

DAYANANDA SAGAR UNIVERSITY



PROJECT REPORT

ON

" HOME AUTOMATION USING IoT"

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE & ENGINEERING

Submitted by

AMOGH G PADUKONE

III Semester, 2019

Under the supervision of

Prof.CVSN REDDY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SCHOOL OF ENGINEERING

DAYANANDA SAGAR UNIVERSITY

KUDLU GATE

BANGALORE - 560068

DAYANANDA SAGAR UNIVERSITY

School of Engineering, Kudlu Gate, Bangalore-560068



CERTIFICATE

This is to certify that Mr./Ms. Amogh G Padukone bearing USN ENG18CS0032 has satisfactorily completed his/her Minor Project as prescribed by the University for the 3rd semester B.Tech programme in Computer Science & Engineering during the year 2019-2020 at the School of Engineering, Dayananda Sagar University, Bangalore.

Date: _____

Supervisor(s)

Chairman

Department of Computer Science & Engineering

DECLARATION

I hereby declare that the work presented in this Mini Project entitled “HOME AUTOMATION USING Iot” has been carried out by me and it has not been submitted for the award of any degree, diploma or the mini project report of any other college or university.

AMOGH G PADUKONE
ENG18CS0032

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of people who made it possible and whose constant guidance and encouragement crown all efforts with success.

I are especially thankful to our **Chairman Dr. Banga M K, Ph.D.**, for providing necessary departmental facilities and moral support and encouragement.

I would like to express my deep sense of gratitude and sincere thanks to our project guide **Prof. Dr. CVSN Reddy**, for initiating me into this project by providing information regarding this project, guidance, support, motivation and patience which helped me for successful completion of my mini project work.

I have received a great deal of guidance and co-operation from my friends and I wish to thank one and all that have directly or indirectly helped me in the successful completion of this project.

Amogh G Padukone

ENG18CS0032

CONTENTS

SL NO.	TOPIC	PAGE NO.
1.	ABSTRACT	7
2.	INTRODUCTION	8
3.	INTERNET OF THINGS	9
4.	SYSTEM DESIGN AND IMPLEMENTATION	10
5.	HARDWARE	
	NodeMCU (ESP8266)	11
	RELAY BOARD	12
	ULN 2803 IC	13
6.	SOFTWARE	
	THE BLYNK APPLICATION	14
	THE IFTTT APPLICATION	15
7.	CODE FOR CONFIGURING NODEMCU	17

8.	CODE FOR WEBSERVER	18
9.	TEST RESULTS	23
10.	CONCLUSION AND REFERENCE	27

ABSTRACT

This paper presents home automation using voice via Google Assistant. Home automation or domotics a term for home automation coined by Jim Hill has been evolving drastically. We saw many home automation technologies introduced over these years from Zigbee automation to Amazon Echo, Google Home and Home from Apple. It has become a craze these days. Google Home price is around 150\$ (USD) with an additional cost of the devices to be connected to, the total cost of the system reaches over 250\$ (USD). Apple Home Kit too is pretty more expensive, over 100\$ (USD) more than the Google Home just for a basic setup. Philips Hue, a smart light which is controlled by the Google Assistant, Amazon Echo and Siri, voice assistant by Apple is priced around 145\$ (USD). Similarly, Belkin's Wemo light is priced around 44\$ (USD) per unit and this can be controlled both by Siri and Google Assistant. So, overall we can see here that to make our home smart we need to invest quite a lot, let's say some 250\$ (USD) for a basic setup. What if we can automate our house within (cost of the Smartphone is not included as it is assumed to be owned by every individual these days) 10\$ (USD) and can control up to 8 appliances using Google Assistant? Well, this paper describes the implementation of such a system. The system is implemented using ordinary household appliances. Natural language voice commands are given to the Google Assistant and with the help of IFTTT (If This Then That) application and the Blynk application the commands are decoded and then sent to the microcontroller, the microcontroller in turn controls the relays connected to it as required, turning the device connected to the respective relay On or OFF as per the users request to the Google Assistant. The microcontroller used is NodeMCU (ESP8266) and the communication between the microcontroller and the application is established via Wi-Fi (Internet).

INTRODUCTION

Home, it is the place where one fancies or desires to be after a long tiring day. People come home exhausted after a long hard working day. Some are way too tired that they find it hard to move once they land on their couch, sofa or bed. So any small device/technology that would help them switch their lights on or off, or play their favorite music etc. on a go with their voice with the aid of their smart phones would make their home more comfortable. Moreover, it would be better if everything such as warming bath water and adjusting the room temperature were already done before they reach their home just by giving a voice command. So, when people would arrive home, they would find the room temperature, the bath water adjusted to their suitable preferences, and they could relax right away and feel cozier and rather, feel more homely. Human assistants like housekeepers were a way for millionaires to keep up their homes in the past. Even now when technology is handy enough only the well to do people of the society are blessed with these new smart home devices, as these devices costs are a bit high. However, not everyone is wealthy enough to be able to afford a human assistant, or some smart home kit. Hence, the need for finding an inexpensive and smart assistant for normal families keeps growing. This paper proposes such inexpensive system. It uses the [1] Google Assistant, the [2] IFTTT application, [3] the Blynk application and [4] the NodeMCU microcontroller as the major components along with a relay board comprising of 4/8 relays along with ULN 2803 IC. Natural language voice is used to give commands to the Google Assistant . All of the components are connected over the internet using WiFi which puts this system under [5] the IoT .

INTERNET OF THINGS

Today, **Internet application development demand is very high**. So **IoT is a major technology** by which we can produce various useful internet applications.

Basically, **IoT is a network in which all physical objects are connected to the internet** through network devices or routers and exchange data. IoT allows objects to be controlled remotely across existing network infrastructure. IoT is a very good and intelligent technique which reduces human effort as well as easy access to physical devices. This technique also has autonomous control feature by which any device can control without any human interaction

“Things” in the IoT sense, is the mixture of hardware, software, data, and services. “Things” can refer to a wide variety of devices such as DNA analysis devices for environmental monitoring, electric clamps in coastal waters, Arduino chips in home automation and many other. **These devices gather useful data with the help of various existing technologies and share that data between other devices.** Examples include Home Automation System which uses Wi-Fi or Bluetooth for exchange data between various devices of home.



SYSTEM DESIGN AND

IMPLEMENTATION

The system design is broken down into two main categories,

- i. **THE HARDWARE** - It has the capability to connect to the router. It would also be able to turn on/off specified devices, such as lights and fans. It is called the 'Control Unit'.
- ii. **THE SOFTWARE** - The Blynk app, the IFTTT app and the Google Assistant constitute the software of the design and these applications would be integrated in the Android device.

The Control Unit comprises of the microcontroller Node MCU and the 4/8 Channel Relay board. Relay board uses ULN 2803 IC to control the relays. The Blynk app on an Android device communicates with the microcontroller and sends the desired signal via the internet. Figure 1 below shows the basic system design architecture.

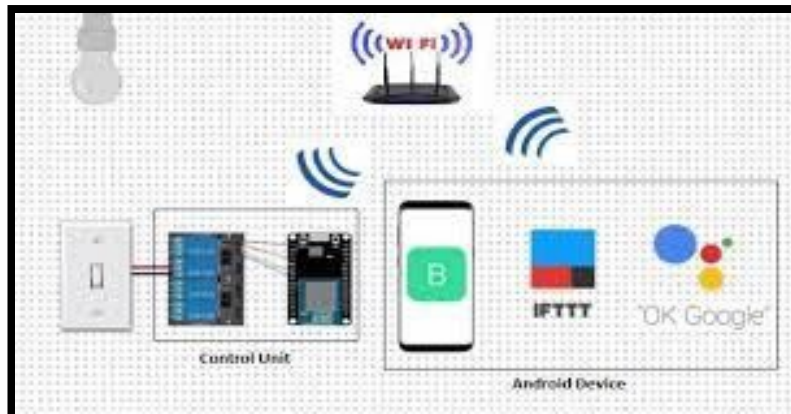


Fig – 1 Basic System Design Architecture

The hardware also called the Control Unit comprises of the Node MCU microcontroller and the Relay board. Node MCU's digital output pins are connected to the Relay pins of the Relay board. Finally, each Relay is connected to an appliance. In the fig- 1 above the second relay is connected to a bulb.

HARDWARE

NodeMCU (ESP8266)

The NodeMCU (Node MicroController Unit) is an open source software and hardware development environment that is built around a very inexpensive System-on-a-Chip (SoC) called the ESP8266. The ESP8266 contains all crucial elements of the modern computer: CPU, RAM, networking (wi-fi), and even a modern operating system and SDK. When purchased at bulk, the ESP8266 chip costs only \$2 USD a piece. That makes it an excellent choice for this system design. The NodeMCU aims to simplify ESP8266 development.

It has two key components:

- i. **An open source ESP8266 firmware** that is built on top of the chip manufacturer's proprietary SDK. The firmware provides a simple programming environment based on eLua (embedded Lua), which is a very simple and fast scripting language with an established developer community. The Lua scripting language is easy to learn. And to add on NodeMCU can be programmed with the Android IDE too.
- ii. **A development kit board** that incorporates the ESP8266 chip on a standard circuit board. The board has a built-in USB port that is already wired up with the chip, a hardware reset button, Wi-Fi antenna, LED lights, and standard-sized GPIO (General Purpose Input Output) pins that can plug into a bread board.

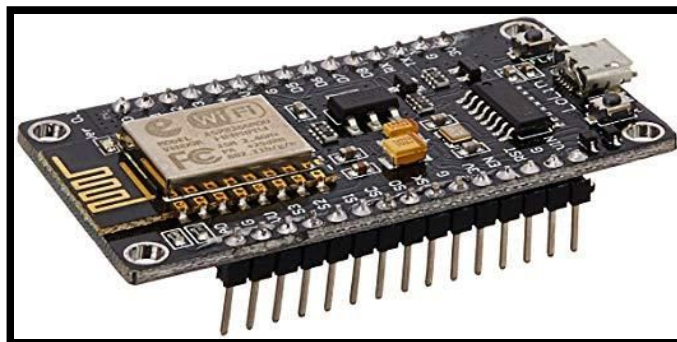


Fig - 2 : NodeMCU Development Board

RELAY BOARD

A relay is an electromagnetic switch. It is activated when a small current of some microampere is applied to it. Normally a relay is used in a circuit as a type of switch, an automatic switch. There are different types of relays and they operate at different voltages. When a circuit is built the voltage that will trigger it has to be considered.

In this system the relay circuit is used to turn the appliances ON/OFF. The high/low signal is supplied from the NodeMCU microcontroller. When a low voltage is given to the relay of an appliance it is turned off and when a high voltage is given it is turned on. The number of appliances can be modified according to the user's requirements. The relay circuit to drive four appliances in the Home automation system is shown below in figure 3.

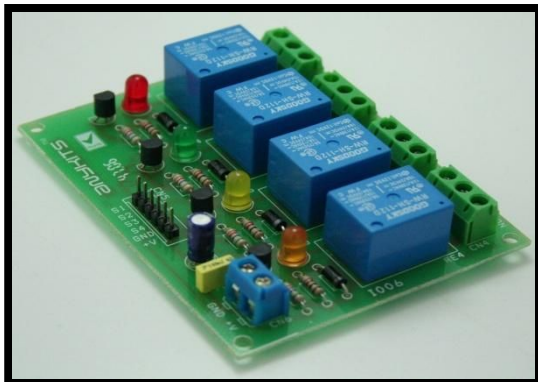


Fig – 3A : 4 Channel Relay Board

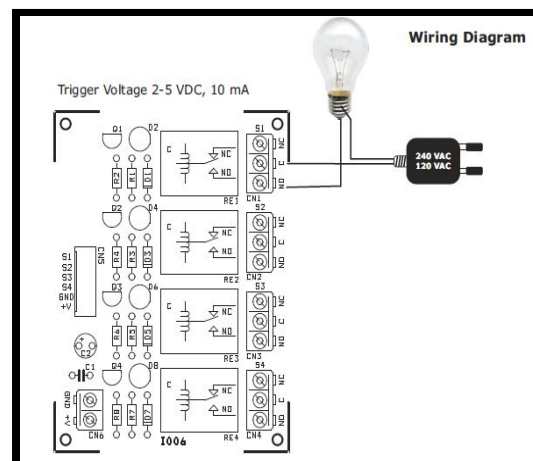


Fig – 3B : Wiring Diagram

ULN 2803 IC

ULN 2803 IC is used as a relay driver. It is a High voltage, high current Transistor Array IC used especially with Microcontrollers where we need to drive high power loads. This IC consists of eight NPN Darlington connected transistors with common Clamp diodes for switching the loads connected to the output and they also provide current amplification. This IC is widely used to drive high loads such Lamps, relays, motors etc. Most of the Chips operates with low level signals such as TTL, CMOS, PMOS, NMOS which operates at the range of (0-5) V and are incapable to drive high power inductive loads. However this chip takes low level input signals (TTL) and uses that to switch off the higher voltage loads that are connected to the output side.

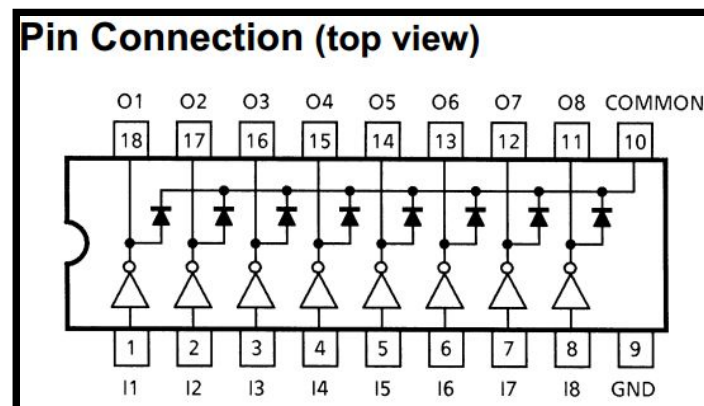


Fig – 4a ULN 2803 IC Pin Connection

A Darlington pair has two transistors that act as a single transistor providing high current gain. In this pair the current amplified by the first transistor is further amplified by the next providing high current to the output terminal.

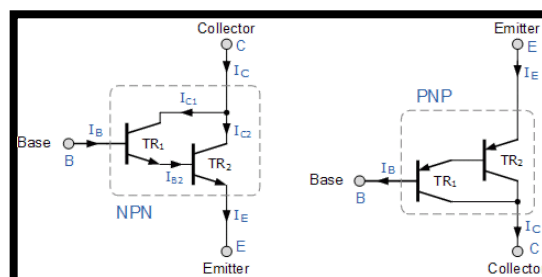


Fig – 4b

SOFTWARE

The software of the system proposed consists of mainly
The Blynk Application and The IFTTT application.

BLYNK APPLICATION

Blynk is a Platform with iOS and Android apps to control Arduino, Raspberry Pi, NodeMCU and several other boards over the Internet. Blynk was designed for the **Internet of Things**. It can control hardware remotely, it can display sensor data, it can store data, visualize it and do many other cool things. Blynk App setup is required; we set it up as per the requirement. We begin by creating a project and then selecting the microcontroller we are using. After which we create the toggle buttons for each relay associated with the digital pins of the microcontroller. Once this is done, Blynk sends an authentication token to the registered email id for this particular project. This token should be noted and saved for its use while programming the NodeMCU and setting up the IFTTT application,

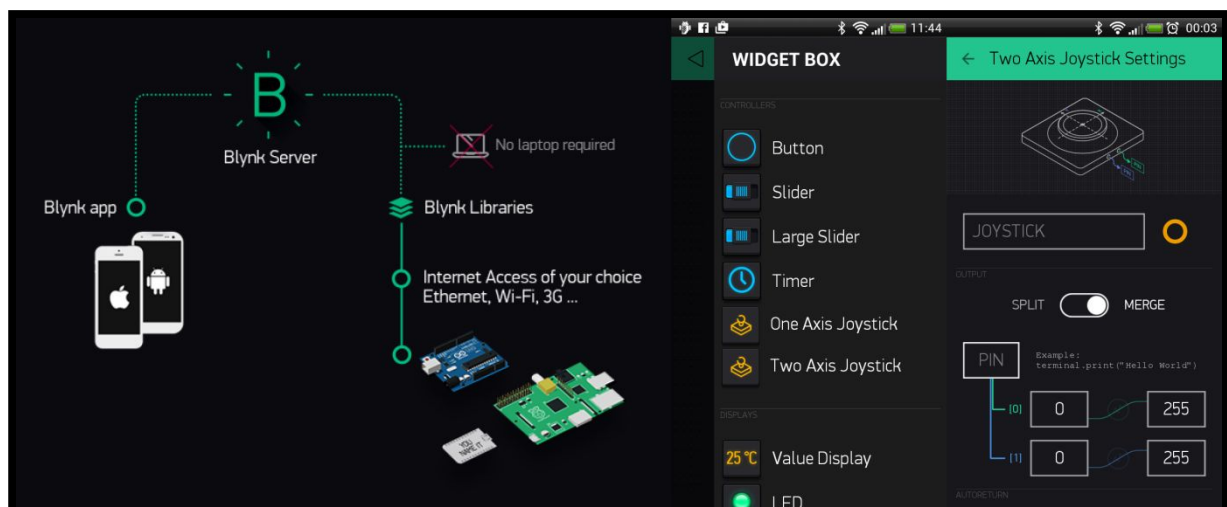


Fig – 5: BLYNK Application

IFTTT APPLICATION

IFTTT derives its name from the programming conditional statement **“if this, then that.”** IFTTT is both a website and a mobile app that launched in 2010 and has the slogan "Put the Internet to work for you". The idea is that you use IFTTT to automate everything from your favourite apps and websites to app-enabled accessories and smart devices. Here, IFTTT application is used to bridge the gap between the Google Assistant commands and the Blynk app.

Setting up the IFTTT application first requires logging in after which we need to create an applet and then **“This”**, i.e. the trigger, here we select Google Assistant and then we will type in the commands to which the Google Assistant should respond and to this command it should control the appliance/relay associated with it. The response command from the Google Assistant can also be typed in as desired. After configuring the trigger, i.e. **“This”** of the application we need to configure the **“That”**. What should be done once the Google Assistant hears the command which we just configured? This is decided by setting **“That”** of the app. We click **“That”** and then select webhooks and click connect. Webhooks will allow us to send commands to the Blynk Server. Now, in the URL we type the IP address of the Blynk server followed by the Authentication token sent by the Blynk and then the pin number of the microcontroller to which the device to be controlled is connected. The URL should be in the following format `http://188.166.206.43/AuthToken/pin/CorrespondingDigit alPinNo`

Then in the method we select ‘PUT’ and the content type is ‘Application/JSON’ and in the body we write [“1”] to turn ON and [“0”] to turn OFF. This creates the action for the trigger i.e. the Google Assistant command. The action taken by it is simply sending a message to the Blynk app to either turn ON or OFF the concerned connected device. Finally, the microcontroller is programmed with the actions it needs to do once it receives the signal from the Blynk application. Before that, the Blynk and the microcontroller should communicate and the communication is done via the internet and since the microcontroller, NodeMCU comes with inbuilt Wi-Fi module, it is programmed to connect to the desired network once plugged in. ‘C’ language is used to program the microcontroller and is programmed in the Arduino IDE .

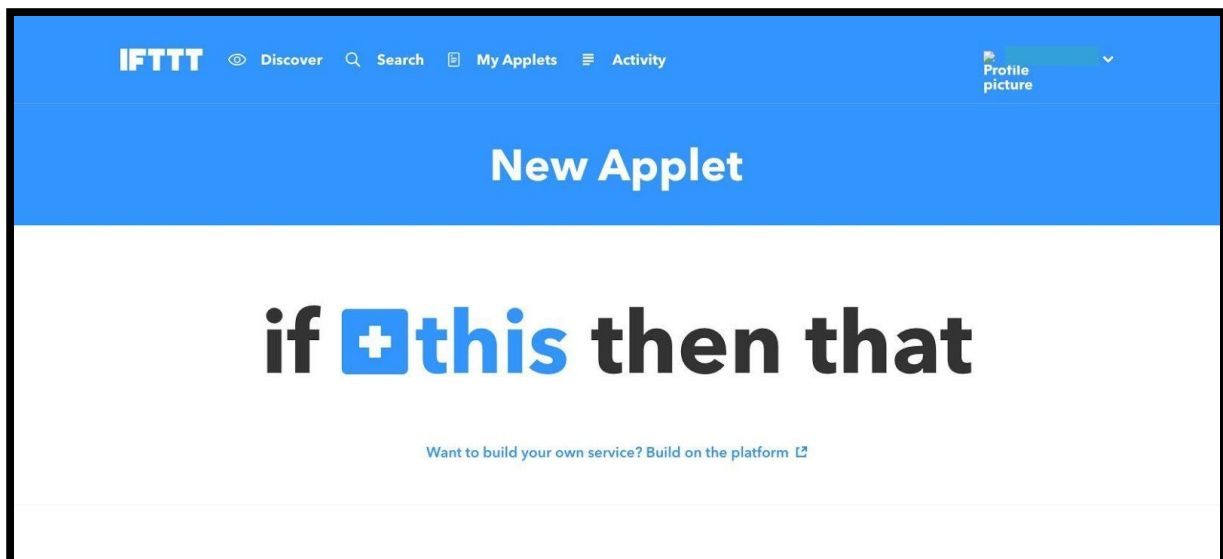


Fig – 6a : Creating a New Applet

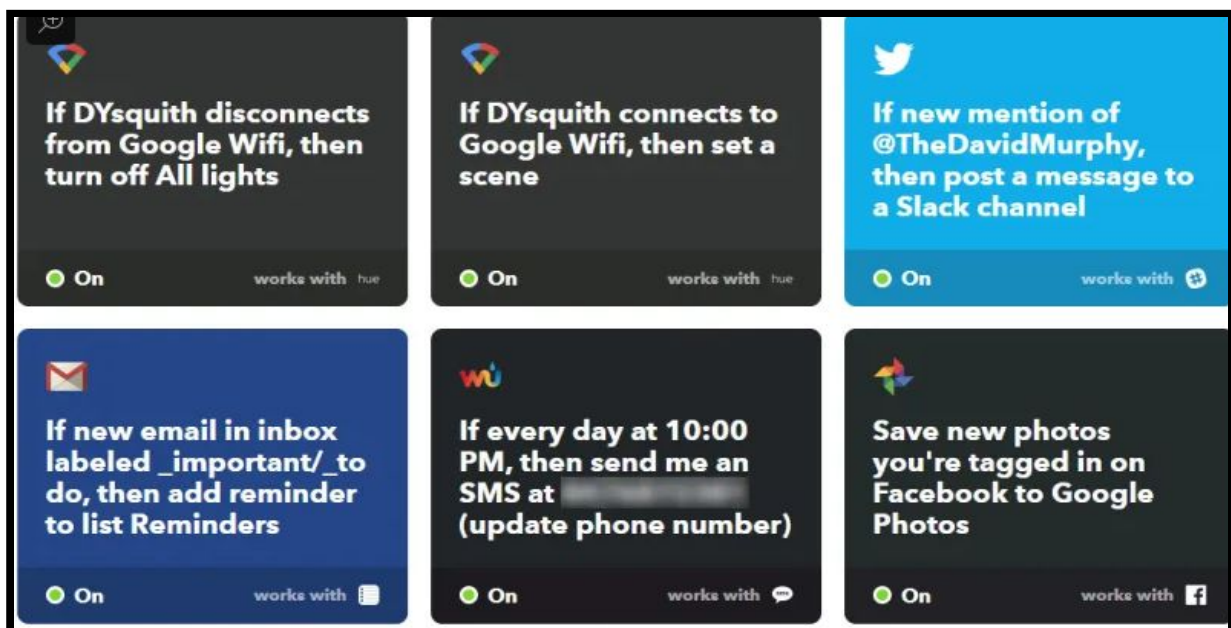


Fig- 6b : A Screenshot of the IFTTT Application after creating applets

CODE FOR CONFIGURING NodeMCU

```
#define BLYNK_PRINT Serial

#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

// You should get Auth Token in the Blynk App.
// Go to the Project Settings (nut icon).
char auth[] = "YourAuthToken";

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "YourNetworkName";
char pass[] = "YourPassword";

void setup()
{
  // Debug console
  Serial.begin(9600);

  Blynk.begin(auth, ssid, pass);
}

void loop()
{
  Blynk.run();
}
```

CODE FOR WEBSERVER

```
// Load Wi-Fi library
#include <ESP8266WiFi.h>

// Replace with your network credentials
const char* ssid    = "Darkshadow";
const char* password = "12345678";

// Set web server port number to 80
WiFiServer server(80);

// Variable to store the HTTP request
String header;

// Auxiliar variables to store the current output state
String output5State = "off";
String output4State = "off";

// Assign output variables to GPIO pins
const int output5 = 12;
const int output4 = 4;

// Current time
unsigned long currentTime = millis();
// Previous time
unsigned long previousTime = 0;
// Define timeout time in milliseconds (example: 2000ms = 2s)
const long timeoutTime = 2000;

void setup() {
  Serial.begin(115200);
  // Initialize the output variables as outputs
```

```

pinMode(output5, OUTPUT);
pinMode(output4, OUTPUT);
// Set outputs to LOW
digitalWrite(output5, LOW);
digitalWrite(output4, LOW);

// Connect to Wi-Fi network with SSID and password
Serial.print("Connecting to ");
Serial.println(ssid);
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
}
// Print local IP address and start web server
Serial.println("");
Serial.println("WiFi connected.");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());
server.begin();
}

void loop(){
    WiFiClient client = server.available(); // Listen for incoming clients

    if (client) { // If a new client connects,
        Serial.println("New Client."); // print a message out in the serial port
        String currentLine = ""; // make a String to hold incoming data from
the client
        currentTime = millis();
        previousTime = currentTime;
        while (client.connected() && currentTime - previousTime <= timeoutTime) {
// loop while the client's connected
            currentTime = millis();
            if (client.available()) { // if there's bytes to read from the client,
                char c = client.read(); // read a byte, then
                Serial.write(c); // print it out the serial monitor
                header += c;
            }
        }
    }
}

```

```

if (c == '\n') {           // if the byte is a newline character
    // if the current line is blank, you got two newline characters in a row.
    // that's the end of the client HTTP request, so send a response:
    if (currentLine.length() == 0) {
        // HTTP headers always start with a response code (e.g. HTTP/1.1 200
OK)
        // and a content-type so the client knows what's coming, then a blank
line:
        client.println("HTTP/1.1 200 OK");
        client.println("Content-type:text/html");
        client.println("Connection: close");
        client.println();

        // turns the GPIOs on and off
        if (header.indexOf("GET /5/on") >= 0) {
            Serial.println("GPIO 5 on");
            output5State = "on";
            digitalWrite(output5, HIGH);
        } else if (header.indexOf("GET /5/off") >= 0) {
            Serial.println("GPIO 5 off");
            output5State = "off";
            digitalWrite(output5, LOW);
        } else if (header.indexOf("GET /4/on") >= 0) {
            Serial.println("GPIO 4 on");
            output4State = "on";
            digitalWrite(output4, HIGH);
        } else if (header.indexOf("GET /4/off") >= 0) {
            Serial.println("GPIO 4 off");
            output4State = "off";
            digitalWrite(output4, LOW);
        }

        // Display the HTML web page
        client.println("<!DOCTYPE html><html>");
        client.println("<head><meta name=\"viewport\"
content=\"width=device-width, initial-scale=1\">");
        client.println("<link rel=\"icon\" href=\"data:.\",>");
        // CSS to style the on/off buttons

```

// Feel free to change the background-color and font-size attributes to fit your preferences

```
client.println("<style>html { font-family: Helvetica; display:
inline-block; margin: 0px auto; text-align: center; }");
client.println(".button { background-color: #195B6A; border: none;
color: white; padding: 16px 40px;");
client.println("text-decoration: none; font-size: 30px; margin: 2px;
cursor: pointer; }");
client.println(".button2 {background-color:
#77878A; }</style></head>");
```

// Web Page Heading

```
client.println("<body><h1>ESP8266 Web Server</h1>");
```

// Display current state, and ON/OFF buttons for GPIO 5

```
client.println("<p>GPIO 5 - State " + output5State + "</p>");
// If the output5State is off, it displays the ON button
if (output5State=="off") {
    client.println("<p><a href=\"/5/on\"><button
class=\"button\">ON</button></a></p>");
} else {
    client.println("<p><a href=\"/5/off\"><button class=\"button
button2\">OFF</button></a></p>");
}
```

// Display current state, and ON/OFF buttons for GPIO 4

```
client.println("<p>GPIO 4 - State " + output4State + "</p>");
// If the output4State is off, it displays the ON button
if (output4State=="off") {
    client.println("<p><a href=\"/4/on\"><button
class=\"button\">ON</button></a></p>");
} else {
    client.println("<p><a href=\"/4/off\"><button class=\"button
button2\">OFF</button></a></p>");
}
client.println("</body></html>");
```

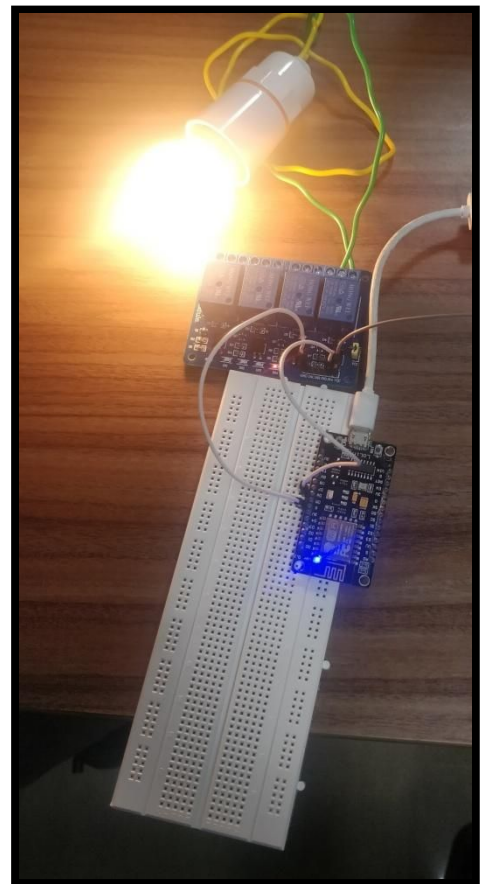
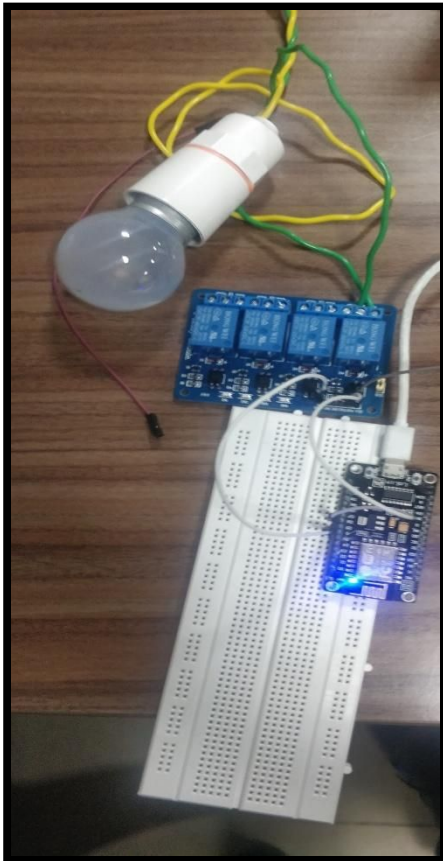
// The HTTP response ends with another blank line

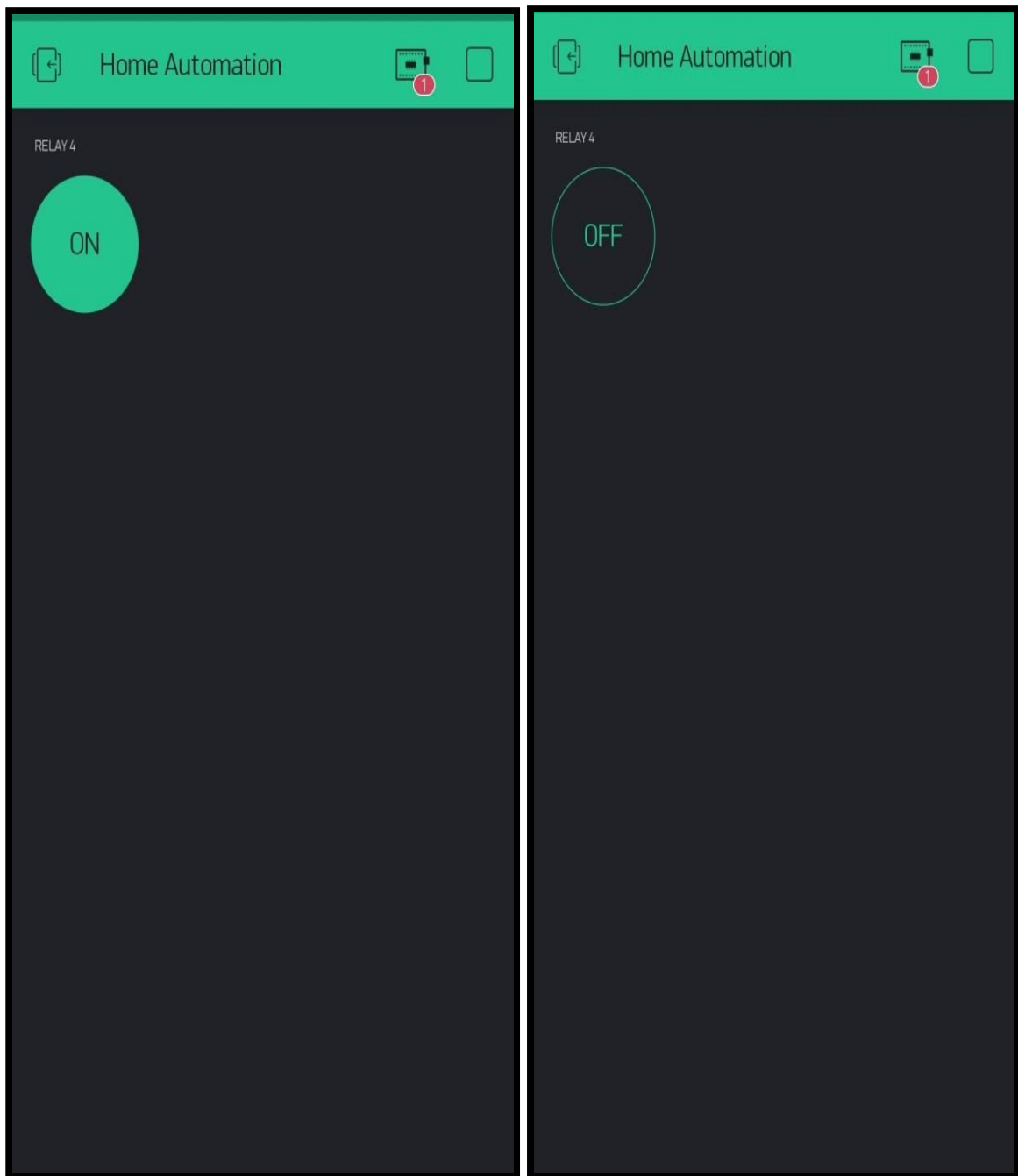
```

        client.println();
        // Break out of the while loop
        break;
    } else { // if you got a newline, then clear currentLine
        currentLine = "";
    }
    } else if (c != '\r') { // if you got anything else but a carriage return
character,
        currentLine += c;    // add it to the end of the currentLine
    }
}
}
// Clear the header variable
header = "";
// Close the connection
client.stop();
Serial.println("Client disconnected.");
Serial.println("");
}
}

```

TEST RESULTS

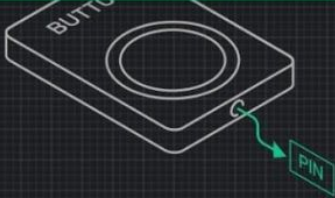




←

Button Settings

i



Relay 4

OUTPUT

D6

1

0

MODE

PUSH

SWITCH

ON/OFF LABELS

OFF

ON

DESIGN

FONT SIZE

T

T

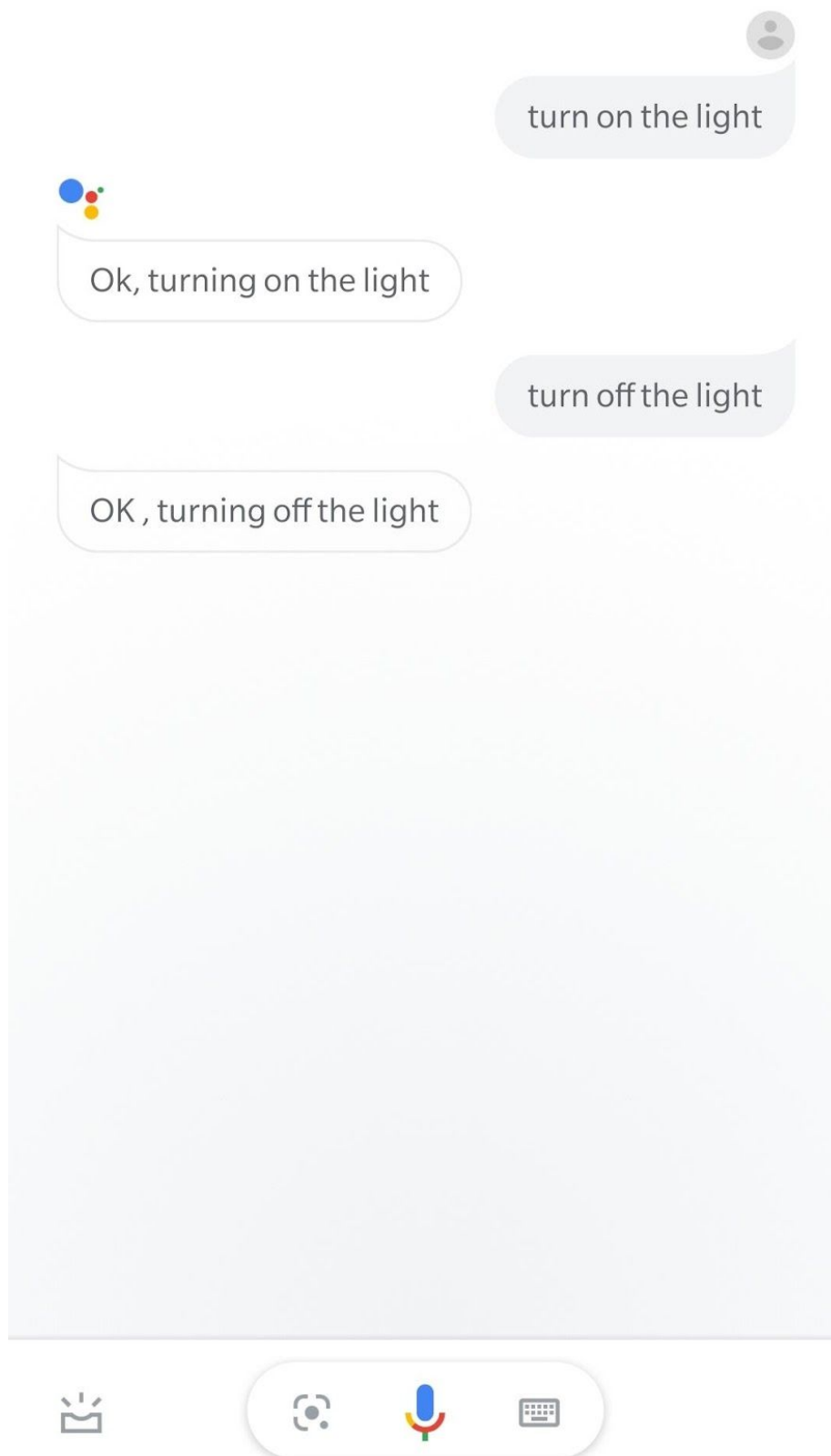
T

TEXT

✕

Delete

25



- All the three methods control the appliance successfully without any errors.

Conclusion

Imagine a future where we can move anything with just our mind. The idea of interfacing minds with machines has long captured the human imagination. The brain is an electrical device and electricity is its common language and this is what allows us to interface the brain to electronic devices. The brain is made up of billions of brain cells called neurons, which use electricity to communicate with each other. The combination of millions of neurons sending signals at once produces an enormous amount of electrical activity in the brain, which can be detected using sensitive medical equipment (such as an EEG), measuring electricity levels over areas of the scalp.

More improvement in this field can lead to control the electronic devices with the state of mind.

REFERENCES

- <https://www.wikipedia.org/>