Project: **ENVIRONMENT MONITORING**

Phase3: Development part 1

INTRODUCTION

Environment monitoring using IoT (Internet of Things) is a technology-driven approach to collect, analyze, and manage data related to various aspects of the environment. IoT devices, such as sensors and actuators, are deployed in the environment to gather real-time information and enable data-driven decision-making.

Components Required

- Esp32
- Dht22

Hardware Development

Hardware development for environmental monitoring using IoT can vary greatly depending on the specific application, environmental conditions, and data requirements. It's essential to plan and design your hardware with these factors in mind to ensure reliable and accurate monitoring.

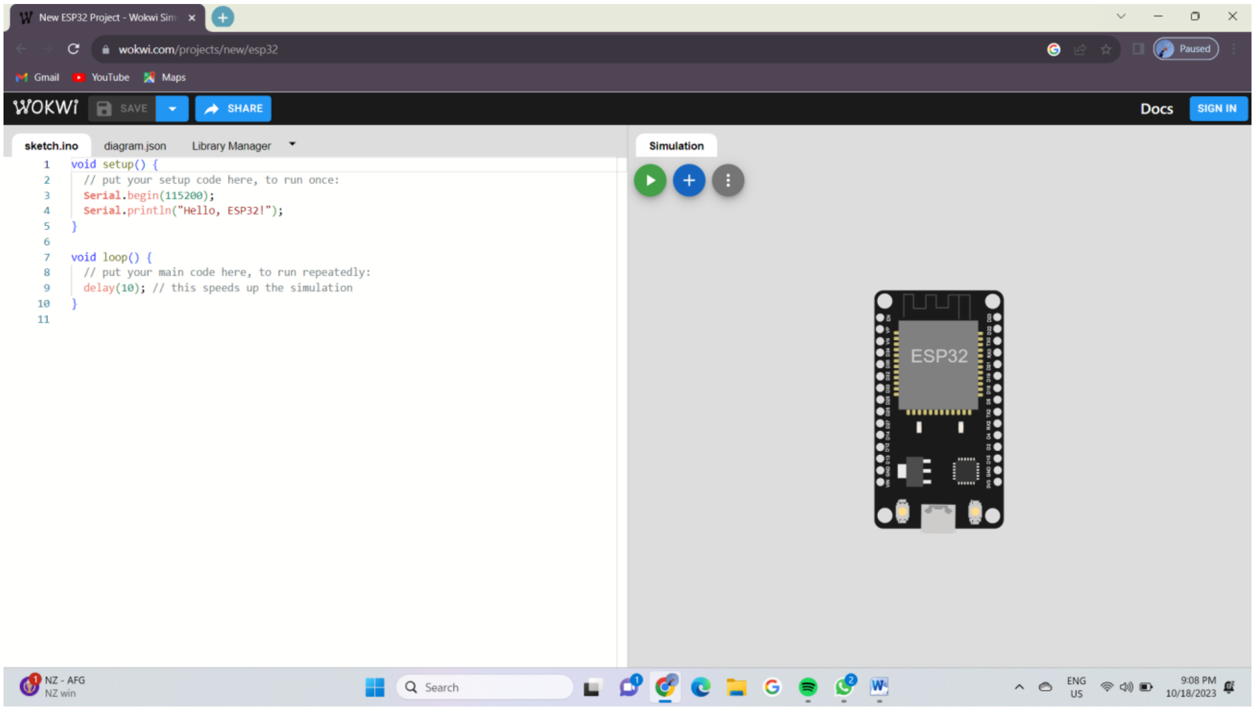
Sensor Selection: Choose appropriate sensors for monitoring parameters like temperature, humidity, air quality, water quality, soil moisture, etc. Select sensors based on the specific environmental factors you want to monitor.

Microcontroller or SoC: Use a microcontroller or System-on-Chip (SoC) such as Arduino, Raspberry Pi, or specialized IoT platforms like ESP8266 or ESP32 to interface with the sensors and manage data transmission.

Power Supply: Ensure a reliable power source for your hardware, which could be batteries, solar panels, or a combination of power sources depending on the deployment environment.

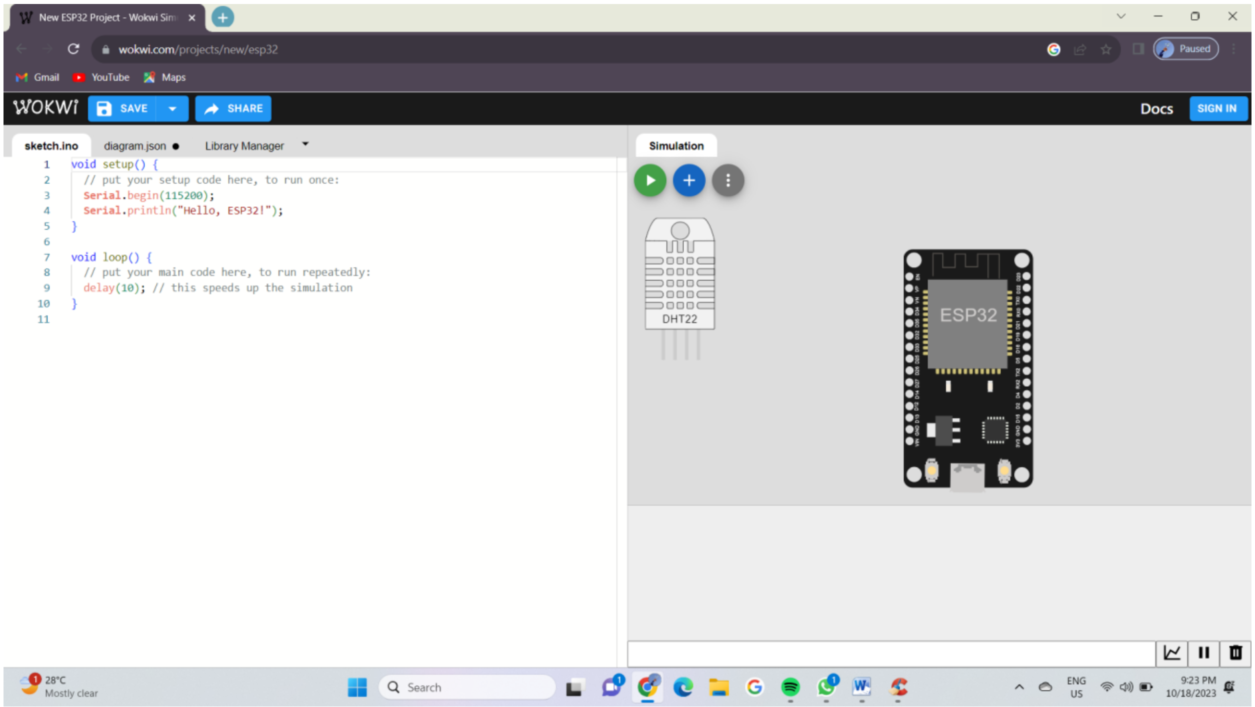
Data Transmission: Integrate communication modules (e.g., Wi-Fi, LoRa, cellular) to send data to a central server or cloud platform. MQTT and HTTP are common protocols for transmitting data.

STEP 1:



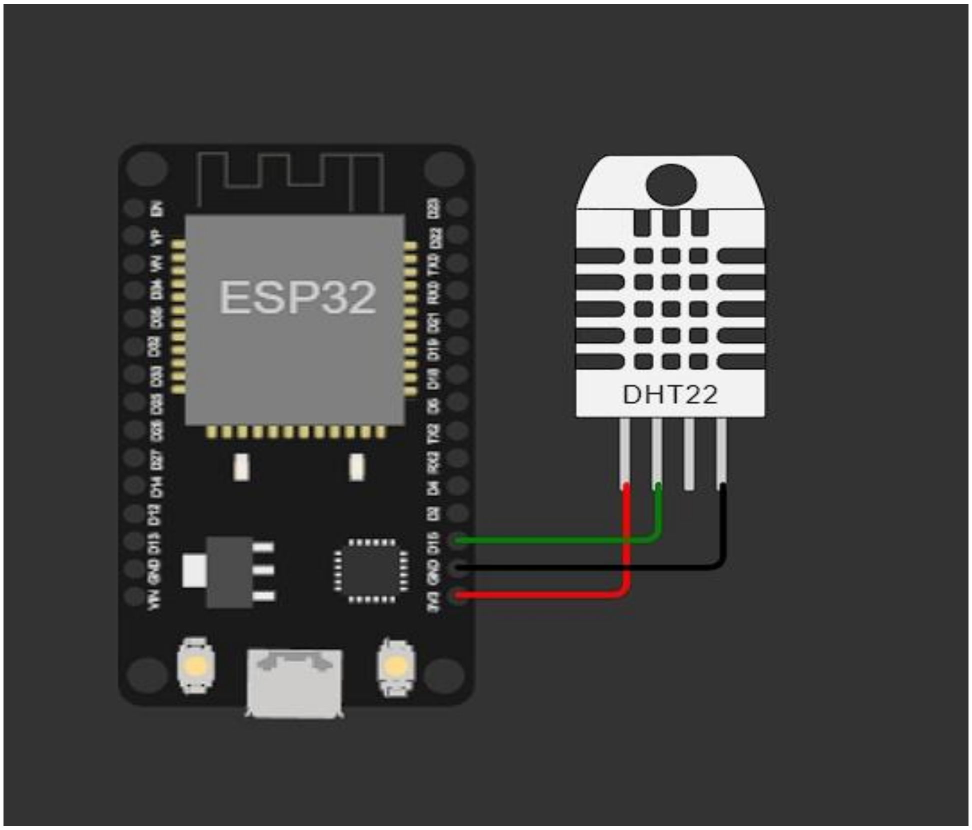
```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  Serial.println("Hello, ESP32!");
}

void loop() {
  // put your main code here, to run repeatedly:
  delay(10); // this speeds up the simulation
}
```

STEP 2:

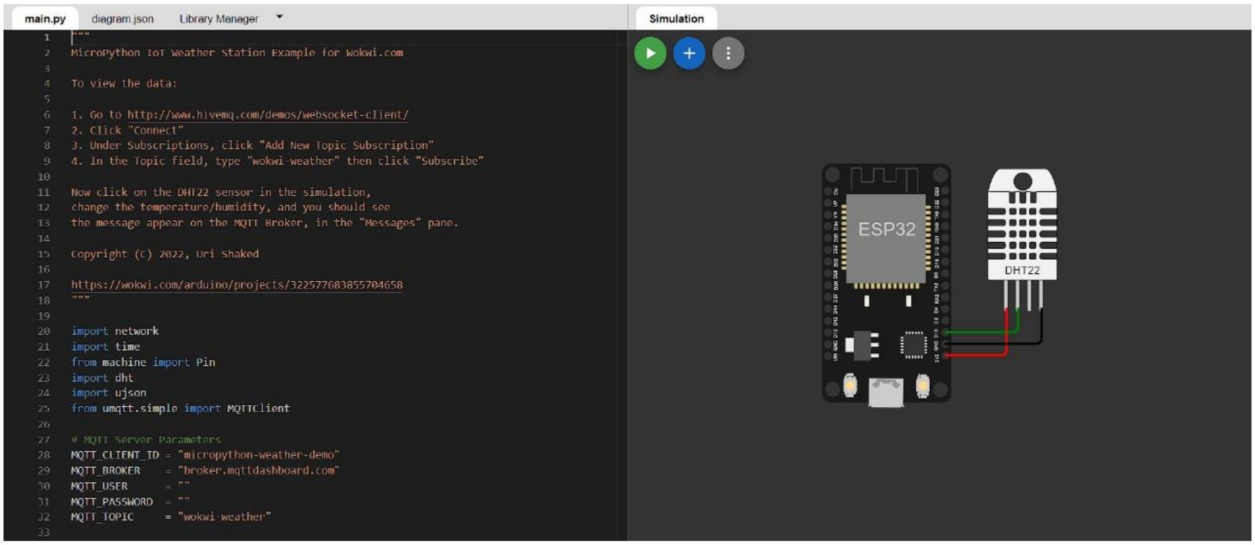


```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  Serial.println("Hello, ESP32!");
}

void loop() {
  // put your main code here, to run repeatedly:
  delay(10); // this speeds up the simulation
}
```

STEP 3:



STEP 4:



```
"""
MicroPython IoT Weather Station Example for Wokwi.com

To view the data:

1. Go to http://www.hivemq.com/demos/websocket-client/
2. Click "Connect"
3. Under Subscriptions, click "Add New Topic Subscription"
4. In the Topic field, type "wokwi-weather" then click "Subscribe"

Now click on the DHT22 sensor in the simulation,
change the temperature/humidity, and you should see
the message appear on the MQTT Broker, in the "Messages" pane.

Copyright (C) 2022, Uri Shaked

https://wokwi.com/arduino/projects/322577683855704658
"""

import network
import time
from machine import Pin
import dht
import ujson
from umqtt.simple import MQTTClient

# MQTT Server Parameters
MQTT_CLIENT_ID = "micropython-weather-demo"
MQTT_BROKER    = "broker.mqttdashboard.com"
MQTT_USER      = ""
MQTT_PASSWORD  = ""
MQTT_TOPIC     = "wokwi-weather"

```

Software Development:

PYTHON SUBSCRIPT:

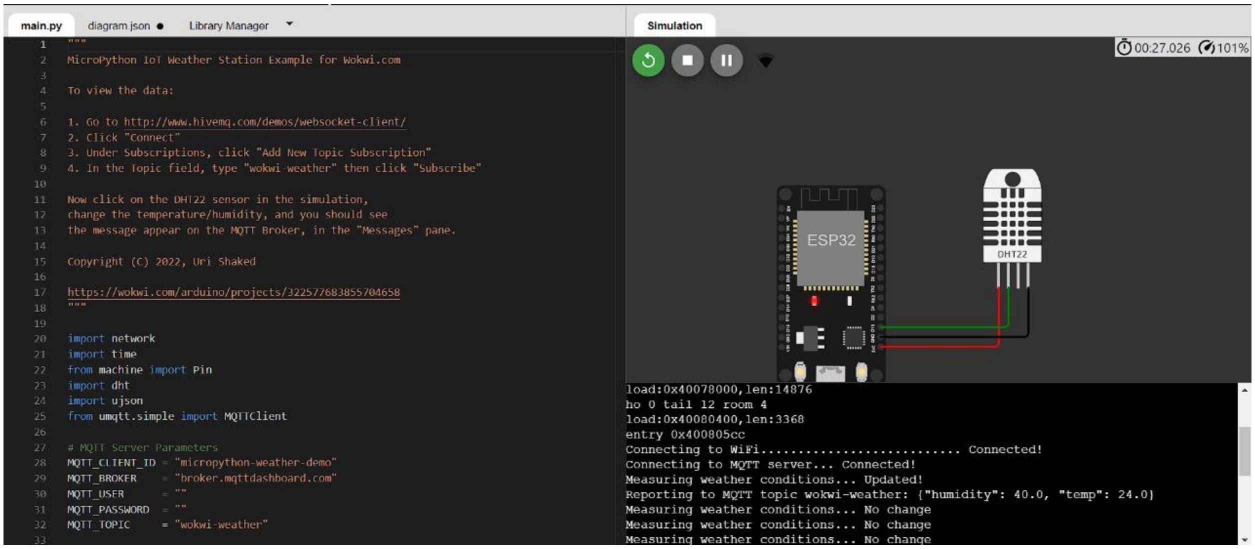
```
import network
import time
From machine import Pin
Import dht
Import ujson
From umqtt.simple import MQTTClient

# MQTT Server Parameters
MQTT_CLIENT_ID = "micropython-weather-demo"
MQTT_BROKER    = "broker.mqttdashboard.com"
MQTT_USER      = ""
MQTT_PASSWORD  = ""
MQTT_TOPIC     = "wokwi-weather"

sensor = dht.DHT22(Pin(15))

print ("Connecting to Wi-Fi", end="")
sta_if = network.WLAN(network.STA_IF)
sta_if.active(True)
sta_if.connect('Wokwi-GUEST', '')
while not sta_if.isconnected():
print(".", end="")
time.sleep(0.1)
print(" Connected!")

print("Connecting to MQTT server... ", end="")
client = MQTTClient (MQTT_CLIENT_ID, MQTT_BROKER, user=MQTT_USER,
password=MQTT_PASSWORD)
client.connect()

print ("Connected!")

prev_weather = ""
while True:
Print ("Measuring weather conditions... ", end="")
sensor.measure()
```

```
message = ujson.dumps({
"temp": sensor. temperature (),
"Humidity": sensor. Humidity (),
})
if message != prev_weather:
print("Updated!")
print("Reporting to MQTT topic {}: {}".format(MQTT_TOPIC, message))
client.publish(MQTT_TOPIC, message)
prev_weather = message
else:
print("No change")
time.sleep(1)
```

simulation output:

```
 (POWERON_RESET), boot: 0x13 (SPI_FAST_FLASH_BOOT)

configsip: 0, SPIWP:0xee

clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00

mode:DIO, clock div:2

load:0x3fff0030,len:4728

load:0x40078000,len:14876

ho 0 tail 12 room 4

load:0x40080400,len:3368

entry 0x400805cc

Connecting to
WiFi......................................................................................................................
..............................................................................................................................
..............................................................................................................................
............... Connected!

Connecting to MQTT server... Connected!

Measuring weather conditions... Updated!

Reporting to MQTT topic wokwi-weather: {"humidity": 40.0, "temp": 24.0}

Measuring weather conditions... No change
Measuring weather conditions... No change
Measuring weather conditions... No change
Measuring weather conditions... No change
Measuring weather conditions... No change
Measuring weather conditions... Updated!
Reporting to MQTT topic wokwi-weather: {"humidity": 80.5, "temp": 48.8}
Measuring weather conditions... Updated!
Reporting to MQTT topic wokwi-weather: {"humidity": 80.5, "temp": -13.8}
Traceback (most recent call last):
  File "main.py", line 62, in <module>
  File "umqtt/simple.py", line 134, in publish
OSError: [Errno 104] ECONNRESET
MicroPython v1.21.0 on 2023-10-05; Generic ESP32 module with ESP32
Type "help ()" for more information
```

OUTPUT



```python
"""
MicroPython IoT Weather Station Example for Wokwi.com

To view the data:

1. Go to http://www.hivemq.com/demos/websocket-client/
2. Click "Connect"
3. Under Subscriptions, click "Add New Topic Subscription"
4. In the Topic field, type "wokwi-weather" then click "Subscribe"

Now click on the DHT22 sensor in the simulation,
change the temperature/humidity, and you should see
the message appear on the MQTT Broker, in the "Messages" pane.

Copyright (C) 2022, Uri Shaked

https://wokwi.com/arduino/projects/322577683855704658
"""

import network
import time
from machine import Pin
import dht
import ujson
from umqtt.simple import MQTTClient

# MQTT Server Parameters
MQTT_CLIENT_ID = "micropython-weather-demo"
MQTT_BROKER    = "broker.mqttdashboard.com"
MQTT_USER      = ""
MQTT_PASSWORD  = ""
MQTT_TOPIC     = "wokwi-weather"
```

```
load:0x40078000,len:14876
ho 0 tail 12 room 4
load:0x40080400,len:3368
entry 0x400805cc
Connecting to WiFi........................... Connected!
Connecting to MQTT server... Connected!
Measuring weather conditions... Updated!
Reporting to MQTT topic wokwi-weather: {"humidity": 40.0, "temp": 24.0}
Measuring weather conditions... No change
Measuring weather conditions... No change
Measuring weather conditions... No change
```