**PHASE 2:INNOVATION**

Creating an indoor air quality monitoring project using IoT involves a combination of hardware components and sensors. Here’s a list of commonly used elements for our project.

**HARDWARE COMPONENTS AND SENSORS:**

1.Microcontroller(raspberry pi or aurdino with a Wi-Fi)

2.Temperature and Humidity sensor:DHT22/DHT11

3.Carbon dioxide(CO2) Sensor:Non-Dispersive Infrared (NDIR) Sensors

4.Volatile Organic Compounds (VOC) Sensor:SGP30

5.Air Quality Sensor : MQ Series Sensors , Bosch BME680

6.Carbon monoxide(CO) Sensor:NDIR CO Sensor

**PROJECT STEPS:**

1. **Sensors:**
   * **Gas Sensors:** Use sensors capable of measuring specific gases like carbon dioxide (CO2), carbon monoxide (CO), volatile organic compounds (VOCs), methane, etc.
   * **Particulate Matter (PM) Sensors:** Measure the concentration of particulate matter in the air, especially PM2.5 and PM10.
   * **Temperature and Humidity Sensors:** Monitor the temperature and humidity levels in the indoor environment.
2. **Microcontroller/Single Board Computer:**
   * Choose a microcontroller or a single-board computer (SBC) to process and transmit the sensor data. Common choices include Raspberry Pi, Arduino, or ESP8266/ESP32.
3. **Connectivity Module:**
   * Integrate a communication module such as Wi-Fi, Bluetooth, or GSM to enable the IoT device to transmit data to a central server or cloud platform.
4. **Power Supply:**
   * Depending on your deployment, you might use a power adapter, batteries, or a combination of both. Ensure the chosen power supply method is reliable and suits the deployment environment.
5. **Enclosure:**
   * Protect your components from environmental factors with a suitable enclosure. This is especially important for outdoor or harsh indoor environments.
6. **Data Storage/Cloud Platform:**
   * Set up a cloud platform (e.g., AWS, Azure, Google Cloud) or a local server to store and analyze the collected data. This allows for remote monitoring and analysis.
7. **User Interface:**
   * Create a user interface for monitoring and interacting with the data. This could be a web dashboard, mobile app, or both, depending on your application.
8. **Power Management:**
   * Implement power management features to optimize energy consumption. This may include sleep modes for sensors or the entire system when not actively collecting data.
9. **Calibration Tools:**
   * Periodically calibrate the sensors to ensure accurate and reliable measurements. Calibration tools and procedures should be part of the system.
10. **Security Measures:**
    * Implement security features to protect the device and data, especially when transmitting data over the internet. This includes encryption, secure communication protocols, and access controls.
11. **Real-time Clock (RTC):**
    * Use an RTC to maintain accurate timestamps, even when the device is powered off. This is crucial for correlating data over time.

**PROGRAM:**

import com.pi4j.io.gpio.\*;

import com.pi4j.temperature.TemperatureScale;

import com.pi4j.temperature.TemperatureSensor;

import com.pi4j.temperature.TemperatureSensorFactory;

import com.pi4j.util.Console;

public class IndoorAirQualityMonitor {

public static void main(String[] args) {

final Console console = new Console();

// Initialize GPIO

GpioController gpio = GpioFactory.getInstance();

GpioPinDigitalInput dhtPin = gpio.provisionDigitalInputPin(RaspiPin.GPIO\_04, PinPullResistance.PULL\_DOWN);

// Initialize DHT22 sensor

TemperatureSensor dhtSensor = TemperatureSensorFactory.getTemperatureSensor(TemperatureScale.CELSIUS, dhtPin);

try {

while (true) {

// Read temperature and humidity from DHT22

double temperature = dhtSensor.getTemperature();

double humidity = dhtSensor.getHumidity();

// Print the data

console.println("Temperature: " + temperature + " °C, Humidity: " + humidity + "%");

// You can further process and store the data as needed

Thread.sleep(5000); // Delay between readings (adjust as needed)

}

} catch (InterruptedException e) {

console.println("Monitoring stopped");

}

}

}