ESP32-C6 Mini

좋습니다. 👍
아까 드린 **최소 학습용 프로젝트**를 **ESP32-C6 mini 보드**용으로 그대로 옮겨서 다시 정리해 드리겠습니다.
(구조/개념은 동일, 라이브러리만 ESP32용으로 변경)

🎯 목표

- ESP32-C6이 AP 모드로 웹서버 실행
- `/` → `index.html` 제공
- `/api/hello` → 문자열 응답
- HTML 버튼 → API 호출 → 화면에 결과 표시

👉 **연결 구조 학습용 최소 예제**

📁 프로젝트 구조

```
minimal_esp32_c6_web/
├─ platformio.ini
├─ src/
│  └─ main.cpp
└─ data/
   └─ index.html
```

⚙️ platformio.ini (ESP32-C6)

> ⚠️ 보드 이름은 보통 `esp32-c6-devkitc-1` 을 사용합니다.
> (mini 모듈도 이 설정으로 대부분 동작)

```ini
[env:esp32c6]
platform = espressif32
board = esp32-c6-devkitc-1
framework = arduino
monitor_speed = 115200
```

---

src/main.cpp (ESP32-C6용)

```cpp
#include <Arduino.h>
#include <WiFi.h>
#include <WebServer.h>
#include <LittleFS.h>

WebServer server(80);

void setup() {
  Serial.begin(115200);

  // AP 모드 시작
  WiFi.mode(WIFI_AP);
  WiFi.softAP("ESP32C6_AP", "12345678");

  Serial.print("AP IP: ");
  Serial.println(WiFi.softAPIP());

  // LittleFS 마운트
  if (!LittleFS.begin(true)) {
    Serial.println("LittleFS mount failed");
  }

  // "/" → index.html 제공
  server.on("/", HTTP_GET, []() {
    File f = LittleFS.open("/index.html", "r");
    if (!f) {
      server.send(500, "text/plain", "index.html not found");
      return;
    }
    server.streamFile(f, "text/html");
    f.close();
  });

  // "/api/hello" → 문자열 응답
  server.on("/api/hello", HTTP_GET, []() {
    server.send(200, "text/plain", "hello from esp32-c6");
  });

  server.begin();
  Serial.println("Server started");
}

void loop() {
  server.handleClient();
}
```

🔹 ESP8266 버전과 차이

ESP8266	ESP32-C6
`#include <ESP8266WiFi.h>`	`#include <WiFi.h>`
`#include <ESP8266WebServer.h>`	`#include <WebServer.h>`
`ESP8266WebServer`	`WebServer`

👉 **개념/구조는 완전히 동일**, 라이브러리 이름만 다름.

🌐 data/index.html

(ESP8266 때와 동일)

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Minimal ESP32-C6 Test</title>
</head>
<body>
  <h2>Minimal ESP32-C6 Web Test</h2>

  <button onclick="sayHello()">Say Hello</button>
  <div id="result"></div>

  <script>
    async function sayHello() {
      const res = await fetch("/api/hello");
      const txt = await res.text();
      document.getElementById("result").innerText = txt;
    }
  </script>
</body>
</html>
```

🚀 업로드 & 실행

1️⃣ 코드 업로드

```
pio run -t upload
```

2️⃣ LittleFS 업로드

```
pio run -t uploadfs
```

3️⃣ 시리얼 모니터

```
pio device monitor
```

출력 예:

```
AP IP: 192.168.4.1
Server started
```

📶 접속 방법

1. WiFi 연결
   - SSID: **ESP32C6_AP**
   - PASS: **12345678**
2. 브라우저 접속

```
http://192.168.4.1/
```

3. **Say Hello** 버튼 클릭 →
   화면에:

```
hello from esp32-c6
```

보이면 성공 🎉

🧠 여기서 학습 포인트 (ESP32-C6에서도 동일)

이 한 쌍만 기억하면 됩니다:

```
server.on("/api/hello", ...);
```

```
fetch("/api/hello");
```

👉 **주소 문자열이 같으면 연결된다.**
ESP8266이든 ESP32-C6든 구조는 완전히 동일.

✅ 한 줄 요약

> 이 예제는
> **ESP32-C6에서 ESP8266과 동일한 방식으로**
> **HTML ↔ JS ↔ C++ API 연결 구조를 학습하기 위한**
> **최소 프로젝트**이다.

이걸로 그대로 실습해 보시고,
막히는 지점이나 “이건 왜 이래?” 싶은 코드가 나오면
그 부분만 가져와서 다시 물어보세요. 👍