

Guía de Clase – Internet de las Cosas (IoT)

Ejercicio: Envío de datos MQTT y almacenamiento en base de datos MySQL (XAMPP)

Broker utilizado: broker.hivemq.com

Objetivo

Configurar un sistema IoT donde un NodeMCU (ESP8266) envía un dato mediante MQTT a un broker y luego este dato se almacena automáticamente en una base de datos MySQL usando un servidor local XAMPP.

Materiales necesarios

- NodeMCU ESP8266
- Conexión WiFi
- PC con XAMPP (Apache y MySQL activos)
- Arduino IDE con librerías ESP8266WiFi y PubSubClient
- Broker MQTT: broker.hivemq.com
- Python con librerías paho-mqtt y requests instaladas

1 Crear la base de datos MySQL

Abrir phpMyAdmin desde XAMPP y ejecutar el siguiente código SQL:

```
CREATE DATABASE iot_db;  
  
USE iot_db;  
  
CREATE TABLE datos_iot (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  valor VARCHAR(50),  
  fecha TIMESTAMP DEFAULT CURRENT_TIMESTAMP  
);
```

2 Crear el archivo PHP receptor

Guardar en la carpeta C:\xampp\htdocs\iot_mqtt con el nombre guardar_dato.php:

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "iot_db";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect_error) {
    die("Error de conexión: " . $conn->connect_error);
}

if (isset($_GET['valor'])) {
    $valor = $_GET['valor'];
    $sql = "INSERT INTO datos_iot (valor) VALUES ('$valor')";
    if ($conn->query($sql) === TRUE) {
        echo "Dato guardado correctamente";
    } else {
        echo "Error al guardar: " . $conn->error;
    }
} else {
    echo "No se recibió ningún dato";
}
$conn->close();
?>
```

3 Probar el archivo PHP

Abrir un navegador y probar la URL:

http://localhost/iot_mqtt/guardar_dato.php?valor=25.4

Debe mostrar: "Dato guardado correctamente"

4 Crear el script MQTT receptor en Python

Guardar el archivo mqtt_to_mysql.py con el siguiente contenido:

```
import paho.mqtt.client as mqtt
import requests
```

```
BROKER = "broker.hivemq.com"
TOPIC = "claseIoT/datos"
URL = "http://localhost/iot_mqtt/guardar_dato.php"
```

```
def on_message(client, userdata, msg):
    valor = msg.payload.decode()
    print(f"Dato recibido: {valor}")
    try:
        r = requests.get(URL, params={'valor': valor})
        print("Servidor:", r.text)
    except Exception as e:
        print("Error enviando al servidor:", e)
```

```
client = mqtt.Client()
client.on_message = on_message
client.connect(BROKER, 1883, 60)
client.subscribe(TOPIC)
```

```
print(f"Escuchando t pico: {TOPIC}")
client.loop_forever()
```

5 Subir el c digo al NodeMCU

Usar el siguiente programa en Arduino IDE:

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

const char* ssid = "TuRedWiFi";
const char* password = "TuClaveWiFi";

const char* mqtt_server = "broker.hivemq.com";
const int mqtt_port = 1883;
const char* topic = "claseIoT/datos";

WiFiClient espClient;
PubSubClient client(espClient);

void setup_wifi() {
    delay(10);
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
```

```

    }
}

void reconnect() {
    while (!client.connected()) {
        if (!client.connect("NodeMCU_Client")) {
            delay(5000);
        }
    }
}

void setup() {
    Serial.begin(115200);
    setup_wifi();
    client.setServer(mqtt_server, mqtt_port);
}

void loop() {
    if (!client.connected()) {
        reconnect();
    }
    client.loop();
    float temperatura = random(20, 30);
    String mensaje = "Temperatura: " + String(temperatura) + " °C";
    client.publish(topic, mensaje.c_str());
    delay(5000);
}

```

6 Verificar el funcionamiento

1. Ejecutar mqtt_to_mysql.py en el PC.
2. Subir el código al NodeMCU y abrir el monitor serial.
3. Verificar en phpMyAdmin los datos guardados en la tabla datos_iot.

Evidencias del estudiante

Agregar capturas de:

- Monitor Serial mostrando envío de datos MQTT.
- Consola de Python mostrando recepción de mensajes.
- phpMyAdmin con los registros insertados.