Instructor Guide – Module 1: Introduction to Microcontrollers

Learning Outcomes

By the end of this module, learners will be able to:

1. **Explain the role and core components of microcontrollers.**
2. **Identify popular microcontroller families and apply programming skills to build more complex projects.**
3. **Recognize real-world applications in robotics, smart devices, automotive systems, IoT, and medical technology.**

Lesson Flow

1. Warm-Up (5–10 minutes)

- **Instructor Guidance:** Begin by engaging students with a question: "What electronic devices do you use daily?"
- **Explanation:** Guide students to recognize that many of these devices rely on microcontrollers.
- **Key Concept:** Introduce the concept: "A microcontroller is the brain of many electronic devices.
- **Instructor Note:** Provide examples of everyday devices like smartphones, washing machines, and cars to illustrate the pervasive use of microcontrollers

2. Core Concepts (20 minutes)

- **Definition:** Present the definition: A microcontroller is a small computer on a chip, designed for specific tasks.
- **Explanation of Components**:
  - **CPU** → executes instructions.
    - **Instructor Guidance:** Explain that the CPU fetches instructions from memory and executes them, performing arithmetic and logical operations.

- **Memory (RAM & Flash)** → workspace vs. permanent storage.
    - **Instructor Guidance:** RAM is used for temporary data storage during program execution, while Flash memory stores the program permanently.
- **I/O Peripherals** → sensors, LEDs, communication.
    - **Instructor Guidance:** I/O peripherals allow the microcontroller to interact with the outside world, reading inputs from sensors and controlling outputs like LEDs.
- **Timers/Counters** → blinking LEDs, measuring button presses.
    - **Instructor Guidance:** Timers and counters are essential for tasks that require precise timing, such as controlling the duration of an LED blink or measuring the time a button is pressed.

- **Activity**: Match each component to a real-world function.
    - **Instructor Guidance:** Have students match components to functions, such as matching the CPU to "executing code" or I/O Peripherals to "reading sensor data."

3. Microcontroller Families (15 minutes)

- Introduce Arduino, ESP32, PIC, STM32.
    - **Instructor Guidance:**
        - **Arduino:** Beginner-friendly, large community support, easy-to-use IDE.
        - **ESP32:** Built-in Wi-Fi and Bluetooth, suitable for IoT projects.
        - **PIC:** Widely used in industry, diverse range of applications.
        - **STM32:** High performance, used in professional applications.
- Compare beginner-friendly vs. advanced options.
    - **Instructor Guidance:** Discuss the trade-offs between ease of use and performance capabilities.
- Show an Arduino Uno board (physical or image).
    - **Instructor Guidance:** Point out key features like digital and analog pins, USB port, and power connector.
- Discussion: Which family would be best for IoT projects?
    - **Instructor Guidance:** Lead a discussion on the factors to consider when choosing a microcontroller for IoT projects, such as connectivity, power consumption, and processing power.

4. Applications & Careers (15 minutes)

- Highlight uses in robotics, smart homes, automotive, IoT, medical devices.
    - **Instructor Guidance:** Provide examples of how microcontrollers are used in each of these areas.
- Group activity: Learners brainstorm devices around them that use microcontrollers.
    - **Instructor Guidance:** Encourage students to think critically about the devices they use every day and how microcontrollers might be used in those devices.
- Connect to career paths: embedded systems, automation, IoT development.
    - **Instructor Guidance:** Discuss the career opportunities available to students with microcontroller skills.

5. Hands-On Project: Temperature Sensor Display (30 minutes)

- **Overview:** Instead of just blinking an LED, students will connect a temperature sensor (e.g., LM35) to the Arduino and display the temperature reading on an LCD screen.
- **Demonstration:** Demonstrate circuit setup step by step: breadboard, LED, resistor, jumper wires, Arduino Uno.
    - **Instructor Guidance:** Explain the purpose of each component and how they are connected.
- Walk through Arduino IDE and code.
    - **Instructor Guidance:** Explain the code line by line, including how to read the temperature sensor, convert the reading to Celsius or Fahrenheit, and display it on the LCD.
- Learners replicate and upload program.
    - **Instructor Guidance:** Provide assistance to students as they build the circuit and upload the code.
- Troubleshooting tips: LED polarity, resistor placement, code errors.
    - **Instructor Guidance:** Help students troubleshoot common problems, such as incorrect sensor polarity, mis wired LCD connections, and code errors.

6. Wrap-Up & Reflection (10 minutes)

- Recap: definition, components, families, applications.
- Ask learners: *"What's one device in your home that likely uses a microcontroller and what does it do?"*
- Preview next module: sensors and actuators.