# ENVIRONMENTAL MONITORING SYSTEM

PHASE-2: INNOVATION

IoT-based environmental monitoring systems use various components to collect data on environmental conditions. These components are crucial for gathering, transmitting, and analyzing data for a wide range of applications, including weather monitoring, air quality assessment, agriculture, and industrial settings.

Here are some essential components for an IoT-based environmental monitoring system: Sensors are the primary components that measure environmental parameters. They can include various types, such as:

- Temperature sensors
- Humidity sensors
- Pressure sensors
- Gas sensors (e.g., for CO2, CO, VOCs)
- Particulate matter sensorsSoil
- moisture sensors
- pH sensors
- Light sensors (for sunlight intensity)
- Rainfall sensors
- Wind speed and direction sensors

## HARDWARE COMPONENTS:

1. **the ESP32**

2. **the DHT22 Sensor**
3. BME280 Sensor
4. Soil Moisture Sensor
5. Ultrasonic Distance Sensor
6. DS18B20 Sensor

Here's a step-by-step guide on how to do that:

1. **Create a New Project on Wokwi**:

   - Visit the Wokwi platform (https://wokwi.com/).
   - Create an account if you haven't already.

2. **Create a New Project**:

   - Click on the "Create New Project" button to start a new project.

3. **Add ESP32 to Your Project**:

   - In the project editor, search for "ESP32."

- Drag and drop the ESP32 component onto your virtual breadboard.

4. **Add DHT22 Simulation**:

- While Wokwi may not support direct simulation of the DHT22 sensor, you can simulate it using a combination of a potentiometer and an analog pin.
- Add a potentiometer (usually under the "Inputs" section in the component list) to the breadboard. This will serve as a stand-in for simulating analog sensor data.
- Connect the middle pin of the potentiometer to an available analog pin on the ESP32 (e.g., GPIO32).
- You can adjust the potentiometer's position to simulate changing sensor values.

5. **Write Arduino Code**:

- Click on the "Code" tab in the project editor.
- Write Arduino code to read analog data from the virtual potentiometer, simulate it as temperature and humidity values, and then display the results. Here's a basic example:

```cpp
```

```
const int potPin = 32; // The analog pin to which the
potentiometer is connected

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

void loop() {
  int sensorValue = analogRead(potPin); // Simulated sensor
reading
  float voltage = (sensorValue / 1023.0) * 3.3; // Convert to
voltage

  // Convert voltage to temperature and humidity (this is just
an example)
  float temperature = voltage * 20; // Adjust the conversion as
needed
  float humidity = voltage * 50;    // Adjust the conversion as
needed

  Serial.print("Temperature: ");
  Serial.print(temperature);
  Serial.println(" °C");
  Serial.print("Humidity: ");
  Serial.print(humidity);
  Serial.println(" %");

  delay(2000);
}
```

6. **Compile and Upload**:

   - After writing your code, click the "Compile" button to check for any compilation errors.
   - If there are no errors, click the "Upload" button to upload the code to the virtual ESP32.

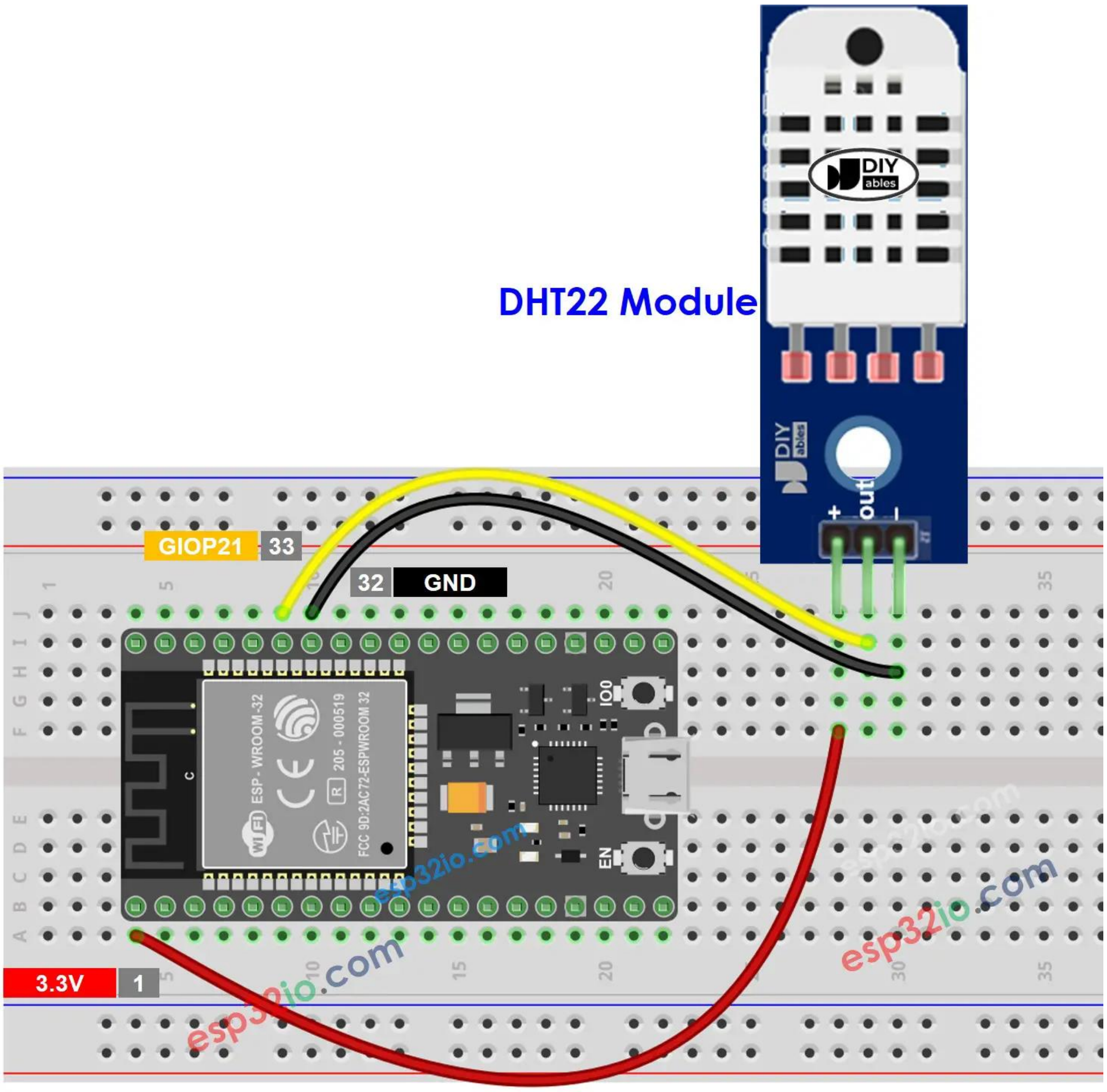
7. **Simulate**:

   - Click on the "Simulate" button to start the simulation.
   - Adjust the potentiometer position to simulate changing temperature and humidity values.
   - The virtual ESP32 will display the simulated temperature and humidity values in the Serial Monitor.

Please note that this is a basic simulation and does not accurately represent the behavior of a DHT22 sensor. It's a workaround to simulate the sensor's behavior on the Wokwi platform. Additionally, my knowledge is based on information available up to September 2021, and there may have been updates or changes to Wokwi since then.

```
#include <Adafruit_Sensor.h>
#include <DHT.h>

#define DHTPIN 2       // Define the GPIO pin where your DHT22 sensor is connected (change this to the actual pin you used)
#define DHTTYPE DHT22  // Change this to DHT11 if you're using that sensor

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(115200);
  dht.begin();
}

void loop() {
  delay(2000);  // Delay between sensor readings

  float humidity = dht.readHumidity();
  float temperature = dht.readTemperature();

  if (isnan(humidity) || isnan(temperature)) {
    Serial.println("Failed to read from DHT sensor!");
  } else {
    Serial.print("Humidity: ");
    Serial.print(humidity);
    Serial.print(" %\t");
    Serial.print("Temperature: ");
    Serial.print(temperature);
    Serial.println(" °C");
  }
}
```

CIRCUIT DIAGRAM:



APPLICATIONS:
The ESP32 microcontroller and the DHT22 (or DHT11) sensor are commonly used for various environmental monitoring applications due to their low cost, versatility, and ease of use. Here are some applications where you can utilize the ESP32 and DHT22 for environmental monitoring:

1. **Home Weather Stations**:

   - Build a home weather station to monitor temperature and humidity both indoors and outdoors. You can use the ESP32 to collect data from multiple DHT22 sensors and display it on a local display or upload it to the cloud for remote monitoring.

2. **Indoor Climate Control**:

   - Use the ESP32 and DHT22 to monitor the indoor climate in homes or offices. This data can be used to control heating, ventilation, and air conditioning systems for optimal comfort and energy efficiency.

3. **Greenhouses**:

   - Monitor temperature and humidity in a greenhouse to ensure that plants receive the ideal growing conditions. The ESP32 can control fans, heaters, and misting systems to maintain the desired environment.

4. **Server Room Monitoring**:

   - Keep tabs on the environmental conditions in server rooms and data centers to prevent equipment overheating. If the temperature or humidity exceeds safe limits, the ESP32 can trigger alarms or alerts.

5. **Wine Cellar Monitoring**:

   - Maintain ideal conditions for wine storage by monitoring temperature and humidity in wine cellars. The ESP32 can alert you if conditions deviate from the optimal range.

6. **Agriculture**:

   - In agriculture, the ESP32 and DHT22 can be used to monitor environmental conditions in crop fields, helping farmers make decisions about irrigation and pest control.

7. **Energy Efficiency**:

   - In homes and commercial buildings, the ESP32 can monitor environmental conditions to optimize energy use. For instance, it can control window blinds based on sunlight and temperature.

8. **Environmental Research**:

   - Researchers can deploy multiple ESP32-DHT22 setups in the field to collect data for environmental studies, such as studying microclimates, tracking climate change, or monitoring wildlife habitats.

9. **Food Storage**:

  - Monitor temperature and humidity in food storage areas, such as refrigerators and warehouses, to ensure that perishable goods are stored under appropriate conditions.

10. **Mushroom Farms**:

  - ESP32-DHT22 setups are used to monitor and control the climate in mushroom farms, ensuring proper growing conditions for different types of mushrooms.

11. **Healthcare**:

  - Monitor and control the climate in healthcare facilities, such as hospitals and laboratories, to maintain the integrity of sensitive equipment and biological samples.

_