Air quality Monitoring

Design of user interface with python:

```
import pandas as pd
import time

# Define the pollutant_data_readings dictionary
pollutant_data_readings = {}

# Define the pollutant_standards dictionary
pollutant_standards = {
    "PM2.5": 25,
    "PM10": 50,
    "Carbon monoxide": 9.0,
    "Nitrogen dioxide": 0.08,
    "Ozone": 0.065,
    "sulphur dioxide": 0.08,
    "ammonia": 9.0,
    "lead": 0.08
}

# Define a function to calculate the AQI
def calculate_aqi(pollutant_data_readings, pollutant_standards):
    aqi_values = {}
    for pollutant in pollutant_data_readings:
        aqi_values[pollutant] = (pollutant_data_readings[pollutant] /
pollutant_standards[pollutant]) * 100
```

```
    return aqi_values

# Get the latest data from the sensor
def get_sensor_readings():
    # This function will return a dictionary of pollutant data readings from the sensor
    return {
        "PM2.5": 20,
        "PM10": 40,
        "Carbon monoxide": 8,
        "Nitrogen dioxide": 0.07,
        "Ozone": 0.06,
        "sulphur dioxide": 0.07,
        "ammonia": 8,
        "lead": 0.07
    }

# Calculate the AQI
def update_aqi():
    pollutant_data_readings = get_sensor_readings()
    aqi_values = calculate_aqi(pollutant_data_readings, pollutant_standards)
    return aqi_values

# Store the AQI values in a DataFrame
aqi_df = pd.DataFrame(update_aqi(), index=["AQI"])

# Display the AQI DataFrame
```

```
print(aqi_df)


# Get the user's input
user_input = input("Press 'p' to display the stored AQI values, or any other key to continue: ")


# Display the stored AQI values if the user enters 'p'
def display_stored_aqi_values():
    print("Stored AQI values:")
    for pollutant in aqi_df.index:
        print(f"{pollutant}: {aqi_df.loc[pollutant, 'AQI']}")


if user_input == "p":
    display_stored_aqi_values()


# Wait for 1 second
time.sleep(1)


# Update the AQI values every second
while True:
    aqi_values = update_aqi()
    aqi_df.update(aqi_values)
    time.sleep(1)
```

Circuit For Air Quality Monitoring

- Electrostatic sensors for particulate matter
- Electrochemical CO Sensors for Carbon monoxide
- Ozone sensors

- Gas sensitive Semiconductor Sensors for nitrogen di oxide and Sulphur di oxide
- Metal oxide sensors for ammonia
- Atomic absorption sensors for lead
- The sensors are connected to the ESP8266 microcontroller using analog and digital pins. The specific pin connections will vary depending on the specific sensors that you are using. The other components of the circuit are
- ESP8266 microcontroller
- Resistors (1Ω, 2Ω)
- Capacitors (various values)
- Voltage regulators (various voltages)
- Sensors (listed above)
- Jumper wires
- Breadboard

The following is the code to be uploaded to ESP8266

Code :

```
import time

import board

import adafruit_esp8266


# Create an ESP8266 object

esp8266 = adafruit_esp8266.ESP8266(board.D4, board.D3)


# Connect to a Wi-Fi network

esp8266.connect('YOUR_WIFI_NETWORK', 'YOUR_WIFI_PASSWORD')


# Define the analog pins that the sensors are connected to

PM25_SENSOR_PIN = board.A0
```

```
CO_SENSOR_PIN = board.A1
OZONE_SENSOR_PIN = board.A2
NO2_SENSOR_PIN = board.A3
SO2_SENSOR_PIN = board.A4
NH3_SENSOR_PIN = board.A5
PB_SENSOR_PIN = board.A6

# Read the sensor values
pm25Reading = esp8266.analog_read(PM25_SENSOR_PIN)
coReading = esp8266.analog_read(CO_SENSOR_PIN)
ozoneReading = esp8266.analog_read(OZONE_SENSOR_PIN)
no2Reading = esp8266.analog_read(NO2_SENSOR_PIN)
so2Reading = esp8266.analog_read(SO2_SENSOR_PIN)
nh3Reading = esp8266.analog_read(NH3_SENSOR_PIN)
pbReading = esp8266.analog_read(PB_SENSOR_PIN)

# Calculate the AQI index for each sensor
pm25Aqi = (pm25Reading / 100) * 100
coAqi = (coReading / 100) * 100
ozoneAqi = (ozoneReading / 100) * 100
no2Aqi = (no2Reading / 100) * 100
so2Aqi = (so2Reading / 100) * 100
nh3Aqi = (nh3Reading / 100) * 100
pbAqi = (pbReading / 100) * 100

# Send the AQI values to a server
# ...

# Delay for 1 second
time.sleep(1)
```