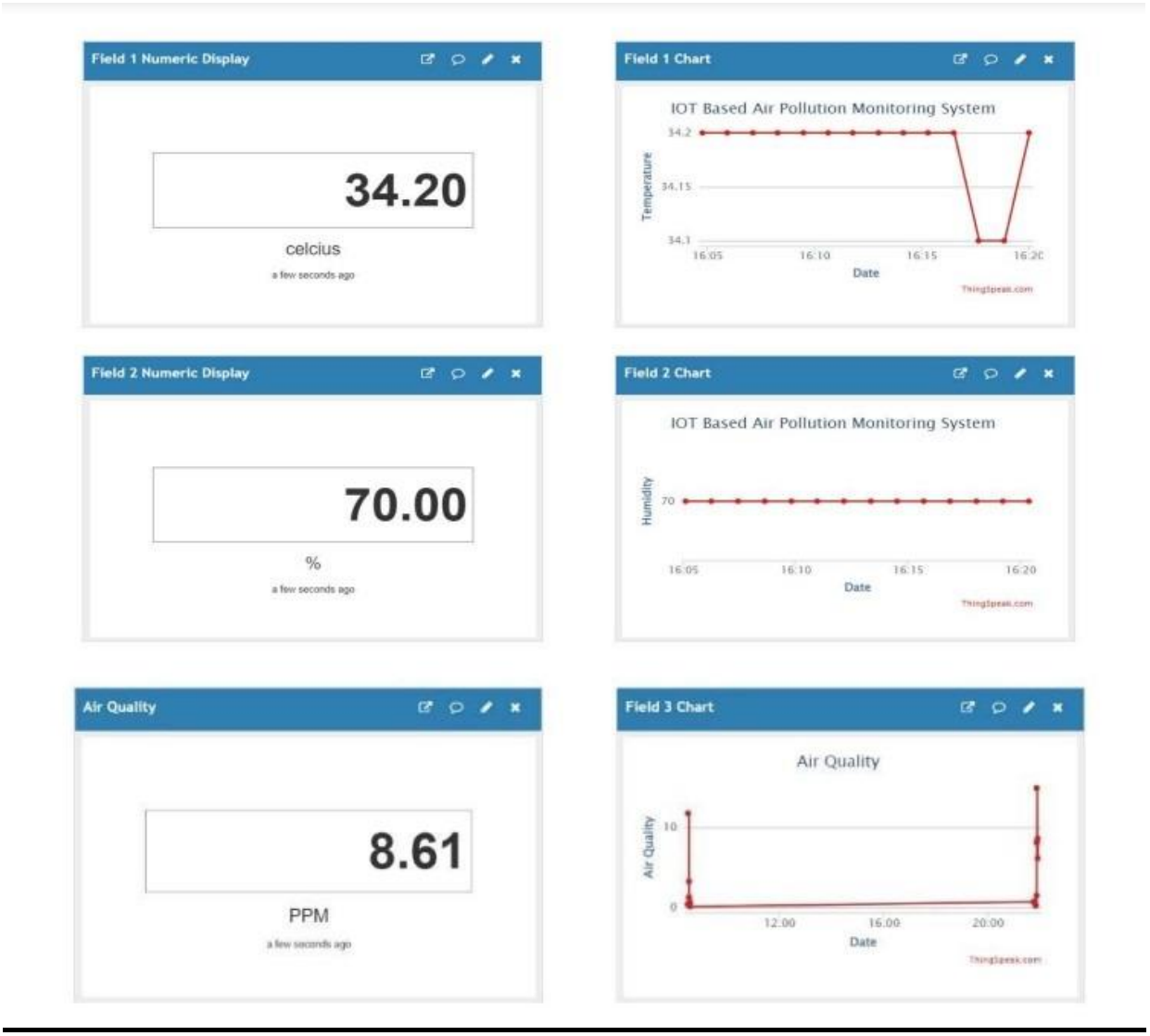
The software codes and the hardware circuits are described in the following chapters

SOFTWARE CODE:

```
void setup()
{
Serial.begin(9600); //Baud rate
19 | P a g e
pinMode(A0,INPUT);
}
void loop()
{
 float sensor_volt; //Define variable for sensor voltage
 float RS_air; //Define variable for sensor resistance

 float R0; //Define variable for R0
 float sensorValue=0.0; //Define variable for analog readings
Serial.print("Sensor Reading = ");
Serial.println(analogRead(A0));
for(int x = 0 ; x < 500 ; x++) //Start for loop
 {
sensorValue = sensorValue + analogRead(A0); //Add analog values of sensor 500 times
 }
sensorValue = sensorValue/500.0; //Take average of readings
sensor_volt = sensorValue*(5.0/1023.0); //Convert average to voltage
RS_air = ((5.0*1.0)/sensor_volt)-1.0; //Calculate RS in fresh air
 R0 = RS_air/3.7; //Calculate R0
Serial.print("R0 = "); //Display "R0"
Serial.println(R0); //Display value of R0
delay(1000); //Wait 1 second
}
```

OBSERVATION IN THINGSPEAK CLOUD:



Challenges we ran into:

- Data Accuracy: Advancements in sensor technology and calibration improve data accuracy.
- Remote Monitoring: Enables remote equipment maintenance and reduces physical visits.
- Data Integration: Improved mechanisms for comprehensive data integration and sharing.
- Data Interpretation: Data visualization tools and public awareness enhance understanding.
- Funding Sources: Diverse funding options support air quality monitoring.
- Privacy Regulations: Implementation of privacy regulations and data anonymization.

• Extreme Weather Preparedness: Robust equipment designs and protective measures.

• Data Security Measures: Cybersecurity protocols safeguard air quality data.

• Public Engagement: Campaigns promote awareness and citizen participation.

• Urban Planning Integration: Air quality data increasingly influences urban planning.

• Interference Mitigation: Improved sensor technology minimizes data interference.

Air quality monitoring benefits:

1. Healthier Environment:

By continuously monitoring air quality, these projects help identify and address pollution sources, leading to improved air quality and a healthier environment for residents.

2. Public Health:

Real-time air quality data allows individuals to take protective measures on days with poor air quality, reducing the risk of respiratory illnesses and other health issues.

3. Data-Driven Decision-Making:

Governments and organizations can make data-driven decisions to implement pollution control measures, urban planning, and traffic management, leading to a cleaner and safer environment.

4. Early Warning Systems:

Air quality monitoring can serve as an early warning system for extreme pollution events, such as wildfires or industrial accidents, enabling timely evacuations and emergency responses.

5. Reduced Healthcare Costs:

Better air quality can lead to reduced healthcare costs, as fewer people suffer from respiratory diseases and related illnesses.

6. Environmental Awareness:

These projects raise awareness about environmental issues and the impact ofhuman activities on air quality, encouraging individuals and businesses to adopt more sustainable practices.

7. Improved Quality of Life:

Clean air contributes to a better quality of life by reducing discomfort andhealth issues caused by pollution.

8. Data for Research:

The collected data can be invaluable for scientific research, allowing scientists to study long-term air quality trends, understand the effects of pollutants, and developmore effective pollution control strategies.

9. Efficient Resource Allocation:

Governments can allocate resources more efficiently by targeting areas withthe worst air quality for pollution control efforts.

10. Community Engagement:

Air quality monitoring projects often involve the community, fostering asense of responsibility and participation in environmental protection.

Conclusion:

the IoT Air Quality Monitoring project has successfully demonstrated the potential to significantly impact public health and environmental awareness. By integrating sensor technology, data analytics, and real-time monitoring, this project has provided valuable insights into air quality conditions. The implementation of this system has the potential to help mitigate the adverse effects of air pollution, inform policy decisions, and empower individuals to make informed choices for healthier living. As we continue to advance in the field of IoT and environmental monitoring, this project serves as a promising step towards a more sustainable and healthier future.