Lampa Interactiva

Introducere

Scopul proiectului este de a realiza o lampa interactiva care ar reactiona la comenzi de la un server la care aceasta ar putea sa se conecteze prin intermediul unui modul Wi-Fi.

Motivatia principala este posibilitea de a extinde functionalitatea lampii prin intermediul serverului.

Utilitatea principala a lampii este de a notifica in prealabil despre posibila ploiae fara a fi nevoie de a verifica vremea pe telefon sau pe internet. Solutia fiind mai eficienta deoarece este un dispozitiv necesar la birou si este foarte usor de observat datorita culorilor.

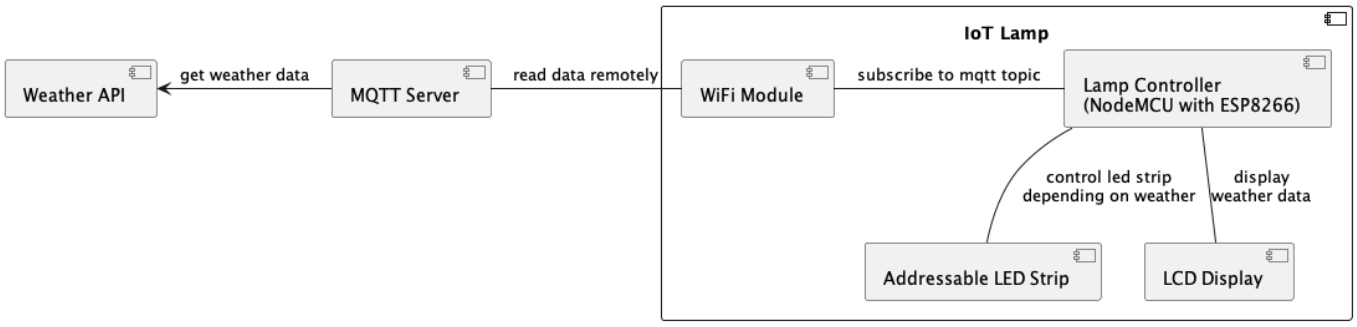
Descriere generală

Proiectul este compus din 2 parti principale: partea de hardware si partea de software.

Partea de software este reprezentata de un server care va fi responsabil de preluarea datelor de la un server meteo si de a publica mesaje pe un broker MQTT. Aceste mesaje vor fi preluate de catre partea de hardware care va fi responsabila de a controla lampa.

Lampa se va conecta la brokerul MQTT si va astepta mesaje de la server. In functie de mesajul primit lampa va schimba culoarea LED-urilor.

De asemenea as vrea sa implementez si afisarea mesajelor pe un display LCD.



Hardware Design

Aici puneţi tot ce ţine de hardware design:

- listă de piese
- scheme electrice (se pot lua şi de pe Internet şi din datasheet-uri, e.g. http://www.captain.at/electronic-atmega16-mmc-schematic.png)
- diagrame de semnal
- rezultatele simulării

Lista piese:

- NodeMCU cu ESP8266
- Banda LED adresabila WS2812B
- Display LCD 16×2
- Aliementator 5V 2A

Software Design

Mediu de dezvoltare: Arduino IDE + VSCode Librarii: ESP8266WiFi, PubSubClient, FastLED, LiquidCrystal_I2C, Wire

Structuri folosit:

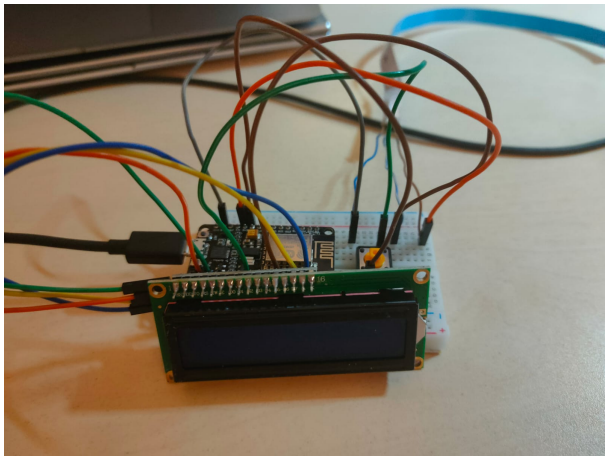
```
// enum pentru modurile de efecte
enum EffectMode { GREEN_STATIC, RED_STATIC, BLUE_STATIC, RAINBOW };

// structura pentru efectul de picaturi de ploaie
struct Raindrop {
  int position;
  int intensity;
};


// coada de mesaje
String messageQueue[MAX_QUEUE_SIZE];

void enqueueMessage(String message) {
  ...
}

String dequeueMessage() {
  ...
}
```

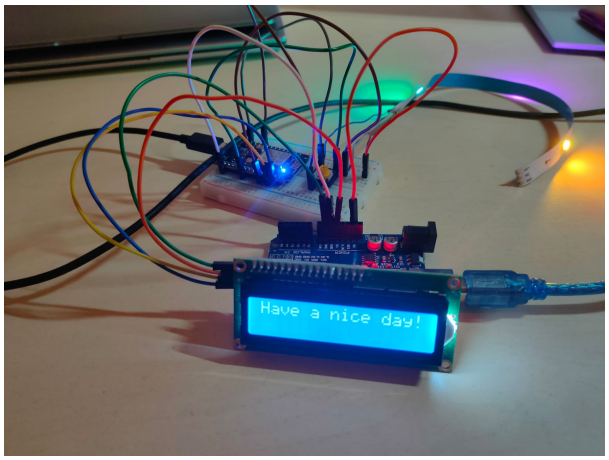
Metode folosite:

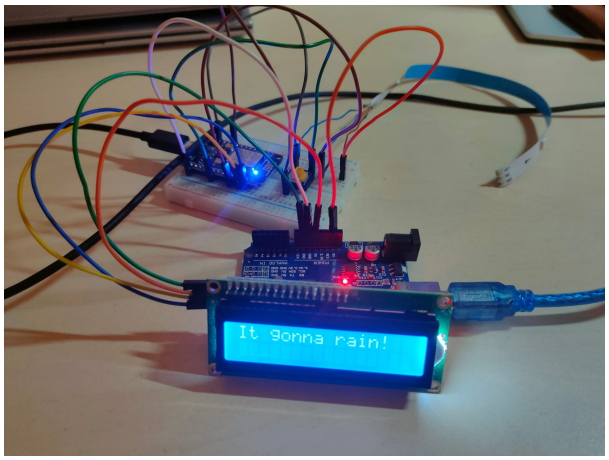
```
// implementarea efectelor
void normalModeEffects() {
```

```
  FastLED.clear();

  if (currentEffectMode == GREEN_STATIC) {
    fill_solid(leds, NUM_LEDS, CRGB(0, 255, 0)); // Green
  }

  if (currentEffectMode == RED_STATIC) {
    fill_solid(leds, NUM_LEDS, CRGB(255, 0, 0)); // Red
  }

  if (currentEffectMode == BLUE_STATIC) {
    fill_solid(leds, NUM_LEDS, CRGB(0, 0, 255)); // Blue
  }

  if (currentEffectMode == RAINBOW) {
    static uint8_t hue = 0;
    fill_rainbow(leds, NUM_LEDS, hue, 255 / NUM_LEDS);
    hue+=3;
  }

  FastLED.show();
}


// bucla principala
void loop() {
  displayNextMessage();

  if (isRaining()) {
    rainEffect();
  } else {
    normalModeEffects();
  }

  if (!client.connected()) {
    reconnect();
  }

  client.loop();
}
```

Rezultate Obţinute







Concluzii

1. Pentru a alimenta banda LED si display-ul LCD am folosit iesirea de 5V de la un Arduino UNO.
2. Pentru server MQTT am folosit Mosquitto.
3. Pentru a trimite mesaje catre server am folosit un script bash care apeleaza comanda mosquitto_pub.
4. Placa NodeMCU cu ESP8266 se conecteaza la brokerul MQTT si asteapta mesaje.
5. In functie de mesajul primit lampa schimba culoarea LED-urilor.
6. Mesajele sunt afisate pe display-ul LCD folosind o coada de mesaje.

Download

Codul sursa: code.txt

Script pentru trimitere mesaje:

```
#!/bin/bash
mosquitto_pub -h 91.121.93.94 -p 1883 -t "weather-project/data" -m "r"
```

Jurnal

Puteți avea și o secțiune de jurnal în care să poată urmări asistentul de proiect progresul proiectului.

Bibliografie/Resurse

* Proiect similar: https://alexgyver.ru/gyverlamp/

Export to PDF