IOT Report

Name: Hamza Iqbal

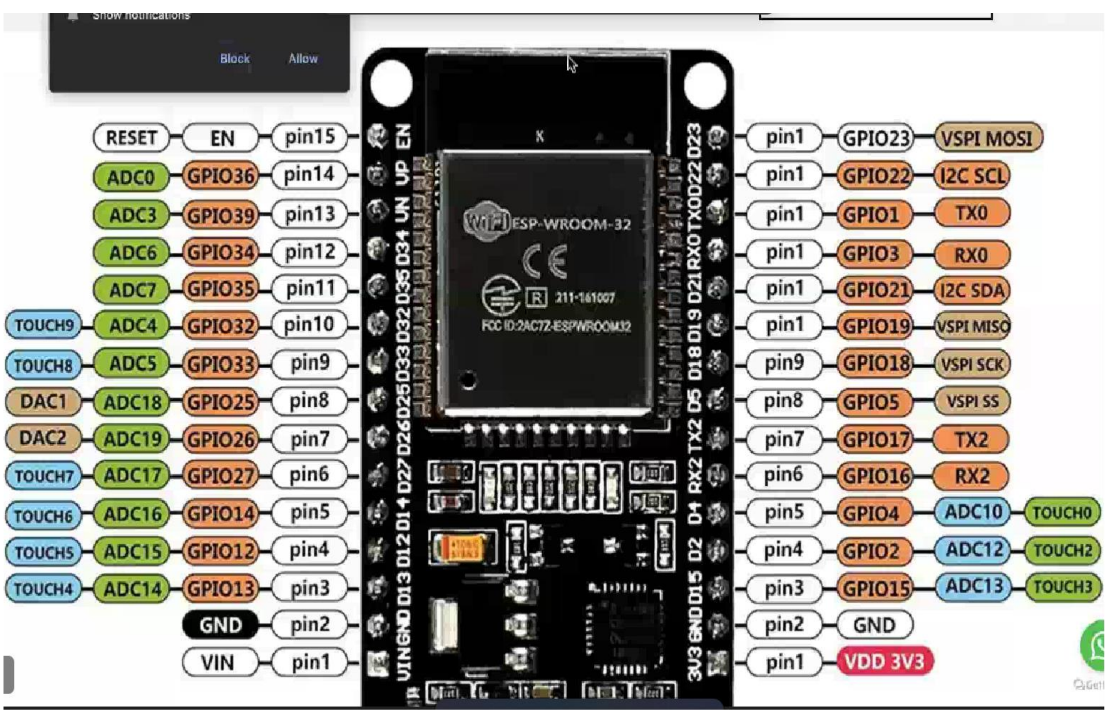
Date: 16-09-23

Objective:

The objective of this project is to establish an IoT system utilizing an ESP32 microcontroller to collect temperature, humidity, and distance data from sensors. This data is then transmitted to a Grafana dashboard via MQTT protocol. The project also involves the use of Docker for efficient container management on a Linux server. The data transmission is facilitated through a local hotspot.

Requirements:

- ESP32 microcontroller
- Temperature and humidity sensor (e.g., DHT22)
- Ultrasonic sensor (e.g., HC-SR04)
- MQTT broker (e.g., Mosquito)
- Grafana installation
- Docker installed on the Linux server.
- Linux operating system
- Local hotspot for data transmission



Procedure:

➢ Hardware:

Connect the temperature and humidity sensor (DHT22) to the ESP32 as per the datasheet instructions. Connect the ultrasonic sensor (HC-SR04) to the ESP32, typically requiring two GPIO pins for trigger and echo. Ensure proper power and ground connections for both sensors and the ESP32.

➢ Programming:

Code : For ultrasonic sensor.

```
  dht.begin();
}

void loop() {
  // Wait a few seconds between measurements.
  delay(2000);

  // Reading temperature or humidity takes about 250 milliseconds!
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  float h = dht.readHumidity();
  // Read temperature as Celsius (the default)
  float t = dht.readTemperature();
  // Read temperature as Fahrenheit (isFahrenheit = true)
  float f = dht.readTemperature(true);

  int d=ultrasonic.getDistance();
  Serial.print("Distance:");
  Serial.print(d);
  Serial.println(" cm");
  delay(100);

  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }

  // Compute heat index in Fahrenheit (the default)
```

Code: For temperature and humidity

```
if (isnan(temp_event.temperature)) {
  Serial.println("Error reading temperature!");
} else {
  Serial.print("Temperature: ");
  Serial.print(temp_event.temperature);
  Serial.println(" °C");
}

if (isnan(humidity_event.relative_humidity)) {
  Serial.println("Error reading humidity!");
} else {
  Serial.print("Humidity: ");
  Serial.print(humidity_event.relative_humidity);
  Serial.println(" %");
}
```

MQTT Broker Setup:

Install Mosquito MQTT broker on your Linux server. Configure Mosquito to listen on a specific port (e.g., 1883). Create MQTT topics for temperature, humidity, and distance data.

Setting Up an MQTT-Based IoT Data Pipeline with Docker Containers:

- **Install and Configure Mosquito MQTT Broker:**

a. Create a Docker container for the Mosquito MQTT broker.
b. Configure MQTT broker settings and ports within the container.

- **Install Grafana and Connect to MQTT Broker:**

a. Create a Docker container for Grafana.
b. Configure Grafana to connect to the MQTT broker.
c. Set up Grafana's data source to communicate with InfluxDB.

- **Create a Container for InfluxDB:**

a. Set up a Docker container for InfluxDB to store MQTT data.
b. Define InfluxDB configurations and expose necessary ports.

- **Implement MQTT Data Relay Script:**

a. Create a Python or Node.js script to receive MQTT data from ESP32.
b. Configure the script to push the data into the InfluxDB container.

- **Deploy Containers with Docker Compose:**

a. Run the containers using Docker Compose with the docker-compose up -d command.
b. Ensure all containers start without errors.

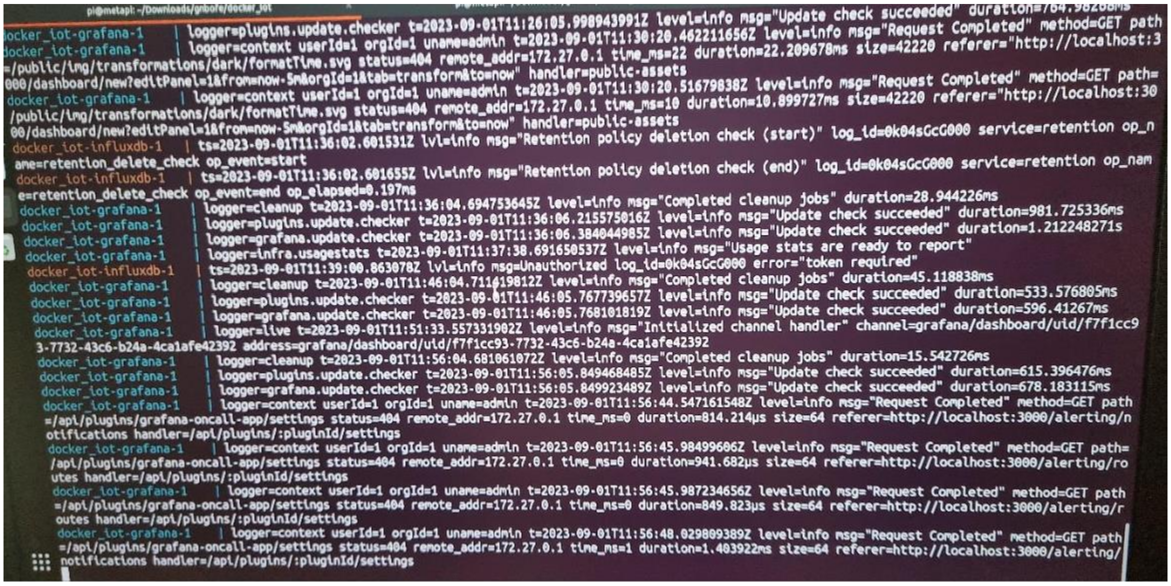
- **Verify Container Status:**

a. Confirm the status of all containers using docker-compose ps.

Testing and monitoring:

- Confirm that the ESP32 is publishing data to the MQTT broker via the local hotspot.
- Verify that data is being stored in the InfluxDB database.
- Access the Grafana dashboard through a web browser to visualize the sensor data.

➢ Visualizations:



➢ Insights:

- Docker provides an efficient and scalable solution for managing containers, ensuring smooth deployment and operation of the IoT system.
- Mosquitto MQTT broker facilitates seamless communication between the ESP32 and the Grafana dashboard.
- InfluxDB serves as a reliable database for storing and retrieving sensor data.

➢ Conclusion:

The project has successfully established an IoT system, integrating sensors with an ESP32 microcontroller. The data collected (temperature, humidity, and distance) is transmitted to a Grafana dashboard via MQTT protocol. Docker containerization enhances the system's scalability and manageability. The utilization of a local hotspot for data transmission ensures remote monitoring and visualization on a Linux server. This comprehensive setup lays a solid foundation for further IoT applications and data-driven.