FuseGuard: Circuit Overload Protection System

Overview

FuseGuard is a smart circuit protection system that monitors current in real-time, detects overloads, and safely disconnects the load using a relay. It uses an ESP32 for control, an ACS712 current sensor for current measurement, and a relay module to cut off the load during overload conditions. The system also displays data on a 16x2 LCD and provides real-time updates via a local web server. A Python script can be used to monitor serial output, and a React web page displays live data.

Components Required

Component	Quantity
ESP32 DevKit v1	1
ACS712 Current Sensor	1
Relay Module (1-Channel)	1
16x2 LCD with I2C Module	1
LED / Buzzer (Optional)	1
220-ohm Resistor (for LED)	1
Breadboard & Jumper Wires	As needed
Power supply for load	1

Circuit Connections

ACS712 Current Sensor:

- VCC -> ESP32 3.3V (or 5V depending on module)

- GND -> ESP32 GND

- OUT -> ESP32 GPIO36 (Analog input)

Relay Module:

- VCC -> ESP32 5V

- GND -> ESP32 GND

- IN -> ESP32 GPIO23

- Relay COM -> Power Source Positive

- Relay NO -> Load Positive

- Load Negative -> Power Source Negative

LCD Display (16x2 with I2C):

- VCC -> ESP32 3.3V

- GND -> ESP32 GND

- SDA -> ESP32 GPIO21

- SCL -> ESP32 GPIO22

Optional LED/Buzzer:

- Positive -> ESP32 GPIO22 via 220-ohm resistor (LED)

- GND -> ESP32 GND

Software Components

Arduino Code

- Reads analog value from ACS712

- Converts to current

- Controls relay based on overload condition

- Displays data on LCD

- Hosts web server to serve JSON data

- Serial prints data for Python monitoring

Python Script

- Connects to ESP32 via serial

- Reads and prints all messages in terminal

React Page

- Fetches data from ESP32 local web server

- Displays current and status

ESP32 WiFi Setup

Replace in code:

```cpp
const char* ssid = "YOUR_WIFI_SSID";

const char* password = "YOUR_WIFI_PASSWORD";
```

Use Serial Monitor to view the IP address of the ESP32 once connected.

Access the data via: `http://<ESP32_IP>/data`

Web Page Access

React app fetches data from `http://<ESP32_IP>/data` and displays:

- Current in Amps

- Status: Normal / Overload

Python Serial Monitoring

Install `pyserial` and use the script to read live data:

```bash
pip install pyserial
```

Script

```python
import serial

PORT = "COM5"  # Replace with your port
BAUD = 115200

ser = serial.Serial(PORT, BAUD, timeout=1)
while True:
    line = ser.readline().decode('utf-8').strip()
    if line:
        print(line)
```

Notes

- Make sure all grounds are connected together

- Ensure ACS712 is rated for the current range you're measuring

- Double-check relay voltage rating for your load

Future Enhancements

- Cloud dashboard (Blynk, Firebase)

- Mobile app integration

- Data logging to SD card or cloud

- SMS/Email alerts for overload

Project by: [Your Name Here]

Date: [Submission Date]