Web-Controlled AC Fan/Light Dimmer using Arduino

A simple yet powerful IoT project that demonstrates how to safely control and dim a 220V AC light bulb or control the speed of a ceiling fan from a local webpage. This project uses an Arduino to handle the precise timing required for phase control dimming/speed of AC home appliances, with a Python Flask server acting as a bridge between the web interface and the Arduino.

Features

- **Web-Based Control:** A clean, modern user interface to control the light from any device on the same local network.
- **Three Power Modes:**
  - **ON:** Full brightness.
  - **HALF:** Dims the light to approximately 50% power.
  - **OFF:** Safely turns the light/fan off.
- **Phase Control Dimming:** Utilizes a zero-cross detection circuit and a TRIAC for smooth and efficient dimming of AC loads.
- **Safe High-Voltage Isolation:** Optocouplers are used to ensure the low-voltage Arduino is safely isolated from the high-voltage AC mains.

Technology Stack

- **Hardware:** Arduino Uno (or compatible), TRIAC (BTA16), Zero-Cross Detector (LTV814), TRIAC Driver (MOC3021), Resistors, Safety Capacitor.
- **Arduino:** C++
- **Web Server:** Python with Flask
- **Communication:** Python pyserial library for USB communication.
- **Frontend:** HTML with Tailwind CSS for styling.
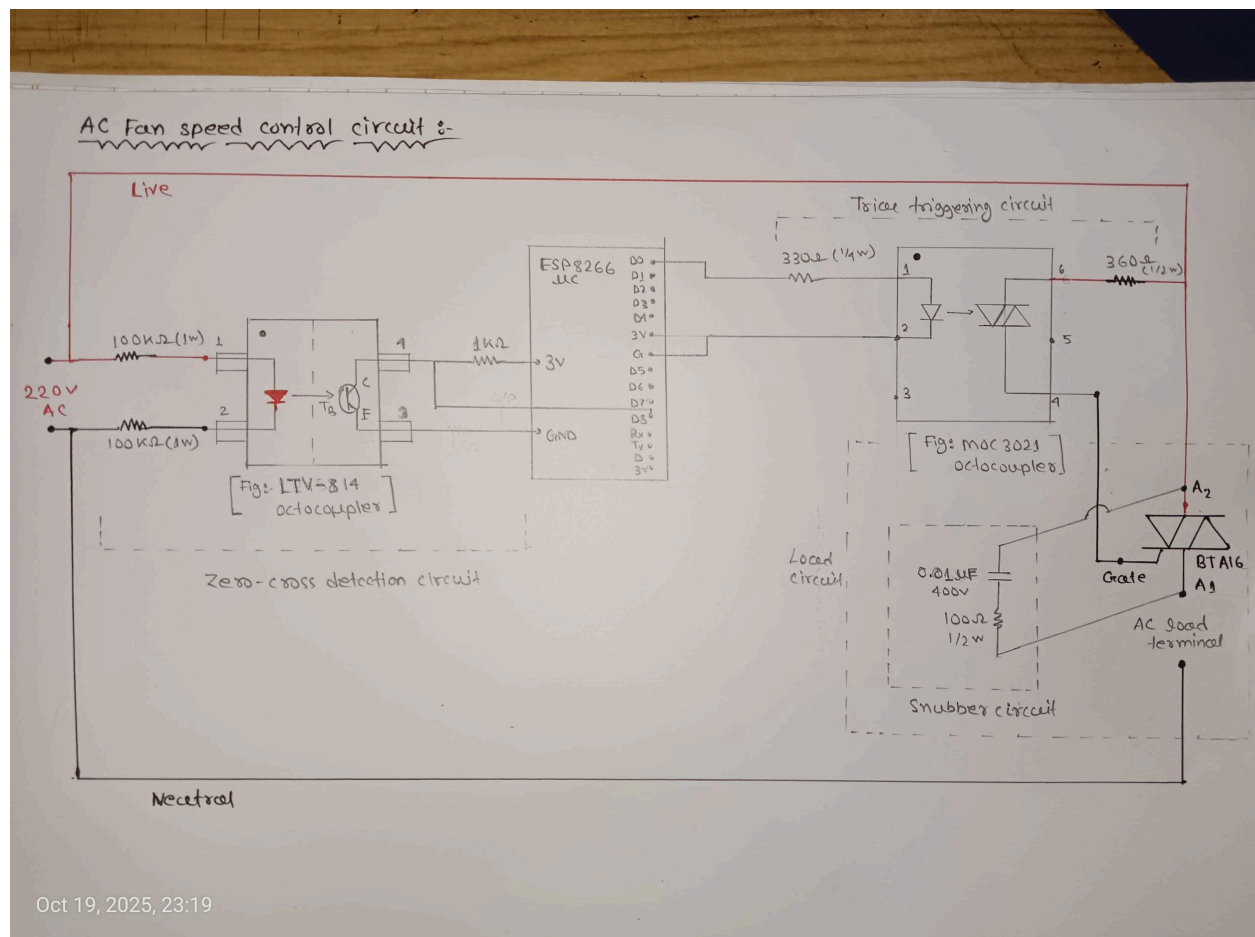
Hardware and Circuit Diagram

Required Components

- 1x Arduino Uno or Nano
- 1x BTA16 TRIAC
- 1x LTV814 Optocoupler (for Zero-Cross Detection)
- 1x MOC3021 Optocoupler (for TRIAC Driving)
- 2x 100kΩ Resistor (1 Watt)
- 1x 1kΩ Resistor (1/4 Watt)
- 1x 330Ω Resistor (1/4 Watt)

- 1x 360Ω Resistor (1/2 Watt)
- 1x 100Ω Resistor (1 Watt - for snubber)
- 1x 0.1µF (100nF), 275V AC **X2-Rated Safety Capacitor** (for snubber)
- An AC light bulb and holder or ceiling fan with the required value of capacitor.
- Connecting wires

    -note that the capacitor used in the actual snubber circuit is of 0.01uf because of lack of components and the resistors in parallel and series connection make up the 100 ohm 1 watt resistance.

Circuit Diagram

This is the hand drawn circuit diagram. This diagram shows the complete circuit, including the zero-cross detector, the TRIAC driver, and the snubber circuit, all connected to the Arduino.

Hardware Details & Theory of Operation

Pin Connections

This table details the critical connections between the components.

From Component	From Pin	To Component	To Pin	Purpose
LTV814 (Zero-Cross)	Pin 1	AC Live (via 100kΩ Resistor)	-	AC Input for detector
LTV814	Pin 2	AC Neutral (via 100kΩ Resistor)	-	AC Input for detector
LTV814	Pin 3 (Emitter)	Arduino	**GND**	Ground reference
LTV814	Pin 4 (Collector)	Arduino	**Pin 2**	Zero-cross signal output
Arduino	**5V**	1kΩ Resistor	Leg 1	Pull-up resistor power
1kΩ Resistor	Leg 2	LTV814	Pin 4	Pulls Pin 2 HIGH at zero-cross
MOC3021 (TRIAC Driver)	Pin 1	Arduino (via 330Ω Resistor)	**Pin 4**	Receives trigger pulse
MOC3021	Pin 2	Arduino	**GND**	Ground reference
MOC3021	Pin 4	BTA16 TRIAC (via 360Ω Resistor)	**Gate**	Delivers trigger to TRIAC
MOC3021	Pin 6	AC Live	-	Powers the MOC3021's internal TRIAC
BTA16 TRIAC	A2	AC Live	-	Main AC terminal
BTA16 TRIAC	A1	AC Load (Light Bulb)	-	Main AC terminal

Physical circuit:

Here is a view of the circuit and components in real life-
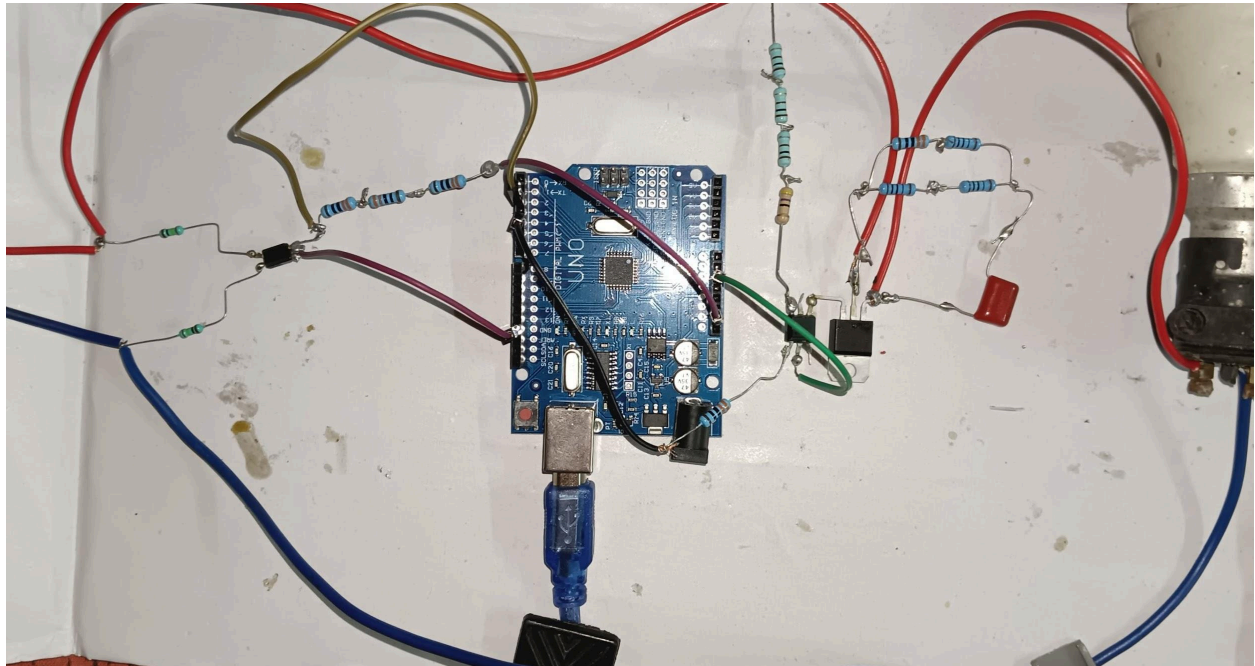
Core Concepts for this project:

What is a TRIAC?

A **TRIAC** (Triode for Alternating Current) is a semiconductor component that acts as a high-power switch for AC electricity. Unlike a simple switch, it can be turned on by a small electrical pulse from a microcontroller. Once triggered, it stays on, allowing current to flow until the current drops to zero (which naturally happens at the end of each AC half-cycle). This ability to be turned on at a precise moment makes it perfect for dimming. In our circuit, the **BTA16** is the main TRIAC that switches the power to the light bulb.

The Zero-Cross Detection Circuit

To dim an AC light, we can't just lower the voltage. Instead, we use a technique called **phase control**, where we turn the light on for only a fraction of each AC half-wave.

- **The Problem:** To do this, the Arduino needs a timing reference. It must know the exact moment the AC wave crosses zero volts so it can start its timer.
- **The Solution:** The **zero-cross detection circuit** provides this reference.
  - The two 100kΩ resistors safely limit the high AC voltage.
  - For most of the AC cycle, current flows through the resistors and turns on the internal LED of the **LTV814 optocoupler**.
  - This light activates the LTV814's internal phototransistor, which pulls the Arduino's Pin

2 LOW.
- At the exact moment the AC voltage crosses zero, the internal LED turns off, the phototransistor turns off, and the external 1kΩ pull-up resistor pulls Pin 2 HIGH.
- The Arduino detects this LOW-to-HIGH signal as an interrupt, which tells it "The AC wave just crossed zero. Start your timer now!"

The Snubber Circuit

- **The Problem:** Fan motors and even the wiring to light bulbs are **inductive loads**. When you abruptly stop the flow of current through an inductor (by the TRIAC turning off at the end of an AC cycle), the inductor's magnetic field collapses and creates a large, sudden voltage spike. This spike can be high enough to falsely re-trigger the TRIAC or even damage it.
- **The Solution:** A **snubber circuit** is a simple filter designed to absorb this spike.
  - It consists of a resistor (100Ω) and a capacitor (0.1μF) connected in parallel with the TRIAC.
  - The capacitor provides a path for the high-frequency voltage spike to bypass the TRIAC, absorbing its energy. The resistor limits the current flow through the capacitor.
  - It is **critical** to use an **X2-rated safety capacitor** here, as it is designed to fail safely (open circuit) when connected directly across the AC mains, preventing a fire hazard.

Software Setup and Installation

Step 1: Program the Arduino

1. **Code:** Use the code from Arduino_Dimmer_Sketch.ino.
2. **Upload:** Connect your Arduino and upload the sketch.
3. **Close IDE:** After uploading, **completely close the Arduino IDE** to free up the serial port.
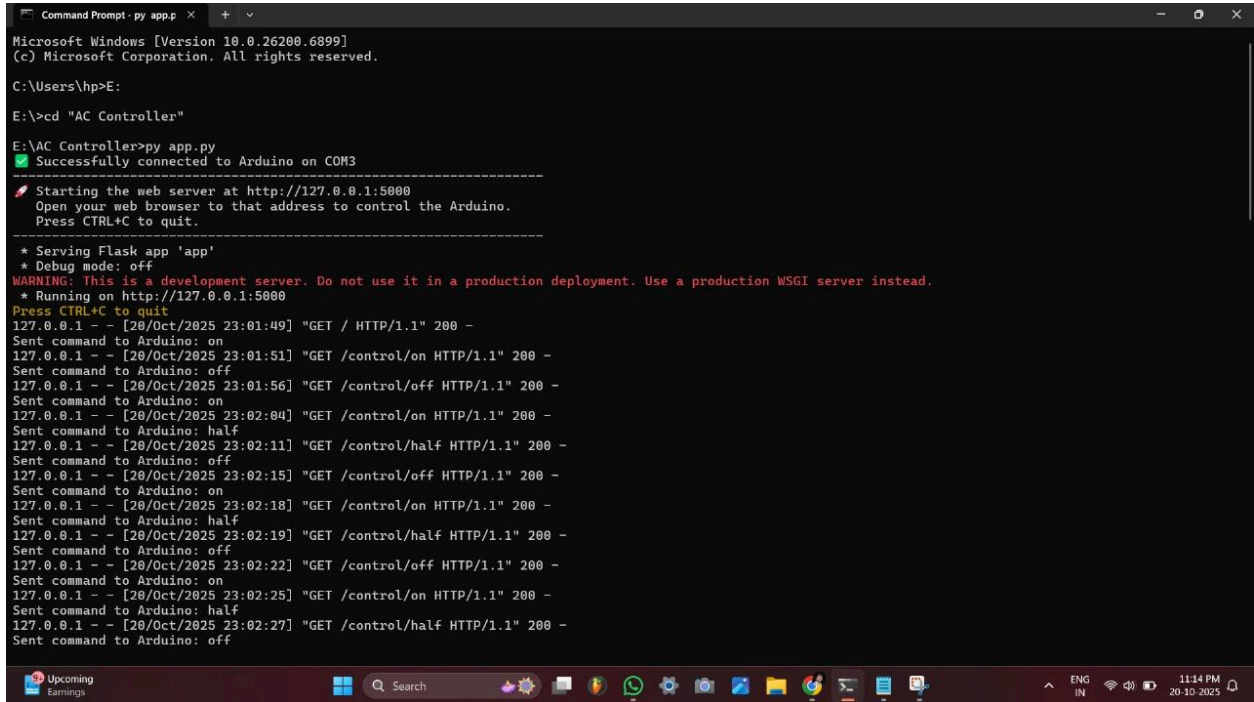
Step 2: Set Up the Python Web Server

The arduino, python server and website(html) code is given in the github page.

1. **Install Libraries:** Open a command prompt and run:
   pip install Flask pyserial

2. **File Structure:** Organize your project folder exactly as follows:
   AC_Controller/
   ├── app.py
   └── templates/
       └── index.html

How to Run

1. **Connect Arduino:** Connect the circuit and Arduino to your computer via USB.
2. **Configure Port:** Open app.py and change SERIAL_PORT to match your Arduino's port.
3. **Start Server:** Open a terminal, navigate to the AC_Controller folder, and run:
   py app.py

This screenshot shows exactly what should be displayed in the terminal after a successful connection.

4. Control the Light or fan: Open a web browser on your computer or any device on the same Wi-Fi network and go to:
5. http://127.0.0.1:5000

The web interface will load, and you can now control your light bulb or fan!
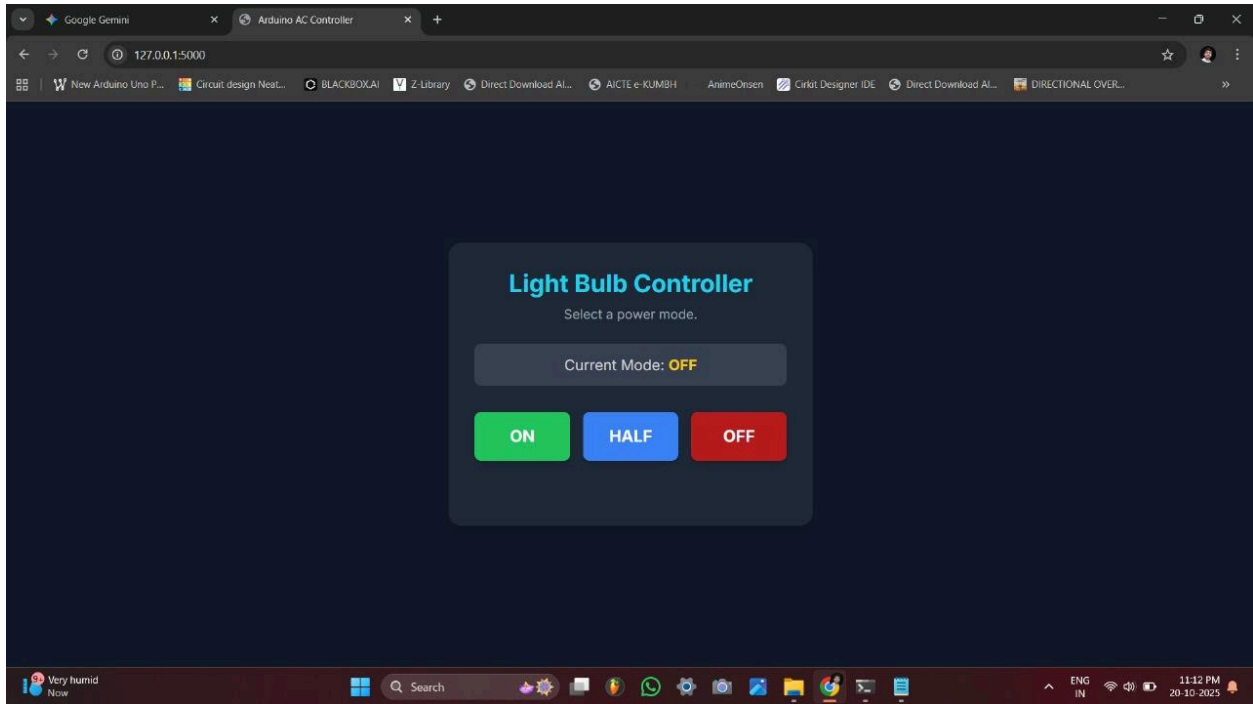
in the above screenshot, the website design and control buttons are displayed.

⚠️ HIGH VOLTAGE WARNING ⚠️

-This project involves working directly with lethal 220V AC mains voltage.

-Never test this circuit on a breadboard. Use a properly soldered perfboard or a custom PCB.

-Always disconnect the power before touching any part of the circuit.

-Use a fuse on the live AC input for safety.

-House the final project in a non-conductive enclosure to prevent accidental contact.

-Safety is your responsibility. Proceed with extreme caution.

✨FUTURE IMPROVEMENTS

-Will Upgrade to ESP8266/ESP32: Replace the Arduino and Python server with a single ESP8266/ESP32 to host the web server directly on the microcontroller.

-Slider Control: Implement a slider on the webpage for fully variable dimming instead of fixed modes.

-MQTT Integration: Add MQTT support to integrate the dimmer into a larger smart home

system.