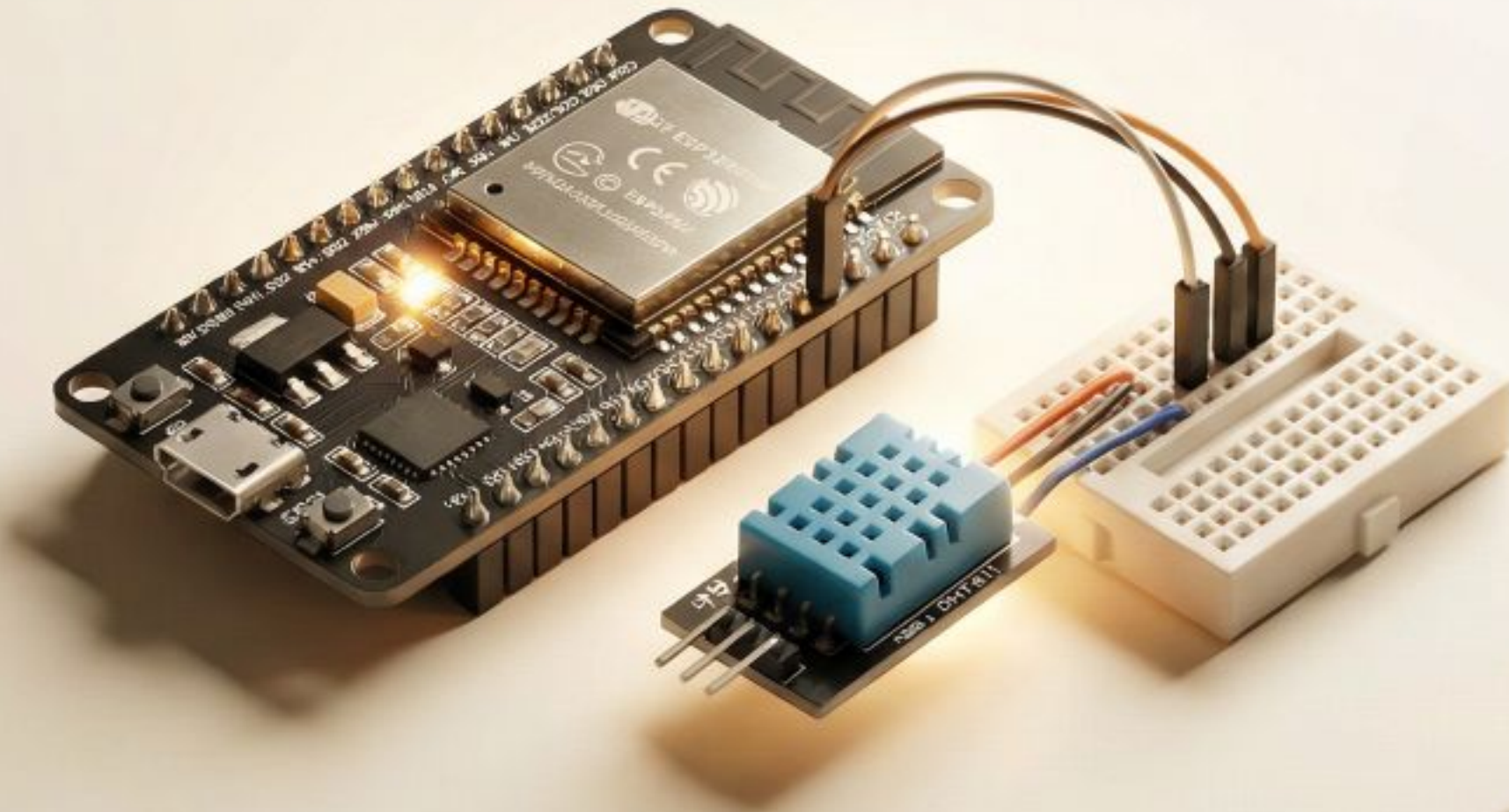


Hands-On IoT Workshop: From Sensors to Smart Systems

3-Hour Workshop | Build. Learn. Innovate.



github.com/eecsclubofficial/workshop_latency26

IoT Around You Right Now

From hostels to classrooms, we interact with connected devices every day.



Smartphone



Your primary connection hub for communication, entertainment, and control.



Smart Light Bulb



Adjustable lighting for study sessions or relaxation.



Fitness Tracker



Monitors activity levels, sleep patterns, and health metrics.



Smart Speaker



Voice-controlled assistant for music, information, and home automation.

How many IoT devices do you interact with daily?



Average per person

Did you know?

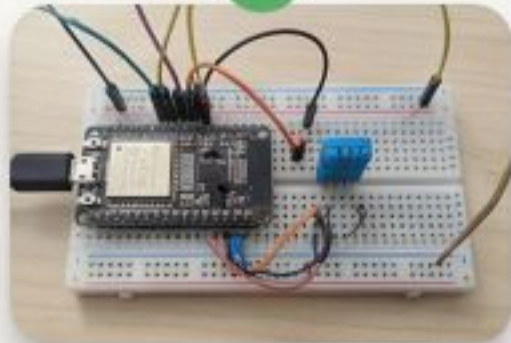
The number of connected devices per person is expected to double in the next few years.

Your Workshop Journey Map



What You'll Take Home Today

Skills & Confidence Checklist



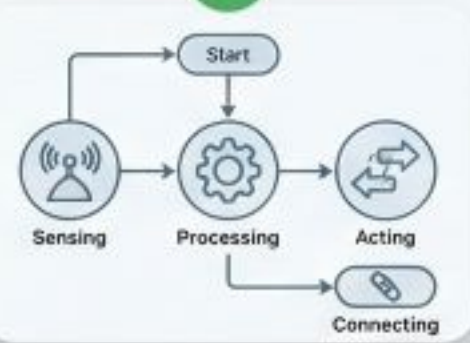
Hardware Understanding

Circuit building & component knowledge



Coding Basics

ESP32 programming & sensor interfacing



IoT Concepts

Connectivity & data flow



Real Problem-Solving

Troubleshooting & debugging skills



A Working Project

Your own built IoT system



github.com/eecsclubofficial/workshop_latency26

Access to All Code & Resources for Continued Learning



Certificate of Completion & Confidence to Build

Tangible Outcomes. Ready to Innovate.

What is IoT?

Internet of Things connects physical devices to the internet enabling data exchange and smart automation.



Bridging the physical and digital worlds through connected systems

Real-World IoT Impact



Smart Agriculture

Moisture sensors save 30-50% water in irrigation.



Home Automation

Intelligent temperature control reduces energy costs.



Security Systems

IR sensors for motion detection and access control.



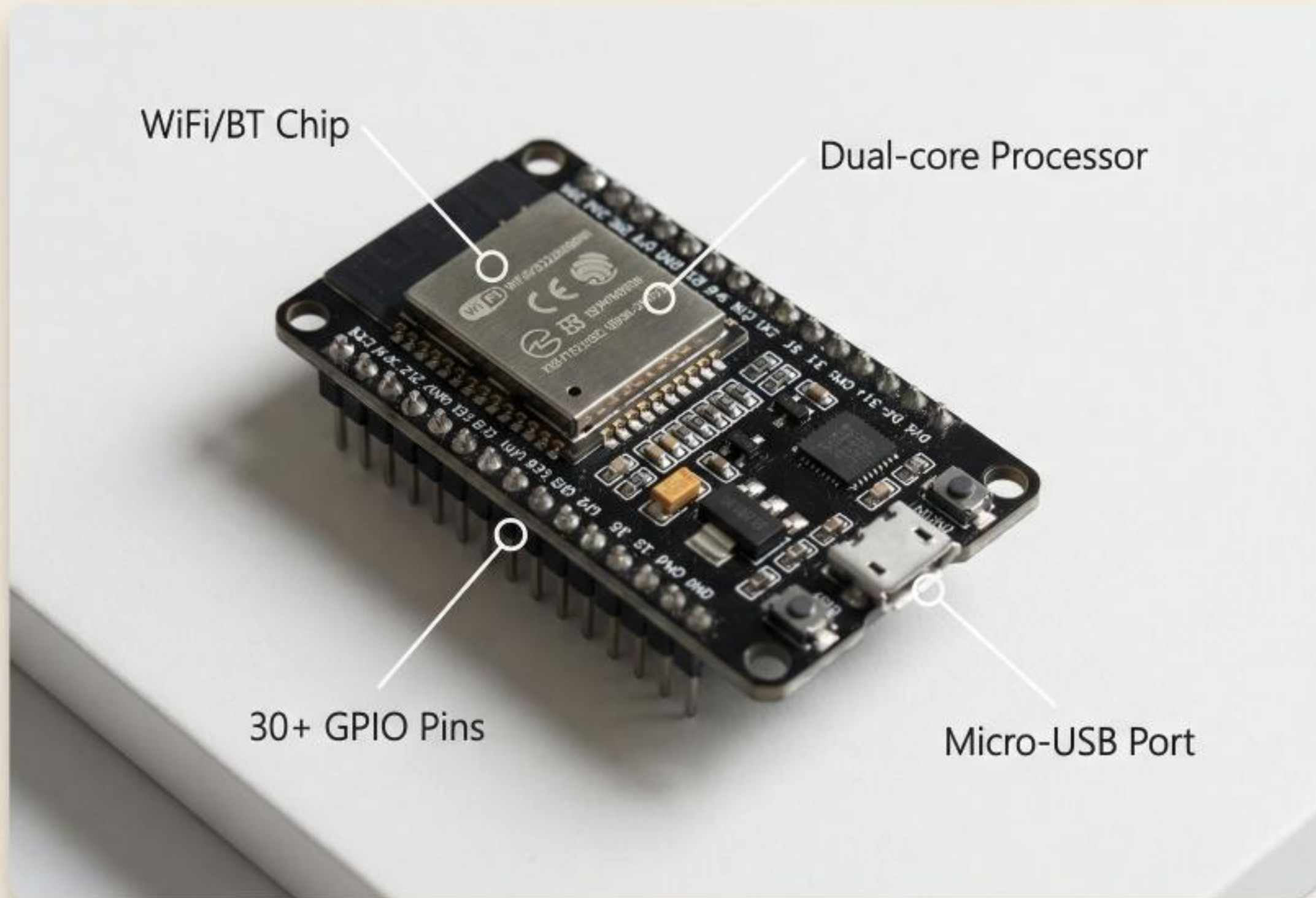
Healthcare

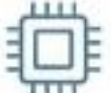
Continuous patient monitoring improves outcomes.

Indian Context

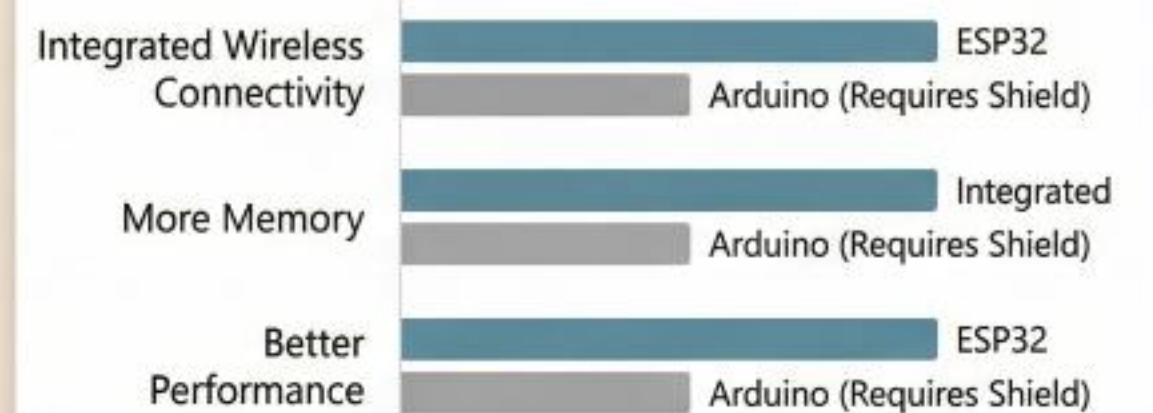
- Local examples & growing market size data. IoT solves real problems.

Meet Your Brain - ESP32



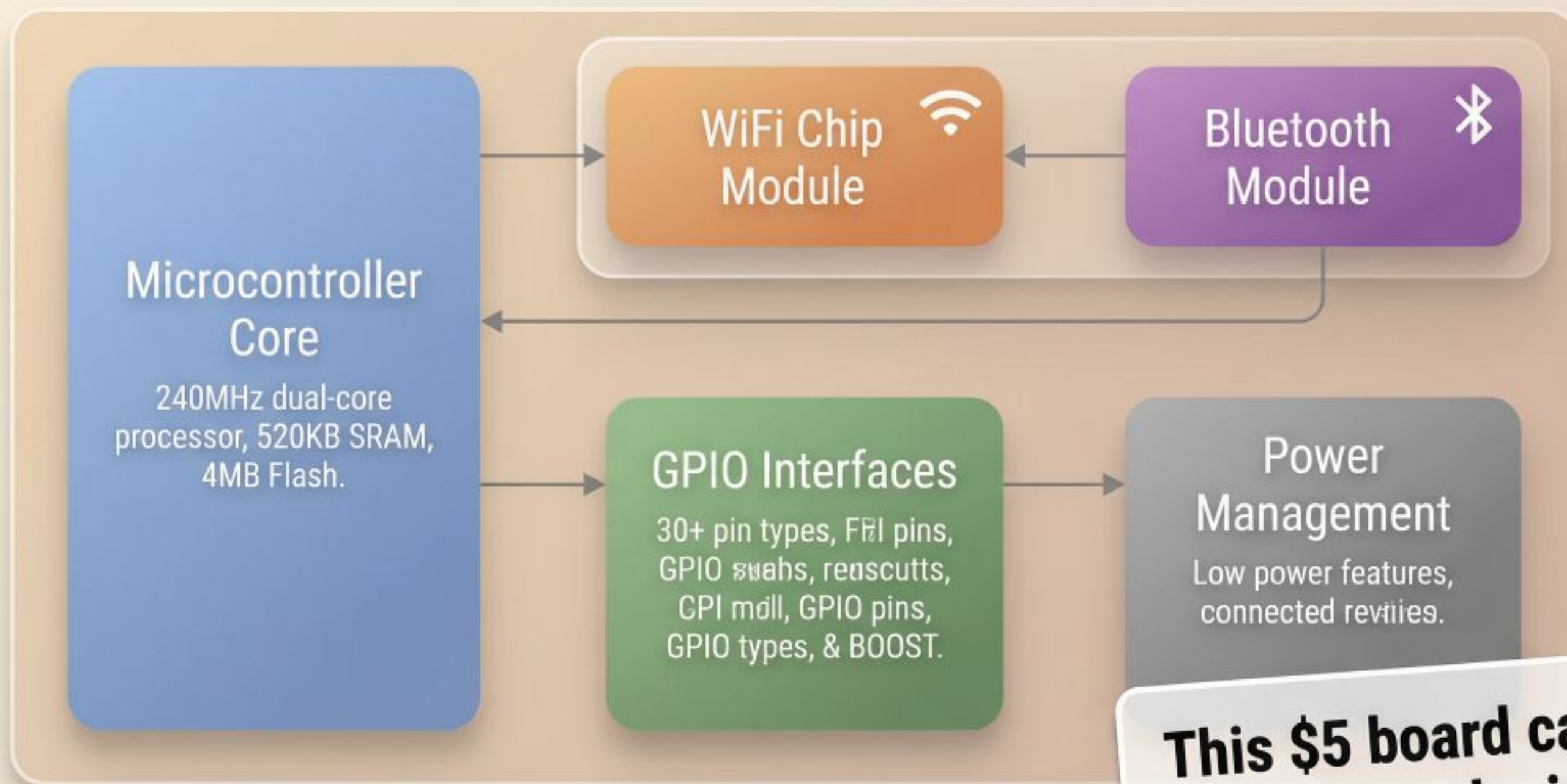
-  Built-in WiFi & Bluetooth
-  Powerful dual-core processor
-  30+ GPIO pins for versatile connections
-  Low power consumption

Why ESP32 > Arduino for IoT



At a similar price point, ESP32 offers superior integrated features for IoT projects.

ESP32 Architecture Simplified

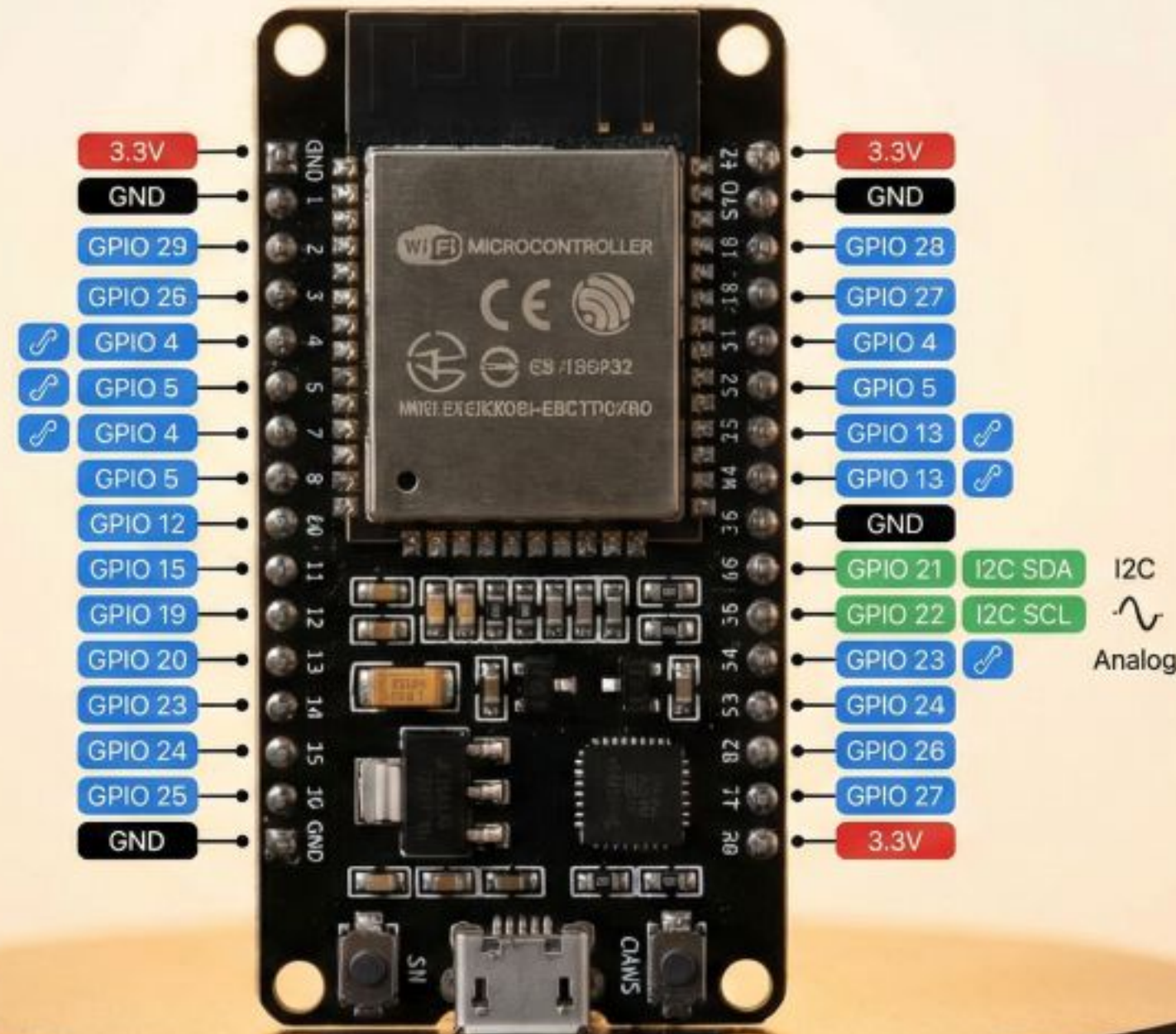


This \$5 board can connect to the internet! 

All components work together on a single chip for seamless IoT integration, empowering students to build connected devices.

Understanding ESP32 Pins

-  Power pins (Red):
3.3V, 5V, GND
-  Ground pins (Black)
-  GPIO pins (Blue):
0-39
-  Special function pins (Green):
I2C, SPI, ADC, DAC



Simple Analogy

Pins are like USB ports - each has a specific job.

Key Pins Today

GPIO 4, 5, 13, 21, 22, 23

Pins can be input-only or support analog reading.

Digital vs Analog Signals

Understanding Data Flow in IoT Systems



Digital Signals



Binary States: 0 or 1

Discrete values for simple states.
Essential for triggers and binary conditions.

- 🖱 Buttons (Push/Toggle) - Instant Response
- 👁 IR Sensors - Object Detection
- 🔊 Buzzers - Audio Alert



Analog Signals



Continuous Range: 0 - 4095 (ESP32)

Smooth, varying values for precise measurements.
Used for sensing environmental conditions.

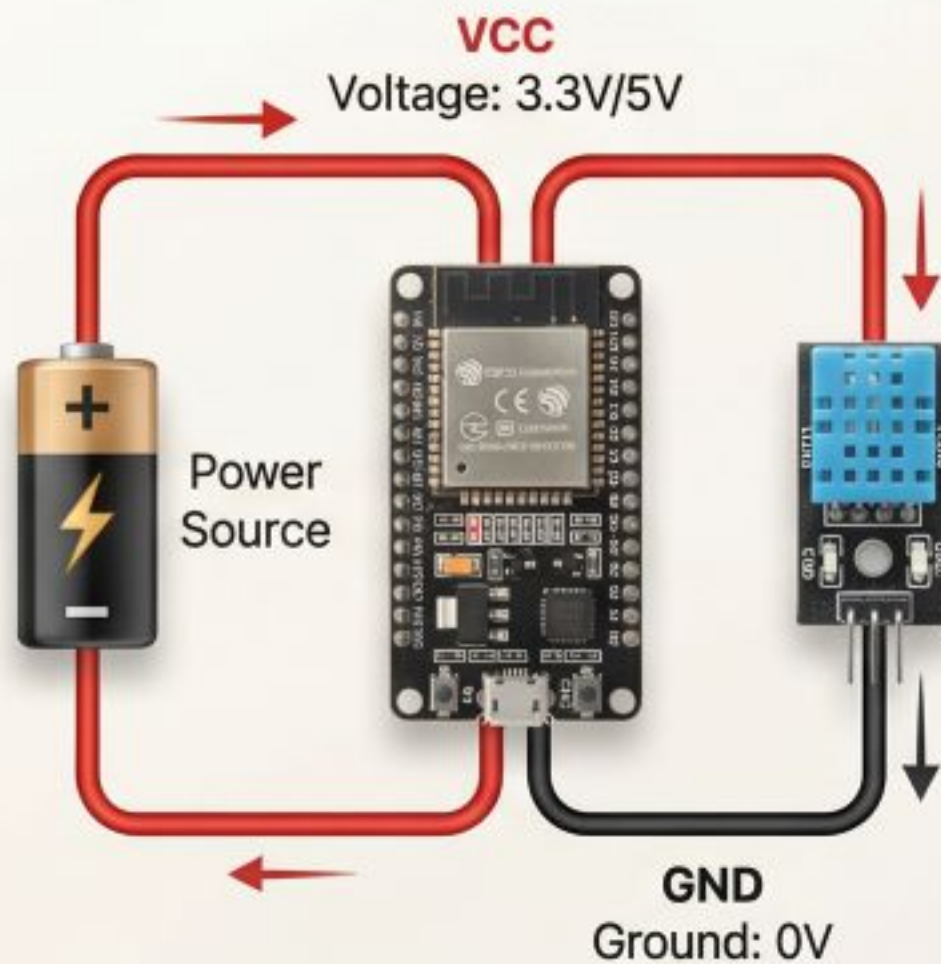
- 🌡 Temperature Sensors (e.g., DHT-11) - Precise Readings
- 💧 Moisture Sensors - Soil Humidity Level
- ☀ Light Sensors (LDR) - Ambient Brightness



ESP32 processes both
signal types for intelligent
decision making.

Power & Ground Basics

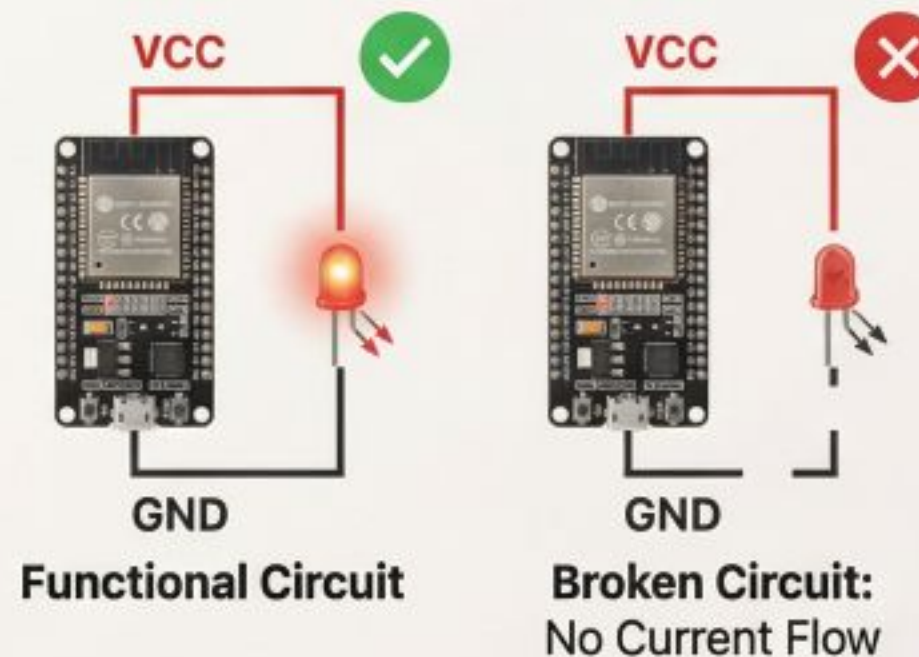
The Complete Circuit Flow



Electricity needs a complete **path** to flow from source to component and back to ground.

Why VCC & GND Matter

Every active component requires a voltage difference to operate. **VCC** provides the potential energy, and **GND** provides the reference point (0V) and return path.

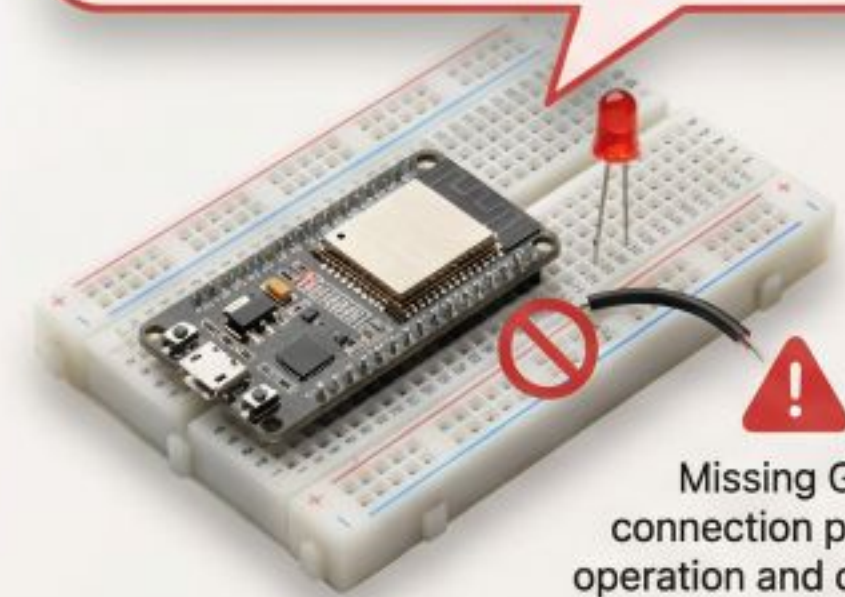


Think of it like water flowing down a hill; VCC is the top, GND is the bottom. No bottom, no flow!

Safety & Common Mistakes



Common Mistake:
"Complete the Circuit!"



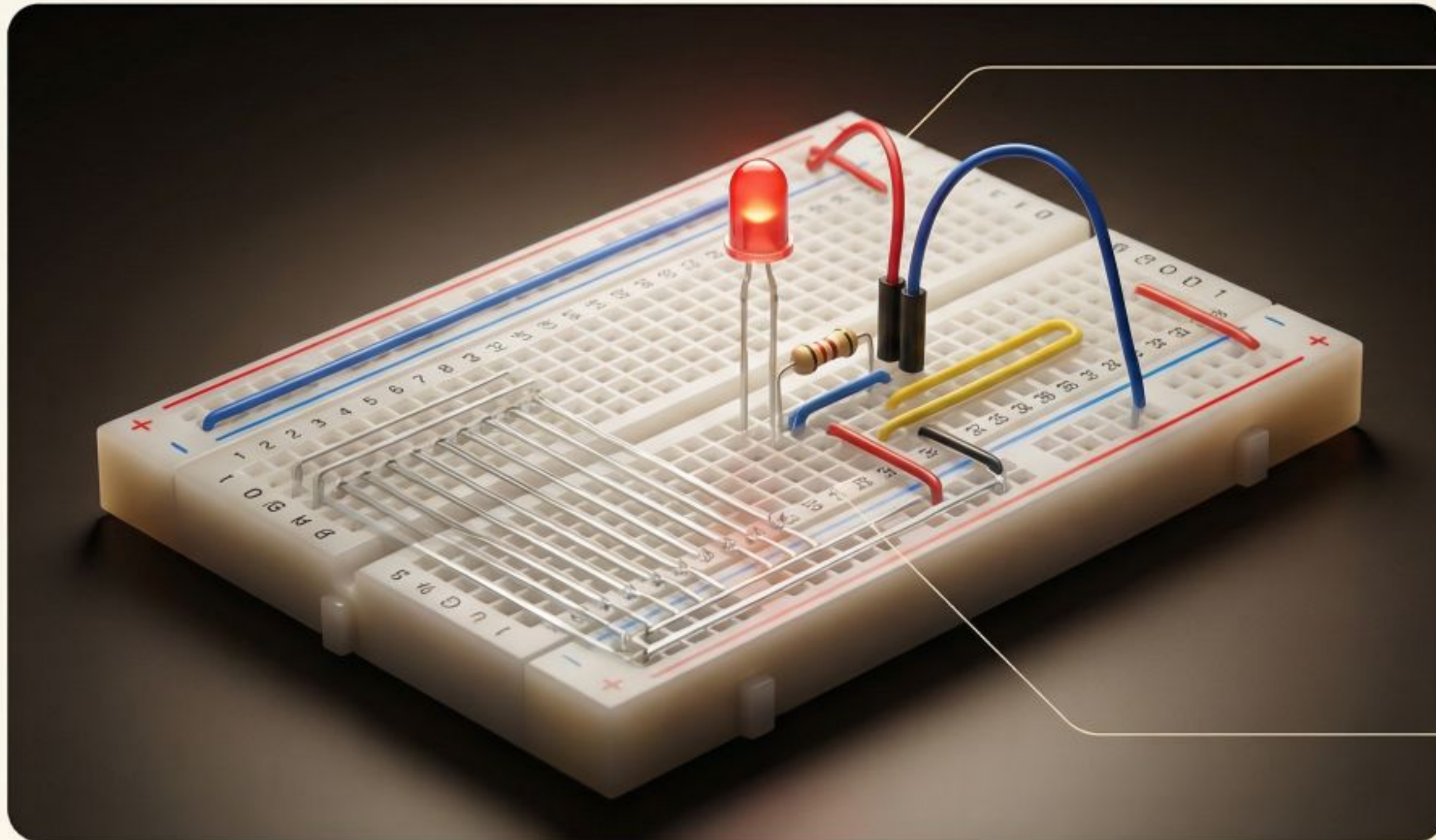
Missing GND connection prevents operation and can cause unpredictable behavior.



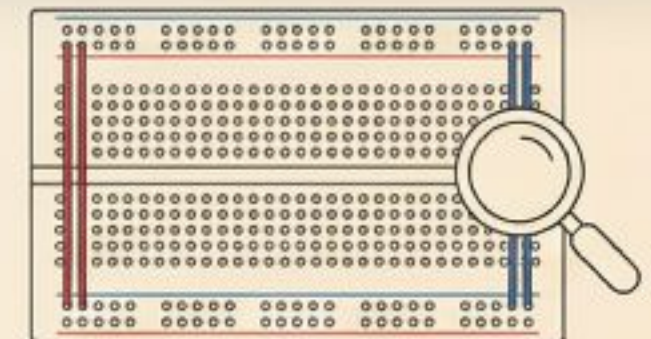
Safety Note: Always verify voltage levels (3.3V vs 5V) before connecting components. Incorrect voltage can damage your board or sensors.

Breadboard Magic

Hands-On IoT Workshop: From Sensors to Smart Systems



-  **INTERNAL STRUCTURE:** Horizontal power rails for VCC/GND, vertical component rows for connections.
-  **QUICK PROTOTYPING:** Enables rapid assembly without soldering.
-  **PRO TIP:** No soldering needed! Components plug directly.
-  **BEST PRACTICES:** Use color-coded wires, keep wiring organized, and double-check connections.



Safety First

Quick safety guidelines with friendly icons. Positive tone: "We're building, not breaking!"



ALWAYS CHECK CORRECT POLARITY BEFORE POWERING.
Prevent damage and short circuits.



NEVER FORCE CONNECTIONS THAT DON'T FIT.
Handle with care to avoid breakage.



DOUBLE-CHECK WIRING BEFORE APPLYING POWER.
Ensure all connections are secure.



KEEP WORKSPACE ORGANIZED.
A clean space creates a safe learning environment.



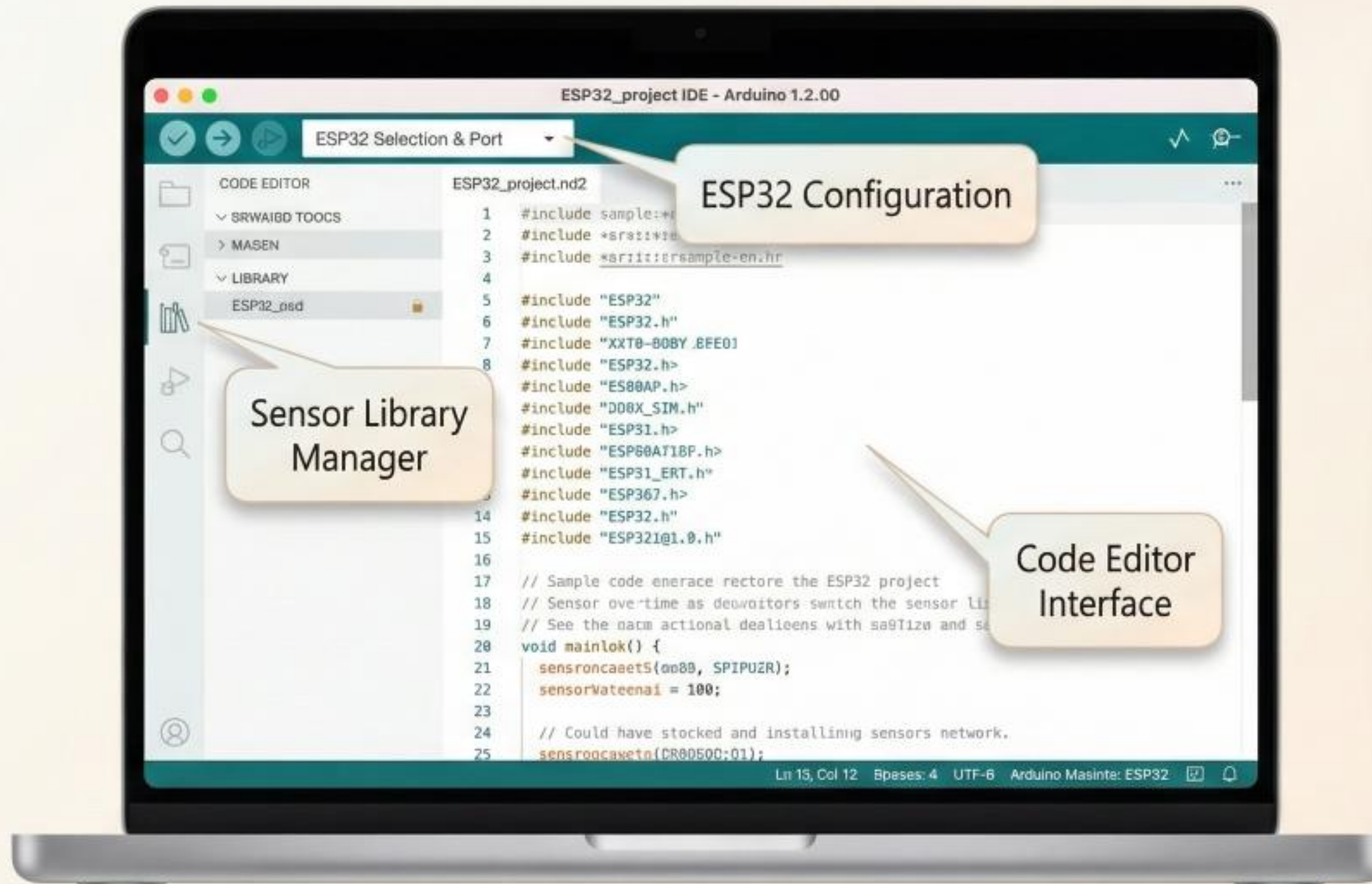
COMMON MISTAKES TO AVOID: Short circuits, Wrong voltage connections, Loose wires.



WE'RE BUILDING, NOT BREAKING! Emergency procedures: If something goes wrong, disconnect power immediately.

Tools You'll Use Today

Your coding companion for the next 3 hours. Get ready to build!



GitHub Repository Access

All project codes, diagrams, and resources. Scan the QR code to explore.



Serial Monitor: Debugging & Data

Crucial for real-time debugging, viewing sensor data, and verifying output.



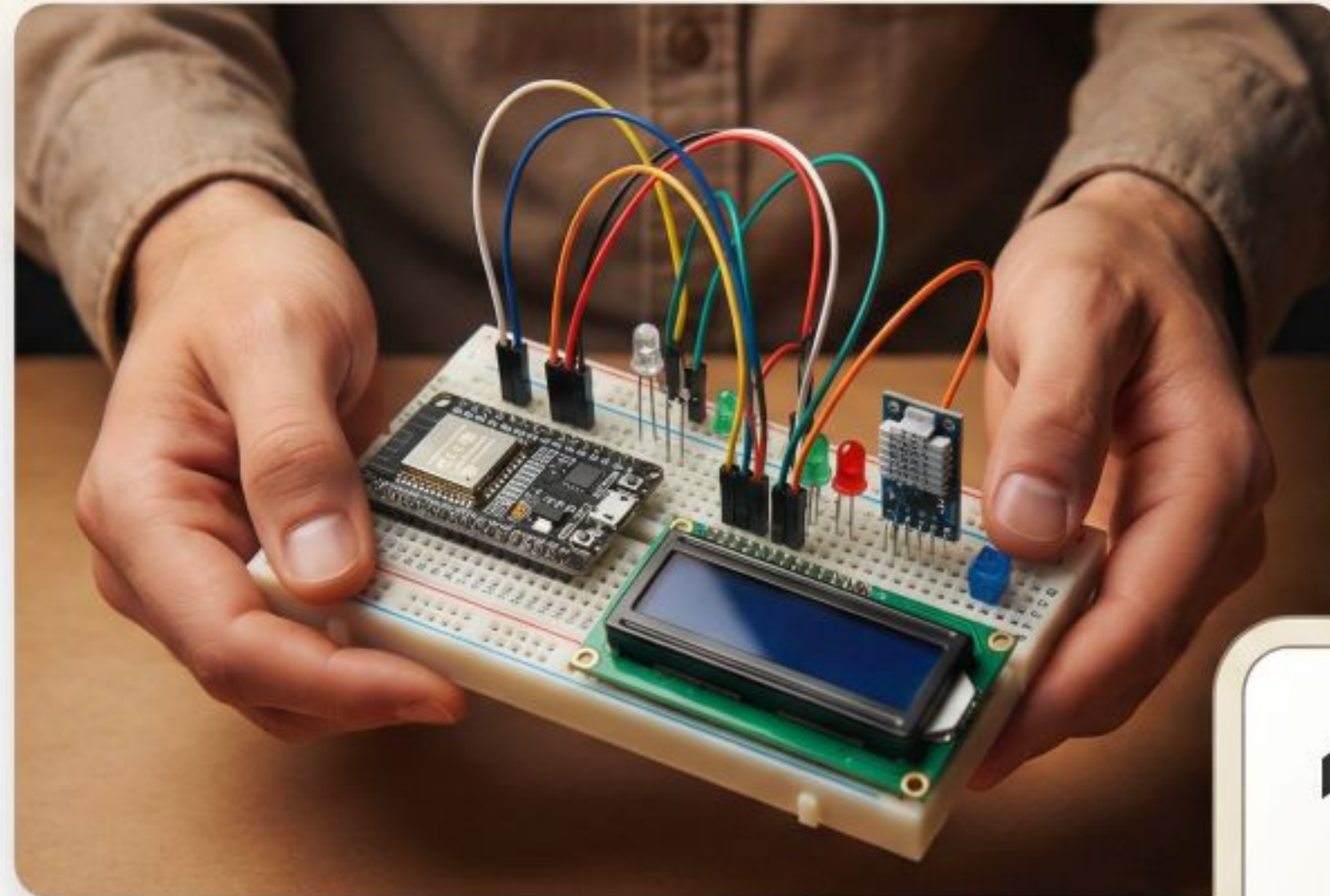
Installation Verification: Check your tools before we begin.

Let's Start Building!

7 Projects. 1 Amazing Journey. Let's Go!

Recap

- ✓ ESP32 Architecture
- ✓ Power & Ground
- ✓ Breadboard Basics



Next Up

Project 1: Hello LCD!
15-Minute Countdown

15:00

Hands-On Session Begins

Project 1: Your First Display - LCD 16x2

Master I2C Communication Protocol | 20 Minutes

Introduction & Applications

First hands-on project. Learn to use a screen in IoT. Real-world use in devices like:



Vending
Machines

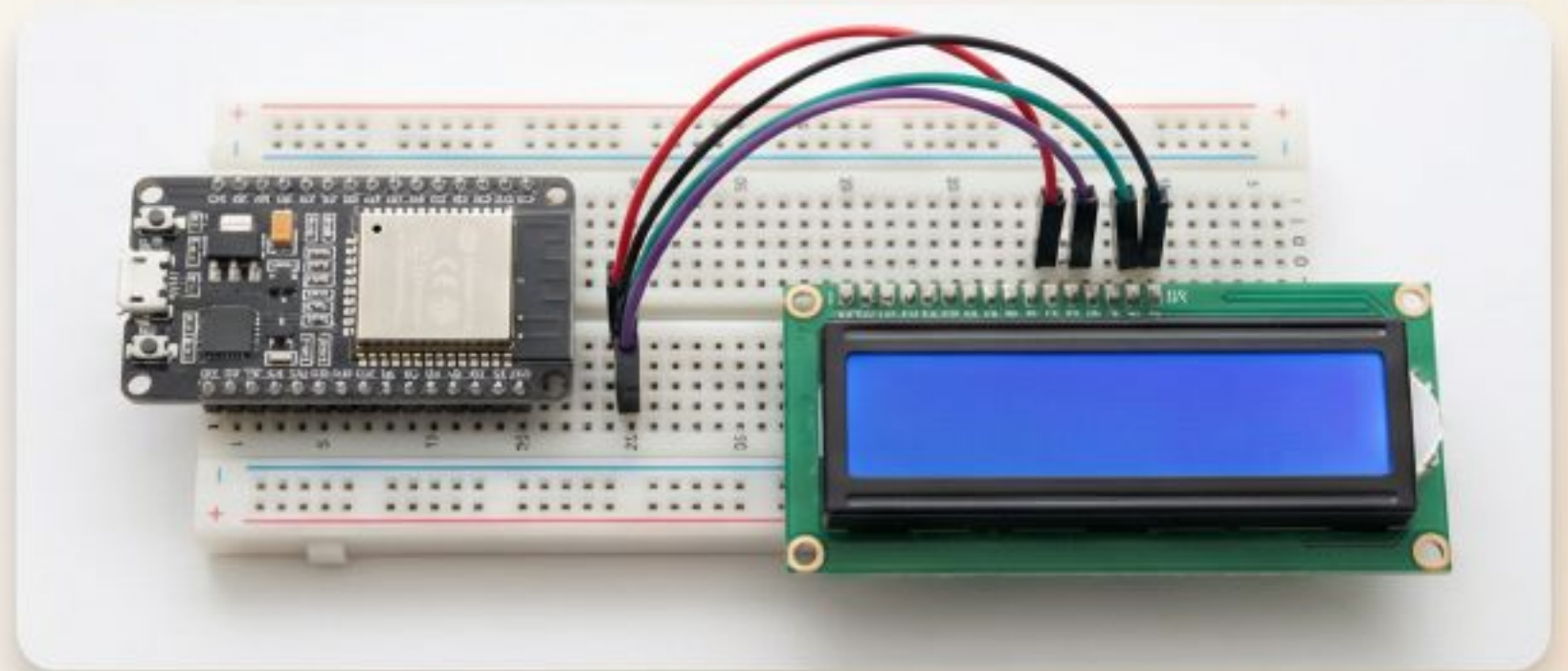


Medical
Devices



ATMs &
Control Panels

Hands-On Build: ESP32 + LCD



Expected Outcome



Time Allocation:
20 MINS

Meet the LCD 16x2 Display

16 columns × 2 rows
of characters

Hello, World!

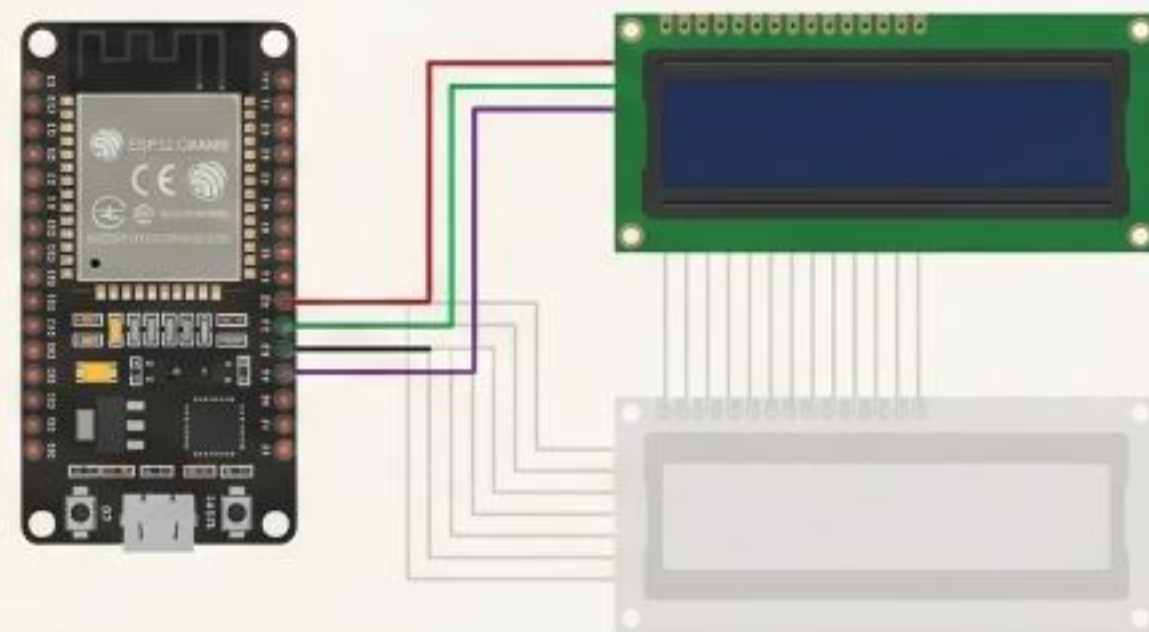
Backlight for visibility



⚡ Operating voltage 5V

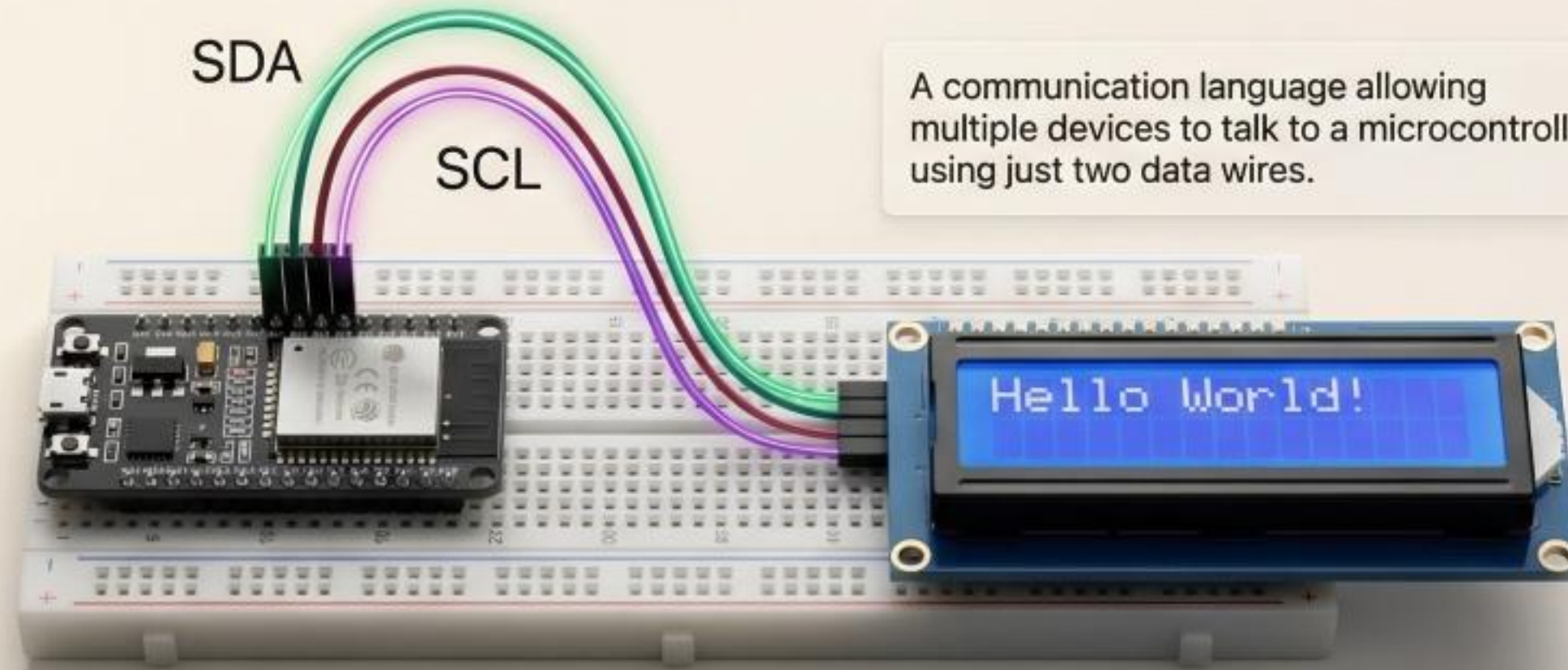
I2C interface
requiring only 4 wires

Why I2C Makes Connection Simple



- ✂ Only 4 pins needed (VCC, GND, SDA, SCL) instead of 12+ for parallel
- 🔧 Simplifies wiring, reduces errors
- 📄 Efficient data transfer for text, numbers, and custom characters

What is I2C Communication?



Analogy:

Like a conference call - only 2 wires for data, 2 for power.



Benefits:

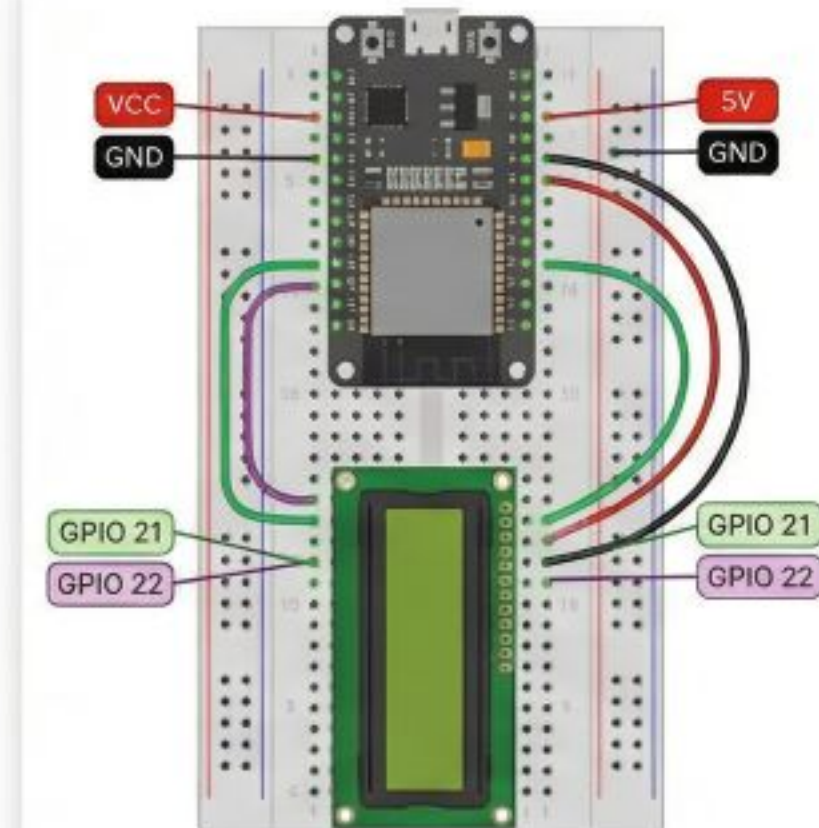
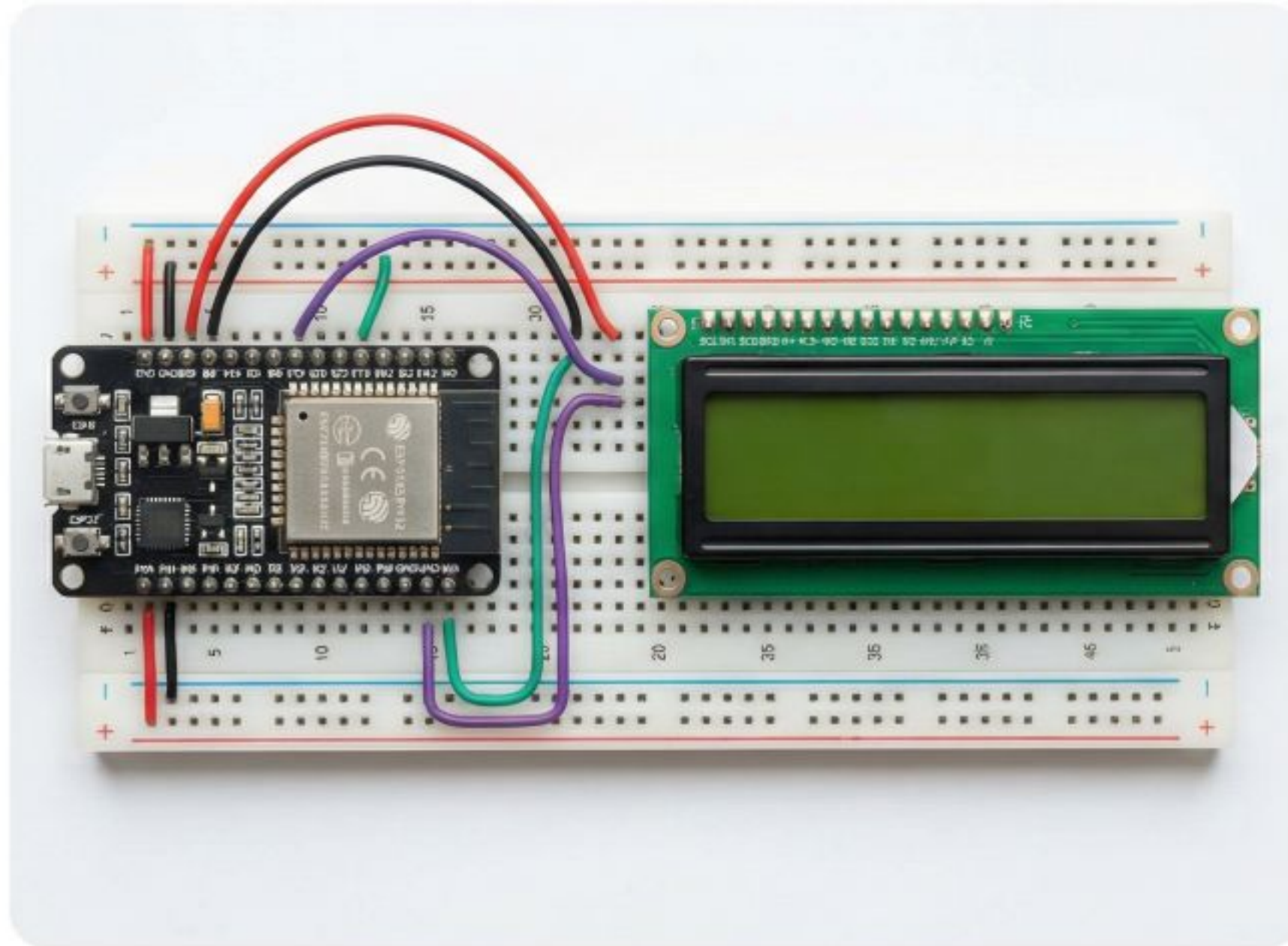
Fewer wires,
Simpler circuits.



Expandable system:

Connect multiple devices on the same bus.

Circuit Diagram - LCD Connection



Pin Connections

VCC → 5V (Red)

GND → GND (Black)

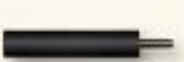
SDA → GPIO 21 (Green)

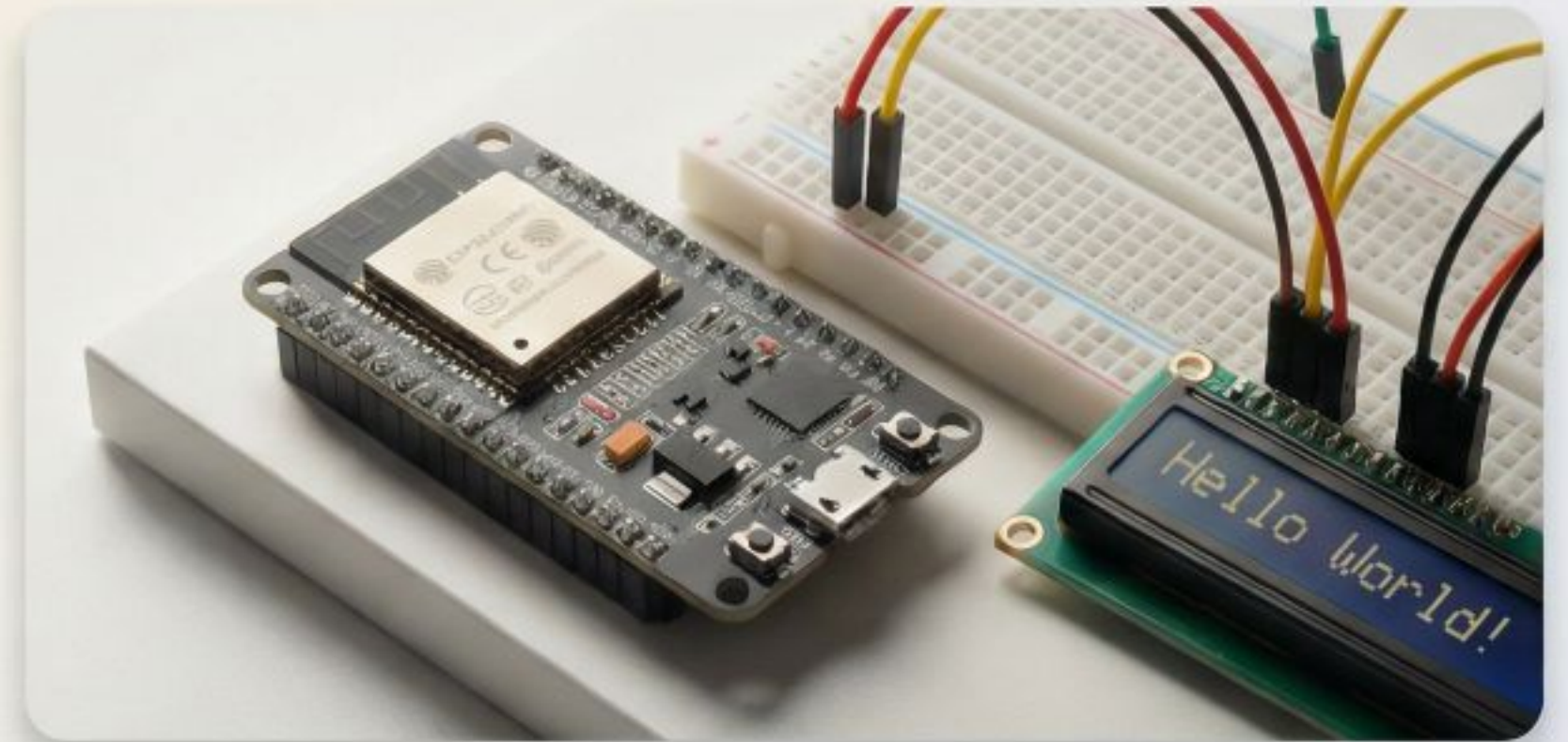
SCL → GPIO 22 (Purple)

Organized wiring on the breadboard for I2C communication.

Connection Checklist

Step-by-Step Connections

- ✓  Connect power VCC to 5V rail (Red)
- ✓  Connect ground GND to GND rail (Black)
- ✓  Connect SDA to GPIO 21 (Green)
- ✓  Connect SCL to GPIO 22 (Purple)



Tips & Verification

- 💡 Identify I2C pins on LCD module: SCL and SDA.
- 🔍 Verify all connections before powering.
Common mistakes: Incorrect polarity, loose wires, wrong GPIO pins.

The Code Walkthrough

```
// Library for LCD  
#include <LiquidCrystal_I2C.h>
```

Library Inclusion: Imports necessary tools for LCD communication.

```
// Set I2C address (usually 0x27) & display size  
LiquidCrystal_I2C lcd(0x27, 16, 2);
```

I2C Address Setup: Defines the display's unique address and dimensions.

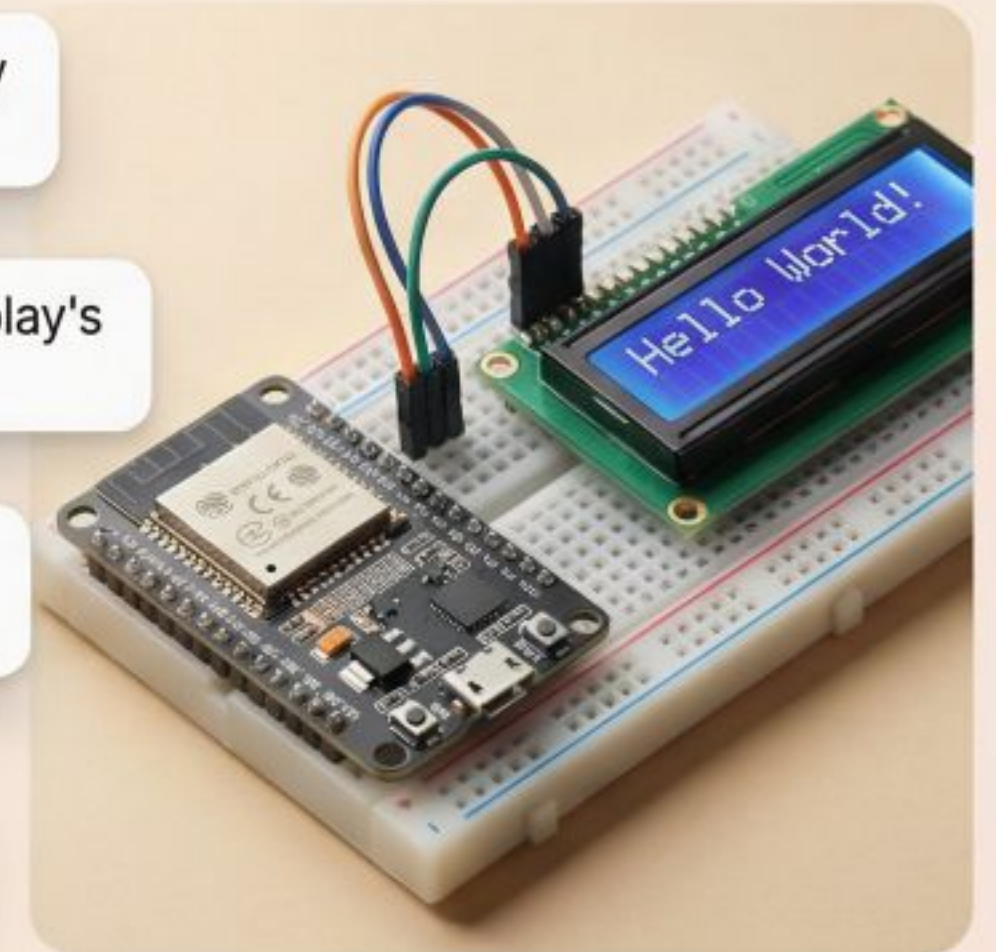
```
void setup() {  
  // Initialize the LCD and turn on backlight  
  lcd.init();  
  lcd.backlight();
```

Display Initialization: Prepares the screen and turns on the light.

```
  // Print a message to the LCD  
  lcd.print("Hello World!");  
}
```

Print Command: Sends the text you want to display.

```
void loop() { /* empty loop */ }
```

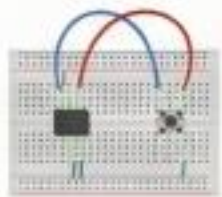


Don't worry about every detail – understanding comes with practice.



Find complete code on GitHub: github.com/eecsclubofficial/workshop_latency26

Hands-On Time!



1. Build the Circuit

Follow the diagram precisely.



2. Upload Code

Fetch from workshop repo.



3. See It Live!

Your first display!

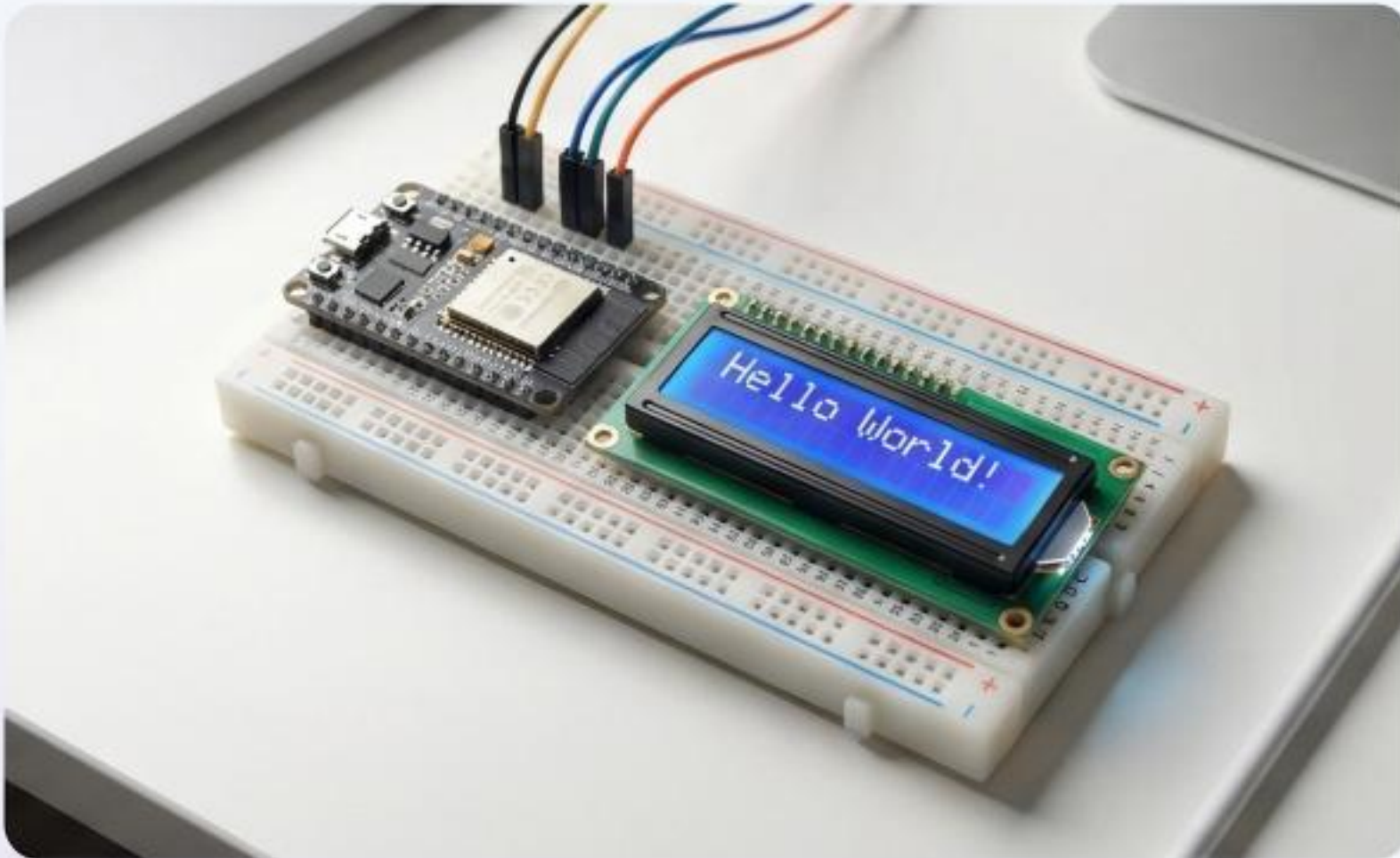


Need help? Raise your hand for instructors.

First-time builders, you've got this! Enjoy the process.

Success Checkpoint & Troubleshooting

What You Should See: Working LCD



The LCD should clearly display 'Hello World!' text. Celebrate your first successful project completion!

Challenge for fast finishers: Can you display your name or a custom message on the screen?

Troubleshooting Quick Guide



No display

Check I2C address in code and wiring.



Dim display

Adjust potentiometer on LCD back.



Garbled text

Check wiring connections and loose wires.



No power

Check power connections to 5V and GND.

Project 2: Feel the Environment – DHT-11 Sensor

Introduction to temperature and humidity sensing.



Real-World Applications



Weather
Stations



Smart
Thermostats



Greenhouse
Monitoring



Server Room
Cooling Systems

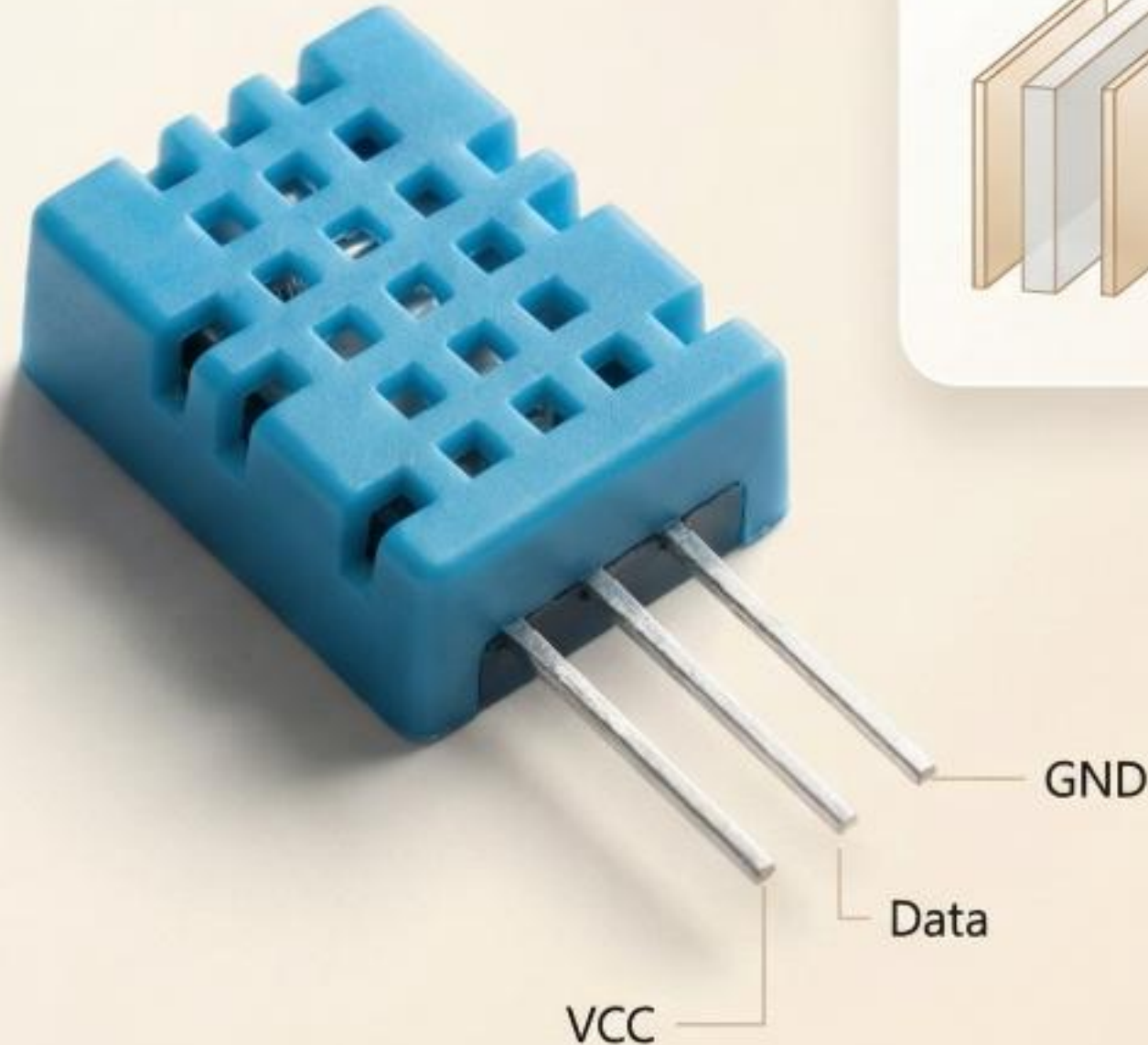
Learning Objectives

- Understand sensor data reading
- Analog vs Digital concepts
- Builds on Project 1 (LCD displays readings)

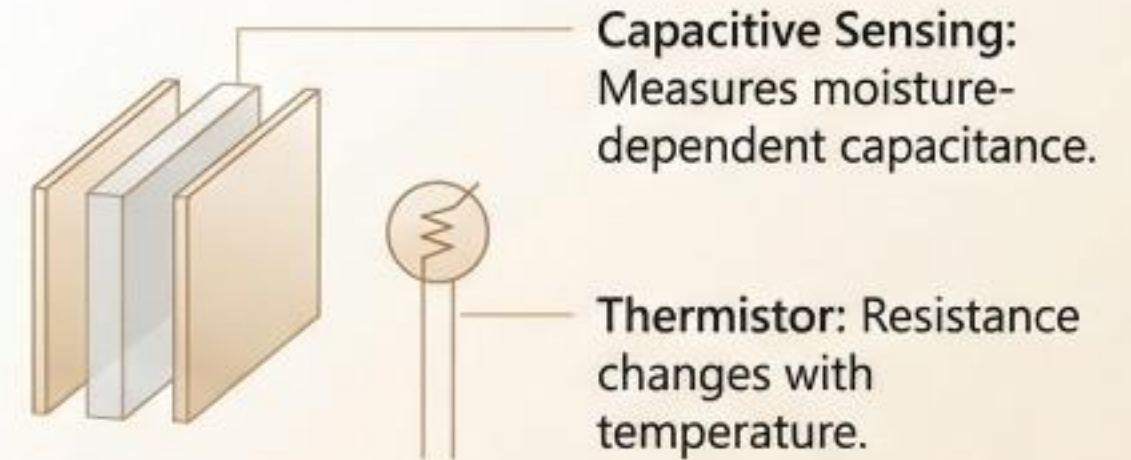
DHT-11 Sensor Deep Dive

What it Measures

-  **Temperature:** 0-50°C ($\pm 2^\circ\text{C}$ accuracy)
-  **Humidity:** 20-90% ($\pm 5\%$ accuracy)



How it Works



Fun Fact

-  Used in millions of smart homes worldwide for climate control.

Why Temperature & Humidity Matter



Agriculture: Crop health & specific ranges.



Healthcare: Critical storage conditions.



Data Centers: Preventing overheating failure.

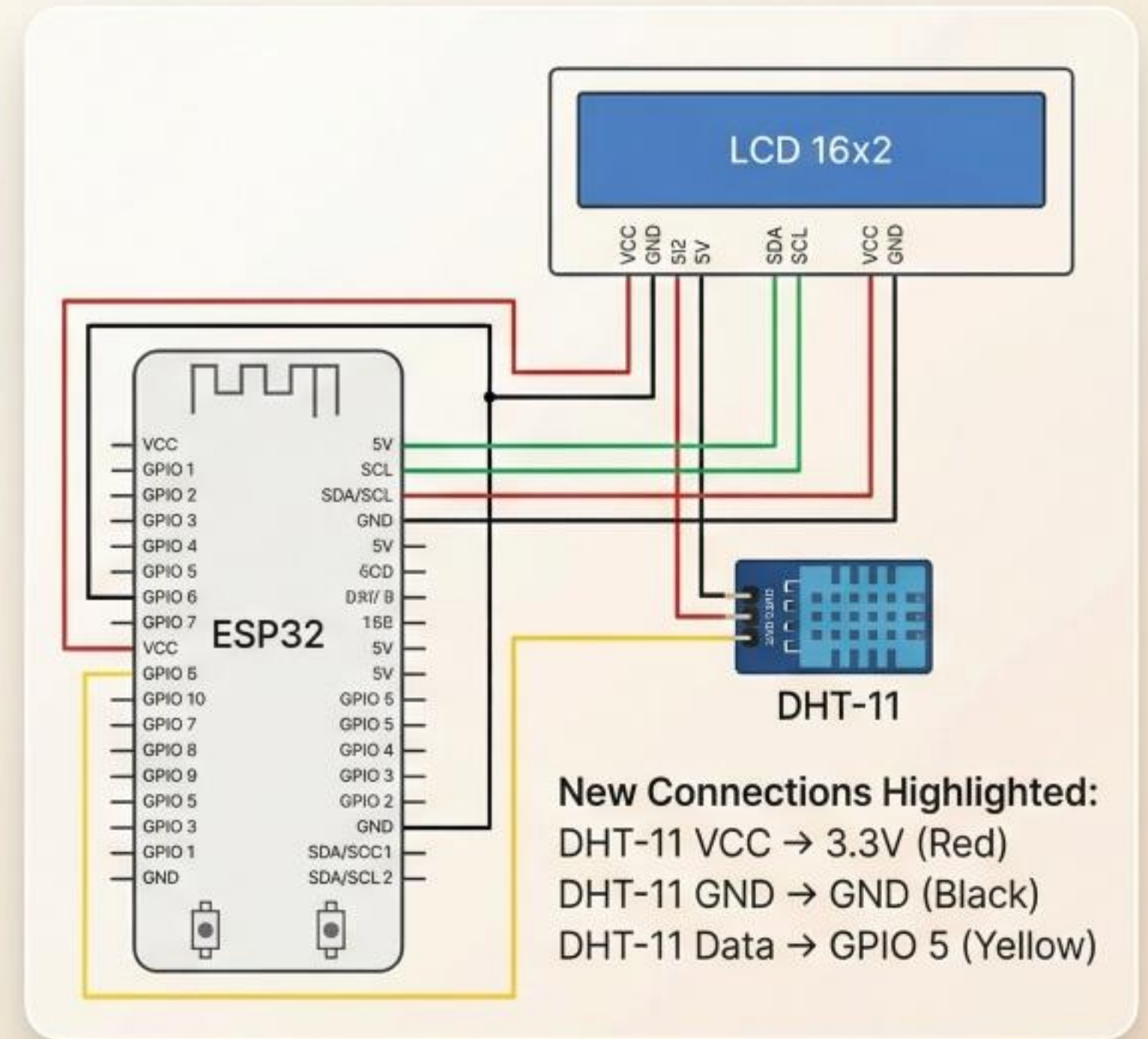
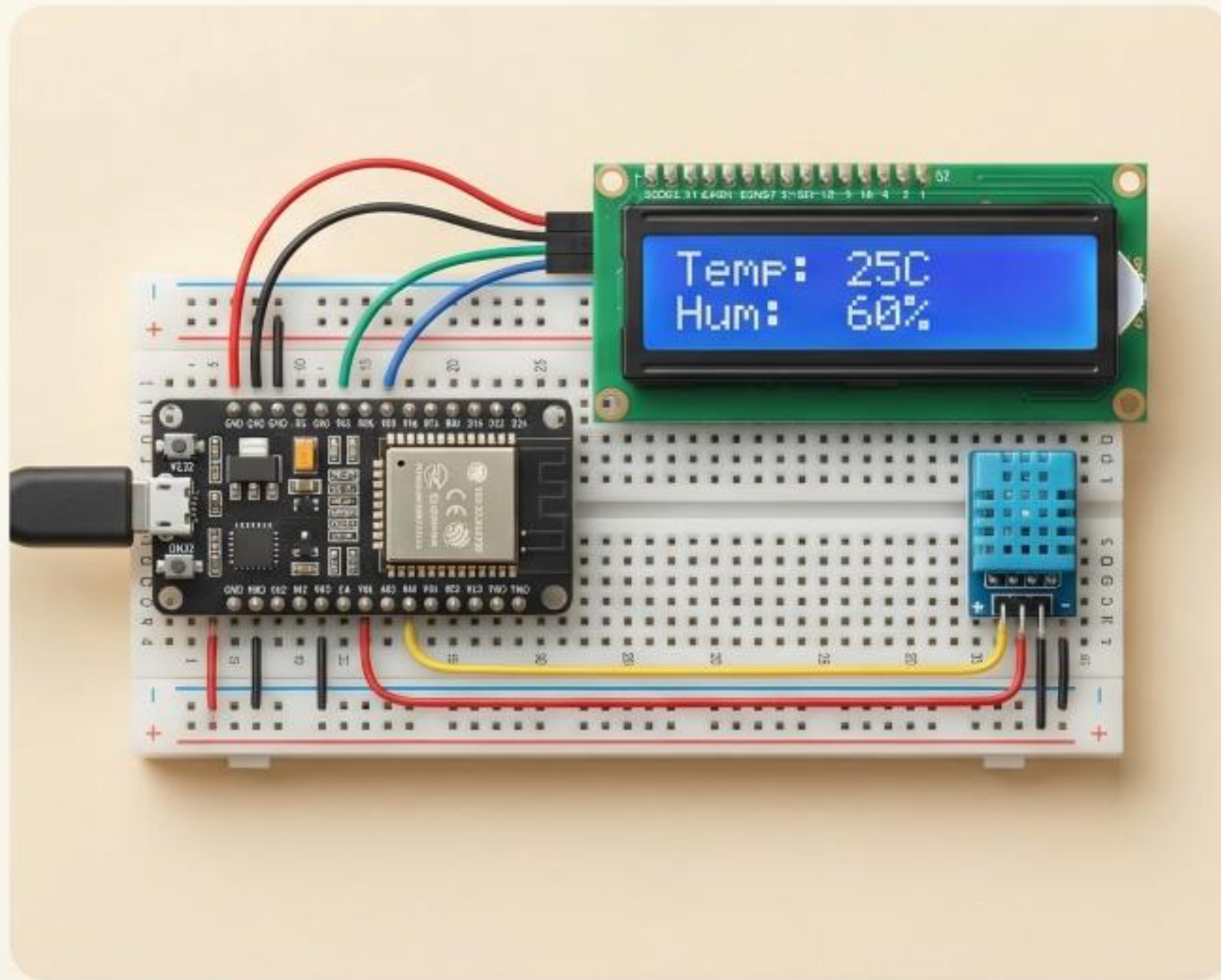


Food Storage & Transport: Quality preservation.

Sensing is the first step to control. →

Connect to Indian context with local examples.

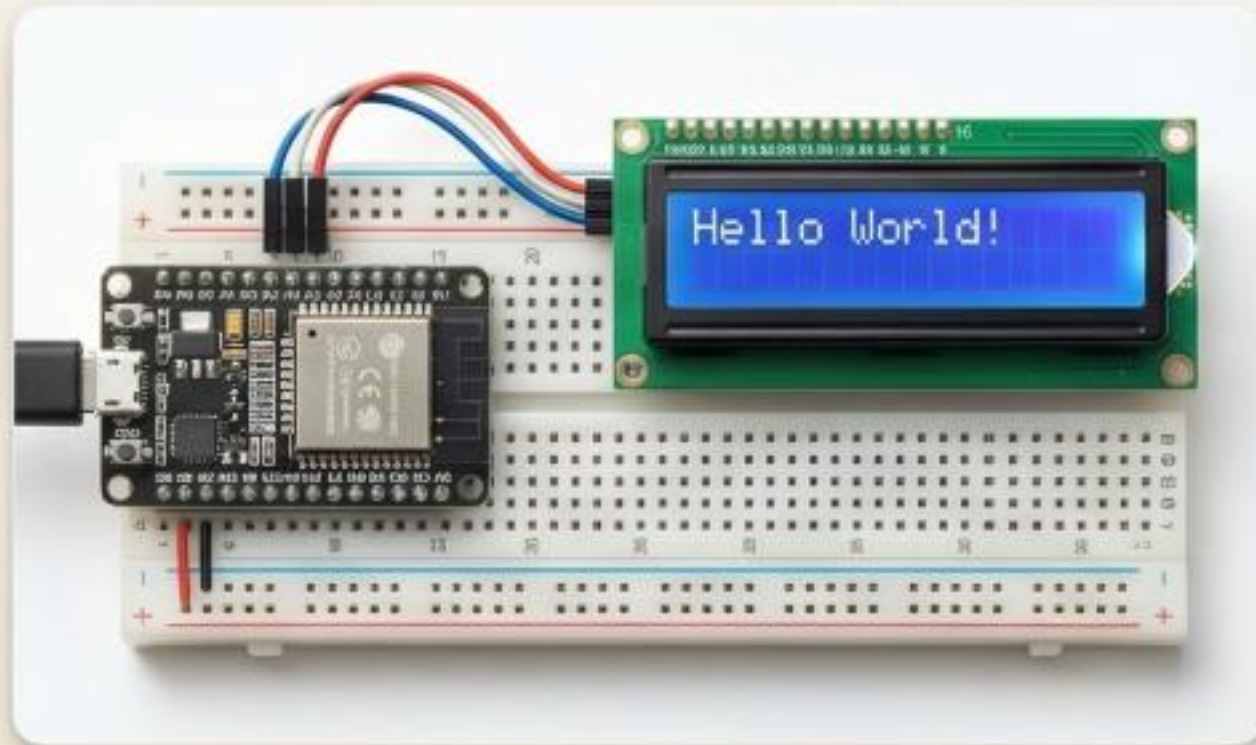
Circuit Diagram - DHT-11 + LCD



Building on Project 1. Keep LCD connected, add DHT-11.

Progressive Building Concept

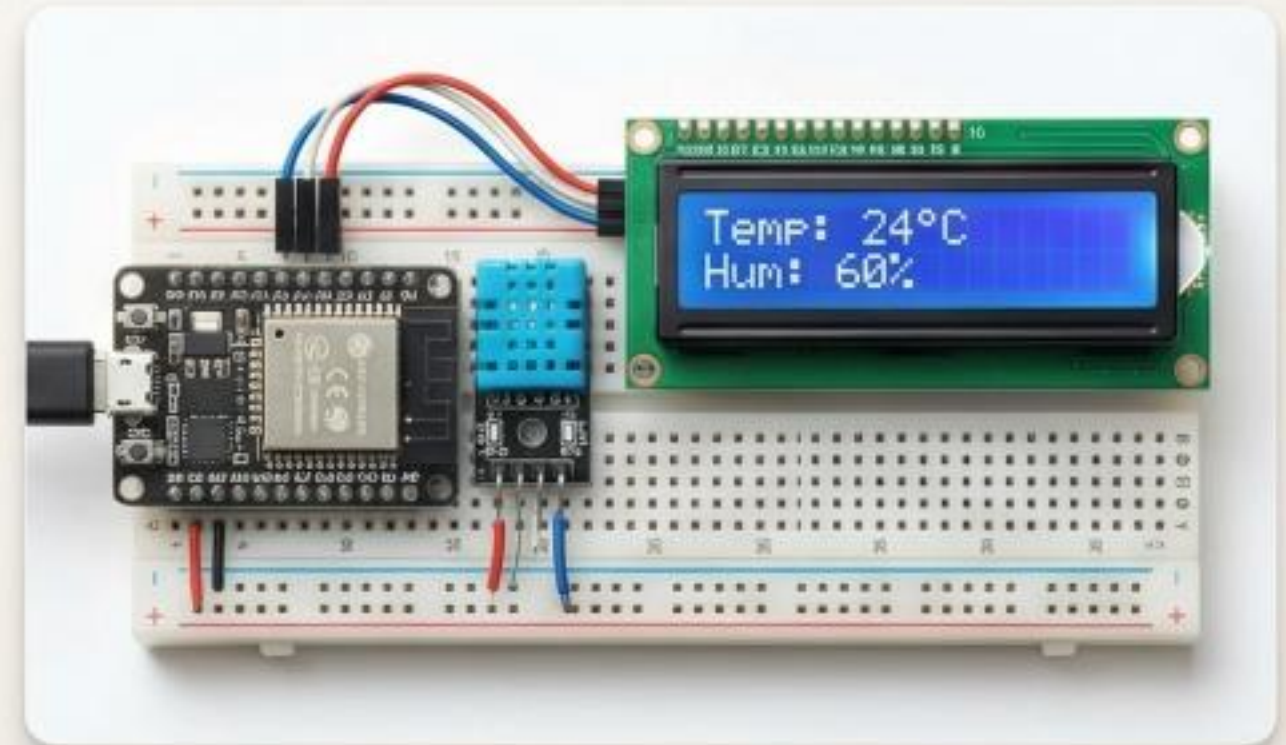
Project 1: Hello LCD!



Foundation: Master I2C display communication. Simple 4-wire setup for visual output.

Add Component
& Code

Project 2: Environment Monitor

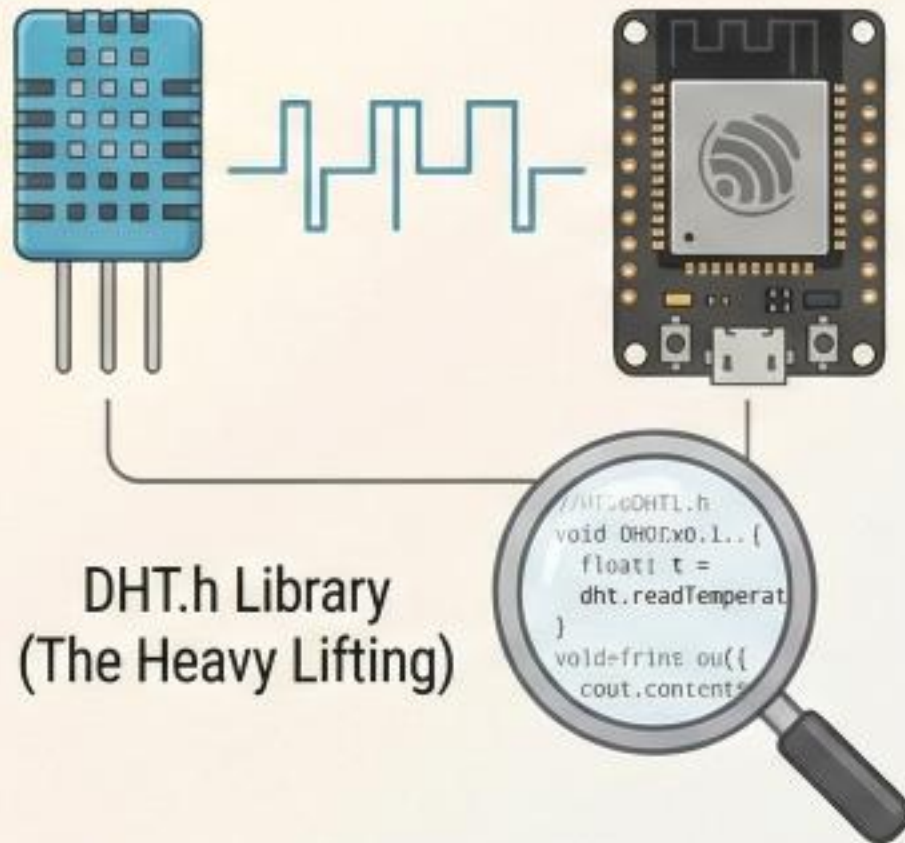


Evolution: "Keep LCD, Add DHT-11". Integrate sensor reading with existing display. Progressive complexity.

Strategic Placement for Organized Wiring
Build upon previous learning, step-by-step.

Understanding Sensor Data Reading

Digital Signal & Library



DHT.h Library
(The Heavy Lifting)

DHT-11 uses a specific digital signal protocol. The DHT.h library handles this complexity for you.

Reading Values (Your Job)

```
1. float t = dht.readTemperature();
```

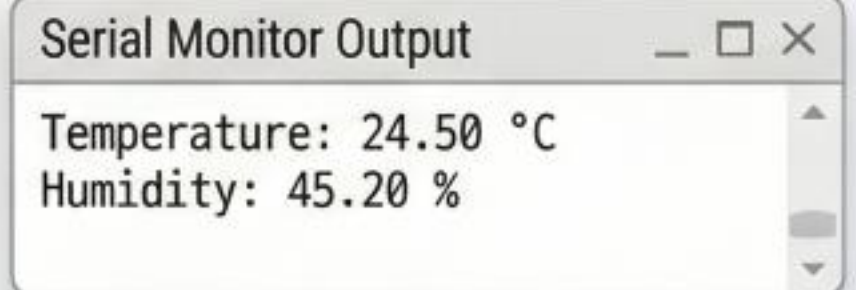
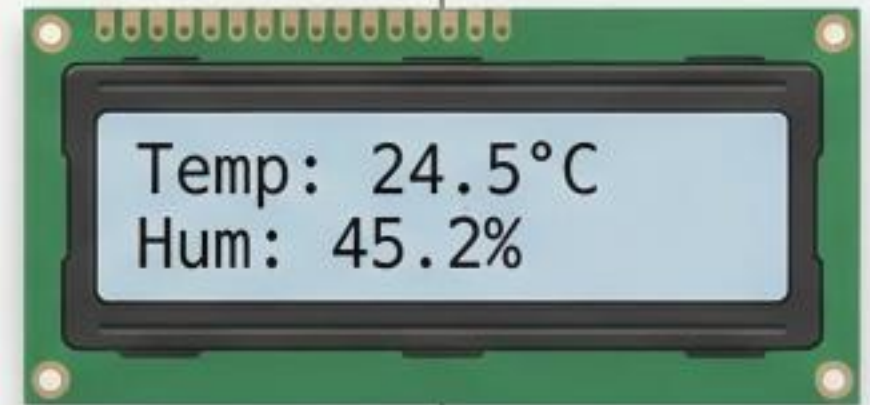
```
2. float h = dht.readHumidity();
```

Use simple functions to read temperature and humidity into variables.

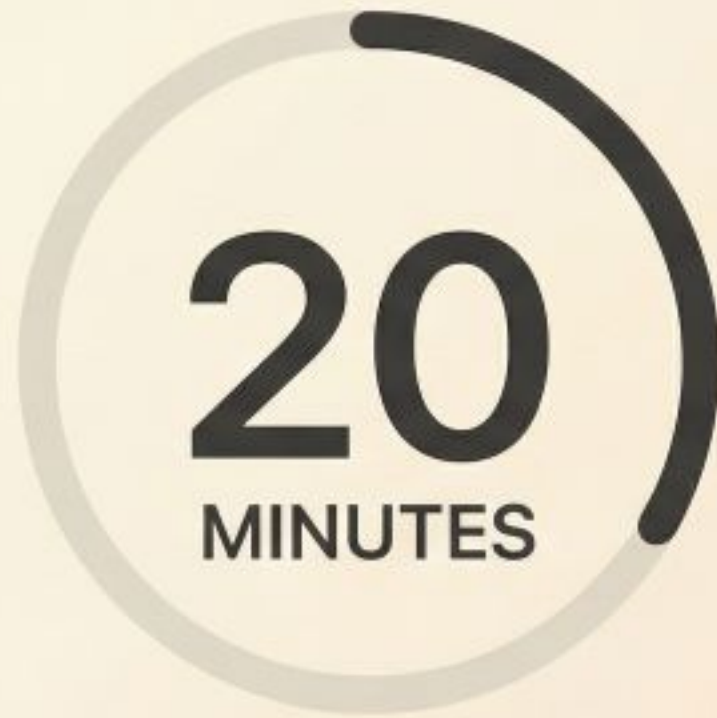
Temperature (°C)	float 24.5
Humidity (%)	float 45.2

Formatting & Display

```
lcd.print("Temp: " + String(t));
```

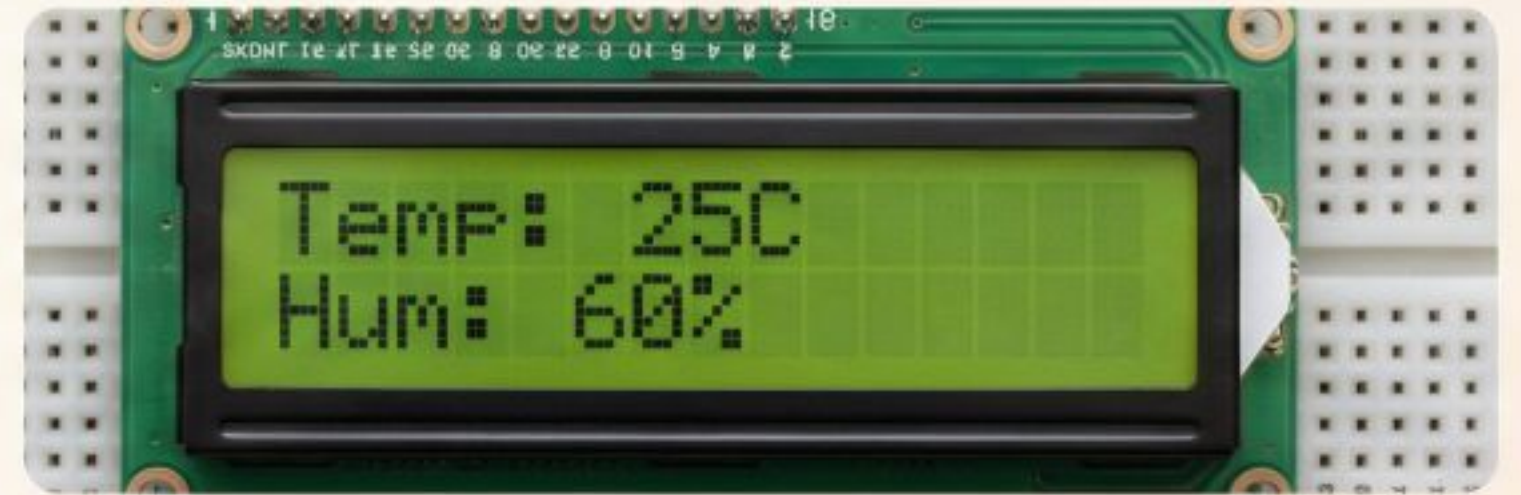


Hands-On Building & Testing



-  1. Add DHT-11 to existing LCD circuit
-  2. Upload DHT-11 code from GitHub
-  3. See temperature and humidity on LCD

Expected output example:



Testing Activity:

Breathe on sensor - watch humidity spike!

Verification Checklist:

- ☐ Code successfully uploaded
- ☐ Sensor values are reasonable & responsive