# Note:

Click on en=reset button to start from begineering. Like to see on log. What is doing on from the begineeings.

# Youtube playlist

https://www.youtube.com/watch?v=yQxk5pGRUYg&list=PLciC3-3T2N8YkcVlGfFBNGvLlwemC4nzt

# Driver link

Cp2102

Silicon laps

**CP210x Windows Drivers**

<https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions/>

<https://www.silabs.com/software-and-tools/usb-to-uart-bridge-vcp-drivers>

# List of components

Please give me a list of videos related to the project I am preparing using Arduino.

**Project Name:** Weather Station

**Components I have:**

* Two ESP32-WROOM-32 boards (one for the sender and one for the receiver)
* DHT sensor
* Two LoRa modules

# If lead not blink to blink lead with code

If your **ESP32 is only showing a red light (usually solid)** and not responding, here’s what it likely means and what you can do:

**🔴 Red LED on ESP32 – Common Meaning**

1. **Power Indicator**
   * A red LED on most ESP32 boards is just a **power indicator** (usually near the USB port).
   * If it's **solid red**, it usually means the board **has power** — this is **normal**.
2. **No Blinking or Serial Output?**  
   If it's only red and:
   * Not blinking
   * Not showing anything in the Serial Monitor
   * Not uploading code

Then there might be an issue with:

* + The **code you uploaded** (e.g., stuck in a loop or crash)
  + **USB cable or port** (try a different one)
  + **Boot mode or flash mode stuck**

**🧪 Try These Steps to Fix It**

## ✅ Step 1: Check the Serial Monitor

* Open Arduino IDE > Tools > Serial Monitor
* Set **baud rate to 115200**
* Press the **RESET (EN)** button on the board
* See if anything prints (like a boot message or crash dump)

## ✅ Step 2: Force Into Boot Mode (for uploading)

Sometimes the ESP32 won’t auto-enter bootloader mode.

* Hold the **BOOT** button (some boards label it “IO0”)
* While holding BOOT, **press and release the RESET (EN)** button
* Keep holding BOOT for 1–2 seconds, then release

Now try uploading your sketch again.

## ✅ Step 3: Upload a Simple Blink Sketch

Try uploading this minimal sketch to recover your board:

void setup() {

pinMode(2, OUTPUT);

}

void loop() {

digitalWrite(2, HIGH);

delay(500);

digitalWrite(2, LOW);

delay(500);

}

**⚠️ If still not working:**

* Try a different **USB cable** (many charge-only cables won’t work for data).
* Try a different **USB port** or **computer**.
* Check that you selected the correct **Board** and **COM port** in Arduino IDE.

Let me know your exact board model (e.g., DOIT ESP32 DevKit v1), and I can give you more specific help!

# is my esp32 will share an hotspot.

Your **ESP32 cannot create a full-featured internet-sharing hotspot** like a smartphone or router. However, it **can act as a Wi-Fi Access Point (AP)** — but only for **local communication**, not for sharing internet.

**🔥 Key Points:**

| **Feature** | **ESP32 Capability** |
| --- | --- |
| Create Wi-Fi hotspot | ✅ Yes (SoftAP mode) |
| Share internet via hotspot | ❌ No |
| Connect other devices (phone, etc.) to ESP32 AP | ✅ Yes |
| Provide internet access to connected devices | ❌ No |
| Connect to Wi-Fi network (Client/STA mode) | ✅ Yes |

**✅ What ESP32 *Can* Do in Hotspot Mode:**

* Create its own **local Wi-Fi network** (Soft Access Point).
* Other devices (phones, laptops) can connect to it via Wi-Fi.
* You can run a **web server** or transfer data between devices locally.

**❌ What ESP32 *Cannot* Do:**

* Share your phone’s or PC’s **mobile data or Wi-Fi internet** through its AP.
* Act like a **real router or mobile hotspot**.

## ✅ Example Use Case for Hotspot Mode (SoftAP):

If you're building a **weather station**, you could:

* Turn ESP32 into a **Wi-Fi hotspot**.
* Let a phone connect to it and **view sensor data** via browser (ESP32 Web Server).

Here’s how to enable AP mode on ESP32:

#include <WiFi.h>

void setup() {

Serial.begin(115200);

// Set up ESP32 as a Wi-Fi Access Point

WiFi.softAP("ESP32\_Weather", "12345678");

Serial.println("AP Started");

Serial.println("IP address: ");

Serial.println(WiFi.softAPIP());

}

void loop() {

// Handle server or sensor tasks

}

Then connect your phone/laptop to Wi-Fi named **"ESP32\_Weather"** (password: 12345678) to interact locally.

**🔄 Alternative:**

If you want the ESP32 to send data online (like to a cloud server or Blynk/ThingSpeak):

* Connect it to your phone’s hotspot (ESP32 in **Station mode**).
* But still: **ESP32 cannot itself provide internet** to other devices.

If you're aiming to **send weather data from ESP32 to your phone or PC**, I can help you set up:

* A **local web server**
* A **mobile app (via Blynk)**
* Or **Bluetooth mode**

Just let me know your use case!

# using only esp32 can i host an server like php.

help me to host simple hello world page

Yes! ✅ You **can host a simple web server directly on the ESP32** — no PHP or external server required.

The ESP32 **does not support PHP** natively (PHP needs a real OS and interpreter), but it can serve **HTML pages** — and you can even mimic simple dynamic behavior using C++ inside Arduino code.

**✅ Your Goal:**

Host a simple Hello, World! web page on ESP32.

**📦 What You Need:**

* ESP32 board
* Arduino IDE with ESP32 board support installed
* No internet required

**🔧 Code to Host a “Hello World” Web Page**

This example creates a **local Wi-Fi hotspot** and serves a basic HTML page at http://192.168.4.1.

#include <WiFi.h>

// Set up access point credentials

const char \*ssid = "ESP32\_HelloWorld";

const char \*password = "12345678";

// Set up web server on port 80

WiFiServer server(80);

void setup() {

Serial.begin(115200);

// Start Wi-Fi in AP (Access Point) mode

WiFi.softAP(ssid, password);

IPAddress IP = WiFi.softAPIP();

Serial.print("Access Point IP: ");

Serial.println(IP);

// Start the server

server.begin();

}

void loop() {

WiFiClient client = server.available(); // Wait for a client to connect

if (client) {

Serial.println("Client connected.");

// Wait until client sends a request

while (!client.available()) {

delay(1);

}

// Read request (not used in this simple example)

String req = client.readStringUntil('\r');

Serial.print("Request: ");

Serial.println(req);

client.flush();

// Send HTTP response

client.println("HTTP/1.1 200 OK");

client.println("Content-type: text/html");

client.println();

// HTML content

client.println("<!DOCTYPE html>");

client.println("<html><head><title>Hello</title></head><body>");

client.println("<h1>Hello, World! 🌍</h1>");

client.println("<p>This page is hosted from your ESP32!</p>");

client.println("</body></html>");

delay(1);

Serial.println("Client disconnected.");

}

}

**📲 How to Use:**

1. Upload the code to your ESP32 using Arduino IDE.
2. Open **Serial Monitor** @ 115200 baud.
3. It will show:
4. Access Point IP: 192.168.4.1
5. On your phone/laptop, connect to Wi-Fi:
   * **SSID**: ESP32\_HelloWorld
   * **Password**: 12345678
6. Open your browser and go to:
7. http://192.168.4.1
8. You’ll see: Hello, World! 🌍

**📌 Notes:**

* This is **not PHP**, but you can simulate dynamic content using C++.
* Want to build more complex pages (CSS/JS/HTML)? You can serve them from:
  + SPIFFS or LittleFS filesystem
  + SD card
  + Or hardcoded like above

**🧪 Next Steps (Optional):**

Would you like me to help you:

* Add **CSS/JS** to the page?
* Create an **HTML form** and handle its response?
* Host files from the ESP32's flash (SPIFFS)?

Just ask!

# Hello world local host

#include <WiFi.h>

// Set up access point credentials

const char \*ssid = "ESP32\_HelloWorld";

const char \*password = "12345678";

// Set up web server on port 80

WiFiServer server(80);

void setup() {

  Serial.begin(115200);

  // Start Wi-Fi in AP (Access Point) mode

  WiFi.softAP(ssid, password);

  IPAddress IP = WiFi.softAPIP();

  Serial.print("Access Point IP: ");

  Serial.println(IP);

  // Start the server

  server.begin();

}

void loop() {

  WiFiClient client = server.available(); // Wait for a client to connect

  if (client) {

    Serial.println("Client connected.");

    // Wait until client sends a request

    while (!client.available()) {

      delay(1);

    }

    // Read request (not used in this simple example)

    String req = client.readStringUntil('\r');

    Serial.print("Request: ");

    Serial.println(req);

    client.flush();

    // Send HTTP response

    client.println("HTTP/1.1 200 OK");

    client.println("Content-type: text/html");

    client.println();

    // HTML content

    client.println("<!DOCTYPE html>");

    client.println("<html>");

    client.println("<head>");

    client.println("<meta name='viewport' content='width=device-width, initial-scale=1'>");

    client.println("<title>ESP32 Weather Station</title>");

    client.println("<style>");

    client.println("body { font-family: Arial, sans-serif; background-color: #f0f8ff; color: #333; text-align: center; padding: 20px; }");

    client.println("h1 { color: #004d99; font-size: 32px; }");

    client.println(".team { margin-top: 30px; }");

    client.println(".team h2 { color: #006699; }");

    client.println(".team p { font-size: 18px; }");

    client.println("marquee { background-color: #004d99; color: white; padding: 10px; font-size: 20px; font-weight: bold; border-radius: 5px; }");

    client.println("</style>");

    client.println("</head>");

    client.println("<body>");

    client.println("<marquee>🔧 Project: ESP32-Based Weather Station</marquee>");

    client.println("<h1>Welcome to Our IoT Project!</h1>");

    client.println("<p>This page is hosted directly from your ESP32 microcontroller.</p>");

    client.println("<div class='team'>");

    client.println("<h2>👨‍💻 Team Members</h2>");

    client.println("<p>• Santosh Khilbahadur<br>");

    client.println("• Saraswoti Rokaya<br>");

    client.println("• Santosh Kumar Shah</p>");

    client.println("</div>");

    client.println("<p><small>© 2025 – Powered by ESP32 Wi-Fi</small></p>");

    client.println("</body>");

    client.println("</html>");

    delay(1);

    Serial.println("Client disconnected.");

  }

}