EE-432L Computer Networks CEP

Spring 2023

Submitted to:

Dr. Naveed Nawaz

Submitted by:

Ali Raza	2021-EE-25
Sheheryar	2021-EE-15

Department of Electrical Engineering

University of Engineering and Technology Lahore

Problem Statement:

In this assignment, you are required to design and develop a smart plug capable of controlling an electrical appliance by transmitting data using MySQL database with the ***Message Queuing Telemetry Transport (MQTT)*** protocol.

Solution:

1- Selection of WIFI chipset module:

We are using the **ESP32-WROOM-32D** chipset for the smart plug for the following reasons:

Low cost: It is easily accessible at the low price of round about **1500pkr.**

Low-power: the ESP32 consumes very little power compared with other microcontrollers 3.3 to 5 Volts.

Wi-Fi capabilities: the ESP32 can easily connect to a Wi-Fi network to connect to the internet (station mode), or create its own Wi-Fi wireless network (access point mode) so we can manage our smart plug through an internet page.

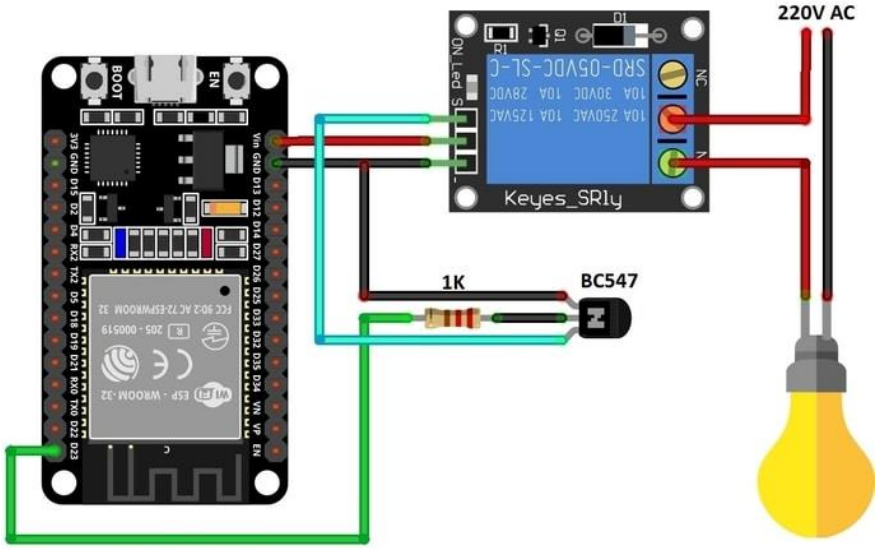
Compatible with the Arduino: It can be programmed in Arduino style in the Arduino IDE.

2- AC-DC converter:

Hi link5V is used to convert the coming 220 VAC into 5 VDC. This 5VDC is provided as a power supply to the ESP32-WROOM32D Wi-Fi chipset. The range of input voltages of the chipset lies between 3.3 to 5 volts.

3- Circuit design:

The following diagram represents the original circuit design. The chipset is provided 5VDC input through the AC-DC converter. Code is embedded into the chip and general-purpose input output pins are used to establish a connection between the chip and the lamp. An active high relay is used between the chipset and the lamp to control the trigger. When an active high signal is provided to the relay it connects the common pin to the normally closed pin and the lamp turns on and when an active low signal is provided to the relay it connects the common signal to the normally closed pin so the lamp turns off.



4- Selection of software:

Arduino IDE is used for the coding of ESP32-WROOM-32D.

VsCode is used for server programming in Python.

5- Webpage:

We have used Hypertext Markup Language (HTML) for developing the webpage. Bootstrap and CSS for styling the webpage. There is a button on the webpage when changes its state is considered as an event then a JavaScript function is called which sends a request with the button's current state to the backend. In the backend, the state is decoded and the corresponding message is published on the topic via MQTT broker.

Server/Publisher Code

```
from http.server import SimpleHTTPRequestHandler
from socketserver import TCPServer
import paho.mqtt.client as mqtt

client = mqtt.Client()
client.connect("broker.hivemq.com", 1883, 60)


class Handler(SimpleHTTPRequestHandler):
    def do_GET(self):
       if self.path == '/':
           self.send_response(200)
           self.send_header('Content-type', 'text/html')
           self.end_headers()

           html = """
                <!DOCTYPE html>
                <html lang="en">
                <head>
                    <meta charset="UTF-8">
                    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
                    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
4bw+/aepP/YC94hEpVNVgiZdgIC5+VKNBQNGCHeKRQN+PtmoHDEXuppvnDJzQIu9"
crossorigin="anonymous">
                <title>Control Your IoT Device</title>
                </head>

                <body style="background-color: #f2f2f2; background-image:
url('simpl.jpg'); background-size: cover; font-family: Arial, sans-serif;
justify-content: center; align-items: center; height: 100vh; margin: 0; padding-
top:150%;">
                        <div class="container" style="color: #333;">
                        <h1 style="color: #1a73e8; font-size: 3.5rem; font-
weight: bold; margin-bottom: 10px;">Control Your Device from Anywhere in the
World</h1>
                        <p style="font-size: 1.5rem; margin-top: 0;">Take charge
of your IoT device with ease.
                        </p>
```

```
                        <div class="form-check form-switch mx-auto"
style="margin-top: 2rem; align-items: center;">
                        <input id="btn" class="form-check-input" type="checkbox"
style="width: 8rem; height: 3rem;>
                        <label class="form-check-label" for="btn" style="font-
size: 1.2rem; font-weight: bold; position: absolute; top: 50%; transform:
translateY(-50%); left: 110%; transition: left 0.3s ease;"></label>
                        </div>

                    <script>
                        function clicked(bid) {
                            var btn = document.getElementById(bid);
                            var xml_req = new XMLHttpRequest();
                            xml_req.open("GET", "/" + btn.checked, true);
                            xml_req.send();
                        }
                    </script>
                    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/js/bootstrap.bundle.min.js
" integrity="sha384-
HwwvtgBNo3bZJJLYd8oVXjrBZt8cqVSpeBNS5n7C8IVInixGAoxmnlMuBnhbgrkm"
crossorigin="anonymous"></script>
                </body>
                </html>
            """
            self.wfile.write(html.encode())
        else:
            btn_id = self.path[1:]
            if btn_id in ['true', 'false']:
                message = '0'
                if btn_id == 'true':
                    message = '1'

                topic = "Bilal_Yousaf"
                client.publish(topic, message)


if __name__ == "__main__":
    with TCPServer(("localhost", 8000), Handler) as httpd:
        httpd.serve_forever()
```

Subscriber Code:

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

const char *broker = "broker.hivemq.com";
const char *topic = "Bilal_Yousaf";
const int port = 1883;

const char *ssid = "Samsung Galaxy J6";
const char *password = "Waneeza@051";

WiFiClient espClient;
PubSubClient client(espClient);

void setup() {
    Serial.begin(115200);
    pinMode(17, OUTPUT);

    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(200);
    }

    client.setServer(broker, port);
    client.setCallback(incoming);
    while (!client.connected()) {
        String client_id = "ESP32-";
        client_id += String(WiFi.macAddress());
        if (client.connect(client_id.c_str()))
            client.subscribe(topic);
        else
            delay(1000);
    }
}

void incoming(char *published_topic, byte *published_message, unsigned int
length) {
    int message;
    for (int i = 0; i < length; i++) {
        message = ((char) published_message[i]) - '0';
    }
    if (message)
        digitalWrite(17, HIGH);
```

```
    else
        digitalWrite(17, LOW);
}


void loop() {
   client.loop();
}
```