# INTRODUCTION TO COMMUNICATION & IOT

## 22AIE211

# "SMART HOME AUTOMATION SYSTEM"

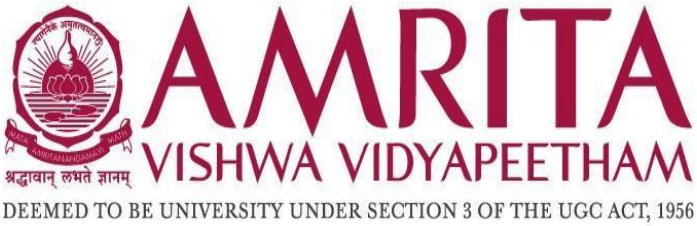
### A REPORT SUBMITTED BY

## GROUP 14 – BATCH (A)


| | |
|---|---|
| **VAMSI** | **CB.EN.U4AIE22028** |
| **THARUN KUMAR** | **CB.EN.U4AIE22068** |
| **SURYA TEJA** | **CB.EN.U4AIE22069** |
| **RAMA KRISHNA PRASAD** | **CB.EN.U4AIE22070** |


**IN PARTIAL FULFILLMENT FOR THE AWARD OF THE DEGREE**

**BACHELOR OF TECHNOLOGY IN CSE (AI)**





**CENTRE FOR COMPUTATIONAL ENGINEERING AND NETWORKING**

**AMRITA SCHOOL OF ARTIFICIAL INTELLIGENCE**

**AMRITA VISHWA VIDYAPEETHAM**

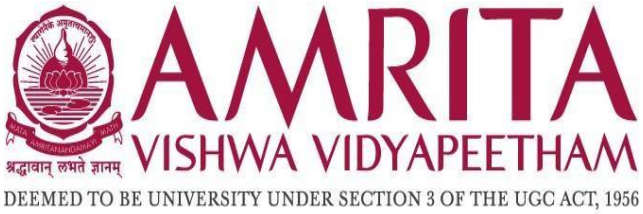
**COIMBATORE-641112(INDIA)**


**JUNE–2024**

# AMRITA VISHWA VIDYAPEETHAM

# AMRITA SCHOOL OF ARTIFICIAL INTELLIGENCE

## COIMBATORE-641112(INDIA)




## BONAFIDE CERTIFICATE

This is to certify that the report entitled "**SMART HOME AUTOMATION SYSTEM"** submitted by Group-14 batch-A (CB.EN.U4AIE22028 – Vamsi, CB.EN.U4AIE22068 – tharun kumar, CB.EN.U4AIE22069 – Surya teja, CB.EN.U4AIE22070 – Rama krishna prasad) for the award of the Degree of Bachelor of Technology in the "CSE(AI)" is a bonafide record of the work carried out by them under our guidance and supervision at Amrita School of Artificial Intelligence, Coimbatore.

MR. JAI SOORAJ

PROJECT GUIDE

DR.K.P.SOMAN

PROFESSOR AND HEAD OF CEN

SUBMITTED FOR THE UNIVERSITY EXAMINATION HELD ON **10/06/2024**

# AMRITA VISHWA VIDYAPEETHAM

# AMRITA SCHOOL OF ARTIFICIAL INTELLIGENCE

## COIMBATORE-641112(INDIA)

# DECLARATION

We, Group-14 of batch-A (CB.EN.U4AIE22028 – Vamsi, CB.EN.U4AIE22068 – tharun kumar, CB.EN.U4AIE22069 – Surya teja, CB.EN.U4AIE22070 – Rama krishna prasad) ,hereby declare that this report entitled**,** "**SMART HOME AUTOMATION SYSTEM"** is the record of the original work done by us under the guidance of Mr.jai Sooraj Sir, Centre for Computational Engineering and Networking, Amrita School of Artificial Intelligence, Coimbatore. To the best of my knowledge this work has not formed the basis for the award of any degree/diploma/associate ship/fellow ship to any candidate in the University.

**PLACE: COIMBATORE**

**DATE: 10-06-2024** **SIGNATURE OF THE STUDENTS**

# ACKNOLEADGMENT

We would like to express our sincere gratitude to our project subject handler, Mr. Jai Sooraj sir, for his invaluable guidance, support, and encouragement throughout this project. His expertise and knowledge in the field were instrumental in the success of this project. We would like to express our profound gratitude to Dr. Soman K P of the CEN department for his contributions to the completion of our project. This project would not have been possible without the help and guidance of every member of our faculty.

# CONTENTS

# TABLE OF FIGURES

# ABSTRACT

This project aims to develop an advanced smart home automation system utilizing a NodeMCU (ESP8266) microcontroller, a 4-channel relay module, and the Blynk mobile application. The system provides dual control for four lights: manual operation through physical push buttons and remote operation via the Blynk app on a smartphone. By integrating the NodeMCU with the Blynk platform, users gain the flexibility to control their home lighting from anywhere, enhancing convenience and energy efficiency.

The hardware setup involves connecting the NodeMCU to a 4-channel relay module, each relay controlling an individual light. Additionally, push buttons are wired to the NodeMCU to allow manual toggling of the lights. The NodeMCU is programmed using the Arduino IDE, incorporating the necessary libraries for Wi-Fi connectivity and Blynk integration. The software setup includes establishing a connection to the Blynk server, enabling real-time control and monitoring through the mobile app.

Key features of this system include the seamless integration of manual and remote control mechanisms, providing users with immediate physical control and the convenience of remote operation. The Blynk app serves as an intuitive interface, allowing users to toggle lights on and off, schedule operations, and receive status updates. This dual-mode control ensures reliability and flexibility, catering to various user preferences and scenarios.

This project showcases a practical and user-friendly solution for smart home lighting automation. It emphasizes ease of installation, scalability, and the potential for further expansion with additional smart home devices and sensors. By leveraging the capabilities of the NodeMCU and the Blynk platform, the project offers an accessible entry point into the realm of smart home technologies, enhancing the overall user experience and promoting energy-efficient practices.

# 1.INRODUCTION

Smart home automation systems have emerged as a result of the transformation in how we interact with our home settings brought about by the Internet of Things (IoT). These systems let customers to remotely operate a variety of household appliances, improving convenience, energy efficiency, and security. In light of this, the current project's goal is to create a flexible smart home lighting automation system by utilizing the Blynk mobile application, a 4-channel relay module, and a NodeMCU (ESP8266) microcontroller.

The main idea behind this project is to combine mobile-based remote control with manual control to provide a versatile and easy-to-use solution for automating lights in homes. The central control unit is a low-cost microcontroller called NodeMCU that has built-in Wi-Fi capabilities. It communicates with the 4-channel relay module to control four lights at once. Every relay channel has a designated light connected to it, and physical push buttons are part of the configuration for manual control.

The system makes use of the Blynk mobile application, a strong and user-friendly platform made for Internet of Things applications, to supplement manual operation. Users may use their cellphones to control the lights by connecting the NodeMCU to the Blynk server. This allows for scheduling, on/off toggling, and real-time monitoring from any location with internet connection. With its dual-mode control, customers may manage their lighting system from a distance or in person, meeting a variety of demands and improving convenience all around.

Blynk's connection with NodeMCU not only makes remote control implementation easier, but it also creates room for future growth. A full smart home environment may be created by simply adding more devices or sensors to the system. Furthermore, the Blynk app's feature-rich functionality and easy-to-use UI make it a great option for novice and expert users alike when it comes to smart home automation.

The goal of this project is to highlight how IoT can be used practically in daily life by illustrating how contemporary technology may change conventional home systems into responsive and intelligent spaces. This project is a great resource for anybody interested in learning more about the possibilities of smart homes since it offers a comprehensive manual for configuring and programming the hardware and software components.


**Figure 1: SMART HOME AUTOMATION**

# 2.PROJECT COMPONENTS

We used a NodeMCU (ESP8266), a 4-channel relay module, push buttons, and lights to complete the smart home automation project. A thorough explanation of each part and how it will be used to carry out the project is provided below.

## 2.1. NODEMCU (ESP8266)

Function: The NodeMCU is a low-cost microcontroller with built-in Wi-Fi capabilities. It serves as the central control unit for the smart home automation system, managing communication between the Blynk app and the relay module.
Connections:
Power: Connected to a 5V power supply.
Ground (GND): Connected to the common ground.
GPIO Pins (D1, D2, D3, D5, D6, D7): Used to interface with the relay module and push buttons.

**Figure 2: NODE MCU**

## 2.2. 4-CHANNEL RELAY MODULE

Function: The relay module is used to control high-voltage devices like lights. Each relay acts as a switch that can be controlled by the NodeMCU to turn the connected lights on or off.
Connections:
VCC: Connected to the 5V power supply.
GND: Connected to the common ground.
IN1, IN2, IN3, IN4: Connected to the GPIO pins on the NodeMCU (D1, D2, D3, D5) to receive control signals.
Common (COM) and Normally Open (NO)
Terminals: Connected to the lights and the AC power supply to control the lights.

**Figure 3: 4CHANNEL RELAY MODULE**

## 2.3. PUSH BUTTONS (S1, S2, S3, S4)

Function: The push buttons provide manual control for the lights. When pressed, they send a signal to the NodeMCU to toggle the state of the corresponding relay.
Connections:
One terminal of each button: Connected to a GPIO pin on the NodeMCU (D3, D6, D7, RX).
Other terminal of each button: Connected to the ground.

**Figure 4: PUSH BUTTONS**

## 2.4. LIGHTS

Function: The lights are the high-voltage devices controlled by the relay module. Each light is connected to a relay that allows the NodeMCU to switch it on or off.
Connections:
Connected to the Normally Open (NO) and Common (COM) terminals of the relays.
AC Supply (Live and Neutral): Connected to the AC mains to provide power to the lights.

**Figure 5: LED**

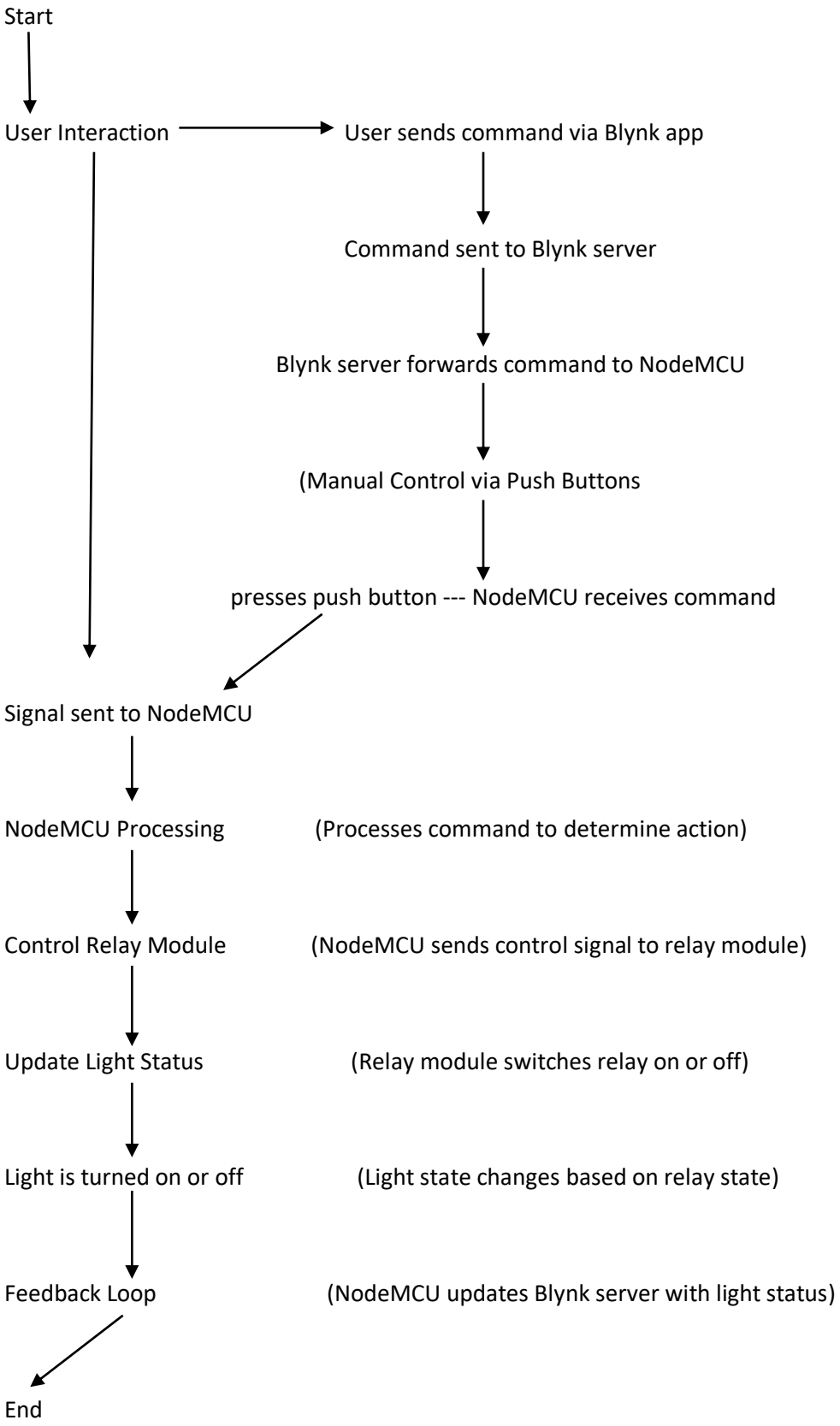
## 2.5. POWER SUPPLY

Function: Provides the necessary power for the NodeMCU and the relay module.
Connections:
5V Output: Connected to the VCC of the NodeMCU and the relay module.
GND: Connected to the common ground of the NodeMCU and relay module.

# 3.DATA FLOW

The data flow in this smart home automation system begins with user interaction, either through the Blynk mobile app or via manual push buttons. When the user interacts with the Blynk app, a command is sent to the Blynk server over the internet, which then forwards the command to the NodeMCU microcontroller via Wi-Fi. Alternatively, when a user presses a push button, a signal is directly sent to the NodeMCU. The NodeMCU processes the received commands or signals to determine the appropriate action, such as turning a light on or off.

Once the NodeMCU has processed the command, it sends control signals to the corresponding channels on the 4-channel relay module. Each relay acts as a switch to control the connected lights based on the signals received from the NodeMCU. If the command is to turn on a light, the relay module switches the relevant relay, completing the circuit and powering the light. Simultaneously, the NodeMCU updates the Blynk server with the current state of the lights, ensuring that the Blynk app reflects the real-time status of the lights, thereby providing feedback to the user. This dual-mode control system allows users to seamlessly operate their home lighting either remotely via the Blynk app or manually through push buttons.


Start
User Interaction
User sends command via Blynk app
Command sent to Blynk server
Blynk server forwards command to NodeMCU
(Manual Control via Push Buttons
presses push button --- NodeMCU receives command
Signal sent to NodeMCU
NodeMCU Processing
(Processes command to determine action)
Control Relay Module
(NodeMCU sends control signal to relay module)
Update Light Status
(Relay module switches relay on or off)
Light is turned on or off
(Light state changes based on relay state)
Feedback Loop
(NodeMCU updates Blynk server with light status)
End

# 4.PROTOCOLS INVOLVED

The NodeMCU smart home automation system, along with a 4-channel relay module and the Blynk app, uses many protocols. The system's components can communicate and transmit data more easily thanks to these protocols. The main procedures involved are as follows:

## 4.1. WI-FI

Usage: The NodeMCU (ESP8266) uses Wi-Fi to connect to the local wireless network.

Function: Wi-Fi enables the NodeMCU to communicate with the Blynk server over the internet. This is essential for remote control functionality, allowing the user to send commands from the Blynk app to the NodeMCU.

## 4.2. HTTP/HTTPS

Usage: Communication between the Blynk app and the Blynk server.

Function: When the user interacts with the Blynk app, commands are sent to the Blynk server using the HTTP/HTTPS protocol. The server processes these commands and forwards them to the NodeMCU. HTTPS is often used to ensure secure communication.

### 4.3. GPIO (GENERAL PURPOSE INPUT/OUTPUT)

Usage: Communication between the NodeMCU and the relay module, as well as the push buttons.

Function: The GPIO pins on the NodeMCU are used to send control signals to the relay module. Each relay is connected to a specific GPIO pin, and the state of these pins determines whether the relays (and thus the connected lights) are turned on or off. Additionally, the push buttons are connected to GPIO pins to detect manual inputs.

The primary protocols used are GPIO for direct hardware control of the relay module and push buttons, HTTP/HTTPS for command transmission from the Blynk app to the Blynk server, and Wi-Fi for network connectivity. Together, these protocols make it possible for the user, the NodeMCU, and the controlled devices to communicate seamlessly, enabling a reliable and adaptable smart home automation system.

# 5.CIRCUIT SETUP

## 5.1. NODEMCU (ESP8266) CONNECTIONS

VCC (5V): Connect to the 5V power supply line.

GND (Ground): Connect to the common ground line.

D1 (GPIO 5): Connect to IN1 on the relay module.

D2 (GPIO 4): Connect to IN2 on the relay module.

D3 (GPIO 0): Connect to IN3 on the relay module and Push Button S1.

D5 (GPIO 14): Connect to IN4 on the relay module.

D6 (GPIO 12): Connect to Push Button S2.

D7 (GPIO 13): Connect to Push Button S3.

RX (GPIO 3): Connect to Push Button S4.

## 5.2. RELAY MODULE CONNECTIONS

VCC (5V): Connect to the 5V power supply line.

GND (Ground): Connect to the common ground line.

IN1: Connect to D1 (GPIO 5) on the NodeMCU.

IN2: Connect to D2 (GPIO 4) on the NodeMCU.

IN3: Connect to D3 (GPIO 0) on the NodeMCU.

IN4: Connect to D5 (GPIO 14) on the NodeMCU.

COM (Common) for Relay 1: Connect to one side of Light 1.

NO (Normally Open) for Relay 1: Connect to the Live wire from the AC supply.

COM (Common) for Relay 2: Connect to one side of Light 2.

NO (Normally Open) for Relay 2: Connect to the Live wire from the AC supply.

COM (Common) for Relay 3: Connect to one side of Light 3.

NO (Normally Open) for Relay 3: Connect to the Live wire from the AC supply.

COM (Common) for Relay 4: Connect to one side of Light 4.

NO (Normally Open) for Relay 4: Connect to the Live wire from the AC supply.

## 5.3. LIGHTS CONNECTIONS

Light 1: Connect one terminal to COM terminal of Relay 1 and the other terminal to the Neutral wire from the AC supply.

Light 2: Connect one terminal to COM terminal of Relay 2 and the other terminal to the Neutral wire from the AC supply.

Light 3: Connect one terminal to COM terminal of Relay 3 and the other terminal to the Neutral wire from the AC supply.

Light 4: Connect one terminal to COM terminal of Relay 4 and the other terminal to the Neutral wire from the AC supply.

## 5.4. PUSH BUTTONS CONNECTIONS

S1: Connect one terminal to D3 (GPIO 0) on the NodeMCU and the other terminal to the common ground line.

S2: Connect one terminal to D6 (GPIO 12) on the NodeMCU and the other terminal to the common ground line.

S3: Connect one terminal to D7 (GPIO 13) on the NodeMCU and the other terminal to the common ground line.

S4: Connect one terminal to RX (GPIO 3) on the NodeMCU and the other terminal to the common ground line.

## 5.5. POWER SUPPLY CONNECTIONS

Live Wire: Connect to the Normally Open (NO) terminals of all four relays.

Neutral Wire: Connect to one side of each light.

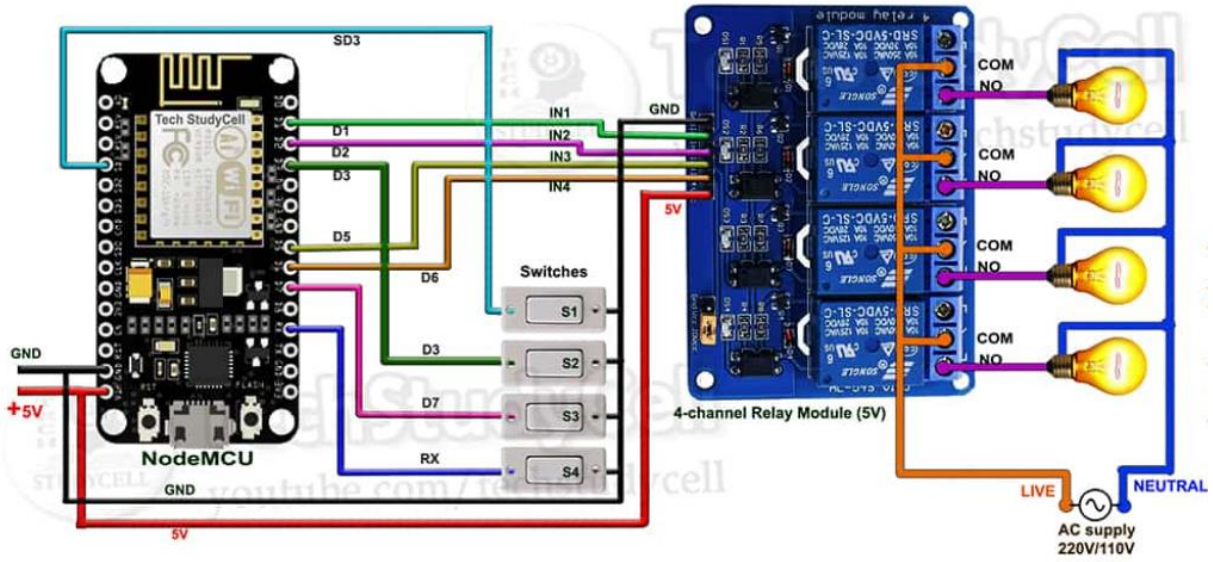



**Figure 6: CIRCUIT DIAGRAM**

# 6.CODE OVERVIEW

```
#define BLYNK_TEMPLATE_ID "TMPL3766ltpPO"
#define BLYNK_TEMPLATE_NAME "IOT Project"
#define BLYNK_AUTH_TOKEN "0eHXcmiDAAzXtKwjTeReOvdHD2U2Fy_q"


#define BLYNK_PRINT Serial


#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>

char auth[] = BLYNK_AUTH_TOKEN;


char ssid[] = "Wokwi-GUEST";
char pass[] = "";

BlynkTimer timer;


#define button1_pin 26
#define button2_pin 25
#define button3_pin 33
#define button4_pin 32

#define relay1_pin 13
#define relay2_pin 12
#define relay3_pin 14
#define relay4_pin 27

int relay1_state = 0;
int relay2_state = 0;
int relay3_state = 0;
int relay4_state = 0;


#define button1_vpin    V1
#define button2_vpin    V2
#define button3_vpin    V3
#define button4_vpin    V4


// This function is called every time the device is connected to the Blynk.Cloud
// Request the latest state from the server
BLYNK_CONNECTED() {
  Blynk.syncVirtual(button1_vpin);
```

```
  Blynk.syncVirtual(button2_vpin);
  Blynk.syncVirtual(button3_vpin);
  Blynk.syncVirtual(button4_vpin);
}


// This function is called every time the Virtual Pin state change
//i.e when web push switch from Blynk App or Web Dashboard
BLYNK_WRITE(button1_vpin) {
  relay1_state = param.asInt();
  digitalWrite(relay1_pin, relay1_state);
}

BLYNK_WRITE(button2_vpin) {
  relay2_state = param.asInt();
  digitalWrite(relay2_pin, relay2_state);
}

BLYNK_WRITE(button3_vpin) {
  relay3_state = param.asInt();
  digitalWrite(relay3_pin, relay3_state);
}

BLYNK_WRITE(button4_vpin) {
  relay4_state = param.asInt();
  digitalWrite(relay4_pin, relay4_state);
}



void setup()
{
  Serial.begin(115200);
  pinMode(button1_pin, INPUT_PULLUP);
  pinMode(button2_pin, INPUT_PULLUP);
  pinMode(button3_pin, INPUT_PULLUP);
  pinMode(button4_pin, INPUT_PULLUP);
  pinMode(relay1_pin, OUTPUT);
  pinMode(relay2_pin, OUTPUT);
  pinMode(relay3_pin, OUTPUT);
  pinMode(relay4_pin, OUTPUT);
  //During Starting all Relays should TURN OFF
  digitalWrite(relay1_pin, HIGH);
  digitalWrite(relay2_pin, HIGH);
  digitalWrite(relay3_pin, HIGH);
  digitalWrite(relay4_pin, HIGH);
  Blynk.begin(auth, ssid, pass);
}

void loop()
{
```

```
  Blynk.run();
  timer.run();
  // You can inject your own code or combine it with other sketches.
  // Check other examples on how to communicate with Blynk. Remember
  // to avoid delay() function!

  listen_push_buttons();
}


void listen_push_buttons(){
    if(digitalRead(button1_pin) == LOW){
      delay(200);
      control_relay(1);
      Blynk.virtualWrite(button1_vpin, relay1_state); //update button state
    }
    else if (digitalRead(button2_pin) == LOW){
      delay(200);
      control_relay(2);
      Blynk.virtualWrite(button2_vpin, relay2_state); //update button state
    }
    else if (digitalRead(button3_pin) == LOW){
      delay(200);
      control_relay(3);
      Blynk.virtualWrite(button3_vpin, relay3_state); //update button state
    }
    else if (digitalRead(button4_pin) == LOW){
      delay(200);
      control_relay(4);
      Blynk.virtualWrite(button4_vpin, relay4_state); //update button state
    }
}

void control_relay(int relay){

  if(relay == 1){
    relay1_state = !relay1_state;
    digitalWrite(relay1_pin, relay1_state);
    Serial.print("Relay1 State = ");
    Serial.println(relay1_state);
    delay(50);
  }

  else if(relay == 2){
    relay2_state = !relay2_state;
    digitalWrite(relay2_pin, relay2_state);
    delay(50);
  }

  else if(relay == 3){
    relay3_state = !relay3_state;
```

```
    digitalWrite(relay3_pin, relay3_state);
    delay(50);
  }

  else if(relay == 4){
    relay4_state = !relay4_state;
    digitalWrite(relay4_pin, relay4_state);
    delay(50);
  }

}
```

This smart home automation project utilizes a NodeMCU (ESP32) to control four relays, which in turn control four lights. The setup allows users to manage the lights via both physical push buttons and the Blynk app, providing flexible control options. The NodeMCU connects to a WiFi network using predefined SSID and password credentials and integrates with the Blynk cloud using authentication tokens. The code initializes serial communication for debugging, sets up the pin modes for buttons and relays, and ensures that all relays are turned off at startup. The relays are controlled by the state of the corresponding virtual pins in the Blynk app or by physical button presses, with the NodeMCU handling the toggling of the relay states and updating the Blynk server to reflect the current status.

The data flow involves the Blynk app sending commands to the Blynk server, which are then relayed to the NodeMCU. The NodeMCU processes these commands, controlling the relays accordingly. Physical button presses are monitored by the NodeMCU, which toggles the relay states and updates the Blynk server to keep the app in sync with the hardware status. This ensures that users can see the current state of the lights in the app, even when they are controlled manually. The code includes functions to handle Blynk connectivity, synchronize virtual pin states upon connection, and manage the relay states based on input from both the app and physical buttons. This integration of hardware and software provides a seamless and efficient way to control home lighting remotely and locally.

# 7.RESULTS

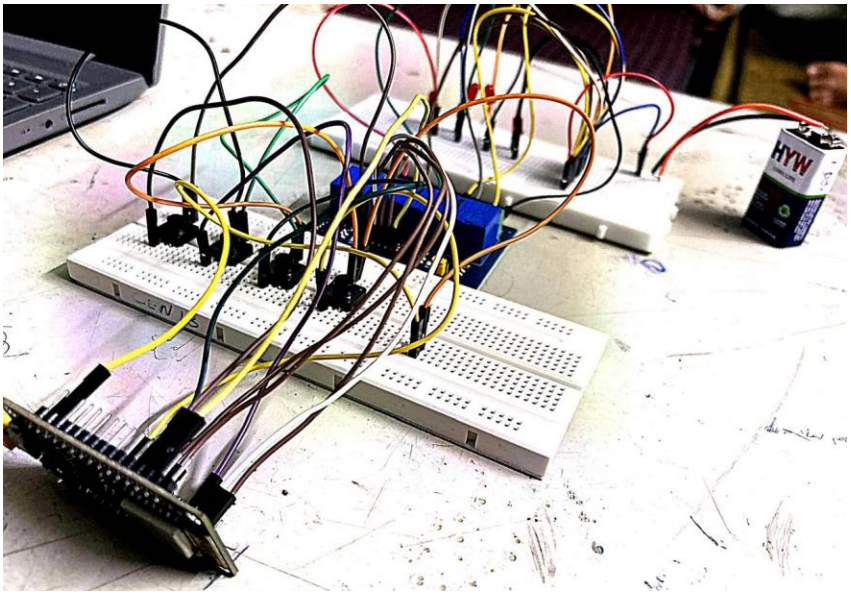



**Figure 7: HARDWARE MODEL**

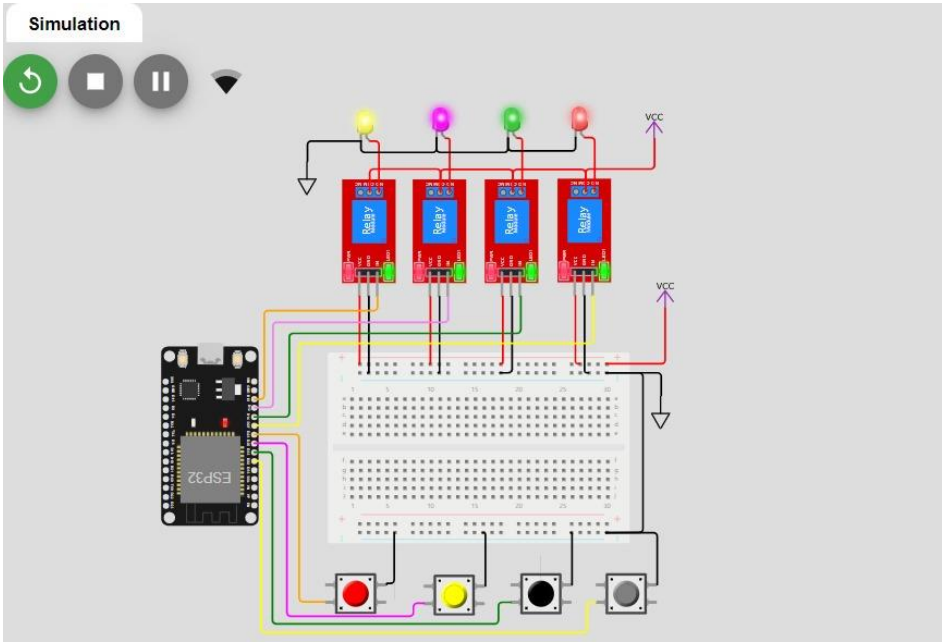



**Figure 8: SOFTWARE MODEL**

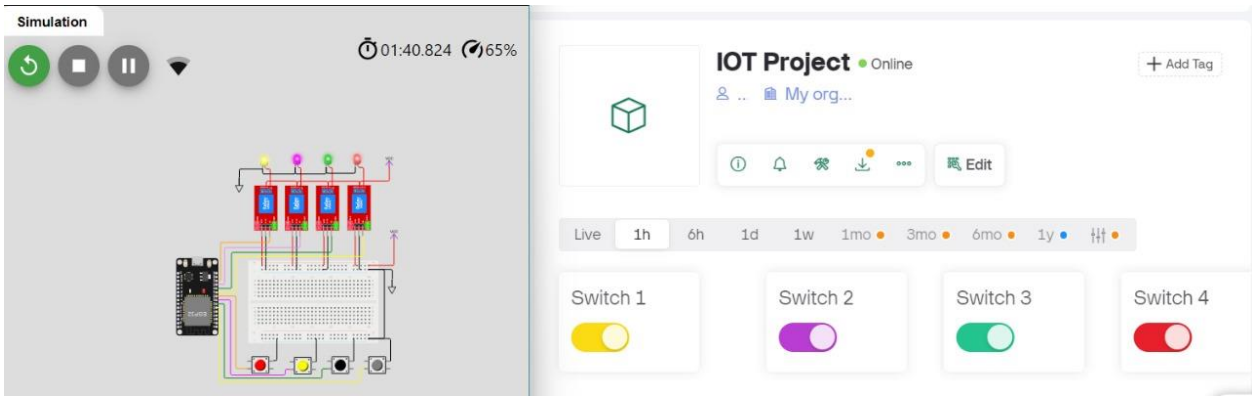



**Figure 9: SOFTWARE MODEL CONTROLLED FROM BLYNK WEBSITE**

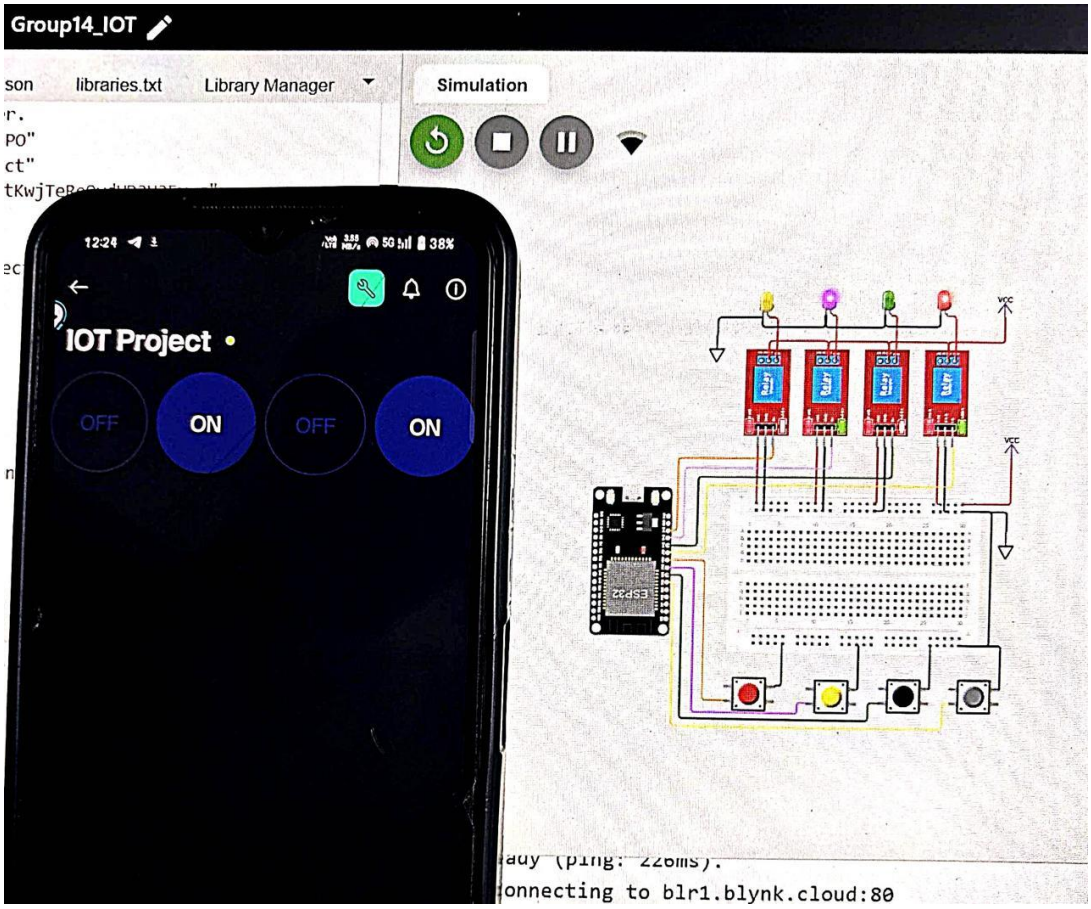



**Figure 10:SOFTWARE MODEL CONTROLLED BY BLYNK APP FROM MOBILE**

# 8.CONCLUSION

This smart home automation project successfully demonstrates the integration of NodeMCU (ESP32) with a 4-channel relay module and the Blynk platform to control home lighting. By enabling both remote control via the Blynk app and manual control through physical push buttons, the project provides a versatile and user-friendly solution for managing household lights. The NodeMCU's ability to connect to WiFi and communicate with the Blynk cloud ensures that users can monitor and adjust the light settings from anywhere, enhancing convenience and accessibility.

The implementation showcases the potential of IoT in creating efficient and smart home environments. It highlights the ease with which existing hardware can be integrated with modern cloud services to achieve real-time control and monitoring. This project not only improves the usability of home lighting systems but also lays a foundation for further expansion into other smart home applications, demonstrating a scalable and practical approach to home automation.