**CHAPTER 1**

**INTRODUCTION**

**Home automation:**

Home automation is building automation for a home, called a smart home or smart house. A home automation system will monitor and/or control home attributes such as lighting, climate, entertainment systems, and appliances. It may also include home security such as access control and alarm systems. When connected with the Internet, home devices are an important constituent of the Internet of Things ("IoT").

A home automation system typically connects controlled devices to a central smart home hub (sometimes called a "gateway"). The user interface for control of the system uses either wall-mounted terminals, tablet or desktop computers, a mobile phone application, or a Web interface that may also be accessible off-site through the Internet.

While there are many competing vendors, there are increasing efforts towards open source systems. However, there are issues with the current state of home automation including a lack of standardized security measures and deprecation of older devices without backwards compatibility.

Home automation has high potential for sharing data between family members or trusted individuals for personal security and could lead to energy saving measures with a positive environmental impact in the future.

The home automation market was worth US$5.77 billion in 2013, predicted to reach a market value of US$12.81 billion by 2020.

In other words, “Home automation” refers to the automatic and electronic control of household features, activity, and appliances. In simple terms, it means you can easily control the utilities and features of your home via the Internet to make life more convenient and secure, and even spend less on household bills.

**How does home automation work?**

Home automation is a network of hardware, communication, and electronic interfaces that work to integrate everyday devices with one another via the Internet. Each device has sensors and is connected through WiFi, so you can manage them from your smartphone or tablet whether you’re at home, or miles away. This allows you to turn on the lights, lock the front door, or even turn down the heat, no matter where you are.

**What are the benefits of home automation?**

The purpose of a home automation system is to streamline how your home functions. Consider some of these benefits:

* *Remote access:* Control your home from mobile devices, including your laptop, tablet, or smartphone.
* *Comfort:* Use home automation to make your home a more comfortable, liveable space. Pre-program your thermostat with your preferred settings so that your home is always at a comfortable temperature, set up smart speakers to play music when you get home from work, or adjust your lights to soften or brighten based on the time of day.
* *Convenience:* Program devices to turn on automatically at certain times, or access their settings remotely from anywhere with an Internet connection. When you don’t have to remember to lock the door behind you or switch off the lights, you can turn your attention to more important things.
* *Increased safety:* Smart fire detectors, carbon monoxide monitors, pressure sensors, and other home automation security features can help protect your home from disaster.
* *Energy efficiency*: Home automation allows you to be more mindful of your power usage. For example, you can save on energy bills by reducing the length of time that lights stay on, or by lowering temperatures when you leave a room.

**What do we do in this project?**

In this project we will control an LED which is connected to ESP8266 with a Telegram bot. As we know about Telegram, it is a messenger app similar to Whatsapp. This enables users to control their ESP8266 via simply typing and sending commands in Telegram by creating a bot. By adding some relays or TRIAC you can make this a home automation project. We control the home automation by using that Telegram bot.

**Requirements:**

**Hardware:**

* ESP8266 NodeMCU
* Lamp
* Jumper wires
* Breadboard
* Relay module
* Battery / Power Supply

**Software:**

* Arduino IDE (v1.6 or above)
* Telegram Bot

We will see more details of each components in the next chapter.

**CHAPTER 2**

**IoT:**

The internet of things, or IoT, is a system of interrelated computing devices, mechanical and digital machines, objects, animals or people that are provided with unique identifiers (UIDs) and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction.

A thing in the internet of things can be a person with a heart monitor implant, a farm animal with a biochip transponder, an automobile that has built-in sensors to alert the driver when tire pressure is low or any other natural or man-made object that can be assigned an Internet Protocol (IP) address and is able to transfer data over a network.

Increasingly, organizations in a variety of industries are using IoT to operate more efficiently, better understand customers to deliver enhanced customer service, improve decision-making and increase the value of the business.

**How does IoT work?**

An IoT ecosystem consists of web-enabled smart devices that use embedded systems, such as processors, sensors and communication hardware, to collect, send and act on data they acquire from their environments. IoT devices share the sensor data they collect by connecting to an IoT gateway or other edge device where data is either sent to the cloud to be analyzed or analyzed locally. Sometimes, these devices communicate with other related devices and act on the information they get from one another. The devices do most of the work without human intervention, although people can interact with the devices -- for instance, to set them up, give them instructions or access the data.

The connectivity, networking and communication protocols used with these web-enabled devices largely depend on the specific IoT applications deployed.

IoT can also make use of artificial intelligence (AI) and machine learning to aid in making data collecting processes easier and more dynamic.

**Advantages of IoT:**

* Cost reduction
* Efficiency and productivity
* Business opportunities
* Customer experience
* Mobility and agility

**Arduino IDE:**

Arduino is an open-source hardware and software company, project, and user community that designs and manufactures single-board microcontrollers and microcontroller kits for building digital devices. Its hardware products are licensed under a CC BY-SA license, while software is licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL), permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially from the official website or through authorized distributors.



Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards ('shields') or breadboards (for prototyping) and other circuits. The boards feature serial communications interfaces, including Universal Serial Bus (USB) on some models, which are also used for loading programs. The microcontrollers can be programmed using the C and C++ programming languages, using a standard API which is also known as the Arduino language, inspired by the Processing language and used with a modified version of the Processing IDE. In addition to using traditional compiler toolchains, the Arduino project provides an integrated development environment (IDE) and a command line tool developed in Go.

**Node MCU ESP8266:**

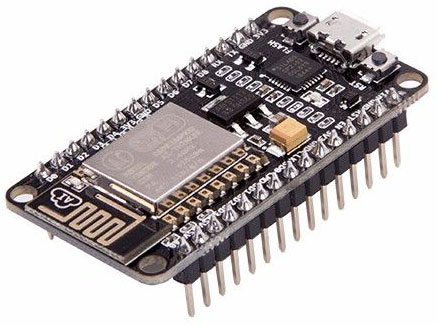
NodeMCU is a low-cost open source IoT platform. It initially included firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which was based on the ESP-12 module. Later, support for the ESP32 32-bit MCU was added.

NodeMCU is an open source firmware for which open source prototyping board designs are available. The name "NodeMCU" combines "node" and "MCU" (micro-controller unit).[8] The term "NodeMCU" strictly speaking refers to the firmware rather than the associated development kits.[citation needed]

Both the firmware and prototyping board designs are open source.

The firmware uses the Lua scripting language. The firmware is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson and SPIFFS. Due to resource constraints, users need to select the modules relevant for their project and build a firmware tailored to their needs. Support for the 32-bit ESP32 has also been implemented.

The prototyping hardware typically used is a circuit board functioning as a dual in-line package (DIP) which integrates a USB controller with a smaller surface-mounted board containing the MCU and antenna. The choice of the DIP format allows for easy prototyping on breadboards. The design was initially based on the ESP-12 module of the ESP8266, which is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core, widely used in IoT applications.



We use ESP-12 version in this project.

**Relay Module:**

The relay is the device that open or closes the contacts to cause the operation of the other electric control. It detects the undesirable condition with an assigned area and gives the commands to the circuit breaker to disconnect the affected area through ON or OFF.

Every electromechanical relay consists of

1. Electromagnet

3. Mechanically movable contact

3. Switching points and

4. Spring.

**One Channel Relay Module:**

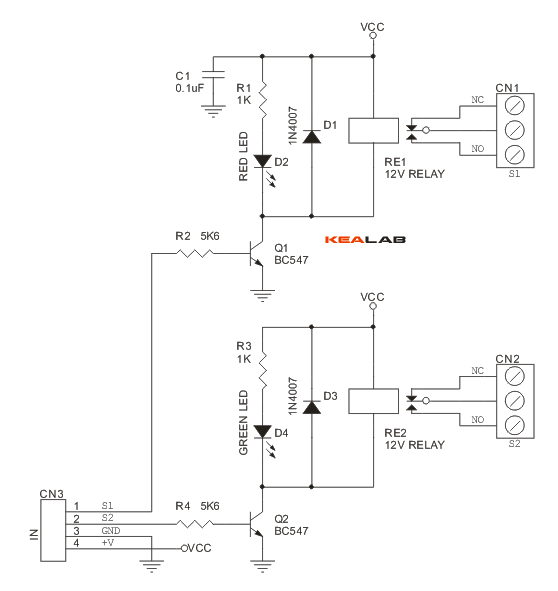


**COM:**common pin

**NO:** Normally open – there is no contact between the common pin and the normally open pin. So, when you trigger the relay, it connects to the COM pin and power is provided to the load.

**NC:** Normally closed – there is contact between the common pin and the normally closed pin. There is always connection between the COM and NC pins, even when the relay is turned off. When you trigger the relay, the circuit is opened and there is no supply provided to the load.

**Relay Module 2-Channel Circuit Diagram:**



.

**APPLICATIONS OF RELAY:**

A. They can be used for both ac and dc systems for protection of ac and dc equipment’s

B. Electromagnetic relays operating speeds which has the ability to operate in milliseconds are also can be possible

C. They have the properties such as simple, robust, compact and most reliable

D. These relays are almost instantaneous. Though instantaneous the operating time of the relay varies with the current. With extra arrangements like dashpot, copper rings.

E. Electromagnetic relays have fast operation and fast reset

**DISADVANTAGES:**

a. High burden level instrument transformers are required (CTs and PTs of high burden is required for operating the electromagnetic relays compared to static relays)

b. The directional feature is absent in electromagnetic relays

c. Requires periodic maintenance and testing unlike static relays

d. Relay operation can be affected due to ageing of the components and dust, pollution resulting in spurious trips

e. Operation speed for an electromagnetic relays is limited by the mechanical inertia of the component.

**CHAPTER 3**

**Installing the libraries in Arduino IDE:**

**Step 1** − First you must have your Arduino board (you can choose your favorite board) and a USB cable. In case you use Arduino UNO, Arduino Duemilanove, Nano, Arduino Mega 2560, or Diecimila, you will need a standard USB cable (A plug to B plug), the kind you would connect to a USB printer as shown in the following image.

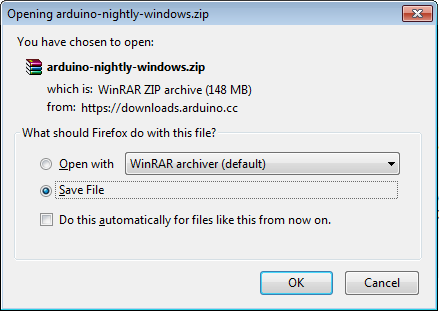


In case you use Arduino Nano, you will need an A to Mini-B cable instead as shown in the following image.



**Step 2 − Download Arduino IDE Software.**

You can get different versions of Arduino IDE from the [Download page](https://www.arduino.cc/en/Main/Software) on the Arduino Official website. You must select your software, which is compatible with your operating system (Windows, IOS, or Linux). After your file download is complete, unzip the file.



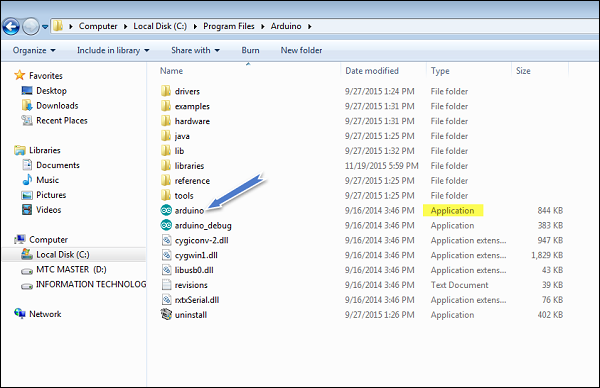
**Step 3 − Power up your board.**

The Arduino Uno, Mega, Duemilanove and Arduino Nano automatically draw power from either, the USB connection to the computer or an external power supply. If you are using an Arduino Diecimila, you have to make sure that the board is configured to draw power from the USB connection. The power source is selected with a jumper, a small piece of plastic that fits onto two of the three pins between the USB and power jacks. Check that it is on the two pins closest to the USB port.

Connect the Arduino board to your computer using the USB cable. The green power LED (labeled PWR) should glow.

**Step 4 − Launch Arduino IDE.**

After your Arduino IDE software is downloaded, you need to unzip the folder. Inside the folder, you can find the application icon with an infinity label (application.exe). Double-click the icon to start the IDE.

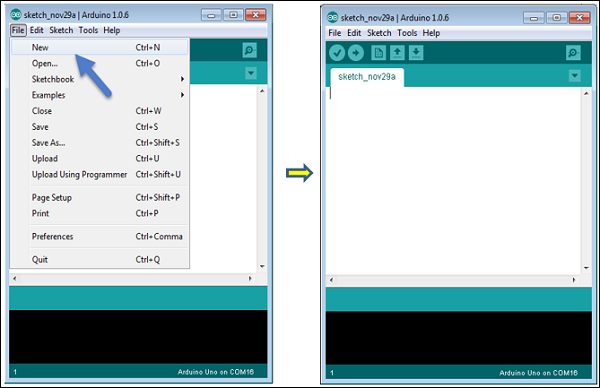


**Step 5 − Open your first project.**

Once the software starts, you have two options −

* Create a new project.
* Open an existing project example.

To create a new project, select File → **New**.



To open an existing project example, select File → Example → Basics → Blink.

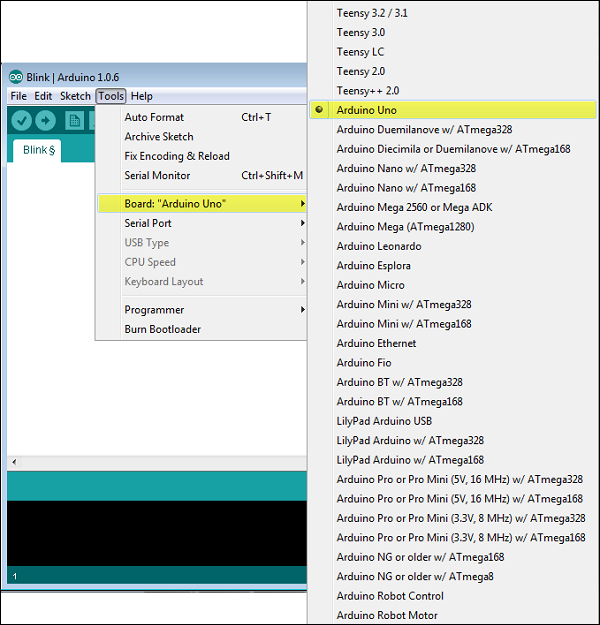
Open Project

Here, we are selecting just one of the examples with the name **Blink**. It turns the LED on and off with some time delay. You can select any other example from the list.

**Step 6 − Select your Arduino board.**

To avoid any error while uploading your program to the board, you must select the correct Arduino board name, which matches with the board connected to your computer.

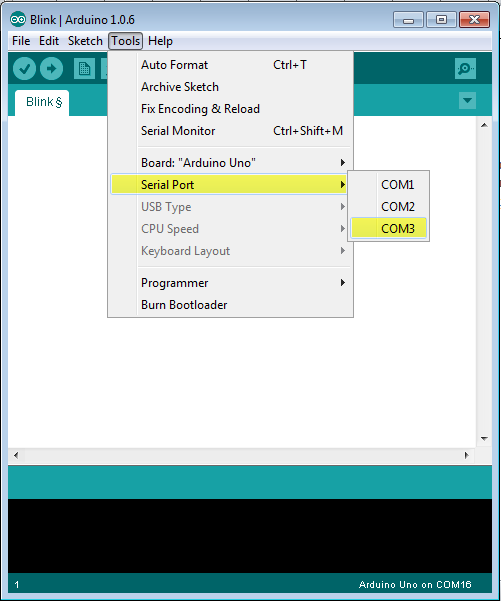
Go to Tools → Board and select your board.



Here, we have selected Arduino Uno board according to our tutorial, but you must select the name matching the board that you are using.

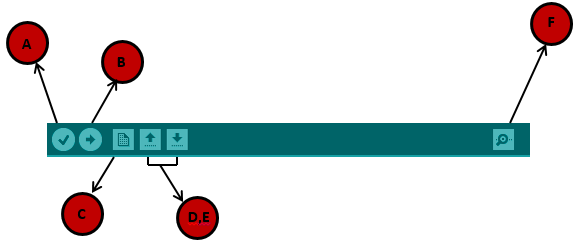
**Step 7 − Select your serial port.**

Select the serial device of the Arduino board. Go to **Tools → Serial Port** menu. This is likely to be COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports). To find out, you can disconnect your Arduino board and re-open the menu, the entry that disappears should be of the Arduino board. Reconnect the board and select that serial port.



**Step 8 − Upload the program to your board.**

Before explaining how we can upload our program to the board, we must demonstrate the function of each symbol appearing in the Arduino IDE toolbar.



**A** − Used to check if there is any compilation error.

**B** − Used to upload a program to the Arduino board.

**C** − Shortcut used to create a new sketch.

**D** − Used to directly open one of the example sketch.

**E** − Used to save your sketch.

**F** − Serial monitor used to receive serial data from the board and send the serial data to the board.

Now, simply click the "Upload" button in the environment. Wait a few seconds; you will see the RX and TX LEDs on the board, flashing. If the upload is successful, the message "Done uploading" will appear in the status bar.

**Setting Up NodeMCU ESP8266 Board into Arduino IDE:**

**Step 1: Connect Your NodeMCU to the Computer**

Use the USB cable to connect your NodeMCU to the computer,you will see the blue onboard LED flicker when powered up, but they will not stay lit.

**Step 2: Install the COM/Serial Port Driver**

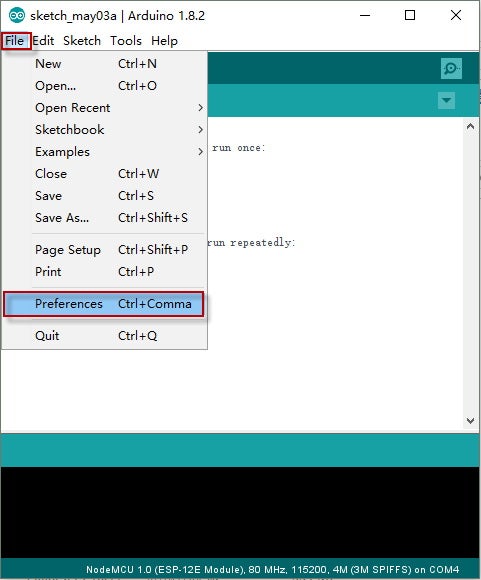
In order to upload code to the ESP8266 and use the serial console, connect any data-capable micro USB cable to ESP8266 IOT Board and the other side to your computer’s USB port.

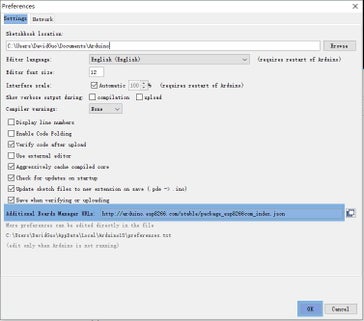
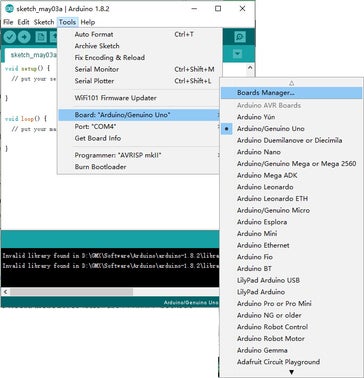
**Step 3: ​Install the Arduino IDE 1.6.4 or Greater**

Download Arduino IDE from Arduino.cc (1.6.4 or greater) – don’t use 1.6.2! You can use your existing IDE if you have already installed it. You can also try downloading the ready-to-go package from the ESP8266-Arduino project, if the proxy is giving you problems

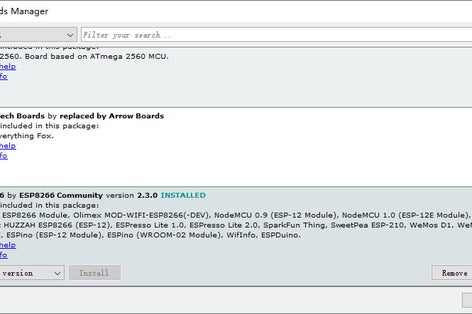
**Step 4: ​Install the ESP8266 Board Package**

Enter http://arduino.esp8266.com/stable/package\_esp8266... into Additional Board Manager URLs field in the Arduino v1.6.4+ preferences (Open Arduino IDE–>File–>Preferences–>Settings). Enter the link and click “OK” to save your changes. Next, use the Board Manager to install the ESP8266 package Enter the Boards Manager and find the board type as below: Scroll the Broads Manager screen down to the bottom, you will see A module called “esp8266 by esp8266 Community” (see following picture), select the latest version and click “Install“. The ESP8266 package has been installed successfully.

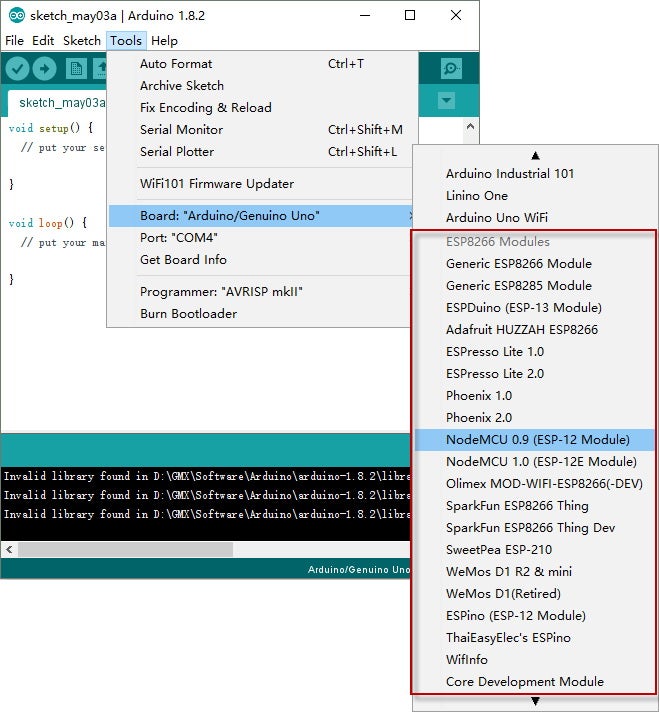
[](https://content.instructables.com/ORIG/F75/FYRT/J7QGGU8U/F75FYRTJ7QGGU8U.jpg?auto=webp&frame=1&fit=bounds&md=1edfd54931b4cbdba26dfda43e918119)

[](https://content.instructables.com/ORIG/F4L/BKVJ/J7QGGU8X/F4LBKVJJ7QGGU8X.jpg?auto=webp&frame=1&fit=bounds&md=15adda4693ff5b192f84b8b3d893cc25)[](https://content.instructables.com/ORIG/FKY/D5SW/J7QGGU8Y/FKYD5SWJ7QGGU8Y.jpg?auto=webp&frame=1&fit=bounds&md=a07a01401c17a724e073dfe27fbbc28f)

[](https://content.instructables.com/ORIG/FO7/1VAY/J7QGGU8Z/FO71VAYJ7QGGU8Z.jpg?auto=webp&frame=1&fit=bounds&md=b5f9697de099389cab098ae9bd5a89d3)

[](https://content.instructables.com/ORIG/FJV/U4SE/J7QGGU98/FJVU4SEJ7QGGU98.jpg?auto=webp&frame=1&fit=bounds&md=09dd54ea8547696a7ee009a8ee7b2714)

**Step 5: Setup ESP8266 Support**





When you’ve restarted, select NodeMCU 0.9 (or NodeMCU 1.0) from the Tools->Board dropdown Config the Board menu and choose the right Port for your device. CPU Frequency：80MHz,Flash Size：4M（3M SPIFFS）,Upload Speed：115200 Now just proceed as the Arduino: Start your sketching! Note: 115200 baud upload speed is a good place to start – later on you can try higher speeds but 115200 is a good safe place to start.

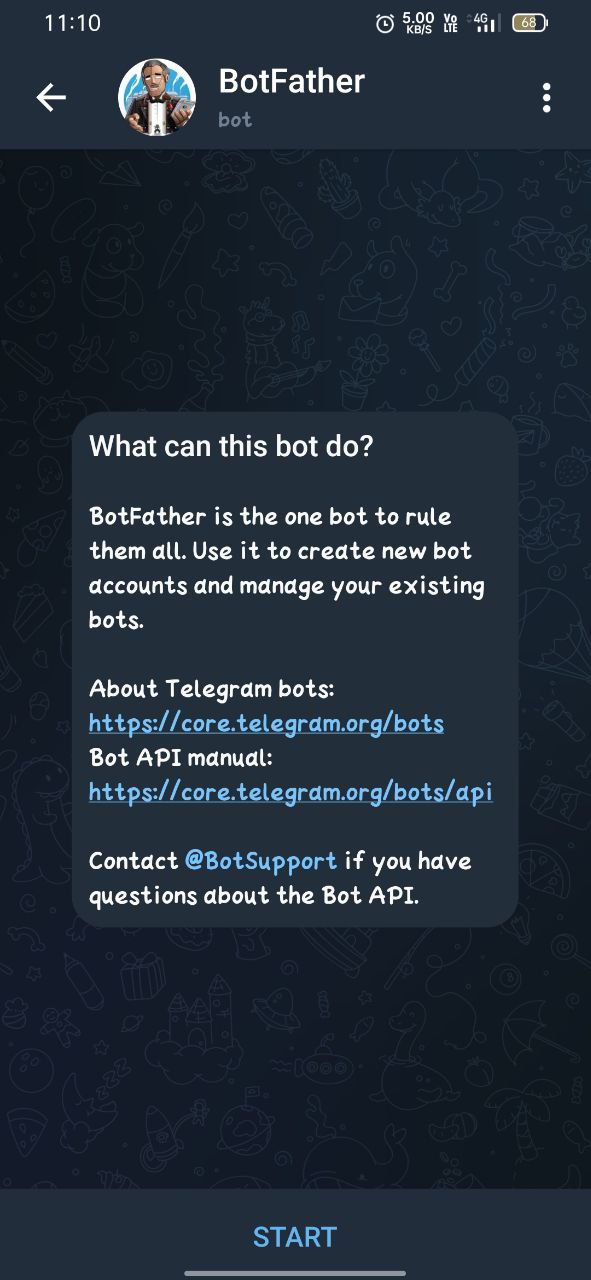
**CHAPTER 4**

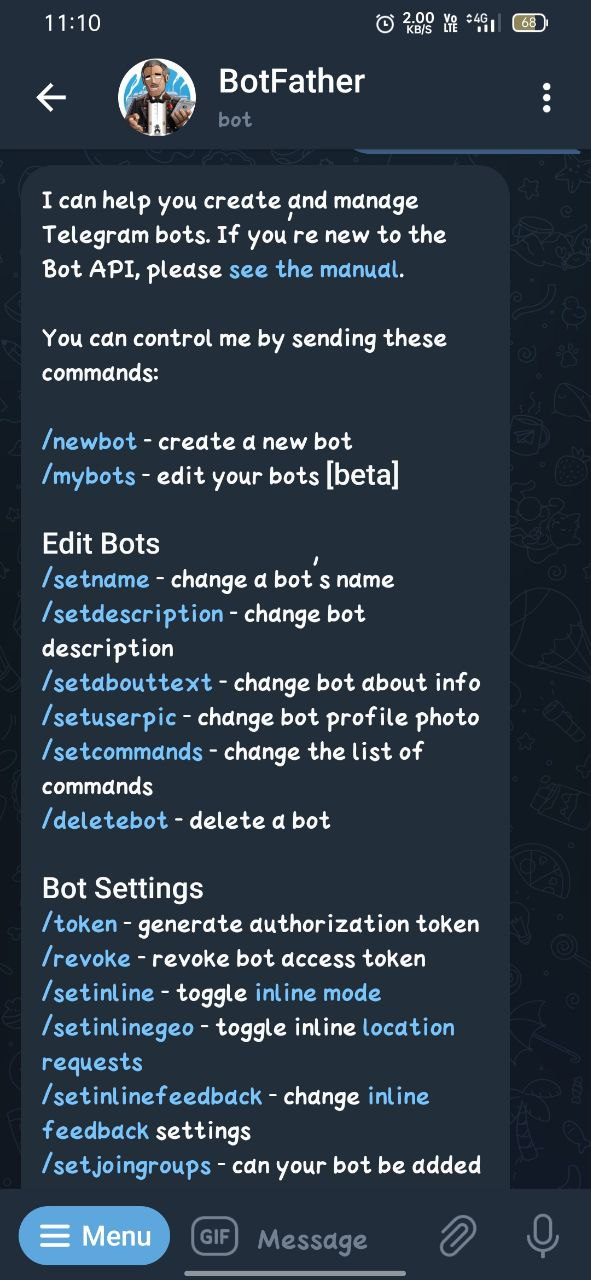
**Creation of Telegram Bot:**

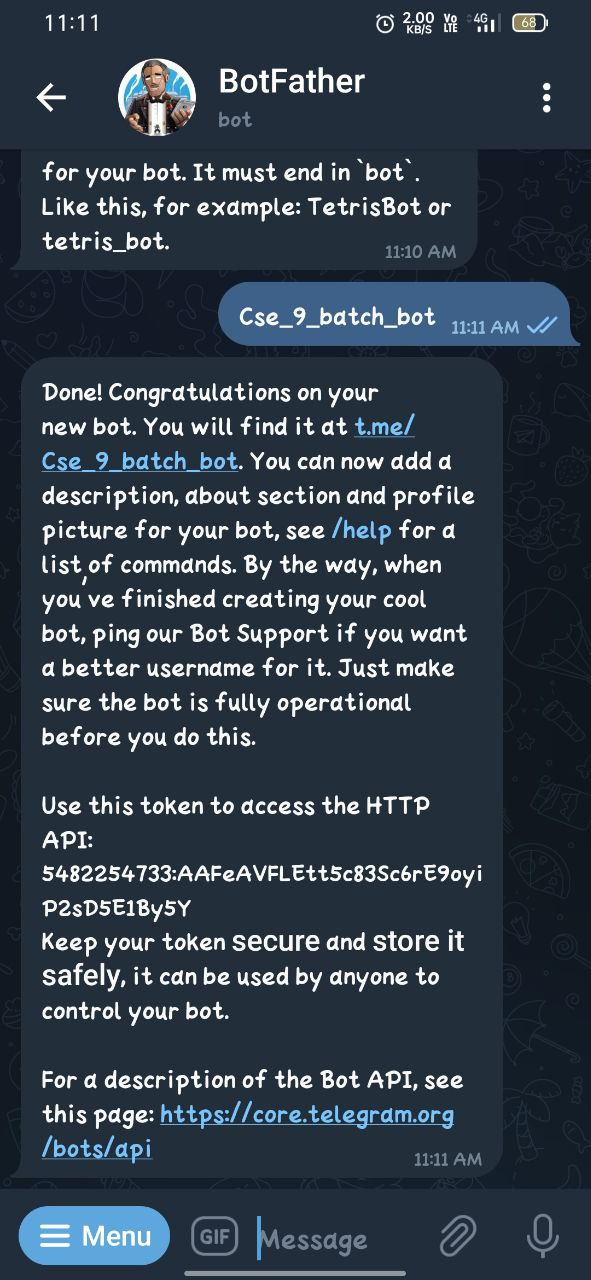
Telegram is a messaging application which is used to send text, images or video messages free of cost. It also allows using other API to create programs to integrate Telegram in their applications. There are special bots which do not need phone number to set up and can be interfaced with any embedded or software application to trigger some event using telegram text messages.

Install “Telegram” app on your smartphone from your play store or apple store. Sign Up if you are going to use it for the first time with your country code and mobile number. It will ask for OTP code for verification.

After successfully sign up, search for BotFather and open it. Open BotFather and click on RESTART for creating your bot as shown in screenshot below. It will show set of codes which can be used to create and edit bots. Type “/newbot” in the text section for creating new bot.







Now you have to choose a name for your bot. Type the name of your bot and click on send. Here I gave “Smart Home 🏡” name to my bot. Now it’s time to choose a unique username for your bot. Here I have given “@Cse\_9\_batch\_bot” username to my bot. This username should be unique. After giving username you will receive a text from BotFather about your new bot and will get a token for the new bot. This token will be used in code for NodeMCU. Click on the link given in the message to go to your new chat box of the bot and then click on Start. You can now send your commands by send a text message from Telegram app.



**CHAPTER 5**

**Algorithm:**

**Step 1:** Start

**Step 2:** Check whether WIFI is connected to NODEMCU , if WIFI connected goto Step3. Else goto Step 9.

**Step 3:** Code in NodeMCU is processed and now telegram bot will be active.

**Step 4:** If the Telegram access token is true then goto Step 5.Else goto Step 9.

**Step 5:** /start

**Step 6:** /helpme

**Step 7:** Now 4 commands will be displayed. /LIGHT\_ON,/LIGHT\_OFF,/helpme,/bye

**Step 8:** If /LIGHT\_ON:

Light will on and bot replies as LIGHT will be ON.

Else If /LIGHT\_OFF:

Light off and bot replies ad LIGHT will OFF.

Else If /helpme :

Goto Step 6.

Else If /bye:

Nothing happens and bot replies as Byee See you soon.

Else:

Goto Step 4.

**Step 9:** End.

**Flow Chart:**

The below is the flow chart designed for our project.

**/LIGHT\_OFF**

Bye. See you soon

Light is now OFF.

Light is now ON

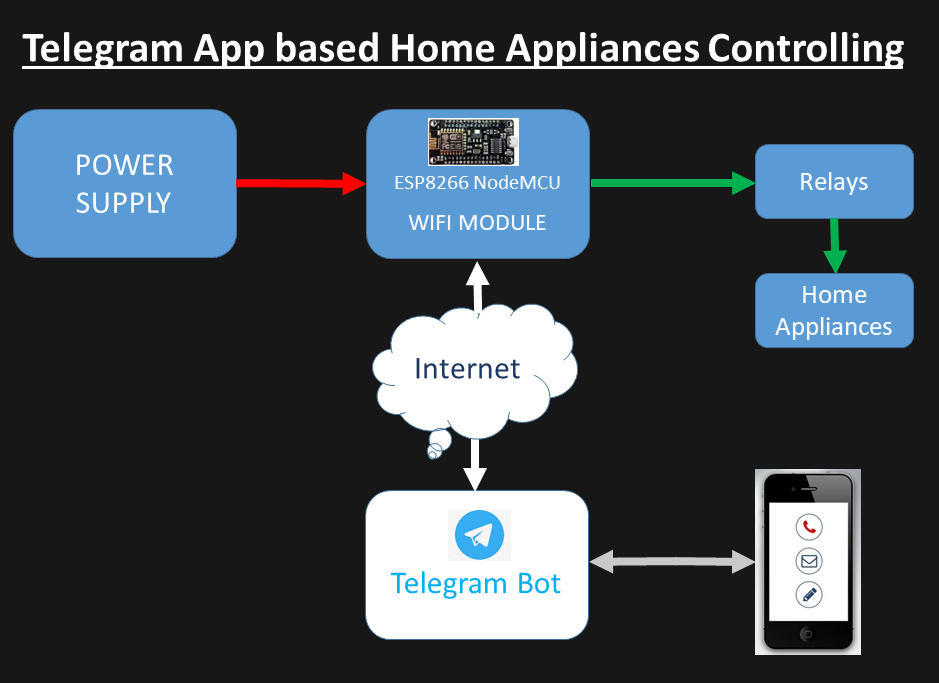
**/bye**

**/help\_me**

**/LIGHT\_ON**

**/helpme**

**/start**



**CHAPTER 6**

### **Code Explanation**

**Complete code with a working video** for this IoT controlled Home Automation is given at the end of this tutorial, here we are explaining the complete program to understand the working of the project.

Replace SSID and password with your Wi-Fi credentials. String token stores the unique token number you got after creating your bot. D0 is the digital output pin for changing the state of the relay.

**String ssid = "\*\*\*\*\*\*\*\*\*\*"; // Replace with your ssid**

**String pass = "\*\*\*\*\*\*\*\*\*\*\*\*"; // Replace with your password**

**String token = "66\*\*\*\*885:A\*G-X\*\*dTYdSCt\*\*\*\*\*\*aCQPCk\*\*\*SL\*\*b4"; // token number of your bot**

**int led = D0; // digital pin on NodeMCU**

myBot.wifiConnect() function takes the SSID and password to connect ESP8266 with the Wi-Fi and function myBot.setTelegramToken() takes the unique token number which further establishes the connection between telegram bot and NodeMCU. On successful connection you get “testConnection OK” on the serial monitor.

**myBot.wifiConnect(ssid, pass);**

**myBot.setTelegramToken(token);**

**if (myBot.testConnection())**

**Serial.println("\ntestConnection OK");**

**else**

**Serial.println("\ntestConnection NOK");**

TBMessage stores the message received from the telegram bot. msg.text.equalsIgnoreCase() checks if the text received matches with string “light on” or “light off”. If the message received is “Light on” it changes the relay state to turn on the lamp and if the message received is “Light off” then it changes the relay state to turn off the lamp. If the text received is different from “light on” and “light off” then it sends welcome message back to the telegram.

**TBMessage msg;**

**if (myBot.getNewMessage(msg)) {**

**if (msg.text.equalsIgnoreCase("LIGHT ON")) {**

**digitalWrite(led, HIGH);**

**myBot.sendMessage(msg.sender.id, "Light is now ON");**

**}**

**else if (msg.text.equalsIgnoreCase("LIGHT OFF")) {**

**digitalWrite(led, LOW);**

**myBot.sendMessage(msg.sender.id, "Light is now OFF");**

**}**

**else {**

**String reply;**

**reply = (String)"Welcome " + msg.sender.username + (String)". Try LIGHT ON or LIGHT OFF.";**

**myBot.sendMessage(msg.sender.id, reply);**

**}**

**}**

Now after completing the code its time to test the system.

**Source Code:**

#include "CTBot.h"

CTBot myBot;

String ssid = "VARSHA"; // REPLACE mySSID WITH YOUR WIFI SSID

String pass = "9866028800";

// REPLACE myPassword YOUR WIFI PASSWORD, IF ANY

String token = "5482254733:AAFeAVFLEtt5c83Sc6rE9oyiP2sD5E1By5Y"; // REPLACE myToken WITH YOUR TELEGRAM BOT TOKEN

uint8\_t led = D0; // the onboard ESP8266 LED.

// If you have a NodeMCU you can use the BUILTIN\_LED pin

// (replace 2 with BUILTIN\_LED)

void setup() {

// initialize the Serial

Serial.begin(115200);

Serial.println("Starting TelegramBot...");

// connect the ESP8266 to the desired access point

myBot.wifiConnect(ssid, pass);

// set the telegram bot token

myBot.setTelegramToken(token);

// check if all things are ok

if (myBot.testConnection())

Serial.println("\ntestConnection OK");

else

Serial.println("\ntestConnection NOK");

// set the pin connected to the LED to act as output pin

pinMode(led, OUTPUT);

digitalWrite(led, HIGH); // turn off the led (inverted logic!)

}

void loop() {

// a variable to store telegram message data

TBMessage msg;

// if there is an incoming message...

if (myBot.getNewMessage(msg)) {

if (msg.text.equalsIgnoreCase("/LIGHT\_ON")) { // if the received message is "LIGHT ON"...

digitalWrite(led, LOW); // turn on the LED (inverted logic!)

myBot.sendMessage(msg.sender.id, "Light is now ON 💡 "); // notify the sender

}

else if (msg.text.equalsIgnoreCase("/LIGHT\_OFF")) { // if the received message is "LIGHT OFF"...

digitalWrite(led, HIGH); // turn off the led (inverted logic!)

myBot.sendMessage(msg.sender.id, "Light is now OFF"); // notify the sender

}

else if (msg.text.equals("/helpme")) {

digitalWrite(led, HIGH);

myBot.sendMessage(msg.sender.id, "Try sending following commands\n");

myBot.sendMessage(msg.sender.id, "/LIGHT\_ON\n");

myBot.sendMessage(msg.sender.id, "/LIGHT\_OFF\n");

myBot.sendMessage(msg.sender.id, "/helpme\n");

myBot.sendMessage(msg.sender.id, "/Byee\n");

}

else if (msg.text.equals("/Byee")){

String r;

r = (String)"Byeeee " + msg.sender.username + (String)" .See you soon 👋 !!! ";

myBot.sendMessage(msg.sender.id, r);

}

else { // otherwise...

// generate the message for the sender

String reply;

reply = (String)"Welcome 😃 " + msg.sender.username + (String)".Try /helpme";

myBot.sendMessage(msg.sender.id, reply); // and send it

}

}

// wait 500 milliseconds

delay(500);

}

**Implementation:**

Here 1st we need to load the code into NodeMCU and then we need to connet the NodeMCU with power and WIFI connection so that it works fine.

After the code gets compiled successfully we need to open the telegram bot.

After opening the Telegram Bot we need to click on START. Now the bot starts replying the user. Bot keeps the commands which the user can use. The user needs to click on the command so that the command will work.

If it is Light On then the light which we connected will ON.

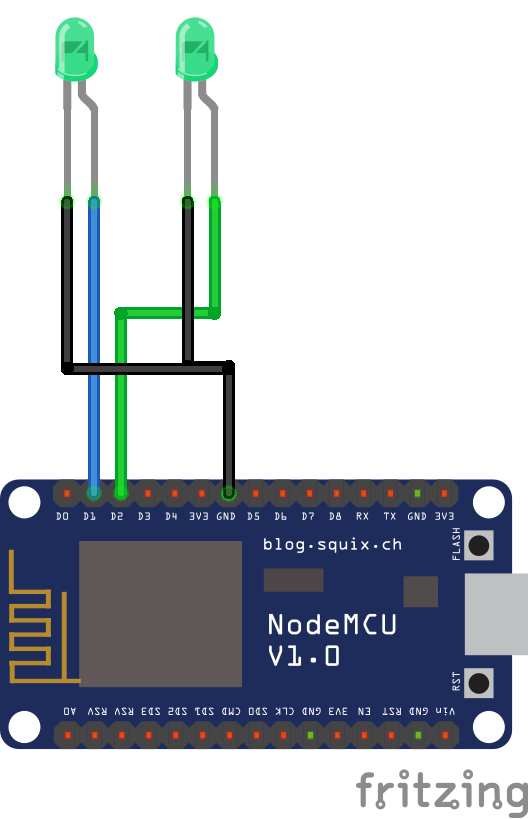
If it is Light Off then the light which we connected will go OFF.

If it is Byee then it Sends off the user with a message.

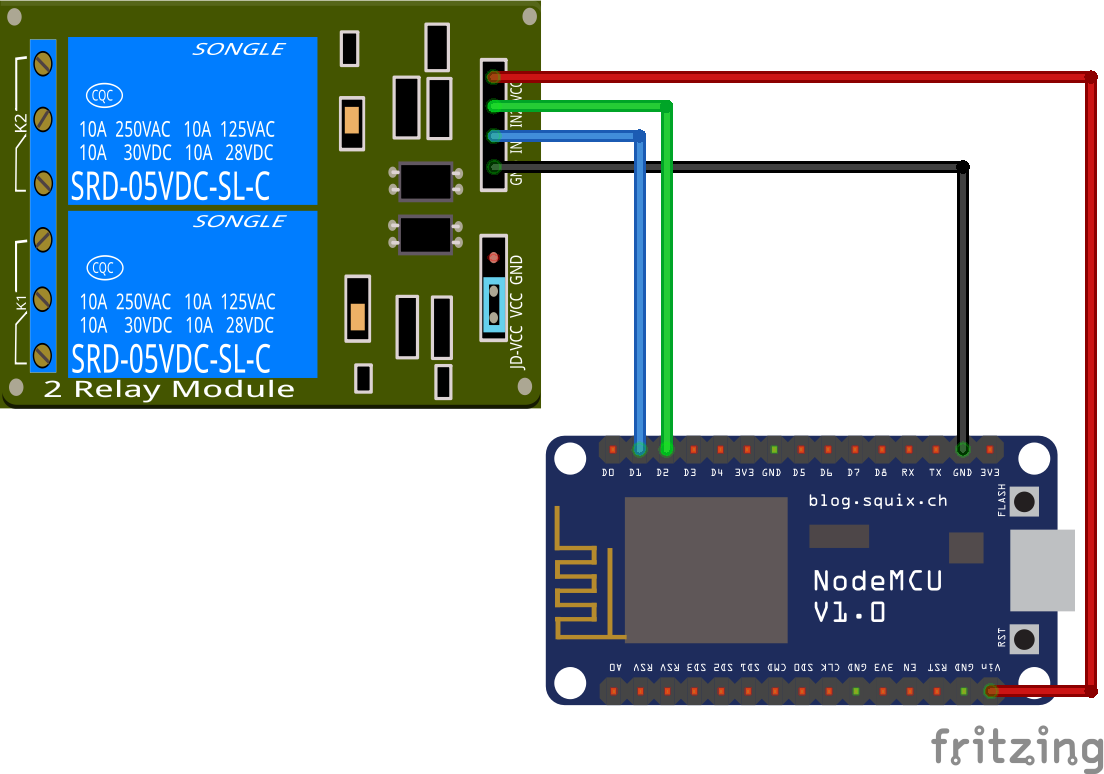
If it is Help me the loop will be continued.

**Circuit Diagrams:**

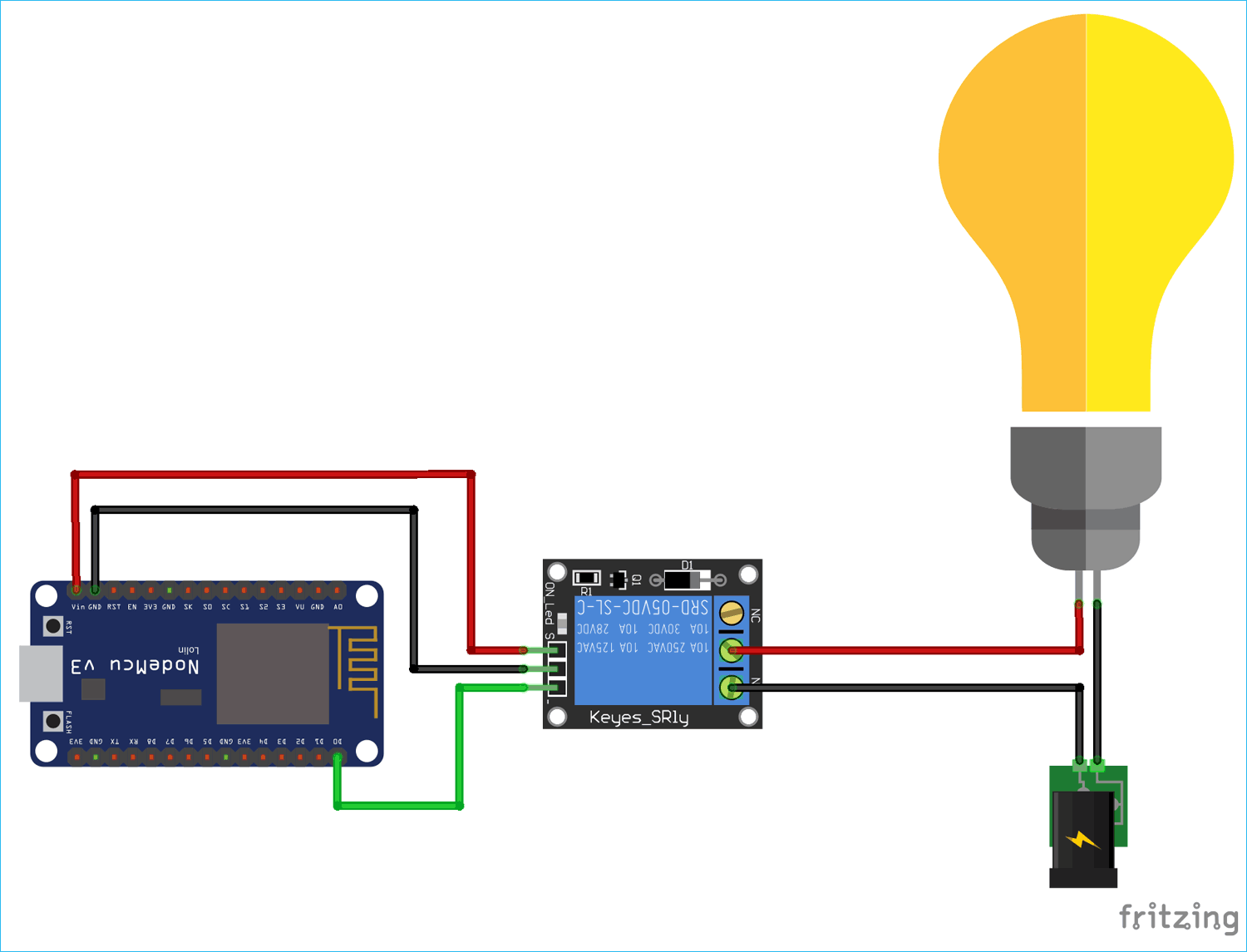
**Node MCU**



**Relay Module:**



**Project Circuit Diagram:**

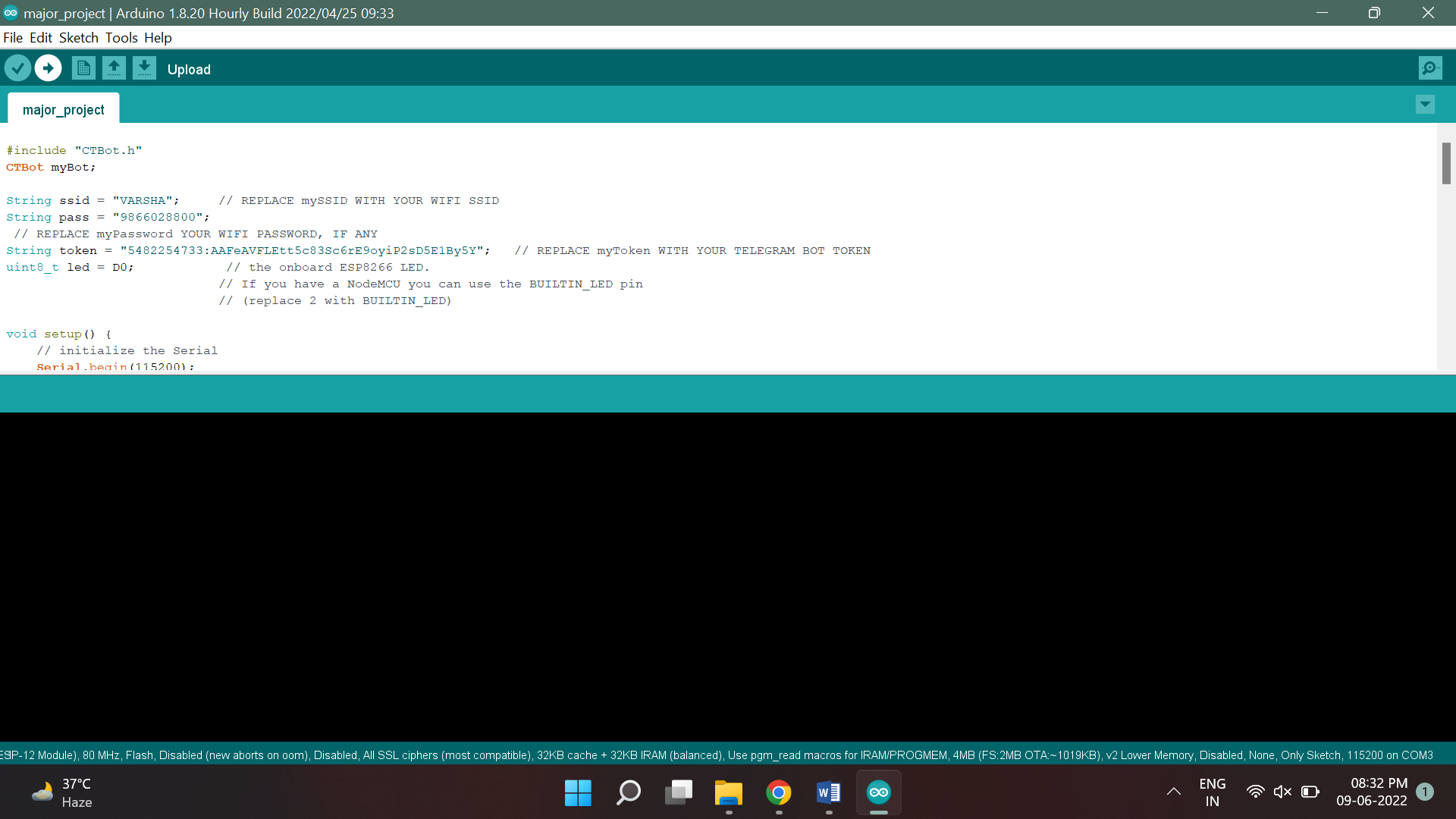


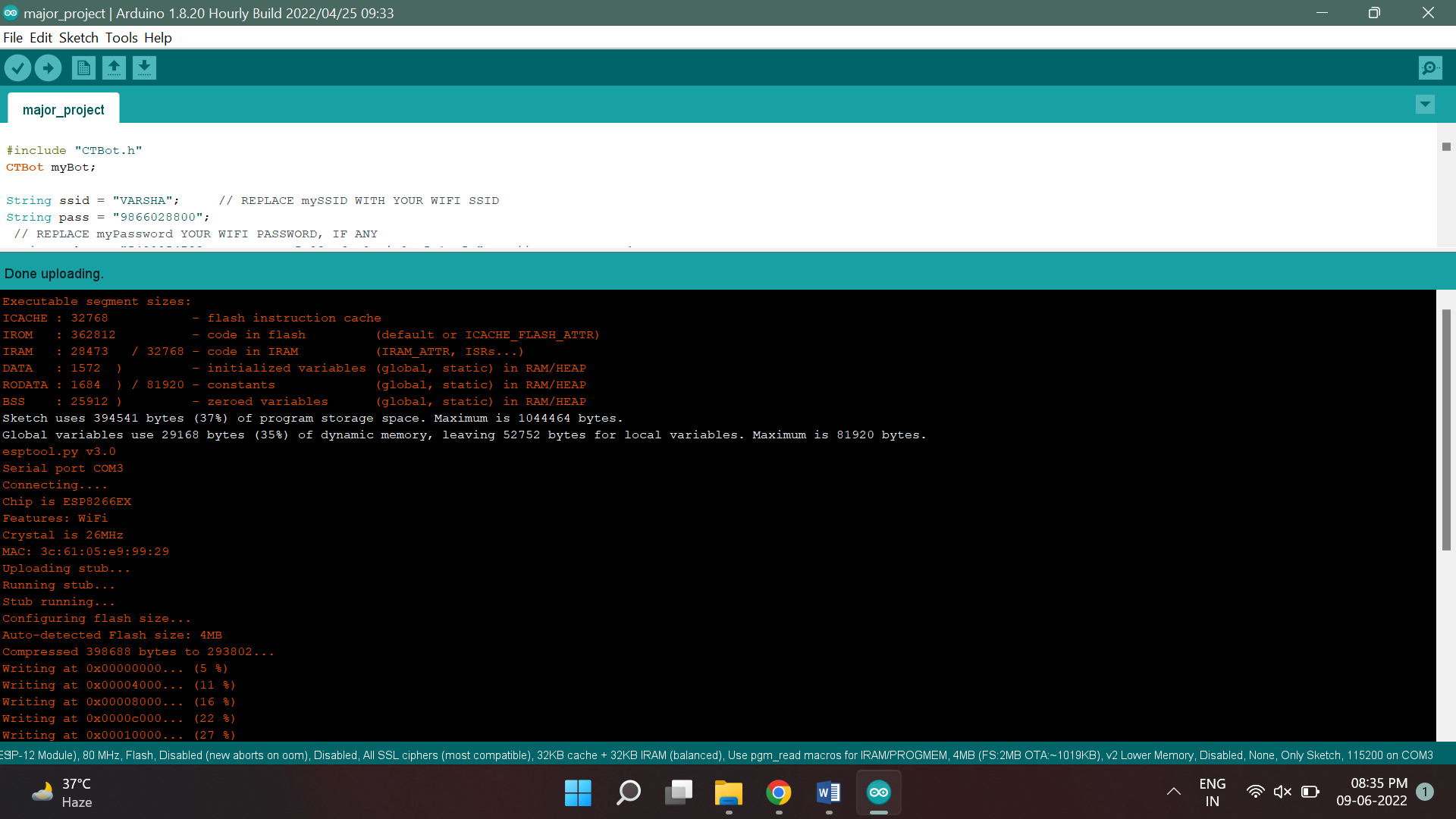
**Output:**

**Software:**

**Arduino IDE:**

Firstly we need to select the board and tools in the arduino which we downloaded earlier.



Here we have uploaded the code . 



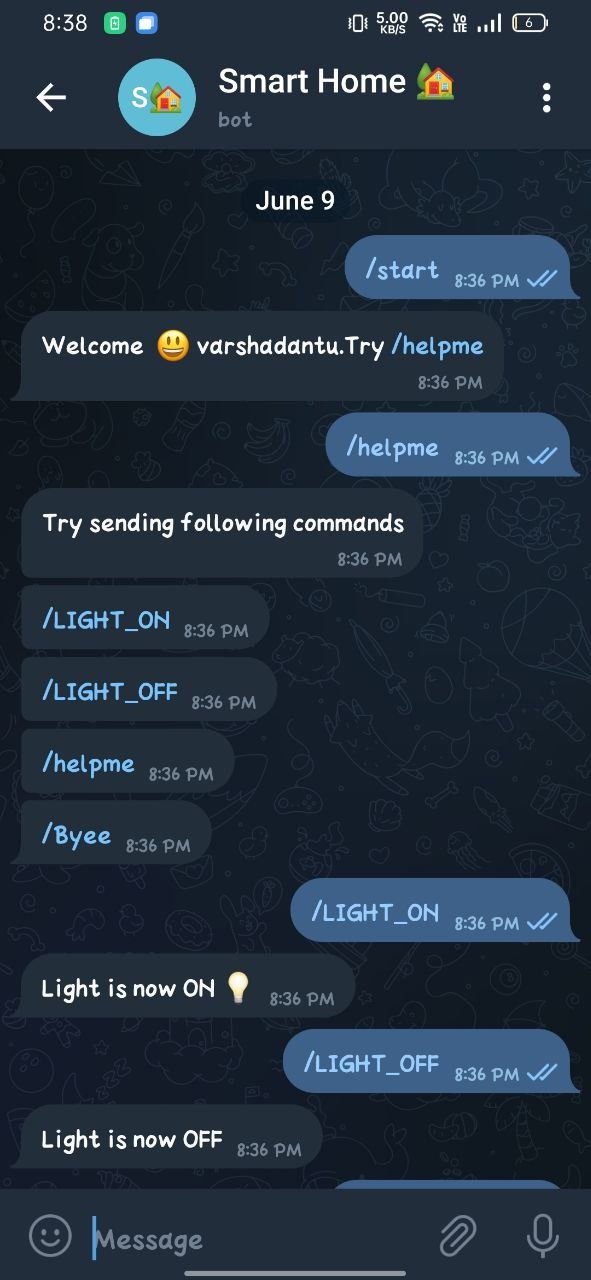
Now the code compiled successfully and done uploading into the NODEMCU.

**Telegram Bot:**

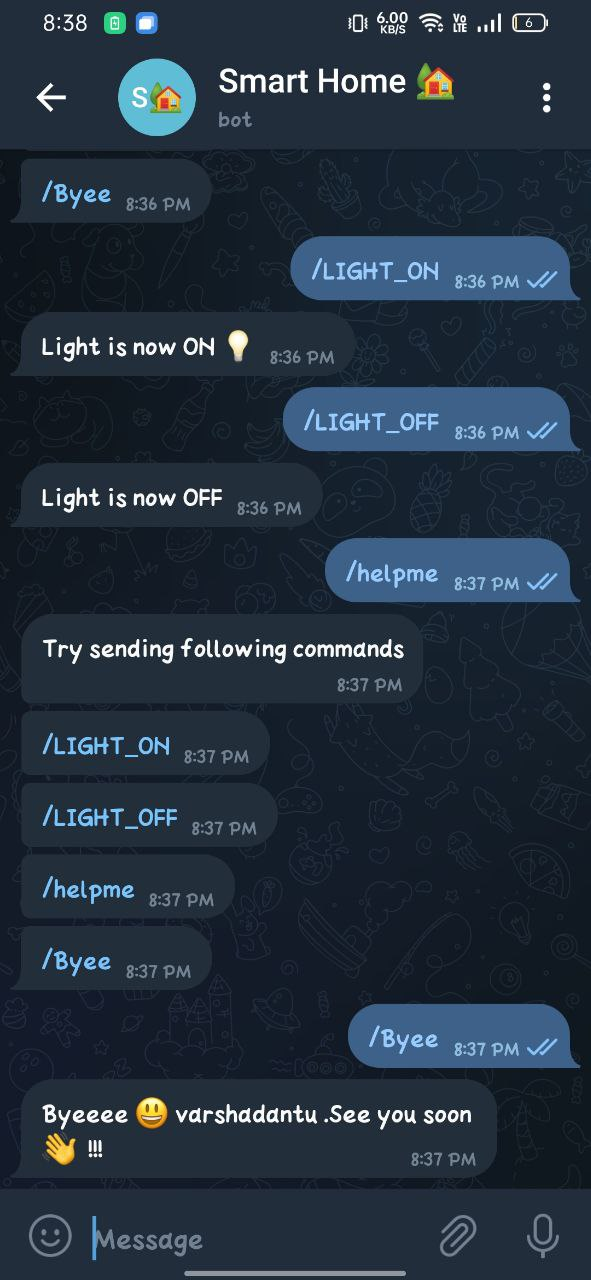


/start 🡪 /helpme 🡪 /LIGHT\_ON 🡪 Light is now ON.

/start 🡪 /helpme 🡪 /LIGHT\_OFF 🡪 Light is now OFF.



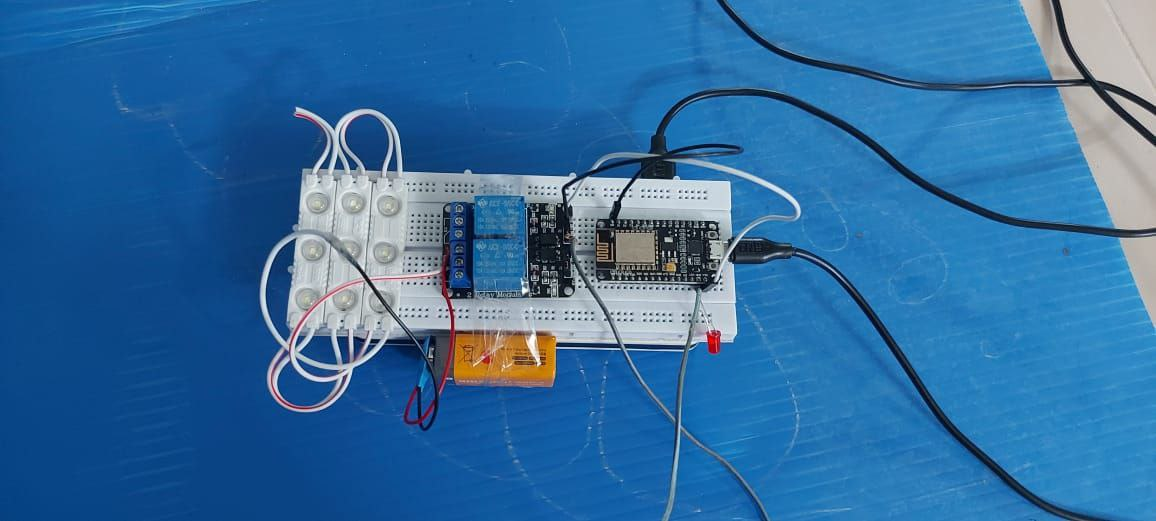
/start🡪 /helpme🡪/Byee 🡪 Byee (username). See you soon.



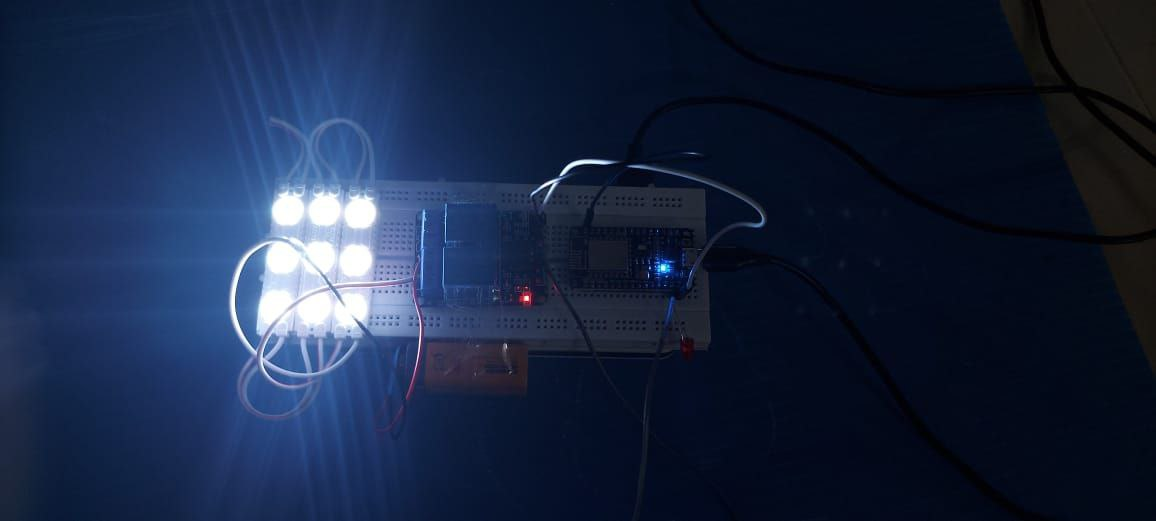
**Hardware:**

Whole figure.

Light Off:



Light On:



**CHAPTER 6**

**Conclusion**

This project is easy to build and can be used effectively. This project responds slowly and this is a major drawback of this project. Else this project is good as you don’t have to install any other app to control your devices. You can edit this code and control 4 or 6 devices.

In a intense study of Internet of Things, I found it to be hypothetical based on its purpose of application. It means IoT provides a lot of automation by connecting things. Connecting things has been made easy by the various sensors and embedding them them to the devices. In my instance of application with telegram messenger, I have used bots to communicate with the connected things in a house. Later by the statistical report given by the bot any user can take the decision on automation of house. With the emerging technologies, I have been successful in exploring the connectivity of various things in a house and also as a network specialist I also reported the pros and cons of connecting things. In future I would see the connected things at a different applicative arena, where it could be used in all contingency situations. It is found that IoT can be implemented and integrated with any software application. So this thesis is just an instance of IoT's implementation using a messenger application.

Now we know how to create a Telegram Bot to interact with the ESP32 or ESP8266. With this bot, you can use your Telegram account to send messages to the ESP and control its outputs. The ESP can also interact with the bot to send responses.

We’ve shown you a simple example on how to control an output. The idea is to modify the project to add more commands to execute other tasks. For example, you can request sensor readings or send a message when motion is detected.

The great thing about using Telegram to control your ESP boards, is that as long as you have an internet connection (and your boards too), you can control and monitor them from anywhere in the world.

**REFERENCES**

1. Ishan Krishna, K. Lavanya,”Intelligent Home Automation System using BitVoicer”, 11th International Conference on Intelligent Systems and Control, 2017.
2. Ramón Alcarria, Diego Martín de Andrés, “A Service-Oriented Monitoring System Based on Rule Evaluation for Home Automation”, IEEE 2016.
3. Hattie Clougherty, Alec Brown, Margaret Stonerock, “Home Automation and Personalization through Individual Location Determination”,IEEE 978-1-5386-1848-6/17/$31.00 2017.
4. ShibliNisar, Muhammad Asadullah, “Home Automation Using Spoken Pashto Digits Recognition”, IEEE 978-1-5090-3310-2/17/$3\.00 2017.
5. Sukhen Das, souvikghosh, RishirajSarker, “A Bluetooth Based Sophisticated Home Automation system Using Smartphone”, international conference on intelligent power and instrumentation, 2016.
6. Juan Carlos de Oliveira, Danilo Henrique Santos,” Chatting with Arduino Platform through Telegram Bot”, IEEE International Symposium on Consumer Electronics, 2016.
7. Vedan rattan vasta , Gopal singh, “Raspberry Pi based Implementation of Internet of Things using Mobile Messaging Application -„Telegram‟”, International Journal of Computer Applications (0975 – 8887), Volume 145 – No.14, July 2016.
8. Ankush B.powar, shasikanth gumhre, “A SURVEY ON IoT APPLICATIONS, SECURITY CHALLENGES AND COUNTER MEASURES”,international conference on computing, dec 19-21, 2016.