Upskill Campus Internship Project

Project Report

IoT based Home Automation System

Submitted By:

Prakhar Tripathi

ABSTRACT

The "Home Automation System" project presented in this abstract leverages the Internet of Things (IoT) technology to enable remote control of home appliances. The system focuses on controlling two LED bulbs and an AC fan using an ESP32 microcontroller. To interface with the appliances, three relay modules are employed due to their ability to handle AC devices effectively.

The Arduino IDE serves as the development environment for programming the microcontroller and managing the system's functionality. A user-friendly web-based dashboard is created, allowing users to control the appliances remotely. By connecting to the same Wi-Fi network as the ESP32, users can simply enter the ESP's IP address in a web browser to access the dashboard. The system utilizes the HTTP protocol for communication between the web interface and the microcontroller.

In addition to the digital aspects, physical implementation is also considered in the project. A custom-designed case is created to house the circuitry, ensuring proper protection and aesthetics. Images of the system and the physical case can be found on the provided GitHub link and in the accompanying documentation.

Power can be supplied to the circuit either through a 3.7V Li-ion battery or by connecting a USB cable to a power bank or a laptop. Furthermore, a user interface is integrated into the physical case, allowing users to control the appliances manually using a switch.

1 Introduction

1.1 Description

The "Home Automation System" project is a sophisticated solution that harnesses the power of Internet of Things (IoT) technology to enable seamless control of home appliances. With a focus on two LED bulbs and an AC fan, the project utilizes an ESP32 microcontroller as the central processing unit.

To facilitate the interaction between the microcontroller and the appliances, three relay modules are employed. These modules are specifically designed to handle AC devices efficiently, ensuring safe and reliable operation. The project is developed using the Arduino Integrated Development Environment (IDE), providing a user-friendly platform for programming the microcontroller and managing the system's functionalities.

The system can be controlled using the dashboard provided as a webpage, through the dashboard, users can remotely control the connected appliances with ease. By connecting their devices to the same Wi-Fi network as the ESP32, users can simply enter the IP address of the microcontroller into a web browser, granting them access to the dashboard. The HTTP protocol is employed for communication between the web interface and the microcontroller, enabling seamless and secure control over the appliances.

1.2 Motivation

The motivation behind the "Home Automation System" project stems from the increasing need for smart and convenient solutions to enhance our living spaces. Traditional manual control of home appliances can be cumbersome and inefficient, often requiring physical interaction and limited flexibility. This project aims to address these limitations by leveraging IoT technology and automation to create a more seamless and personalized user experience.

One of the key motivations for this project is to provide users with the ability to control their home appliances remotely. With the busy and

fast-paced lifestyles that many people lead, being able to adjust lighting, temperature, and other appliances from anywhere within the Wi-Fi network range offers unparalleled convenience. Whether it's turning off forgotten lights while away from home or pre-cooling a room before arrival, remote control empowers users to effortlessly manage their living spaces.

Energy efficiency is another important aspect driving the motivation behind this project. By integrating sensors, artificial intelligence algorithms, and automation, the system can optimize energy consumption based on user preferences and environmental conditions. This not only reduces electricity bills but also contributes to a more sustainable and eco-friendly lifestyle. The ability to monitor and control energy usage in real-time provides valuable insights and encourages responsible energy management practices.

2 Components

2.1 ESP32

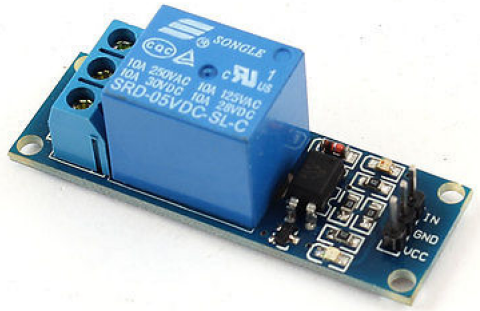
ESP32 is a powerful, low-cost, Wi-Fi and Bluetooth-enabled microcontroller developed by Espressif Systems. It is designed for use in

Internet of Things (IoT) applications and is based on the Xtensa LX6 CPU architecture. The ESP32 includes dual-core processors, a variety of communication interfaces, and an extensive set of peripherals, making it ideal for a wide range of IoT applications. The ESP32 features a rich set of features, including:

- **Dual-core processors:** The ESP32 includes two Tensilica Xtensa LX6 processors, providing high performance and flexibility.
- **Wi-Fi and Bluetooth connectivity:** The ESP32 includes built-in Wi-Fi and Bluetooth modules, providing seamless wireless connectivity.
- **Wide range of communication interfaces:** The ESP32 supports a wide range of communication interfaces, including SPI, I2C, UART, CAN, and Ethernet.
- **Rich set of peripherals:** The ESP32 includes a rich set of peripherals, including ADC, DAC, I2S, and SDIO.
- **Low power consumption:** The ESP32 features a low-power mode that enables it to operate for extended periods on battery power.
- **Support for various operating systems:** The ESP32 supports various operating systems, including FreeRTOS, Zephyr, and MicroPython.

The ESP32 is widely used in IoT applications, such as home automation, industrial automation, smart cities, and wearable devices. Its low cost, small size, and rich set of features make it an ideal choice for a wide range of IoT applications.

2.2 Single Channel Relay Module

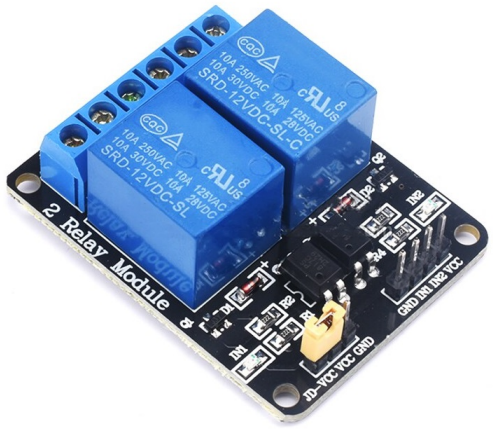


A single channel relay module is an electronic component used to control the power supply to a single electrical circuit or device. It acts as a switch that can be controlled by a low-voltage signal, allowing it to control higher voltage or current circuits. The module consists of a relay, a control circuit, and necessary connectors.

The relay itself is an electromechanical device that consists of a coil, an armature, and a set of contacts. When an electrical signal is applied to the coil, it generates a magnetic field that attracts the armature, causing the contacts to close or open, depending on the relay type. This mechanism allows the relay module to control the flow of electricity to the connected device.

The control circuit of the single channel relay module is designed to provide the necessary voltage and current to activate the relay coil. It typically accepts a low-voltage control signal, such as from a microcontroller or a digital output pin of a microcomputer. When the control signal is activated, the control circuit energizes the relay coil, enabling the module to switch the higher voltage or current circuit.

2.3 Dual Channel Relay Module



A dual channel relay module is an electronic component used to control the power supply to two separate electrical circuits or devices simultaneously. It operates on the same principle as a single channel relay module but provides the capability to control two independent channels or circuits.

The dual channel relay module consists of two relays, each with its own control circuit, and the necessary connectors to interface with external devices. The relays function as individual switches that can be controlled independently using low-voltage signals.

The control circuitry of the dual channel relay module typically includes input terminals for each channel, along with ground and power supply terminals. These input terminals receive the control signals, usually from a microcontroller or a digital output pin of a microcomputer. The ground and power supply terminals provide the necessary voltage and reference potential for the control circuits.

On the output side, the module has two sets of terminals for each channel: common (COM), normally open (NO), and normally closed (NC). The COM terminal of each channel connects to the power supply or load, while the NO or NC terminals connect to the other side of the load or the power source, depending on the desired switching behavior.

Activating the control signal for a specific channel energizes its relay,

thereby connecting or disconnecting the COM terminal with the corresponding NO or NC terminal.

2.4 AC Fan

One of the appliances to be controlled by the home automation system

2.5 Led Bulbs

One of the appliances to be controlled by the home automation system

2.6 Rocker Switch

A rocker switch is a simple and versatile electromechanical component commonly used for toggling electrical circuits on and off. Designed with a user-friendly rocker mechanism, it is ideal for a wide range of applications, including controlling the power supply, switching modes, or selecting options. In the Smart Helmet project, a rocker switch can be employed to activate or deactivate the system as needed. A rocker switch is a practical and reliable choice for the Smart Helmet project due to its simple operation, compact design, and high durability. Its versatility and ease of installation make it an ideal component for controlling the activation and deactivation of the helmet's system.

2.7 3.7 v Lithium Ion Rechargeable Battery

Li-ion batteries are a popular type of rechargeable battery widely used in various electronic devices, including smartphones, laptops, cameras, and portable electronics.

The 3.7V rating of a Li-ion battery represents its average voltage during discharge. It is important to note that the actual voltage of a fully charged Li-ion battery can range from 4.2V to 4.3V, while the voltage during discharge can drop to around 3.0V or lower, depending on the specific battery chemistry and discharge characteristics.

Li-ion batteries have several advantages over other battery chemistries, including high energy density, low self-discharge rate, and long cycle life. They provide a reliable and efficient power source for various applications. The 3.7V Li-ion battery, in particular, is commonly used in low-power electronic devices that require a compact and lightweight power solution.

3 IDE Used

3.1 Arduino

The Arduino IDE (Integrated Development Environment) is a software application used for programming and developing projects with Arduino boards. It provides a user-friendly platform for writing, compiling, and uploading code to Arduino microcontrollers.

The Arduino IDE is designed to be simple and accessible, making it suitable for beginners and experienced developers alike. It supports the Arduino programming language, which is a variant of C/C++ with simplified syntax and libraries specifically tailored for Arduino boards.

Key features of the Arduino IDE include:

Code Editor: The IDE offers a text editor where users can write their Arduino code. It provides syntax highlighting, auto-completion, and indentation to aid in code development.

Code Verification: The IDE has a built-in compiler that checks the syntax and structure of the code for errors. It helps identify and highlight any mistakes or issues before uploading the code to the Arduino board.

Library Manager: Arduino libraries contain pre-written code that simplifies the development process. The IDE includes a Library Manager, allowing users to easily search, install, and manage libraries required for their projects.

Board Manager: Different Arduino boards have different specifications and functionalities. The IDE's Board Manager allows users to select the specific Arduino board they are working with, ensuring compatibility and proper compilation of the code.
Serial Monitor: The Serial Monitor feature enables communication between the Arduino board and the computer. It allows users to send and receive data, view debug information, and monitor the behavior of their projects in real-time.
Overall, the Arduino IDE simplifies the development process for Arduino projects by providing an intuitive interface, essential tools, and a wide range of libraries. It empowers users to write and upload code to their Arduino boards, bringing their ideas to life in a user-friendly and efficient manner.

4 Working

The "Home Automation System" project operates by utilizing an ESP32 microcontroller and a dual-channel relay module to control two LED bulbs and an AC fan. The working of the project involves several steps:

- **Hardware Setup:** The microcontroller, relay module, and associated components are connected according to the project's circuit diagram. The relay module is connected to the ESP32, with each channel of the module corresponding to a specific appliance.
- **Software Programming:** The Arduino IDE is used to write the code for the ESP32 microcontroller. The code includes the necessary libraries and functions to control the relay module and establish communication with the web-based dashboard.
- **Communication Setup:** The ESP32 is connected to a local Wi-Fi network, allowing it to communicate with devices on the same network. The microcontroller obtains an IP address, which is required to access the web-based dashboard.

- **Power Management:** The 3.7 V lithium polymer rechargeable battery provides a reliable power source for the Smart Helmet's components. Its high energy density, lightweight design, and rechargeable nature ensure continuous operation and extended battery life.
- **Web-Based Dashboard:** A user-friendly web interface is created using HTML, CSS, and JavaScript. The dashboard provides control buttons or switches for the LED bulbs and AC fan, allowing users to turn them on/off or adjust their settings.
- **Control Logic:** The code running on the ESP32 microcontroller continuously listens for commands received from the web-based dashboard. When a user interacts with the dashboard, such as toggling a switch, the corresponding command is sent to the ESP32.
- **Relay Control:** Upon receiving a command, the microcontroller activates the appropriate relay channel based on the user's input. For example, if the user wants to turn on an LED bulb, the microcontroller triggers the relay associated with that bulb, allowing current to flow and turning the bulb on.
- **Feedback and Status Update:** The microcontroller provides feedback to the web-based dashboard, indicating the status of the appliances. For instance, if an LED bulb is turned on, the dashboard reflects the updated status accordingly.
- **Power Supply:** The circuit can be powered either by a 3.7V Li-ion battery or through a USB cable connected to a power bank or laptop. The power source supplies the necessary voltage to the ESP32 and relay module for their operation.

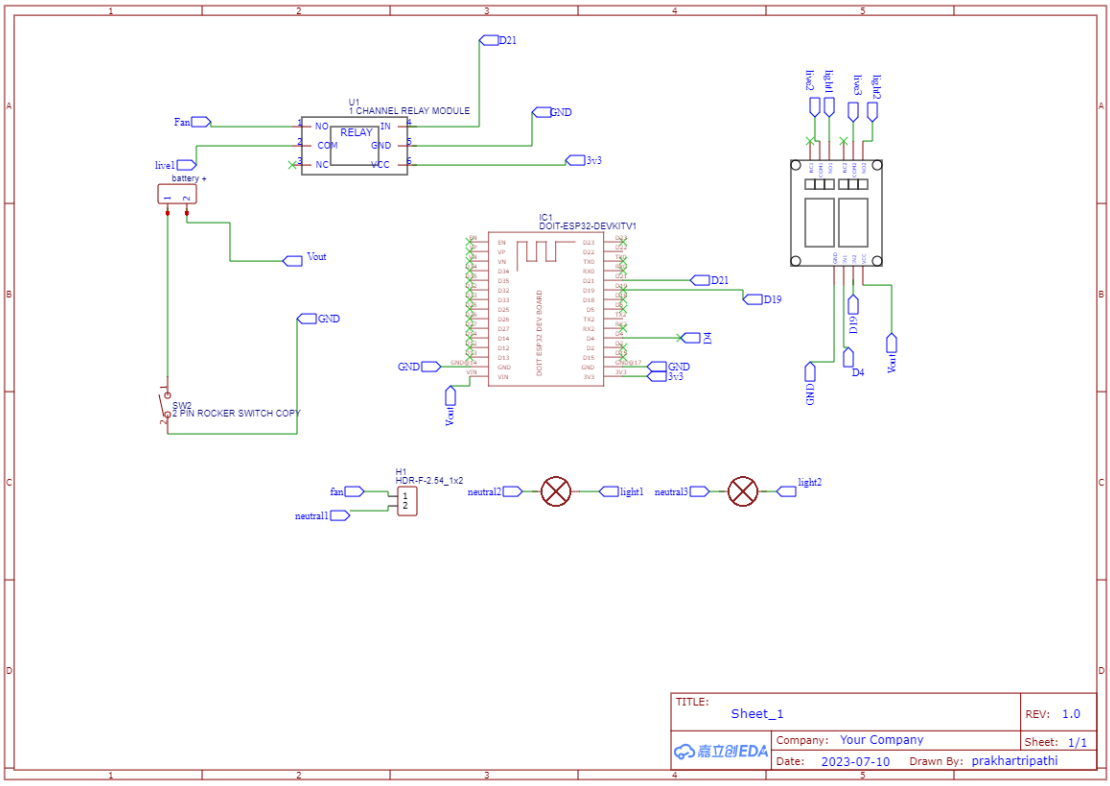
5 Designing Process

The design process of the Smart Safety Helmet involves planning, execution, and testing across two main domains: hardware and software.

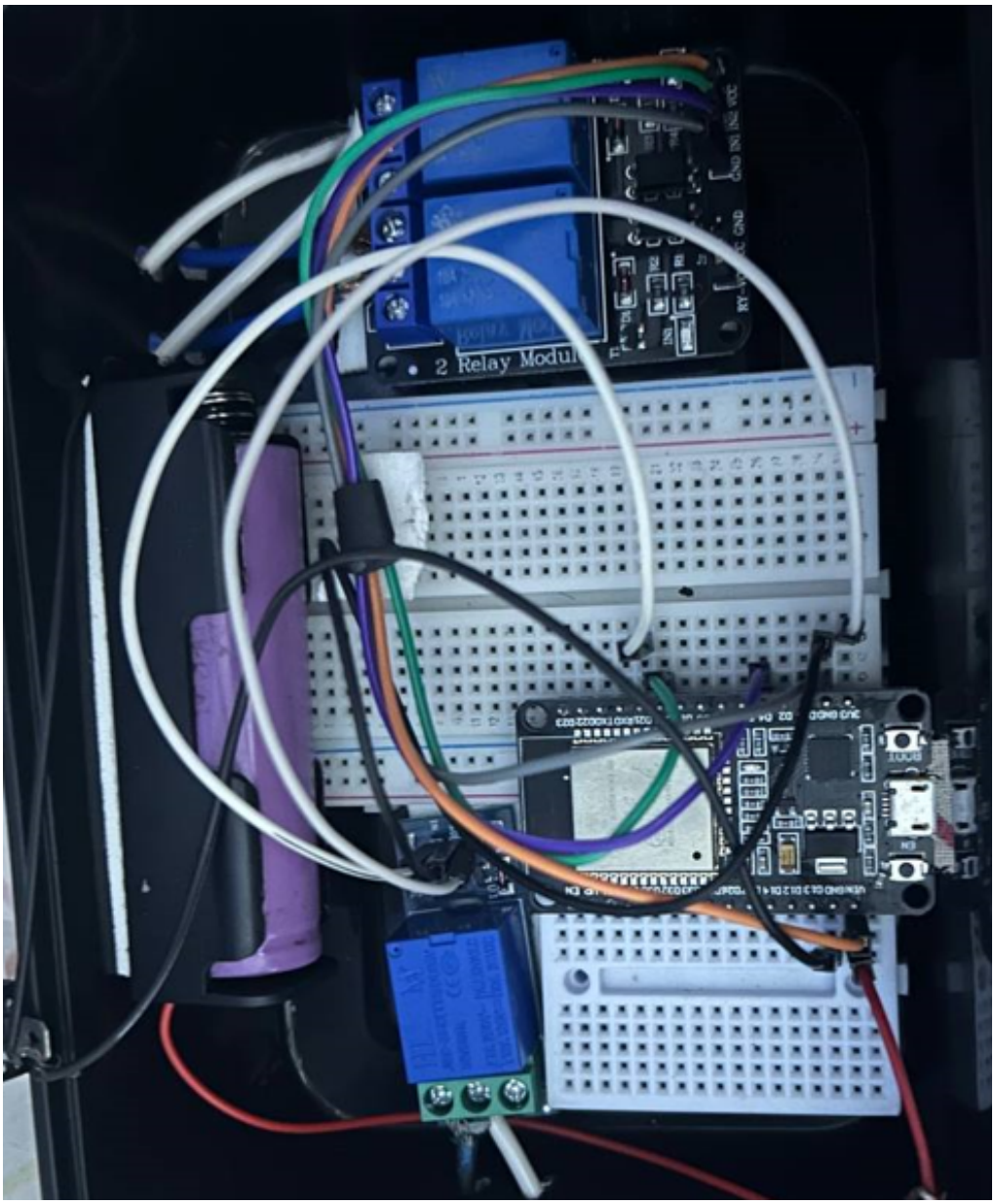
5.1 Hardware Design

The hardware design process entails the appropriate selection of components, their effective integration, and testing to ensure the system's functionality. Here's a detailed explanation:

- **Component Selection:** The first step involves choosing the right hardware like the ESP32 Microcontroller,Relay Modules, 3.7V lithium ion rechargeable battery, rocker switch, and the appliances. Make sure that these components align with the project's technical requirements such as power consumption, performance, and size.
- **Circuit Design:** Next, a schematic diagram illustrating the connections between the microcontroller, relay modules, appliances, battery, and switch needs to be designed. This design should ensure efficient power management and reliable data transfer. Precaution: Ensure the circuit is designed accurately to avoid short circuits and component failures.



- **Prototyping:** I have used a breadboard for the circuit


2 Relay Modul

5.2 Software Design

5.3 Code

Home Automation System:

```
#include <WiFi.h>

const char* ssid     = "Redmi Note 9 Pro";
const char* password = "9e65a49241be";

WiFiServer server(80);

String header;
String Device1State = "off";
String Device2State = "off";
String Device3State = "off";

const int Device1 = 4;
 const int Device2 = 19;
 const int Device3 = 21;

unsigned long currentTime = millis();
unsigned long previousTime = 0;
const long timeoutTime = 2000;

void setup() {
  Serial.begin(9600);

  pinMode(Device1, OUTPUT);
   pinMode(Device2, OUTPUT);
  pinMode(Device3, OUTPUT);

  digitalWrite(Device1, LOW);
   digitalWrite(Device2, LOW);
   digitalWrite(Device3, LOW);

  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
```

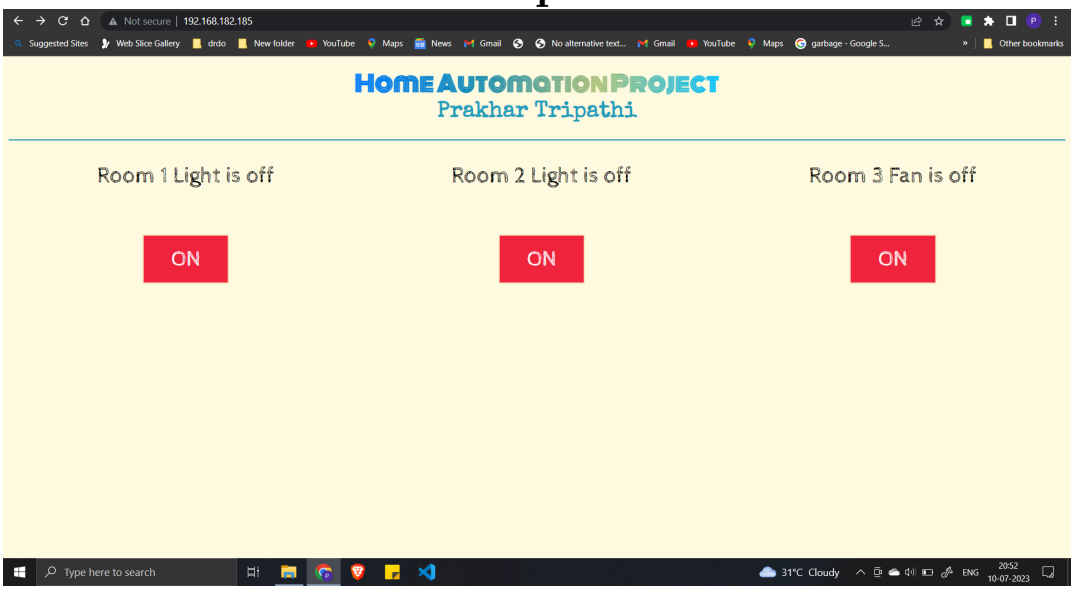
```
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected.");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  server.begin();
}

void loop(){
  WiFiClient client = server.available();

  if (client) {
    Serial.println("New Client.");
    String currentLine = "";
    currentTime = millis();
    previousTime = currentTime;
    while (client.connected() && currentTime - previousTime <
      currentTime = millis();
      if (client.available()) {
        char c = client.read();
        Serial.write(c);
        header += c;
        if (c == '\n') {
            if (currentLine.length() == 0) {
            client.println("HTTP/1.1 200 OK");
            client.println("Content-type:text/html");
            client.println("Connection: close");
            client.println();

            if (header.indexOf("GET /1/on") >= 0) {
              Serial.println("Device 1 on");
              Device1State = "on";
              digitalWrite(Device1, LOW);
            }
```

```
            digitalWrite(Device1, LOW);
          }
          else if (header.indexOf("GET /1/off") >= 0) {
            Serial.println("Device 1 off");
            Device1State = "off";
            digitalWrite(Device1, HIGH);
          }

            else if (header.indexOf("GET /2/on") >= 0) {
             Serial.println("Device 2 on");
              Device2State = "on";
             digitalWrite(Device2, LOW);
           } else if (header.indexOf("GET /2/off") >= 0) {
             Serial.println("Device 2 off");
             Device2State = "off";
             digitalWrite(Device2, HIGH);
           }
           else if (header.indexOf("GET /3/on") >= 0) {
             Serial.println("Device 3 on");
             Device3State = "on";
             digitalWrite(Device3, LOW);
           }
           else if (header.indexOf("GET /3/off") >= 0) {
             Serial.println("Device 3 off");
             Device3State = "off";
            digitalWrite(Device3, HIGH);
           }

          // Display the HTML web page
          client.println("<!DOCTYPE html>");
          client.println("<html lang=\"en-US\">");
          client.println("<head><meta name=\"viewport\" content=\"width=device-width, initial-scale=1\">");
              client.println("<link rel=\"stylesheet\"href=\"https://fonts.googleapis.com/css2?family=Tilt Prism\">");
               client.println("<link rel=\"stylesheet\" href=\"https://fonts.googleapis.com/css?family=Sofia\">");
                client.println("<link rel=\"stylesheet\"href=\"https://fonts.googleapis.com/css2?family=Moirai One\">");
    client.println("<link rel=\"stylesheet\"href=\"https://fonts.googleapis.com/css2?family=Special Elite\">");
```

```
                client.println("<link rel=\"stylesheet\"href=\"https://fonts.googleapis.com/css2?family=Moirai One\">");
     client.println("<link rel=\"stylesheet\"href=\"https://fonts.googleapis.com/css2?family=Special Elite\">");
    client.println("<link rel=\"stylesheet\" href=\"https://fonts.googleapis.com/css2?family=Foldit\">");
    client.println("<link rel=\"stylesheet\" href=\"https://fonts.googleapis.com/css2?family=Monoton\">");
    client.println("<link rel=\"stylesheet\" href=\"https://fonts.googleapis.com/css2?family=Rock 3D\">");
    client.println("<link rel=\"stylesheet\" href=\"https://fonts.googleapis.com/css2?family=Cabin Sketch\">");
    client.println("<link rel=\"stylesheet\" href=\"https://fonts.googleapis.com/css2?family=Cinzel Decorative\">");
    client.println("<link rel=\"stylesheet\" href=\"https://fonts.googleapis.com/css2?family=Fredericka the Great\">");
    client.println("<link rel=\"stylesheet\" href=\"https://fonts.googleapis.com/css2?family=Pacifico\">");
    client.println("<link rel=\"stylesheet\" href=\"https://fonts.googleapis.com/css2?family=Lobster\">");


          client.println("<style>html { font-family: Helvetica; display: inline-block; margin: 0px auto; text-align: center;}");
          client.println("Body { background-color:#fefae0;}");
          client.println("header { text-align:center; color:#219ebc; font-family:Special Elite; margin:auto;border-bottom:2px solid #168aad; line-hei
          client.println(".button { background-color: #ef233c; border: 2px solid wheat; color: white; padding: 16px 40px; font-family:Cabin Sketch;")
          client.println("text-decoration: none; font-size: 30px; margin: 2px; cursor: pointer;}");
           client.println(".button2 {background-color: #8ecae6; color:#d90429}");
client.println(".main {font-family: Monoton; background: linear-gradient(to right, #007bff, #118ab2, #adc178, #03c0ff);-webkit-text-fill-color: transpa
           client.println(".one,.two,.three {display:flex; flex-direction:column;justify-content:center;align-items:center;}");
           client.println("p {font-size:2em; font-family:Cabin Sketch;}");
           client.println(".devices {display:flex; align-items:center; justify-content:space-around;}");
           client.println("@media(max-width:600px){.devices{display:flex;flex-direction:column;align-items:center; justify-content:center;}}</style><

          client.println("<body><header><h1><span class=\"main\">Home Automation Project</span><br>Prakhar Tripathi</h1></header>");


          client.println("<div class=\"devices\"><div class=\"one\"><p>Room 1 Light is " + Device1State + "</p>");
          if (Device1State=="off") {
            client.println("<p><a href=\"/1/on\"><button class=\"button\">ON</button></a></p></div>");
          } else {
```

```
                client.println("<p><a href=\"/1/off\"><button class=\"button button2\">OFF</button></a></p></div>");
              }

              client.println("<div class=\"two\"><p>Room 2 Light is " + Device2State + "</p>");
              if (Device2State=="off") {
                client.println("<p><a href=\"/2/on\"><button class=\"button\">ON</button></a></p></div>");
              } else {
                client.println("<p><a href=\"/2/off\"><button class=\"button button2\">OFF</button></a></p></div>");
              }

              client.println("<div class=\"three\"><p>Room 3 Fan is " + Device3State + "</p>");
              if (Device3State=="off") {
                client.println("<p><a href=\"/3/on\"><button class=\"button\">ON</button></a></p></div>");
              } else {
                client.println("<p><a href=\"/3/off\"><button class=\"button button2\">OFF</button></a></p></div>");
              }


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

              client.println();
              break;
            } else {
              currentLine = "";
            }
          } else if (c != '\r') {
            currentLine += c;
          }
        }
      }
      header = "";
      client.stop();
      Serial.println("Client disconnected.");
      Serial.println("");
    }
}
```

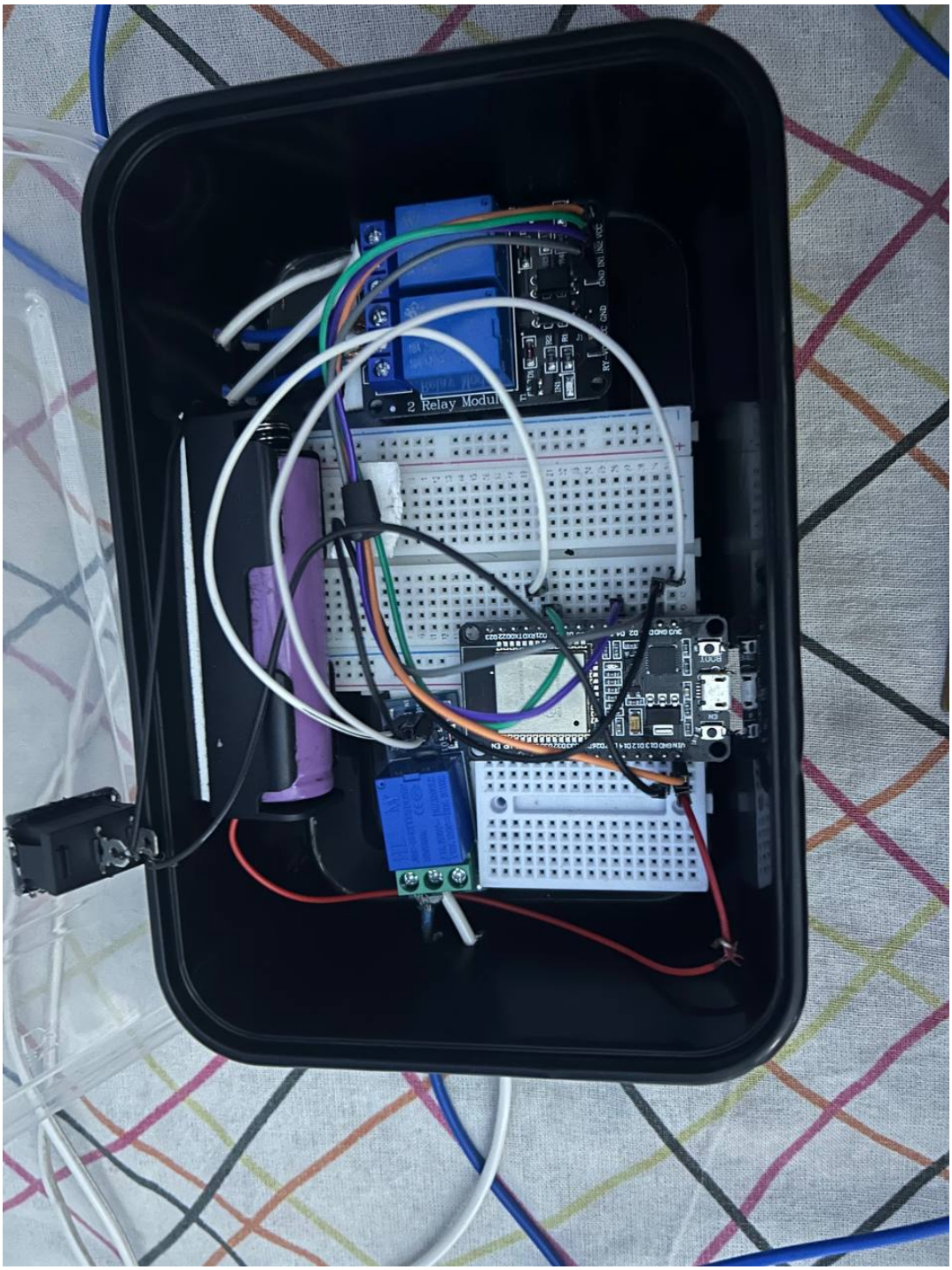
Output:

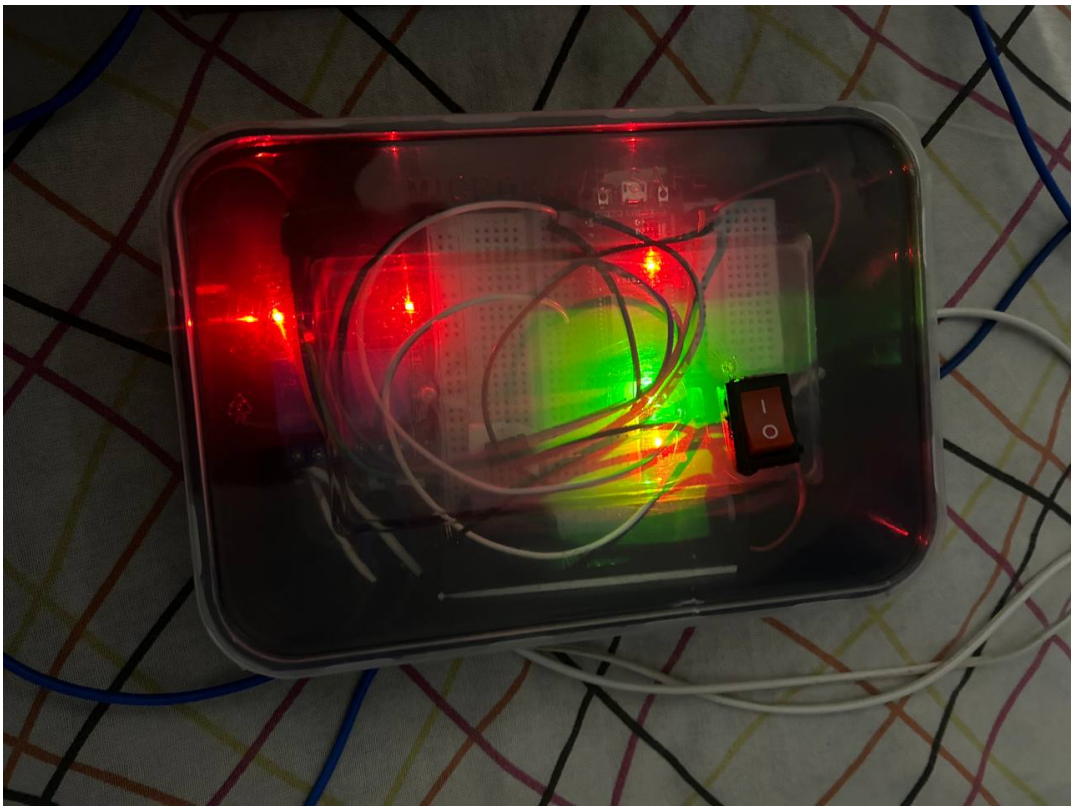



8:42
192.168.182.185
Home
Automation
Project
Prakhar Tripathi
Room 1 Light is off
ON
Room 2 Light is off
ON
50

– Outer Case of the system:

I have used a tiffin box to make the case for the circuit, Using Screwdriver and candle i made the required holes so that the wires coming out of the relay modules can easily pass through the case

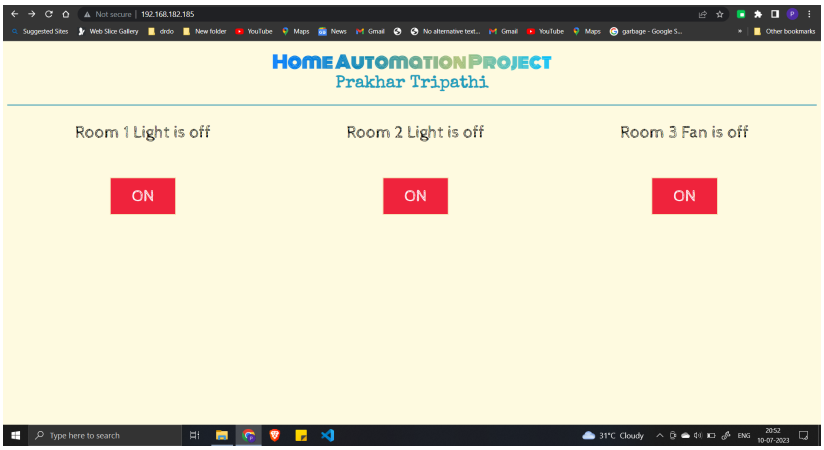

2 Relay Modul


I
O

(a) WHEN OFF

(b) WHEN ON

Figure 1: FINAL PRODUCT



(a) DESKTOP DASHBOARD

(b) MOBILE DASHBOARD

6 To Upskill Campus, IoT Academy and the mentors

I would like to express my heartfelt gratitude to **Upskill Campus, IoT Academy and the mentors** for their assistance and resources during the project. Though I encountered some challenges along the way, the support I received from the resources was invaluable.

7 References

→ http://dvgadre.blogspot.com/2018/09/so-you-want-to-build-electroni project_25.html

→ The IoT academy Embedded Systems Training Basic Electronics Part1 module
IoT academy Embedded Systems Training Basic Electronics Part2 module
IoT academy Embedded Systems Training Basic Electronics Part3

→ https://www.youtube.com/watch?v=zJ-LqeX$_f LU$

8 Personal Details

* **Name:** Prakhar Tripathi

* **Email Id:** Prakhar.tripathi.1402@gmail.com

* **Phone No:** 9654937307