# National Textile University, Faisalabad

## Department of Computer Science

| | |
|---|---|
| Name: | Huzaifa Shafique |
| Class: | BS Artificial Intelligence |
| Reg No: | 22-NTU-CS-1348 |
| Activity: | Lab 13 Report |
| Course Code: | AIE-3079 |
| Course Name: | Internet of Things Fundamentals |
| Submitted To: | Nasir Mahmood |
| Submission Date: | 20-May-2025 |

# Lab 13 Tasks

Run the Arduino-based code to publish DHT sensor data to the Mosquitto MQTT broker.

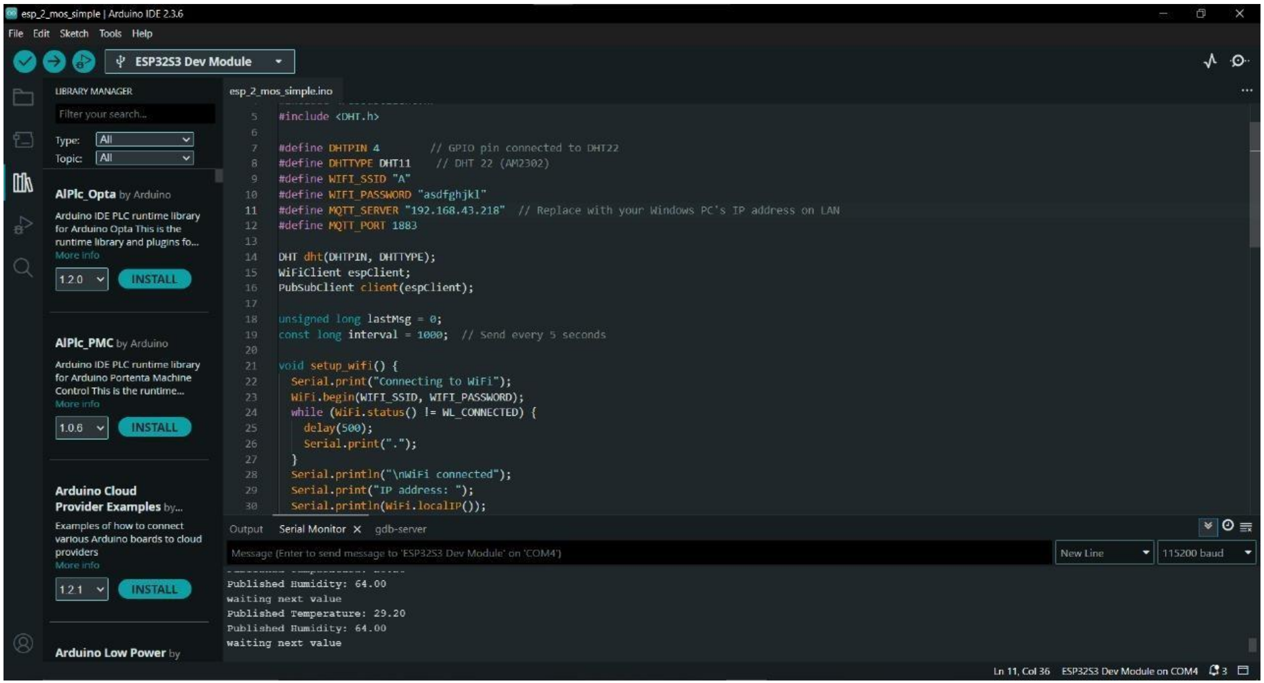
Code:

```
#include <WiFi.h>

#include <PubSubClient.h>

#include <DHT.h>


#define DHTPIN 4        // GPIO pin connected to DHT22

#define DHTTYPE DHT11    // DHT 22 (AM2302)

#define WIFI_SSID "A"

#define WIFI_PASSWORD "asdfghjkkl"

#define MQTT_SERVER "192.168.160.157"  // Replace with your Windows PC's IP address on LAN #define

MQTT_PORT 1883


DHT dht(DHTPIN, DHTTYPE);

WiFiClient espClient;

PubSubClient client(espClient);


unsigned long lastMsg = 0; const long interval = 1000;  //

Send every 5 seconds


void setup_wifi() {

  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);

Serial.print("Connecting to WiFi"); while

(WiFi.status() != WL_CONNECTED) {   delay(500);

  Serial.print(".");

}

Serial.println();

Serial.println("Connected to WiFi");
Serial.println(WiFi.localIP());  // Print IP to confirm connection


}


void reconnect() {   while
(!client.connected()) {
    Serial.print("Attempting MQTT connection...");
String clientId = "ESP32Client-";     clientId +=
String(random(0xffff), HEX);     if
(client.connect(clientId.c_str())) {
      Serial.println("connected");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());       Serial.println("
try again in 5 seconds");       delay(5000);
    }
  }
}


void setup() {   Serial.begin(115200);   dht.begin();
setup_wifi();   client.setServer(MQTT_SERVER,
MQTT_PORT);
}


void loop() {   if (!client.connected())
{     reconnect();

  }
client.loop();
unsigned long
```

```
now = millis();
if (now -
lastMsg >
interval) {
lastMsg = now;
float
temperature =
dht.readTemper
ature();     float
humidity =
dht.readHumidit
y();

  if (isnan(temperature) || isnan(humidity)) {
Serial.println("Failed to read from DHT sensor!");       return;
  }

  String tempStr = String(temperature, 2);
  String humStr = String(humidity, 2);

  client.publish("esp32/dht/temp", tempStr.c_str());     client.publish("esp32/dht/hum",
humStr.c_str());

  Serial.print("Published Temperature: ");
  Serial.println(tempStr);
  Serial.print("Published Humidity: ");     Serial.println(humStr);
  Serial.println("waiting next value");
 }
```

```
}
```

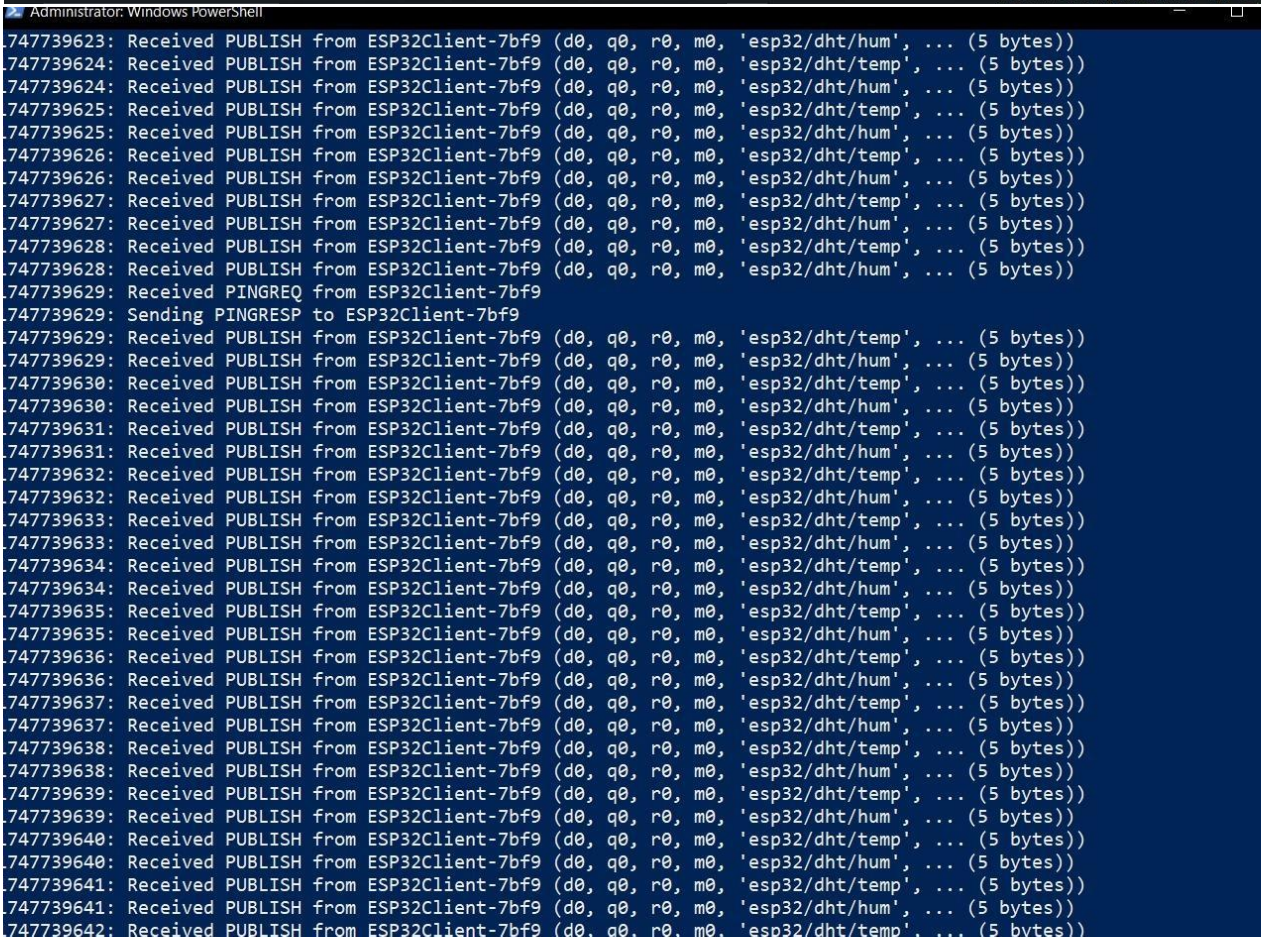
Output:



esp_2_mos_simple | Arduino IDE 2.3.6
File Edit Sketch Tools Help
ESP32S3 Dev Module
LIBRARY MANAGER
Filter your search...
Type: All
Topic: All
AIPlc_Opta by Arduino
Arduino IDE PLC runtime library for Arduino Opta This is the runtime library and plugins fo...
More info
1.2.0
INSTALL
AIPlc_PMC by Arduino
Arduino IDE PLC runtime library for Arduino Portenta Machine Control This is the runtime...
More info
1.0.6
INSTALL
Arduino Cloud Provider Examples by...
Examples of how to connect various Arduino boards to cloud providers
More info
1.2.1
INSTALL
Arduino Low Power by
esp_2_mos_simple.ino
Output
Serial Monitor
gdb-server
Message (Enter to send message to 'ESP32S3 Dev Module' on 'COM4')
New Line
115200 baud
Ln 11, Col 36
ESP32S3 Dev Module on COM4
3

```
#include <DHT.h>

#define DHTPIN 4        // GPIO pin connected to DHT22
#define DHTTYPE DHT11    // DHT 22 (AM2302)
#define WIFI_SSID "A"
#define WIFI_PASSWORD "asdfghjkl"
#define MQTT_SERVER "192.168.43.218"  // Replace with your Windows PC's IP address on LAN
#define MQTT_PORT 1883

DHT dht(DHTPIN, DHTTYPE);
WiFiClient espClient;
PubSubClient client(espClient);

unsigned long lastMsg = 0;
const long interval = 1000;  // Send every 5 seconds

void setup_wifi() {
  Serial.print("Connecting to WiFi");
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nWiFi connected");
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());
```

```
Published Humidity: 64.00
waiting next value
Published Temperature: 29.20
Published Humidity: 64.00
waiting next value
```



```
747739623: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/hum', ... (5 bytes))
747739624: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/temp', ... (5 bytes))
747739624: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/hum', ... (5 bytes))
747739625: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/temp', ... (5 bytes))
747739625: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/hum', ... (5 bytes))
747739626: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/temp', ... (5 bytes))
747739626: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/hum', ... (5 bytes))
747739627: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/temp', ... (5 bytes))
747739627: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/hum', ... (5 bytes))
747739628: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/temp', ... (5 bytes))
747739628: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/hum', ... (5 bytes))
747739629: Received PINGREQ from ESP32Client-7bf9
747739629: Sending PINGRESP to ESP32Client-7bf9
747739629: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/temp', ... (5 bytes))
747739629: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/hum', ... (5 bytes))
747739630: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/temp', ... (5 bytes))
747739630: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/hum', ... (5 bytes))
747739631: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/temp', ... (5 bytes))
747739631: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/hum', ... (5 bytes))
747739632: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/temp', ... (5 bytes))
747739632: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/hum', ... (5 bytes))
747739633: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/temp', ... (5 bytes))
747739633: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/hum', ... (5 bytes))
747739634: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/temp', ... (5 bytes))
747739634: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/hum', ... (5 bytes))
747739635: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/temp', ... (5 bytes))
747739635: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/hum', ... (5 bytes))
747739636: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/temp', ... (5 bytes))
747739636: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/hum', ... (5 bytes))
747739637: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/temp', ... (5 bytes))
747739637: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/hum', ... (5 bytes))
747739638: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/temp', ... (5 bytes))
747739638: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/hum', ... (5 bytes))
747739639: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/temp', ... (5 bytes))
747739639: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/hum', ... (5 bytes))
747739640: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/temp', ... (5 bytes))
747739640: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/hum', ... (5 bytes))
747739641: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/temp', ... (5 bytes))
747739641: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/hum', ... (5 bytes))
747739642: Received PUBLISH from ESP32Client-7bf9 (d0, q0, r0, m0, 'esp32/dht/temp', ... (5 bytes))
```

INFLUX DB:

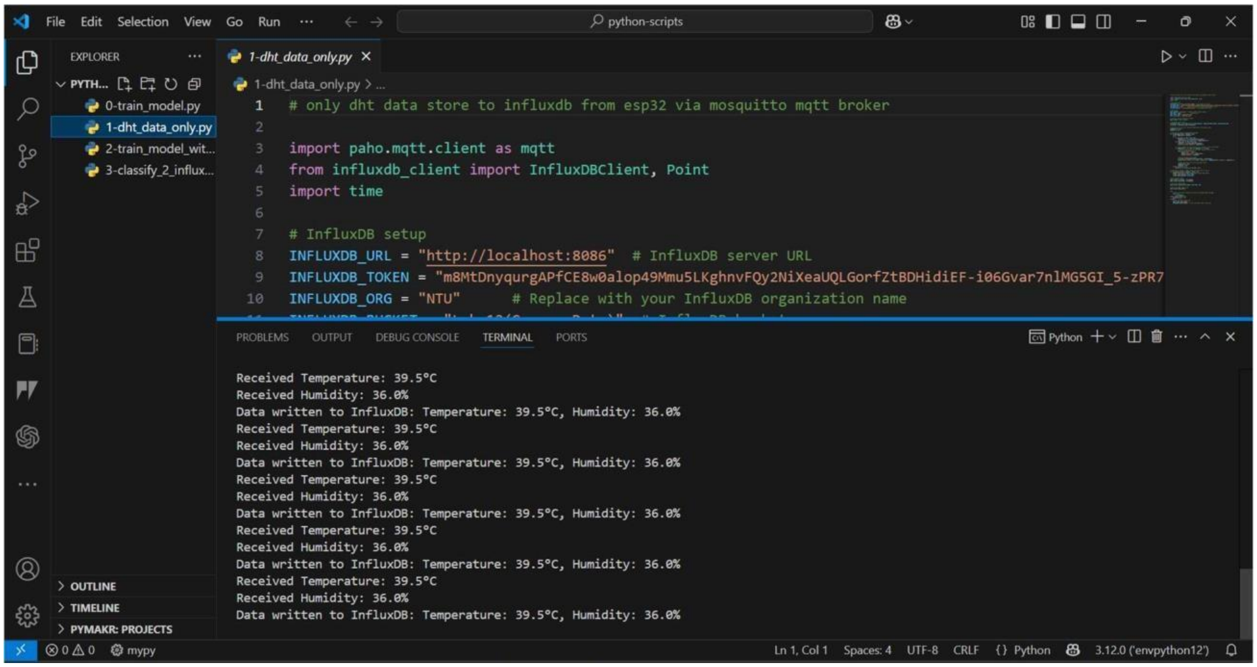
# only dht data store to influxdb from esp32 via mosquitto mqtt broker

```
import paho.mqtt.client as mqtt from influxdb_client

import InfluxDBClient, Point import time


# InfluxDB setup

INFLUXDB_URL = "http://localhost:8086"  # InfluxDB server URL

INFLUXDB_TOKEN =
"mCsHohIF0w9mrRYssecDddjOCpXwIxwuoA0FsQDiDGUUd1h15l00tcbdPbgaGErogubCy4BL5JPFXEHJ05ziWw
==# Replace with your InfluxDB token
INFLUXDB_ORG = "Huziafa"      # Replace with your InfluxDB organization name

INFLUXDB_BUCKET = "Lab_13(Sensor_Data)"  # InfluxDB bucket name


# MQTT setup

MQTT_BROKER = "localhost"  # ESP32's MQTT broker address

MQTT_PORT = 1883               # MQTT port

MQTT_TOPIC_TEMP = "esp32/dht/temp"

MQTT_TOPIC_HUM = "esp32/dht/hum"


# Create a client instance for MQTT mqtt_client =

mqtt.Client()


# InfluxDB client setup influxdb_client = InfluxDBClient(url=INFLUXDB_URL, token=INFLUXDB_TOKEN,

org=INFLUXDB_ORG) write_api = influxdb_client.write_api()


# Flag to track if we've received temperature and humidity data temperature = None

humidity = None

# Function to handle incoming MQTT messages

def on_message(client, userdata, msg):      global

temperature, humidity
```

```
    try:
        if msg.topic == MQTT_TOPIC_TEMP:
            temperature = float(msg.payload.decode())
print(f"Received Temperature: {temperature}°C")         elif msg.topic
== MQTT_TOPIC_HUM:
            humidity = float(msg.payload.decode())            print(f"Received
Humidity: {humidity}%")

        # If both temperature and humidity are received, write to InfluxDB         if
temperature is not None and humidity is not None:            # Create a data point for
InfluxDB using the Point class            point = Point("dht_data") \
                .tag("device", "esp32") \
                .field("temperature", temperature) \
                .field("humidity", humidity)

            # Write the data to InfluxDB            write_api.write(bucket=INFLUXDB_BUCKET, record=point)
print(f"Data written to InfluxDB: Temperature: {temperature}°C, Humidity: {humidity}%")

            # Reset the values to avoid duplicate writes
temperature = None            humidity = None
except Exception as e:         print(f"Error processing
message: {e}")

# Function to connect to MQTT broker and subscribe to topics def
on_connect(client, userdata, flags, rc):     print(f"Connected to MQTT
broker with result code {rc}")     client.subscribe(MQTT_TOPIC_TEMP)
client.subscribe(MQTT_TOPIC_HUM)

# Set up MQTT client mqtt_client.on_connect = on_connect
mqtt_client.on_message = on_message
```

```
# Connect to MQTT broker mqtt_client.connect(MQTT_BROKER,

MQTT_PORT, 60)


# Start the MQTT client loop mqtt_client.loop_start()

 try:

    # Keep the program running to listen for incoming MQTT messages

while True:

        time.sleep(1)  except

KeyboardInterrupt:

print("Exiting...") finally:

    # Stop the MQTT client loop     mqtt_client.loop_stop()

influxdb_client.close()  # Close InfluxDB client connection
```

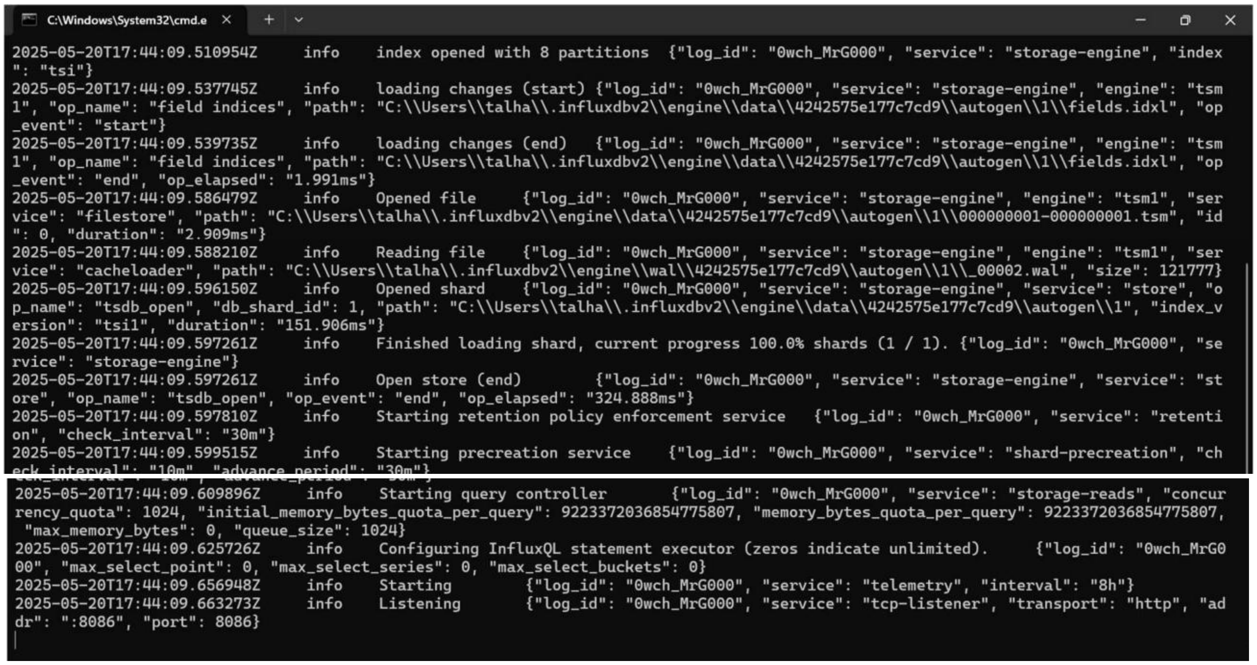
## Output:



```
# only dht data store to influxdb from esp32 via mosquitto mqtt broker

import paho.mqtt.client as mqtt
from influxdb_client import InfluxDBClient, Point
import time

# InfluxDB setup
INFLUXDB_URL = "http://localhost:8086"  # InfluxDB server URL
INFLUXDB_TOKEN = "m8MtDnyqurgAPfCE8w0alop49Mmu5LKghnvFQy2NiXeaUQLGorfZtBDHidiEF-i06Gvar7nlMG5GI_5-zPR7
INFLUXDB_ORG = "NTU"      # Replace with your InfluxDB organization name
```

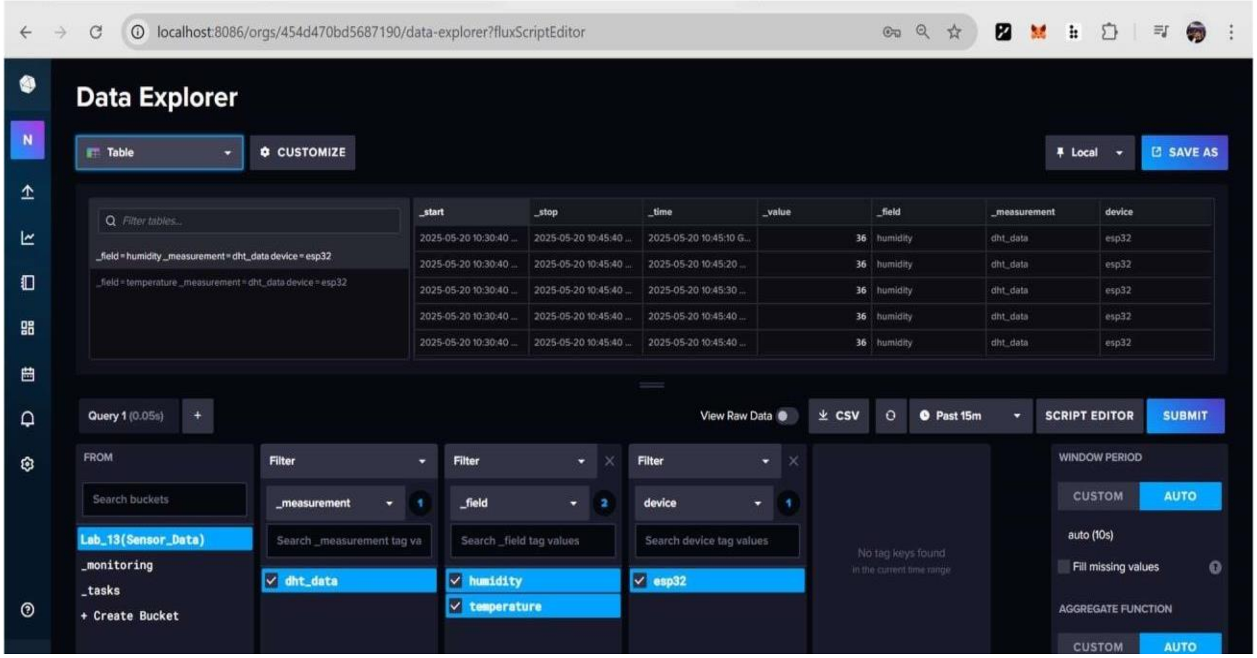
```
Received Temperature: 39.5°C
Received Humidity: 36.0%
Data written to InfluxDB: Temperature: 39.5°C, Humidity: 36.0%
Received Temperature: 39.5°C
Received Humidity: 36.0%
Data written to InfluxDB: Temperature: 39.5°C, Humidity: 36.0%
Received Temperature: 39.5°C
Received Humidity: 36.0%
Data written to InfluxDB: Temperature: 39.5°C, Humidity: 36.0%
Received Temperature: 39.5°C
Received Humidity: 36.0%
Data written to InfluxDB: Temperature: 39.5°C, Humidity: 36.0%
Received Temperature: 39.5°C
Received Humidity: 36.0%
Data written to InfluxDB: Temperature: 39.5°C, Humidity: 36.0%
```

Starting InfluxDB port



```
2025-05-20T17:44:09.510954Z     info    index opened with 8 partitions  {"log_id": "0wch_MrG000", "service": "storage-engine", "index
": "tsi"}
2025-05-20T17:44:09.537745Z     info    loading changes (start) {"log_id": "0wch_MrG000", "service": "storage-engine", "engine": "tsm
1", "op_name": "field indices", "path": "C:\\Users\\talha\\.influxdbv2\\engine\\data\\4242575e177c7cd9\\autogen\\1\\fields.idxl", "op
_event": "start"}
2025-05-20T17:44:09.539735Z     info    loading changes (end)   {"log_id": "0wch_MrG000", "service": "storage-engine", "engine": "tsm
1", "op_name": "field indices", "path": "C:\\Users\\talha\\.influxdbv2\\engine\\data\\4242575e177c7cd9\\autogen\\1\\fields.idxl", "op
_event": "end", "op_elapsed": "1.991ms"}
2025-05-20T17:44:09.586479Z     info    Opened file     {"log_id": "0wch_MrG000", "service": "storage-engine", "engine": "tsm1", "ser
vice": "filestore", "path": "C:\\Users\\talha\\.influxdbv2\\engine\\data\\4242575e177c7cd9\\autogen\\1\\000000001-000000001.tsm", "id
": 0, "duration": "2.909ms"}
2025-05-20T17:44:09.588210Z     info    Reading file    {"log_id": "0wch_MrG000", "service": "storage-engine", "engine": "tsm1", "ser
vice": "cacheloader", "path": "C:\\Users\\talha\\.influxdbv2\\engine\\wal\\4242575e177c7cd9\\autogen\\1\\_00002.wal", "size": 121777}
2025-05-20T17:44:09.596150Z     info    Opened shard    {"log_id": "0wch_MrG000", "service": "storage-engine", "service": "store", "o
p_name": "tsdb_open", "db_shard_id": 1, "path": "C:\\Users\\talha\\.influxdbv2\\engine\\data\\4242575e177c7cd9\\autogen\\1", "index_v
ersion": "tsi1", "duration": "151.906ms"}
2025-05-20T17:44:09.597261Z     info    Finished loading shard, current progress 100.0% shards (1 / 1). {"log_id": "0wch_MrG000", "se
rvice": "storage-engine"}
2025-05-20T17:44:09.597261Z     info    Open store (end)        {"log_id": "0wch_MrG000", "service": "storage-engine", "service": "st
ore", "op_name": "tsdb_open", "op_event": "end", "op_elapsed": "324.888ms"}
2025-05-20T17:44:09.597810Z     info    Starting retention policy enforcement service   {"log_id": "0wch_MrG000", "service": "retenti
on", "check_interval": "30m"}
2025-05-20T17:44:09.599515Z     info    Starting precreation service    {"log_id": "0wch_MrG000", "service": "shard-precreation", "ch
eck_interval": "10m", "advance_period": "30m"}
2025-05-20T17:44:09.609896Z     info    Starting query controller       {"log_id": "0wch_MrG000", "service": "storage-reads", "concur
rency_quota": 1024, "initial_memory_bytes_quota_per_query": 9223372036854775807, "memory_bytes_quota_per_query": 9223372036854775807,
 "max_memory_bytes": 0, "queue_size": 1024}
2025-05-20T17:44:09.625726Z     info    Configuring InfluxQL statement executor (zeros indicate unlimited).     {"log_id": "0wch_MrG0
00", "max_select_point": 0, "max_select_series": 0, "max_select_buckets": 0}
2025-05-20T17:44:09.656948Z     info    Starting        {"log_id": "0wch_MrG000", "service": "telemetry", "interval": "8h"}
2025-05-20T17:44:09.663273Z     info    Listening       {"log_id": "0wch_MrG000", "service": "tcp-listener", "transport": "http", "ad
dr": ":8086", "port": 8086}
```

InfluxDB Dashboard



localhost:8086/orgs/454d470bd5687190/data-explorer?fluxScriptEditor

Data Explorer

_field = humidity _measurement = dht_data device = esp32

_field = temperature _measurement = dht_data device = esp32

| _start | _stop | _time | _value | _field | _measurement | device |
|---|---|---|---|---|---|---|
| 2025-05-20 10:30:40 ... | 2025-05-20 10:45:40 ... | 2025-05-20 10:45:10 G... | 36 | humidity | dht_data | esp32 |
| 2025-05-20 10:30:40 ... | 2025-05-20 10:45:40 ... | 2025-05-20 10:45:20 ... | 36 | humidity | dht_data | esp32 |
| 2025-05-20 10:30:40 ... | 2025-05-20 10:45:40 ... | 2025-05-20 10:45:30 ... | 36 | humidity | dht_data | esp32 |
| 2025-05-20 10:30:40 ... | 2025-05-20 10:45:40 ... | 2025-05-20 10:45:40 ... | 36 | humidity | dht_data | esp32 |
| 2025-05-20 10:30:40 ... | 2025-05-20 10:45:40 ... | 2025-05-20 10:45:40 ... | 36 | humidity | dht_data | esp32 |

FROM
Search buckets
Lab_13(Sensor_Data)
_monitoring
_tasks
+ Create Bucket

Filter: _measurement 1
Search _measurement tag va
☑ dht_data

Filter: _field 2
Search _field tag values
☑ humidity
☑ temperature

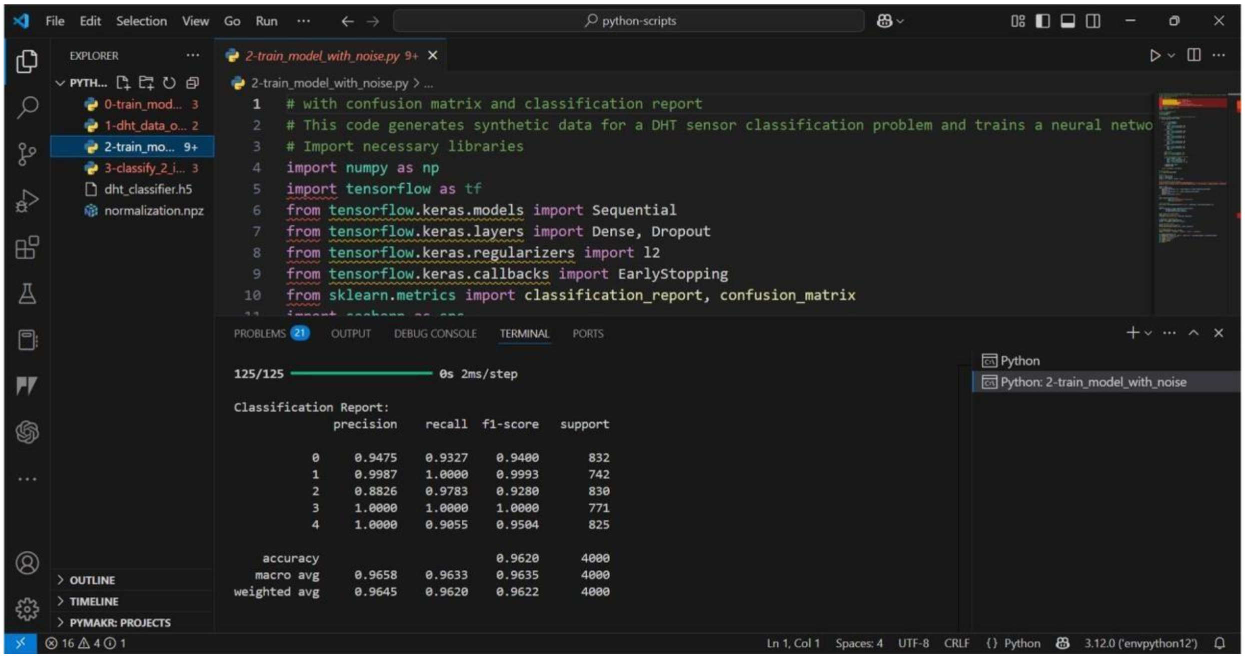
Filter: device 1
Search device tag values
☑ esp32

No tag keys found
in the current time range

WINDOW PERIOD
CUSTOM AUTO
auto (10s)
□ Fill missing values

AGGREGATE FUNCTION
CUSTOM AUTO

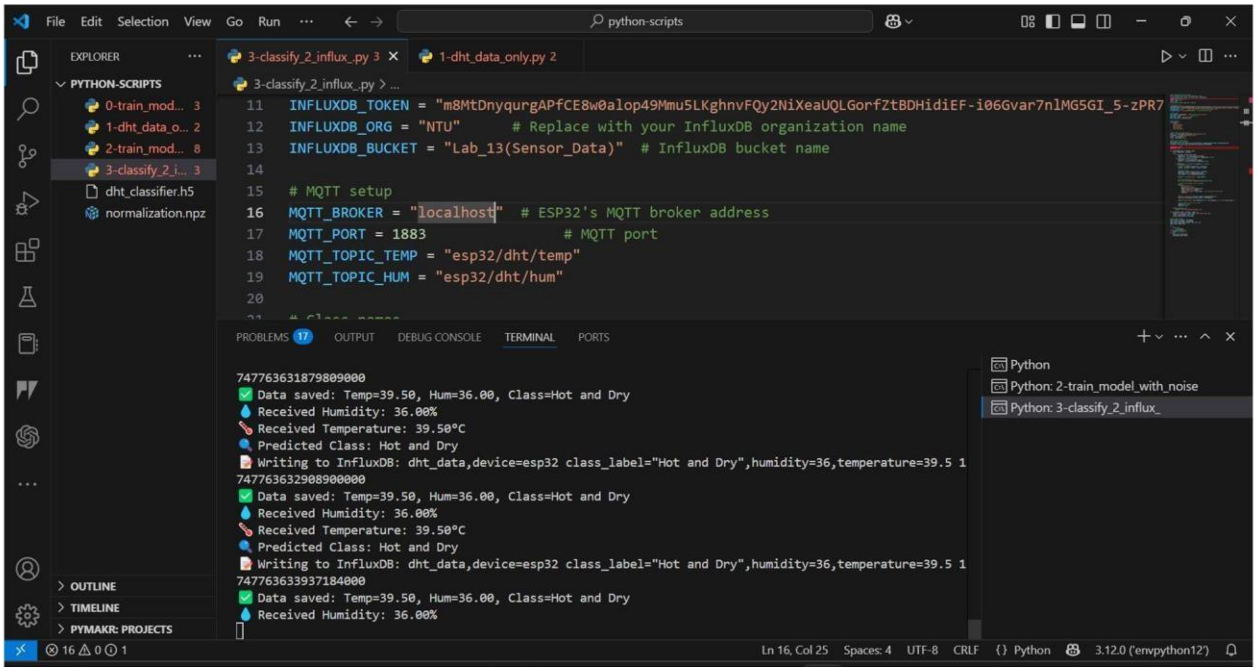
Run 2-train_model_with_noise.py and record the confusion matrix and classification report.



```python
# with confusion matrix and classification report
# This code generates synthetic data for a DHT sensor classification problem and trains a neural netwo
# Import necessary libraries
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras.regularizers import l2
from tensorflow.keras.callbacks import EarlyStopping
from sklearn.metrics import classification_report, confusion_matrix
```

```
125/125 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step

Classification Report:
              precision    recall  f1-score   support

           0     0.9475    0.9327    0.9400       832
           1     0.9987    1.0000    0.9993       742
           2     0.8826    0.9783    0.9280       830
           3     1.0000    1.0000    1.0000       771
           4     1.0000    0.9055    0.9504       825

    accuracy                         0.9620      4000
   macro avg     0.9658    0.9633    0.9635      4000
weighted avg     0.9645    0.9620    0.9622      4000
```

Execute 3-classify_2_influx.py and verify InfluxDB data for temperature, humidity, and classification results.



```python
INFLUXDB_TOKEN = "m8MtDnyqurgAPfCE8w0alop49Mmu5LKghnvFQy2NiXeaUQLGorfZtBDHidiEF-i06Gvar7nlMG5GI_5-zPR7
INFLUXDB_ORG = "NTU"      # Replace with your InfluxDB organization name
INFLUXDB_BUCKET = "Lab_13(Sensor_Data)"  # InfluxDB bucket name

# MQTT setup
MQTT_BROKER = "localhost"  # ESP32's MQTT broker address
MQTT_PORT = 1883                # MQTT port
MQTT_TOPIC_TEMP = "esp32/dht/temp"
MQTT_TOPIC_HUM = "esp32/dht/hum"

```

```
747763631879809000
✅ Data saved: Temp=39.50, Hum=36.00, Class=Hot and Dry
💧 Received Humidity: 36.00%
🌡️ Received Temperature: 39.50°C
🔍 Predicted Class: Hot and Dry
📝 Writing to InfluxDB: dht_data,device=esp32 class_label="Hot and Dry",humidity=36,temperature=39.5 1
747763632908900000
✅ Data saved: Temp=39.50, Hum=36.00, Class=Hot and Dry
💧 Received Humidity: 36.00%
🌡️ Received Temperature: 39.50°C
🔍 Predicted Class: Hot and Dry
📝 Writing to InfluxDB: dht_data,device=esp32 class_label="Hot and Dry",humidity=36,temperature=39.5 1
747763633937184000
✅ Data saved: Temp=39.50, Hum=36.00, Class=Hot and Dry
💧 Received Humidity: 36.00%
```