

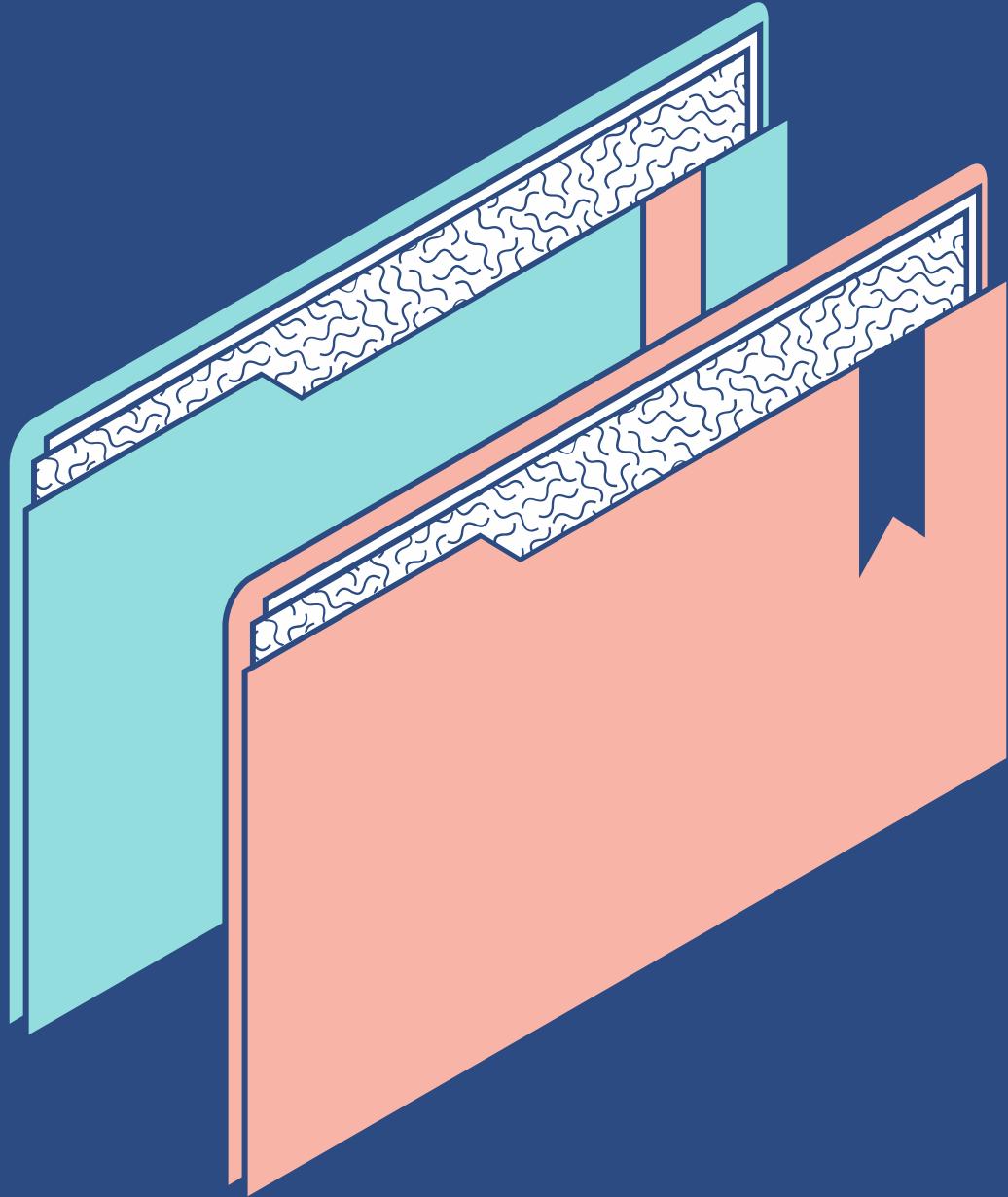


BÁO CÁO GIỮA KÌ MÔN HỌC: IOT VÀ
ỨNG DỤNG

Dải đèn LED trang trí phản ứng âm nhạc

Nhóm bài tập lớn: 12

Nội dung



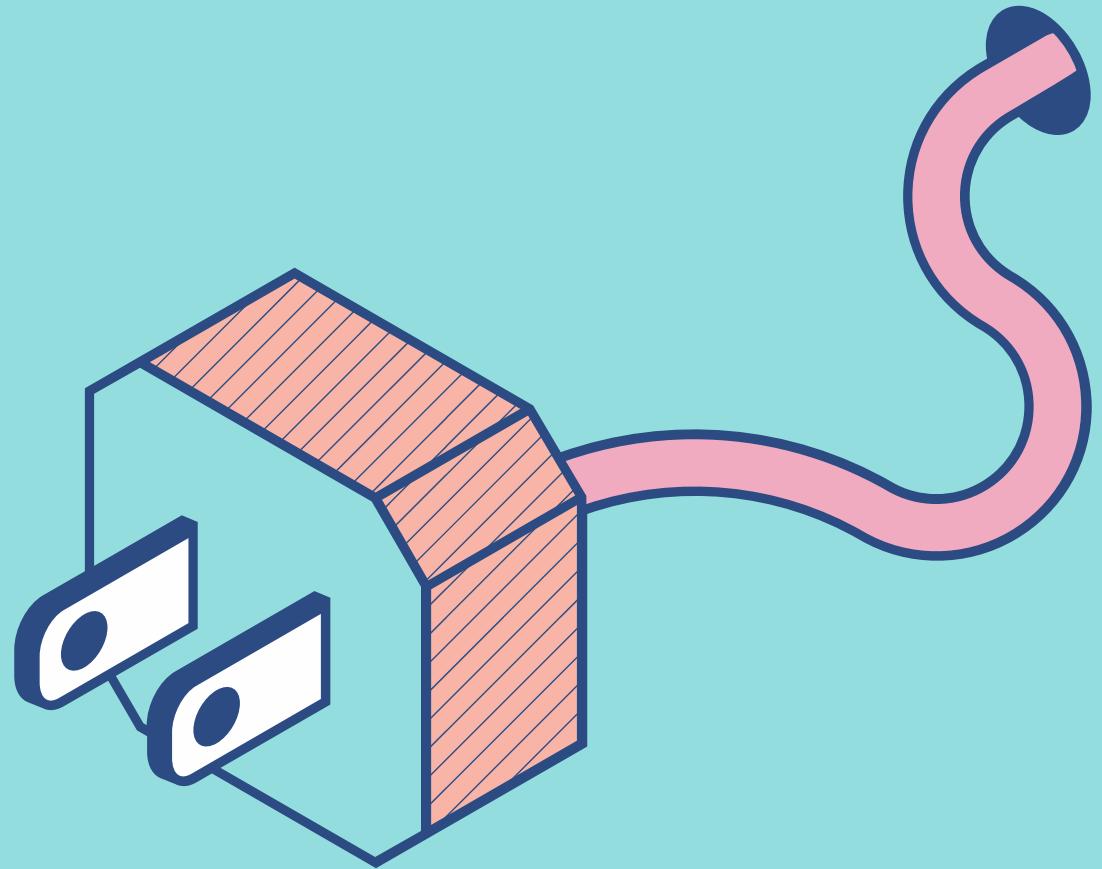
- I. GIỚI THIỆU ĐỀ TÀI
- II. PHÂN TÍCH YÊU CẦU
- III. TỔNG QUAN HỆ THỐNG & PHẦN CỨNG
- IV. THIẾT KẾ PHẦN MỀM & CÔNG NGHỆ

I. GIỚI THIỆU ĐỀ TÀI

1.1 Mục tiêu dự án

Dự án “Dải đèn LED thông minh phản ứng âm nhạc” nhằm tạo ra một hệ thống có khả năng:

- Tự động thay đổi màu sắc, độ sáng và hiệu ứng của dải đèn LED dựa trên tín hiệu âm thanh.
- Phản ứng theo nhịp điệu (beat) của nhạc để tạo ra trải nghiệm ánh sáng sinh động.
- Có thể điều khiển thủ công qua web hoặc app di động, ví dụ bật/tắt đèn, chọn hiệu ứng, thay đổi chế độ nhạc.



I. GIỚI THIỆU ĐỀ TÀI

1.2 Phạm vi hệ thống

- Thiết bị phần cứng: dải LED, vi điều khiển (ESP32), cảm biến âm thanh.
- Phần mềm nhúng: chạy trên ESP32 để nhận tín hiệu âm thanh, xử lý, điều khiển LED.
- Giao diện điều khiển: web app và mobile app giúp người dùng chọn chế độ hoạt động, màu sắc, hiệu ứng.
- Kết nối: sử dụng WiFi để giao tiếp giữa thiết bị và web/app.

I. GIỚI THIỆU ĐỀ TÀI

1.2 Mô tả tổng quan

Chế độ 1: Sử dụng cảm biến âm thanh

Bước 1: Thu âm thanh từ môi trường

Bước 2: Xử lý tín hiệu âm thanh trên ESP32

Bước 3: Xử lý logic hiệu ứng LED

Bước 4: Hiển thị hiệu ứng trên dải LED

Bước 5: Điều khiển & giám sát qua web/app

Bước 6: Cập nhật trạng thái và lưu cấu hình



I. GIỚI THIỆU ĐỀ TÀI

1.2 Mô tả tổng quan

Chế độ 2: Nhận dữ liệu âm thanh, phân tích nhạc từ thiết bị điều khiển

Bước 1: Phát hiện âm thanh và phân tích tại thiết bị điều khiển

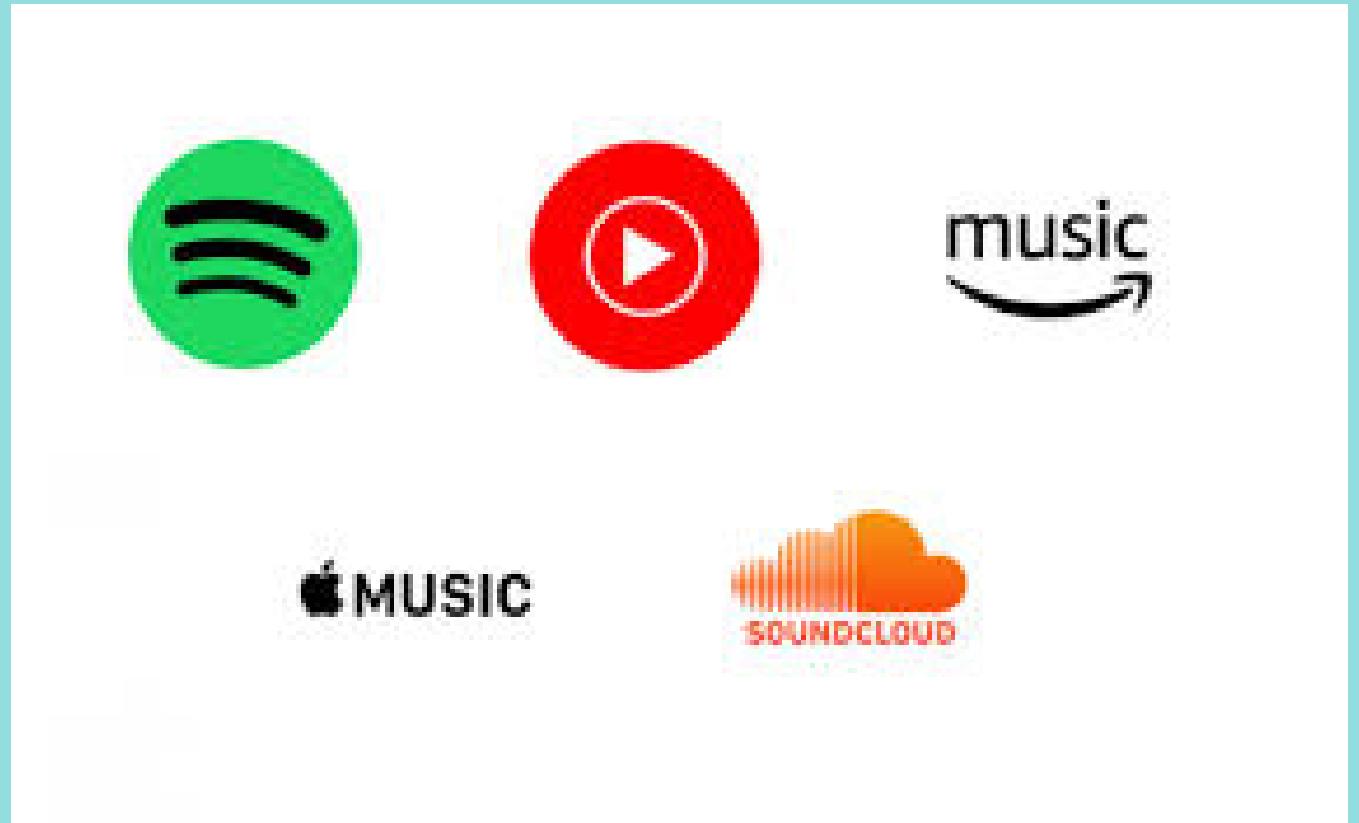
Bước 2: Gửi dữ liệu điều khiển đến ESP32

Bước 3: Xử lý dữ liệu tại ESP32

Bước 4: Hiển thị hiệu ứng trên dải LED

Bước 5: Điều khiển & giám sát qua web/app

Bước 6: Cập nhật trạng thái và lưu cấu hình



II. PHÂN TÍCH YÊU CẦU

2.1 Yêu cầu chức năng (Functional Requirements)

- Hệ thống thu nhận tín hiệu âm thanh
- Điều khiển dải đèn LED
- Kết nối Wi-Fi
- Điều khiển từ xa qua Web
- Hiển thị trạng thái hệ thống
- Giao tiếp giữa thiết bị và Web
- Cập nhật firmware từ xa (OTA)
- Lưu cấu hình người dùng

II. PHÂN TÍCH YÊU CẦU

2.2 Yêu cầu phi chức năng (Non-functional Requirements)

- Thời gian phản hồi ≤ 1 giây
- Xử lý tín hiệu âm thanh real-time
- Tự động kết nối lại khi mất mạng
- Dữ liệu không bị mất khi lỗi tạm thời
- Xác thực khi truy cập Web
- OTA an toàn
- Hỗ trợ nhiều thiết bị cùng lúc
- Giao diện thân thiện, dễ dùng
- Hệ thống dễ cập nhật
- Hỗ trợ nhiều trình duyệt



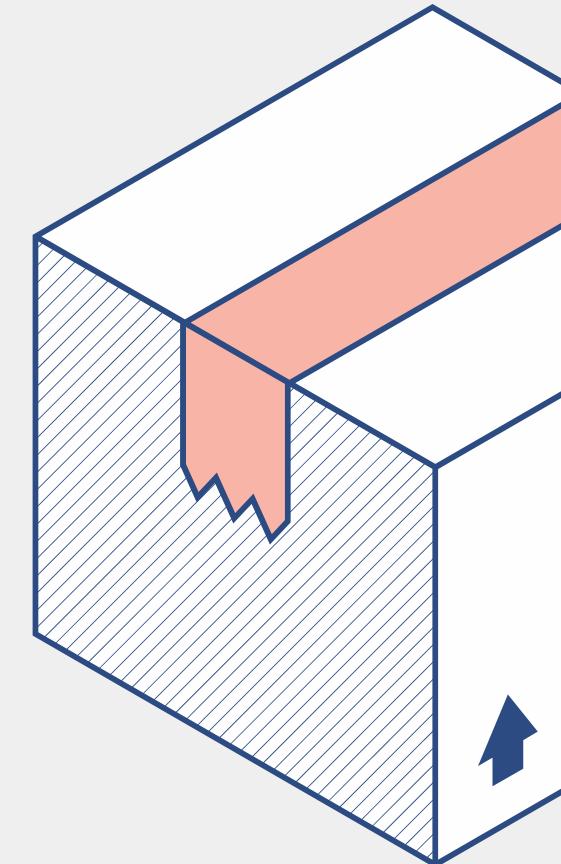
TỔNG QUAN HỆ THỐNG & PHẦN CỨNG

THÀNH PHẦN PHẦN CỨNG
SƠ ĐỒ KẾT NỐI
GIAO THỨC MẠNG VÀ TRUYỀN THÔNG
TỔNG QUAN HỆ THỐNG

III. THIẾT KẾ HỆ THỐNG VÀ PHẦN CỨNG

3.1 Thành phần phần cứng

Tên linh kiện	Mã	Chức năng
Ví điều khiển	ESP32 CP2102 38P typeC Wifi	Điều khiển toàn bộ hệ thống, đọc mic, xuất tín hiệu điều khiển LED.
Microphone module	MAX4466	Thu âm, khuếch đại tín hiệu âm thanh đưa vào ESP32.
Dải LED RGB	WS2812B DC5V PCB IP67 1m, 74leds /m	LED RGB điều khiển từng bóng, hiển thị hiệu ứng theo lập trình.
Nguồn điện 5V	Nguồn led 5V 8A	Cấp nguồn ổn định cho dải LED và ESP32.
Breadboard	MB-102 830 lỗ	Bảng thử mạch tạm, dùng để nối linh kiện mà không cần hàn.
Dây nối (jumper wire)	Đực-đực, đực-cái, cái cái	Dùng để nối các chân linh kiện và breadboard.
Điện trở	330R	Giảm nhiễu cho chân điều khiển dữ liệu của LED.
Tụ điện	10000 uF 63V	Ổn định nguồn, chống sụt áp và xung áp khi LED đổi màu.
Cáp USB Type-C / Micro USB	Tùy theo board ESP32	Nạp code cho ESP32 và cấp nguồn khi cần thiết.

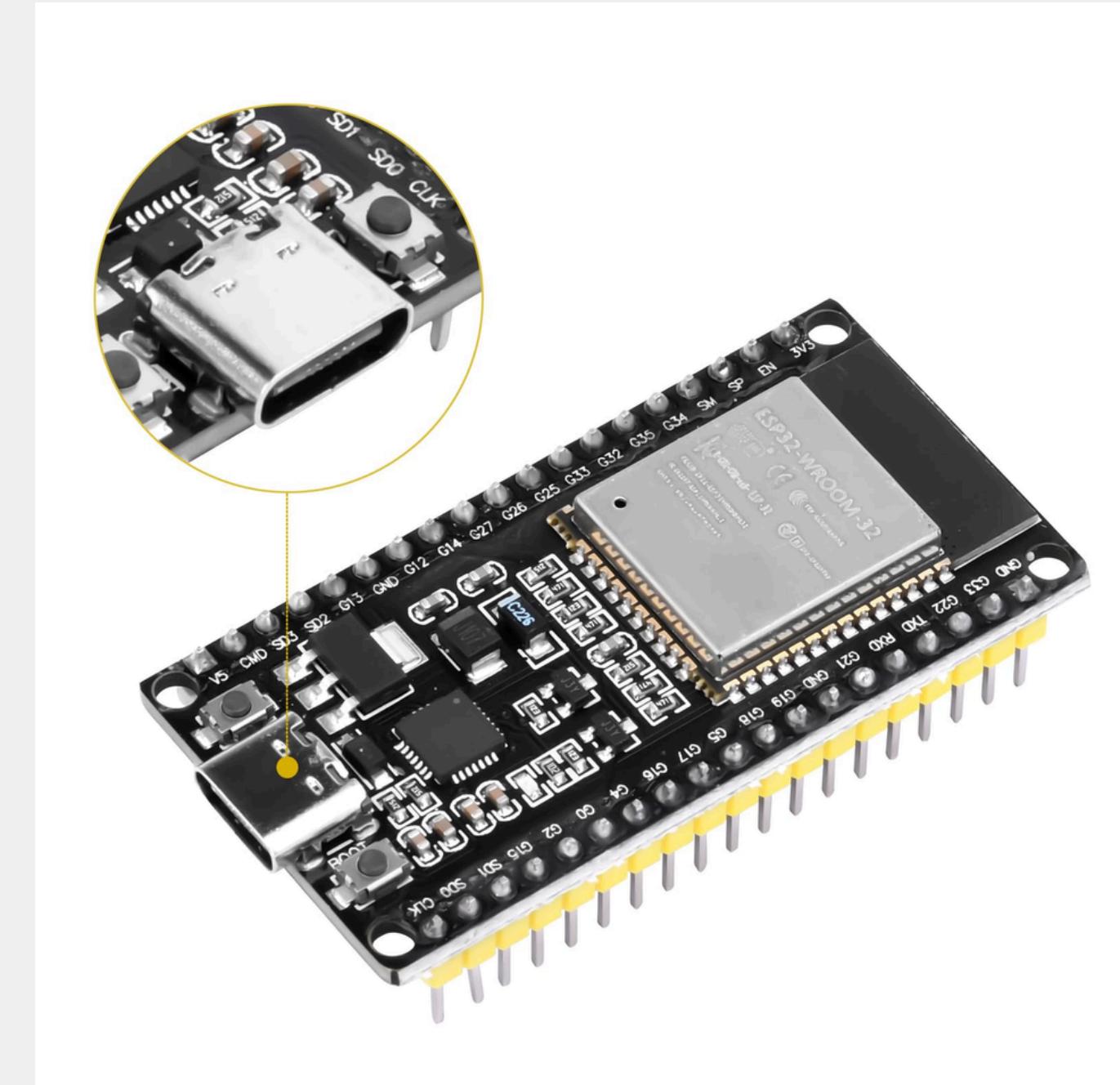


III. THIẾT KẾ HỆ THỐNG VÀ PHẦN CỨNG

3.1 Thành phần phần cứng

ESP32 CP2102 38P Type-C là bo mạch phát triển ESP32 sử dụng:

- WiFi + Bluetooth BLE tích hợp có thể cập nhật code và kết nối app điện thoại,...
- 38 chân GPIO có thể kết nối nhiều module.
- Cổng Type-C giúp nạp code nhanh, dễ dàng và bền bỉ.
- Chip Dual-Core 240 MHz giúp xử lý tốt các tác vụ phức tạp



III. THIẾT KẾ HỆ THỐNG VÀ PHẦN CỨNG

3.1 Thành phần phần cứng

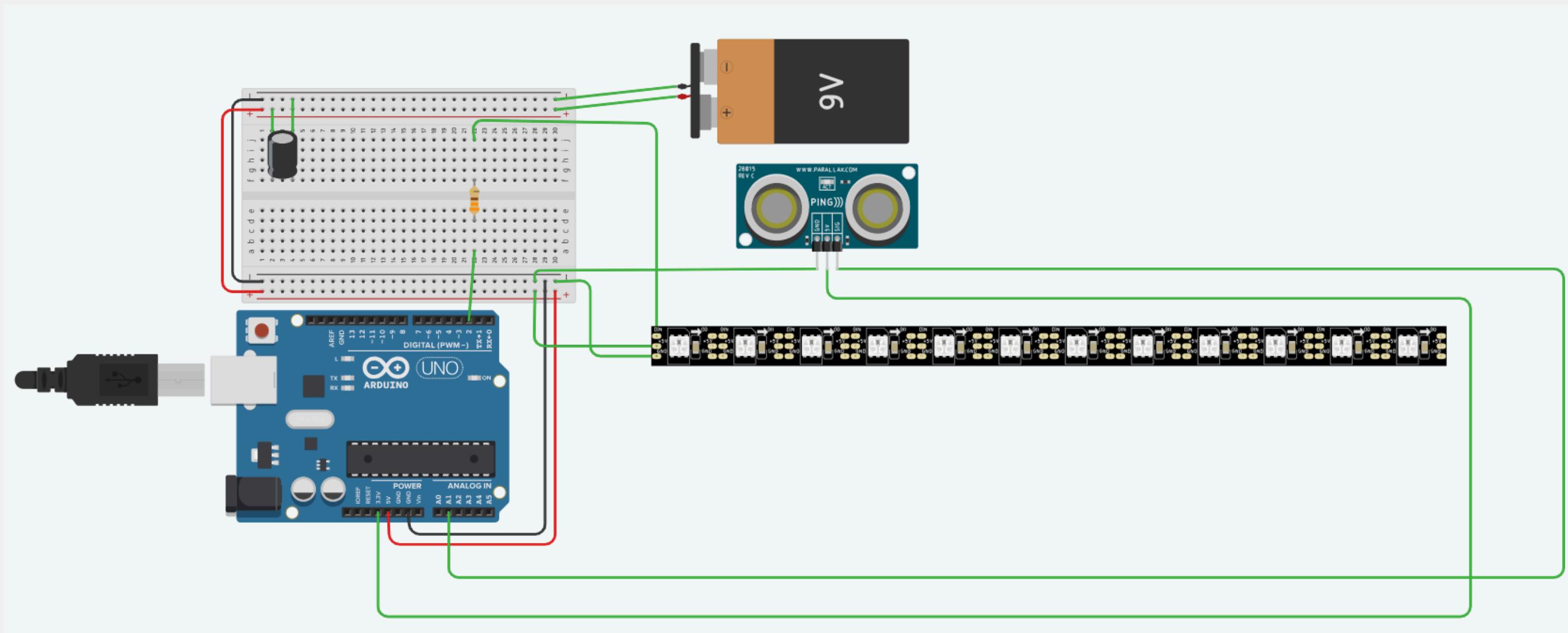
MAX4466 là module microphone khuếch đại âm thanh sử dụng :

- Micro electret có độ nhạy cao có thể thu tiếng nói, tiếng nhạc, âm thanh môi trường rõ ràng.
- Op-amp MAX4466 khuếch đại tín hiệu sóng âm, giảm nhiễu.
- Chiết áp điều chỉnh độ khuếch đại (Gain) cho phép tăng/giảm độ nhạy thu âm.
- Ngõ ra Analog giúp đọc sóng âm dưới dạng điện áp để xử lý FFT, hiển thị LED, nhận dạng âm thanh, v.v.
- Nguồn hoạt động 2.4-5.5V dễ dàng kết nối với ESP32.



III. THIẾT KẾ HỆ THỐNG VÀ PHẦN CỨNG

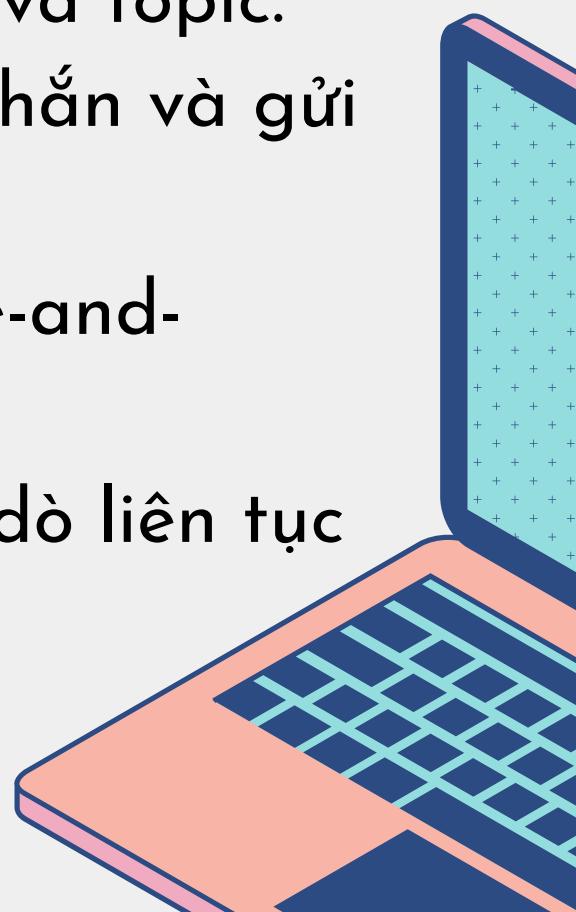
3.2 Sơ Đồ Kết Nối



III. THIẾT KẾ HỆ THỐNG VÀ PHẦN CỨNG

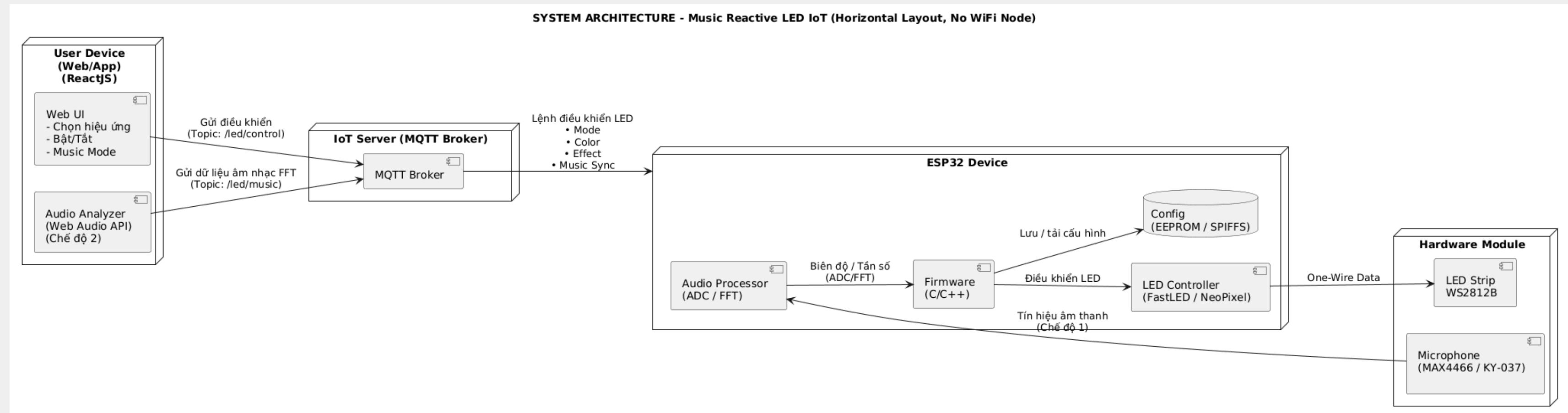
3.3 Giao Thức mạng và truyền thông

- MQTT (Message Queuing Telemetry Transport) là giao thức truyền thông nhẹ (lightweight) dùng trong IoT để các thiết bị giao tiếp với nhau thông qua cơ chế Publish/Subscribe (Pub/Sub).
- Bao gồm 3 thành phần chính là Publisher (thiết bị gửi dữ liệu), MQTT Broker và Subscriber (thiết bị nhận dữ liệu)
- Các đặc điểm nổi bật:
 - Tách dời không gian: Publisher & Subscriber không cần biết nhau, chỉ cần biết broker và topic.
 - Tách rời thời gian: Publisher gửi dữ liệu ngay cả khi subscriber offline. Broker giữ tin nhắn và gửi khi subscriber kết nối.
 - Bất đồng bộ: Publisher gửi xong là quên luôn, không cần chờ subscriber phản hồi (fire-and-forget).
 - Giao tiếp nhanh: Thời gian thực, gần như nhận lệnh ngay lập tức → không cần thăm dò liên tục như HTTP.



III. THIẾT KẾ HỆ THỐNG VÀ PHẦN CỨNG

3.3 Tổng Quan Hệ thống



IV. THIẾT KẾ PHẦN MỀM VÀ CÔNG NGHỆ SỬ DỤNG



4.1 Tổng quan

Chế độ 1 - Micro Mode:

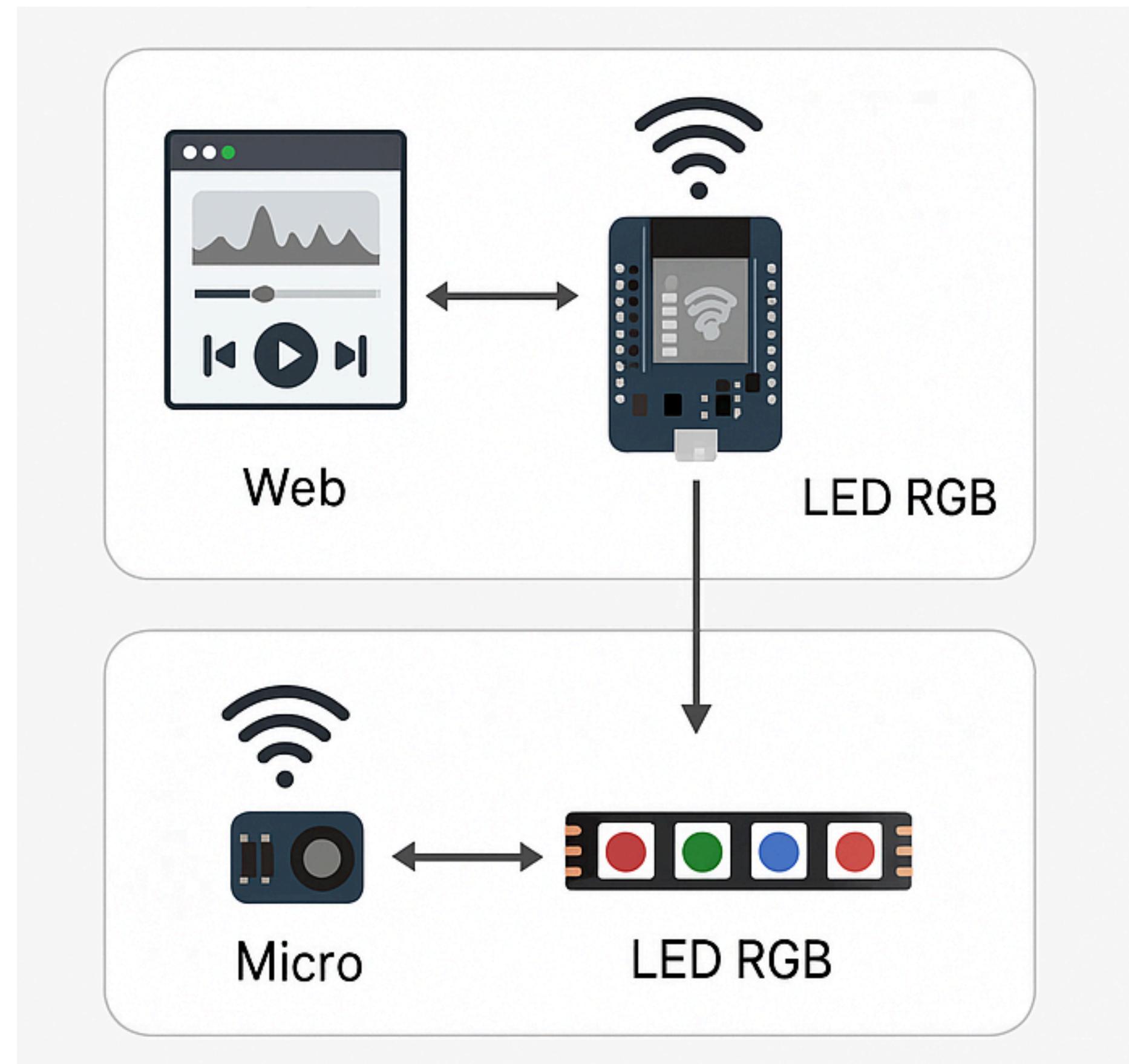
- ESP32 thu âm qua microphone, xử lý (lọc nhiễu, biên độ/ tần số), điều khiển WS2812B

Chế độ 2 - Wifi Stream Mode:

- Web/App phân tích âm thanh bằng Web Audio, gửi gói dữ liệu ánh sáng qua MQTT/WebSocket

Giao diện Web/App cho phép:

- Chọn chế độ, hiệu ứng, màu sắc, độ sáng
- Đồng bộ trạng thái 2 chiều



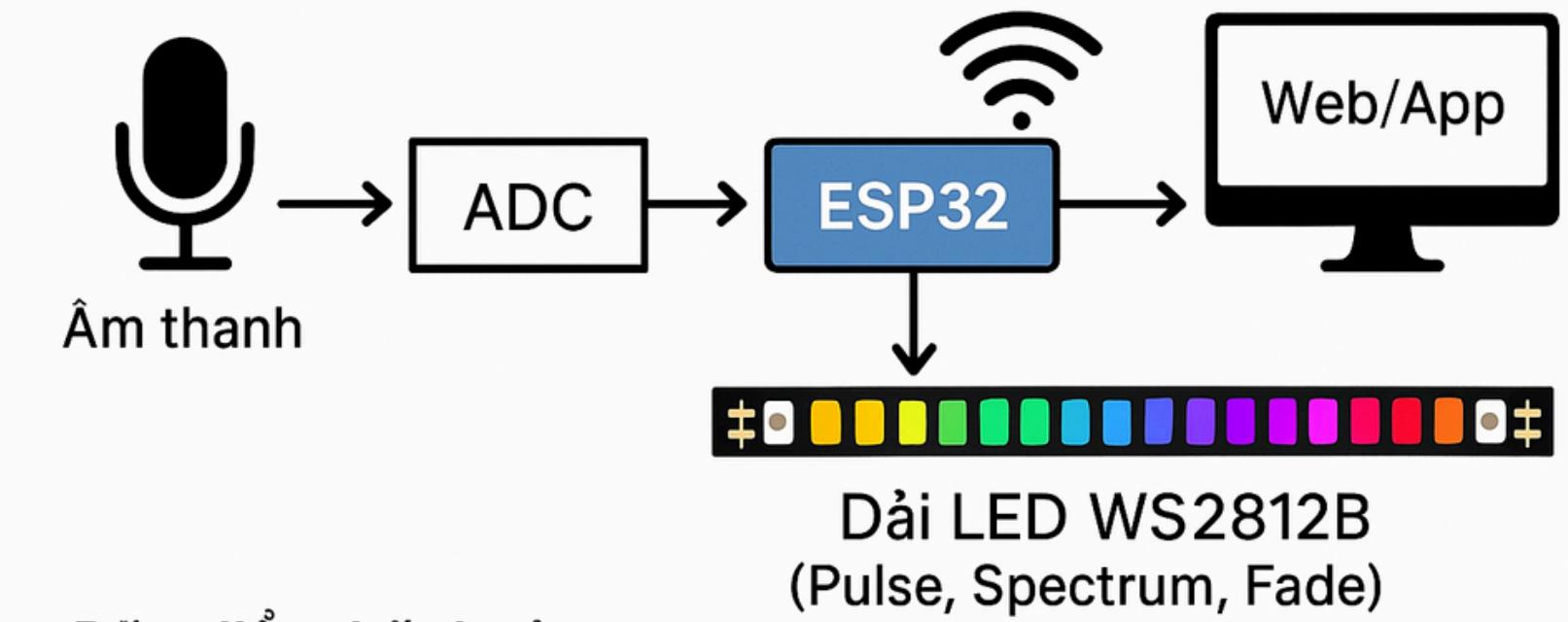
4.2 Hai chế độ hoạt động

Chế độ 1 - Micro Mode

1. Quy trình:

- a. Micro thu âm thanh môi trường → tín hiệu analog → ADC ESP32.
- b. ESP32 xử lý tín hiệu số (lọc nhiễu, phân tích biên độ, FFT).
- c. Ánh xạ dữ liệu âm thanh → hiệu ứng LED (Pulse (Nhấp nháy theo nhịp), Spectrum (Phổ tần số), Fade (Chuyển màu mượt)).
- d. Hiển thị trên dải LED WS2812B.
- e. Người dùng điều khiển qua Web/App (WebSocket/MQTT).

Chế độ 1 – Micro Mode

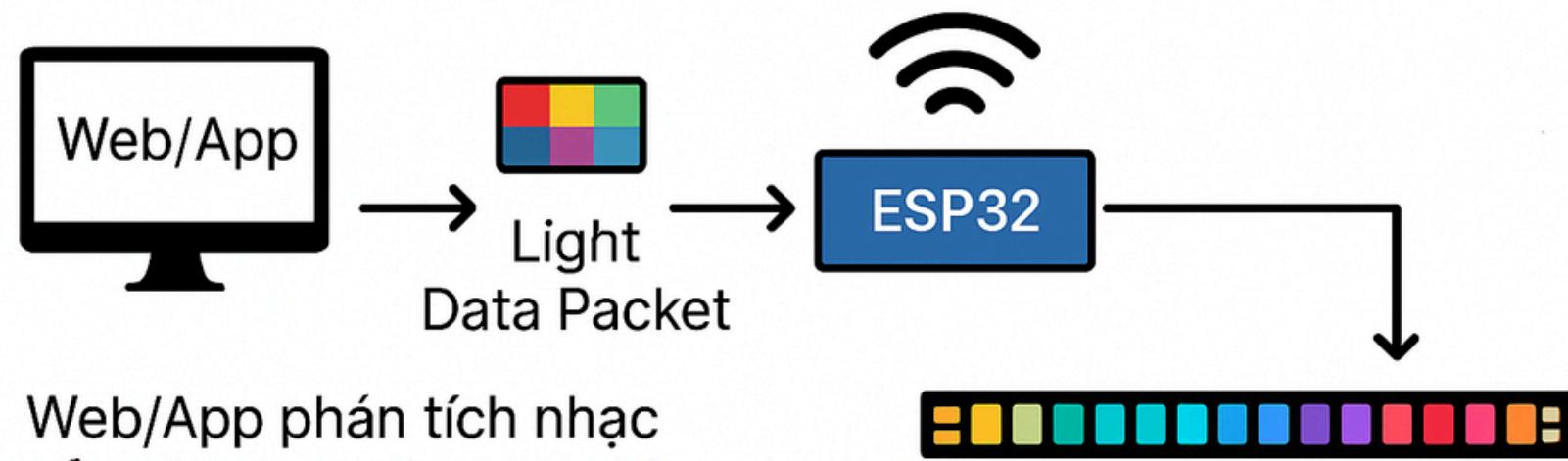


Đặc điểm kỹ thuật:

- Tần số lấy mẫu: 8–11 kHz
- Độ trễ hiển thị: < 50 ms
- Thuật toán: lọc nhiễu, phát hiện beat, FFT 256/512 điểm

4.2 Hai chế độ hoạt động

Chế độ 2 – Wifi Stream Mode



Web/App phân tích nhạc bằng Web Audio API → lấy phổ tần, biên độ, BPM

Đặc điểm kỹ thuật:

- Định mạn gói: {ts, bpm, bands:[bass,mid,treble], effect, brightness, speed}
- Độ trễ tổng: < 200 ms
- Hỗ trợ nhiều thiết bị LED qua một giao diện

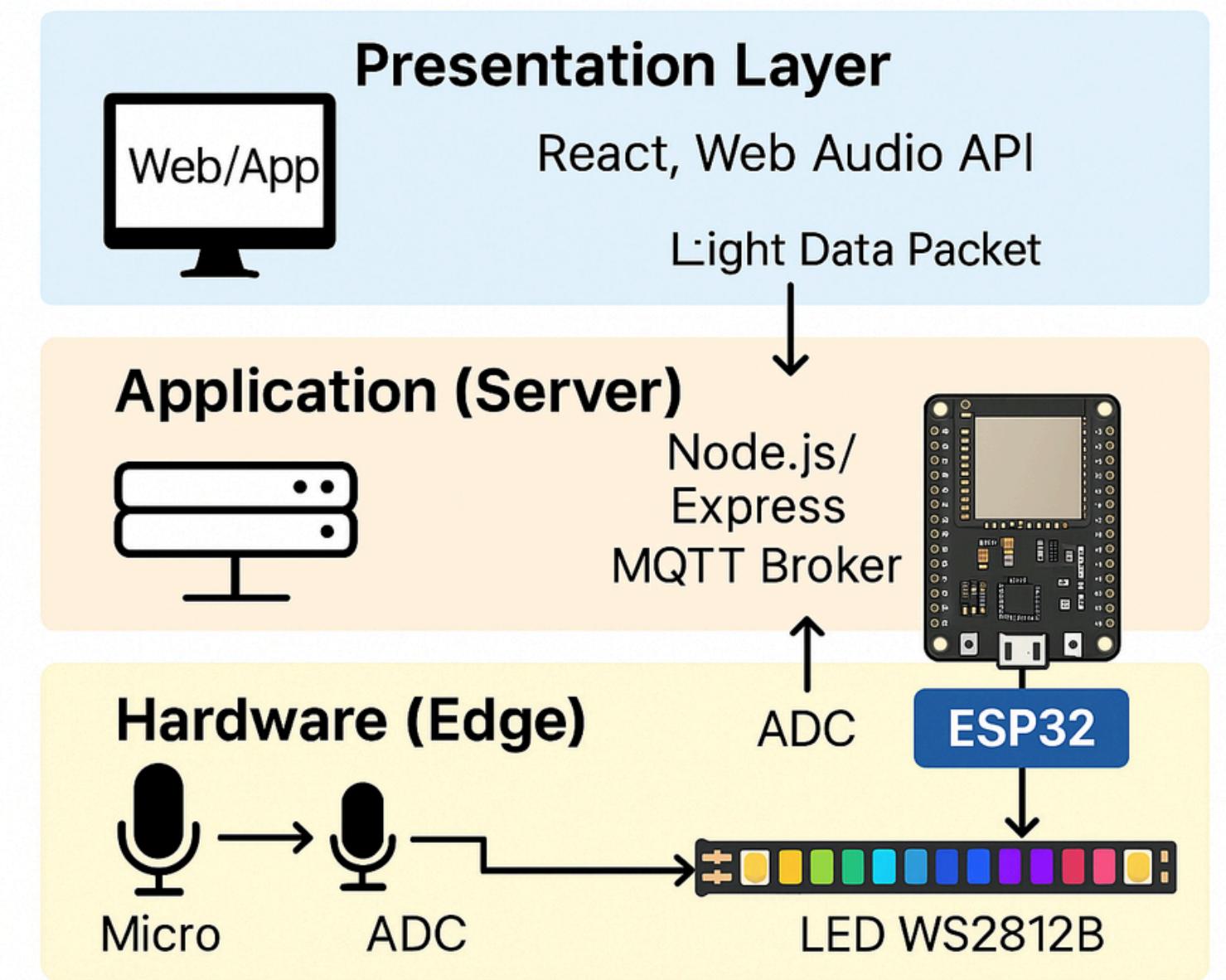
Chế độ 2 - Wifi Stream Mode

1. Quy trình:

- Web/App phân tích nhạc bằng Web Audio API → lấy phổ tần, biên độ, BPM.
- Tạo gói dữ liệu ánh sáng (Light Data Packet – thời gian, BPM, các băng tần, hiệu ứng, độ sáng và tốc độ.) → gửi ESP32 qua MQTT/WebSocket.
- ESP32 giải mã gói → cập nhật LED theo dữ liệu.
- Đồng bộ trạng thái hai chiều giữa ESP32 và giao diện.

4.3 Kiến trúc tổng thể

- Presentation (Web/App): React (SPA) + Web Audio API + MQTT/WebSocket client.
- Application: Node.js/Express, MQTT Broker (Mosquitto), WebSocket Gateway, Auth + OTA Service, Logging.
- Hardware (Edge): ESP32 (Arduino) + Micro (Mode 1) + LED WS2812B.
- Kênh dữ liệu:
 - Chế độ 1 (Micro Mode): Mic → ADC → DSP (ESP32) → LED; điều khiển/telemetry qua MQTT.
 - Chế độ 2 (Wifi Stream): Web/App → phân tích → Light Data Packet → MQTT/WebSocket → ESP32 → LED.



4.3 Kiến trúc tổng thể

4.3.1 Presentation Layer (Lớp trình bày)

- Đây là lớp giao diện người dùng (UI/UX), nơi người dùng tương tác trực tiếp với hệ thống.
- Vai trò:
 - Hiển thị trạng thái thiết bị (màu LED, chế độ, kết nối).
 - Cho phép điều khiển: bật/tắt, chọn chế độ (Micro/Wifi), chọn hiệu ứng, điều chỉnh độ sáng, tốc độ.
 - Phân tích nhạc (ở chế độ Wifi Stream) bằng Web Audio API để lấy phổ tần, BPM, biên độ.
- Công nghệ sử dụng:
 - React (SPA): xây dựng giao diện web hiện đại, phản hồi nhanh.
 - Web Audio API: phân tích tín hiệu âm thanh trên trình duyệt.
 - MQTT/WebSocket client: gửi lệnh điều khiển và nhận trạng thái từ ESP32 theo thời gian thực.

4.3 Kiến trúc tổng thể

4.3.2 Application Layer (Lớp ứng dụng - tùy chọn)

- Đây là lớp trung gian (server) để quản lý logic phức tạp, nhiều thiết bị, bảo mật, và OTA.
- Vai trò:
 - Quản lý kết nối nhiều thiết bị LED cùng lúc.
 - Cung cấp API cho Web/App.
 - Xác thực người dùng (Auth), quản lý token.
 - Hỗ trợ cập nhật firmware từ xa (OTA).
 - Lưu cấu hình người dùng, lịch sử điều khiển.
- Công nghệ sử dụng:
 - Node.js/Express: xây dựng server API.
 - MQTT Broker (Mosquitto/EMQX): giao tiếp thời gian thực với ESP32.
 - WebSocket Gateway: cho LAN hoặc fallback khi MQTT không khả dụng.
 - Logging: ghi lại sự kiện, lỗi, telemetry.

4.3 Kiến trúc tổng thể

4.3.3 Hardware Layer (Lớp phần cứng)

- Đây là lớp thiết bị thực tế, nơi xử lý tín hiệu và điều khiển LED.
- Thành phần:
 - ESP32: vi điều khiển chính, có Wi-Fi, thực hiện DSP (Micro Mode) hoặc nhận dữ liệu từ Web/App (Wifi Stream Mode).
 - Microphone: thu âm thanh môi trường (chế độ Micro Mode).
 - Dải LED WS2812B: hiển thị màu sắc và hiệu ứng ánh sáng.
- Vai trò:
 - Xử lý tín hiệu âm thanh (lọc nhiễu, FFT, phát hiện beat).
 - Điều khiển LED theo hiệu ứng.
 - Giao tiếp với Web/App qua Wi-Fi (MQTT/WebSocket).

4.4 Thiết kế giao diện (UI/UX)

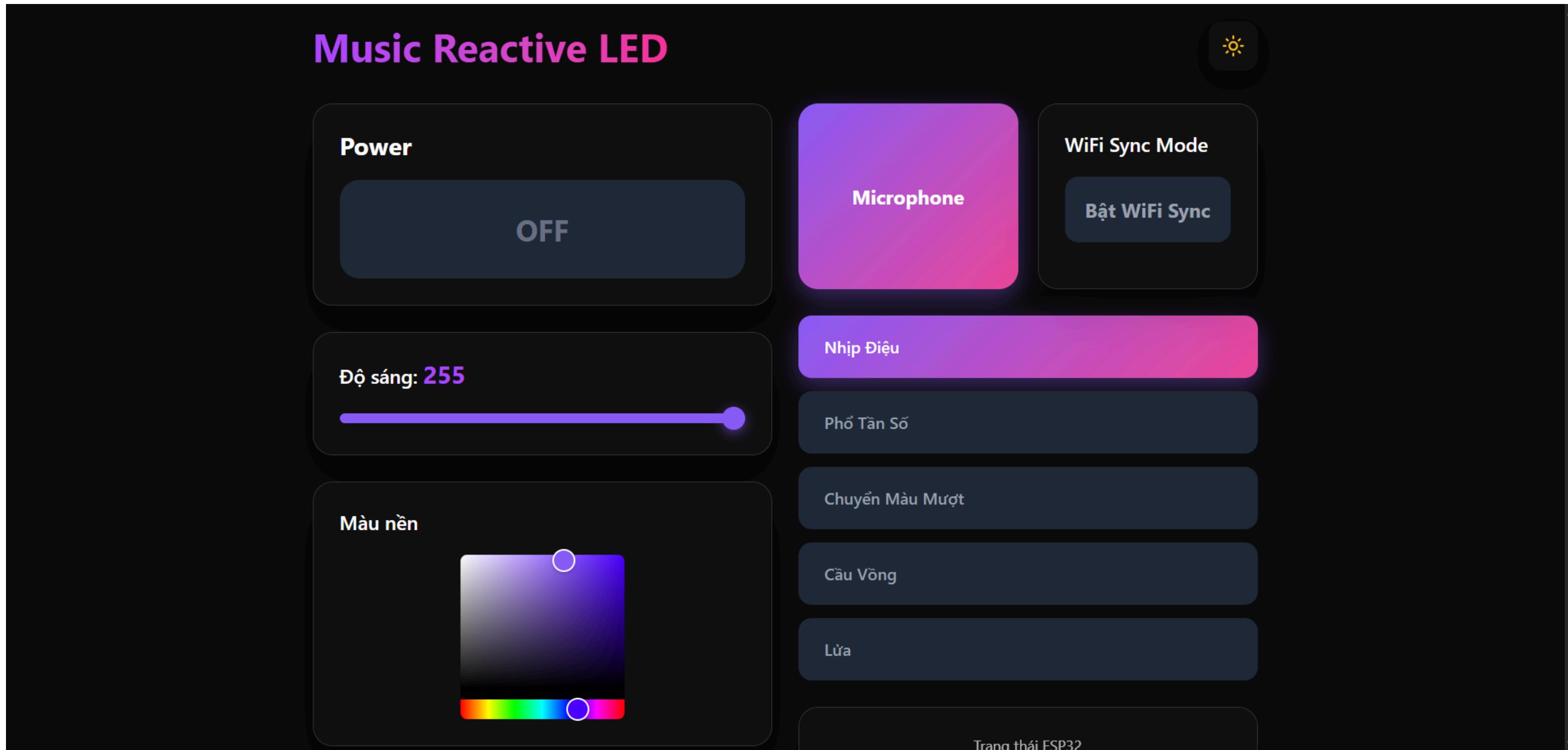
Nguyên tắc & mục tiêu UX

- Ngay lập tức thấy phản hồi (real-time preview, trạng thái kết nối).
- Đơn giản hóa lựa chọn (preset + chỉnh sâu “Advanced”).
- Khả năng mở rộng (nhiều thiết bị, nhóm thiết bị).
- Khả năng truy cập (dark mode, responsive).

Cấu trúc màn hình

- Dashboard: Tổng quan thiết bị, trạng thái, nhanh chọn chế độ & hiệu ứng.
- Device Detail: Điều khiển sâu từng dải LED (thanh sliders, palette, effect params).
- Music Analyzer (Mode 2): Visualization phổ tần, BPM, bands realtime.
- Settings: Wi-Fi, MQTT broker, OTA, quyền truy cập.
- Logs/Diagnostics: Ping, RSSI, latency, firmware version.

4.4 Thiết kế giao diện (UI/UX)



Công nghệ sử dụng

- Firmware: Arduino IDE/PlatformIO, thư viện ArduinoFFT, FastLED/Adafruit_NeoPixel.
- Frontend: ReactJS, Web Audio API.
- Backend: Node.js + Express, MQTT Broker.
- Giao thức: MQTT/WebSocket cho real-time, HTTP cho cấu hình.

Công nghệ sử dụng

```
Attaching to mosquitto
mosquitto | 1763517157: mosquitto version 2.0.22 starting
mosquitto | 1763517157: Config loaded from /mosquitto/config/mosquitto.conf.
mosquitto | 1763517157: Opening ipv4 listen socket on port 1883.
mosquitto | 1763517157: Opening ipv6 listen socket on port 1883.
mosquitto | 1763517157: Opening websockets listen socket on port 9001.
mosquitto | 1763517157: mosquitto version 2.0.22 running
mosquitto | 1763517158: New connection from 172.18.0.1:40582 on port 1883.
mosquitto | 1763517158: New client connected from 172.18.0.1:40582 as mqttjs_e10317ee (p2, c1, k60).
```

Chạy broker trên local bằng mosquitto

- Mosquitto là một MQTT Broker - thành phần trung gian trong giao thức MQTT.
- Nó cho phép các thiết bị publish (gửi) và subscribe (nhận) thông điệp theo mô hình Pub/Sub.

Công nghệ sử dụng

```
PS D:\Coding\PTIT\BTL-IOT---Music-Reactive-LED-Strip\Source code\server> node index.js
Server chạy tại http://localhost:3000
Web kết nối: GnozRpcF5t1xGWcYAAAB
MQTT Broker kết nối OK
Subscribed: led/control/#  
Subscribed: led/status
Subscribed: led/config/save
Web kết nối: GmeA0ktvP8sXHuESAAAD
Web → ESP32: led/control/color #7951d8
MQTT → Web: led/control/color #7951d8
Web → ESP32: led/control/color #7850d7
MQTT → Web: led/control/color #7850d7
Web → ESP32: led/control/color #764ed2
MQTT → Web: led/control/color #764ed2
Web → ESP32: led/control/color #764fd0
MQTT → Web: led/control/color #764fd0
Web → ESP32: led/control/color #744dcd
MQTT → Web: led/control/color #744dcd
```

Server nodejs giao tiếp giữa web/app với thiết bị

Mối quan hệ giữa phần cứng và phần mềm

Phần cứng	Phần mềm tương ứng	Mô tả tương tác
ESP32	Arduino Firmware	Đọc tín hiệu, điều khiển LED, xử lý dữ liệu.
Microphone	ADC Module	Cung cấp tín hiệu âm thanh analog cho ESP32.
LED WS2812B	FastLED Library	Nhận tín hiệu điều khiển để đổi màu, hiển thị hiệu ứng.
Web/App	ReactJS + MQTT	Gửi lệnh điều khiển và nhận phản hồi từ ESP32.
Cloud Server	Node.js / Firebase	Lưu trữ cấu hình và dữ liệu người dùng (tùy chọn).

Hướng phát triển tiếp

- Tích hợp với các music streaming services: Youtube, Spotify, Apple Music,...
- Đồng bộ đa thiết bị: nhiều dải LED cùng một bài nhạc, clock sync (NTP).
- Tích hợp nhà thông minh: Home Assistant, automation theo ngữ cảnh.
- Ứng dụng thực tế: sân khấu mini, quán cà phê, preset theo playlist.

**Thank you for
listening!**