**HOME AUTOMATION USING GOOGLE ASSISTANT**

A PROJECT REPORT

submitted by

**ANJU K V**

**KSD15EC016**

**SANOJ C R**

**KSD15EC061**

**SHILPA T**

**KSD15EC067**

to

the APJ Abdul Kalam Technological University

in partial fulfillment of the requirements for the award of the Degree

of

Bachelor of Technology

In

*Electronics and Communication Engineering*

****

**Department of Electronics and Communication Engineering**

LBS College of Engineering Kasaragod

Povval

MAY 2019

**DECLARATION**

We undersigned hereby declare that the project report “Home Automation Using Google Assistant”, submitted for partial fulfillment of the requirements for the award of degree of Bachelor of Technology of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by us under supervision of Ms. Dilna C. This submission represents our ideas in our own words and where ideas or words of others have been included, We have adequately and accurately cited and referenced the original sources. We also declare that we have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in our submission. We understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

Povval

14-05-2019 ANJU K V

SANOJ C R

SHILPA T

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

**LBS COLLEGE OF ENGINEERING KASARAGOD, POVVAL**

****

**CERTIFICATE**

This is to certify that the report entitled ‘**HOME AUTOMATION USING GOOGLE ASSISTANT’** submitted by **ANJU K V, SANOJ C R, SHILPA T,** to APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Electronics and Communication Engineering is a bonafide record of the project work carried out by them under our guidance and supervision. This report in any form has not been submitted to any other University or Institute for any purpose.

Head of The Department Project Guide

**Dr. Mary Reena K E Ms. Dilna C**

Associate Professor Assistant Professor

Dept of ECE, LBSCEK Dept of ECE, LBSCEK

Place: Povval

Date: 14-05-2019

**ACKNOWLEDGEMENT**

We have taken efforts in this project. However it would not have been possible without the kind support and help of many individuals. We would like to extend our sincere thanks to all of them. We express our gratitude to our Head of The Department, Dr. Mary Reena K E for providing the opportunity to do this project. We are highly indebted to Ms. Dilna C for her guidance and constant supervision as well as for providing necessary information regarding the project. Our thanks and appreciation to all our colleagues in developing the project and people who have willingly helped us out with their abilities.

**ABSTRACT**

World’s demand for electricity had grown 85% between 2010 and 2017 this increase is more than today’s total use of electricity in India, USA, Japan, Australia combined. We can’t decrease the electricity growth rate but we can lessen the amount of electricity wasted each year by turning off our home appliances when not in use. This project presents a design and prototype of Home Automation system that will use ESP8266 Wi-Fi module as a network provider in connecting with other appliances. The proposed system has two main components. The main part is NodeMCU which consists of a microcontroller and Wi-Fi module. Through Wi-Fi module a web server can be added to the module which will help in controlling of devices over Internet. One server can manage many hardware interface modules as long as it exists on Wi-Fi network coverage. It supports a wide range of home automation devices like power management components, and security components. We want to make this automation system centralized and artificially intelligent. Further we will connect the specific home to our database and it can be accessed from anywhere through a specific IP address or website. Also, an application ( app ) would be developed which will allow the user to control their devices using the Google Assistant.

CONTENTS

Contents Page No

ACKNOWLEDGEMNENTS i

ABSTRACT ii

LIST OF TABLES iv

LIST OF FIGURES v

Chapter 1. INTRODUCTION

1.1 Literature survey 2

1.2 Block diagram 3

1.3 Block diagram description 3

Chapter 2. HARDWARE SPECIFICATIONS

2.1 NodeMCU – ESP8266 4

2.1.1 Digital I/O pins

2.1.2 GPIO Overview

2.1.3 The ESP8266 as a microcontroller – Software

2.1.4 NodeMCU GPIO for Lua

2.1.5 Blink test using NodeMCU

2.2 Relay board 10

2.2.1 Four channel Relay Board

2.3 Servo Sg90 12

2.4 DHT 11 12

Chapter 3. SOFTWARE SPECIFICATIONS

3.1 Blynk Application 14

3.2 Google Assistant 15

3.3 IFTTT Application 17

3.4 Arduino IDE 18

Chapter 4. CIRCUIT SETUP

4.1 Configuring Blynk Application 20

4.2 Uploading the Code to NodeMCU 21

4.3 Hardware Assembly 23

4.4 Connecting Google Assistant (using IFTTT) 24

Chapter 5. CIRCUIT DIAGRAM & WORKING

Chapter 6. EXPERIMENTAL SETUP WITH RESULTS

Chapter 7. COST AND EXPENDITURE

Chapter 8. CONCLUSION

Chapter 9. FUTURE SCOPE

REFERENCE

**LIST OF TABLES**

No Title Page No.

2.1 Bootable modes 6

2.2 GPIO Overview 7

2.3 IO index for internal GPIO 8

7.1 Cost and expenditure 31

**LIST OF FIGURES**

No Title Page No.

1.1 Home automation 1

1.2 Block diagram of home automation system 3

2.1 NodeMCU 5

2.2 Pin diagram explanation of NodeMCU 5

2.3 Program for Blynk test 9

2.4 Putting board into boot load mode 9

2.5 Connecting LED to the board 10

2.6 Indication of LED blinking 10

2.7 Relay board 11

2.8 Servo Sg90 12

2.9 DHT 11 13

3.1 Functioning of the Blynk Application 15

3.2 Google Assistant in smartphone 17

3.3 Programming in Arduino IDE 18

4.1 Creating new project 20

4.2 Creating buttons 21

4.3 Programme for connecting Blynk and NodeMCU 23

4.4 Hardware assembly 24

4.5 Creating trigger in IFTTT for Google assistant 26

4.6 Setting in IFTTT 27

5.1 Circuit diagram of home automation system 28

6.1 Green bulb gets turned ON by saying command 29

6.2 Green and white bulbs gets turned ON by saying command 29

6.3 Green and white bulbs gets turned OFF by saying command 29

6.4 Controlling servo motor to control regulator 29

**CHAPTER-1**

**INTRODUCTION**

Home automation involves the control and automation of lighting, heating, ventilation, air conditioning and security, as well as home appliances. Wi-Fi is often used for remote monitoring and control. Home devices, when remotely monitored and controlled via Internet is a part of Internet of things. Modern systems generally consist of switches and sensors connected to a central hub called a gateway from which the system is controlled with a user interface that is interacted either with a mobile phone software, tablet, computers or a web interface, often but not always via internet cloud services.



Figure 1.1 Home automation

We know world’s demand for electricity is growing day by day. On the other side electricity is being waste because we forget to turn off our home appliances not in use. This project describes about controlling home appliances using Google Assistant and thereby reducing the amount of electricity being wasted. Today all people have smart phone and internet facility. So by using this we can control our appliances easily. Mainly based on Google Assistant and NodeMCU. Google Assistant is used for saying commands so we can control devices using speech. NodeMCU is used as a network provider. It is also highly reliable and efficient for the aged people and differently abled person who cannot reach the switch and adding convenience to our daily life.

* 1. **LITERATURE SURVEY**

1. **Review of Related Literature**

When people think about home automation, most of them may imagine living in a smart home: One remote controller for every household appliance, cooking the rice automatically, starting air conditioner automatically, heating water for bath automatically and shading the window automatically during night. To some extent home automation equals to smart home. They both bring out smart living condition and make our life more convenient and fast. Early home automation began with labor-saving machines. Self-contained electric or gas powered home appliance became viable in the 1900s with the introduction of electric power distribution led to the introduction of washing machine (1904), water heater (1889), refrigerator, sewing machines, dishwashers and clothes dryers. As per our survey currently there exists system neither at cheaper rates nor easy to handle. Various systems are hard to install, difficult to use and maintain. Current systems are generally proprietary, closed and not very user friendly Based on Arduino or GSM or low cost home security system and home automation system. Home automation using NodeMCU is already exist and no much user friendly.

1. **Review of Foreign Study**

Tan, Lee and Soh (2002) proposed the development of an Internet-based system to allow monitoring of important process variables from a distributed control system (DCS). That paper proposed hardware and software design considerations which enable the user to access the process variables on the DCS, remotely and effectively rent designations. Potamitis, Georgila, Fakotakis, and Kokkinoss, G. (2003) suggested the use of speech to interact remotely with the home appliances to perform a particular action on behalf of the user.

The approach is inclined for people with disability to perform real-life operations at home by directing appliances through speech. Voice separation strategy is selected to

take appropriate decision by speech recognition.

In the year 2006, S. M. AnamulHaque S, S. M. Kamruzzaman and Md. Ashraful Islam proposed a system entitled “A System for Smart-Home Control of Appliances Based on Time and Speech Interaction” that controls the home appliances using the personal computer. This system is developed by using the Visual Basic 6.0 as programming language and Microsoft voice engine tools for speech recognition purpose. Appliances can be either controlled by timer or by voice command. Jawarkar, Ahmed, Ladhake, and Thakare (2008) propose remote monitoring through mobile phone involving the use of spoken commands. The spoken commands are generated and sent in the form of text SMS to the control system and then the microcontroller on the basis of SMS takes a decision of a particular task. Prof. Era Johri Dept. Of Information and Technology K.J. Somaiya College of Engineering VIDYAVIHAR, MUMBAI in (2001) have successfully completed the project on “Remote Controlled Home Automation”.

* 1. **BLOCK DIAGRAM**

**SERVO MOTOR**

**BLYNK**

**APPLICATION**

**IFTTT**

**GOOGLE ASSISTANT**

**RELAY**

**MODULE**

**HOME**

**APPLIANCES**

**DHT 11**

**NODEMCU**

**REGULATOR**

Figure.1.2 Block diagram of home automation system

* 1. **Block diagram description**

Here the Blynk application with in the smart phone is the controlling device. NodeMCU act as a network server. We can’t connect the Google assistant to NodeMCU directly so we are using Blynk application. Google assistant cannot directly understand foreign command, so to solve this we use another intermediate application /website called IFTTT. Here we are using 4 channel relay module to connect NodeMCU to electrical appliances. We are also used a servo motor to control the regulator thereby controlling the speed of fan. DHT 11 is used to determine the temperature and humidity.

**CHAPTER-2**

**HARDWARE SPECIFICATIONS**

**2.1 NODEMCU -ESP8266**

NodeMCU is an open source IoT platform. It is a SOC which consists a microcontroller and a Wi-Fi module. The term "NodeMCU" by default refers to the firmware. The firmware uses the Lua scripting language. Hardware is based on the ESP-12 module.

The ESP8266 is the name of a micro controller designed by Espressif Systems. The ESP8266 itself is a self-contained Wi-Fi networking solution offering as a bridge from existing micro controller to Wi-Fi and is also capable of running self-contained applications. This module comes with a built in USB connector and a rich assortment of pin-outs. With a micro USB cable, you can connect NodeMCU devkit to your laptop and flash it without any trouble, just like Arduino. It is also immediately breadboard friendly.

**Specifications**

* Single board microcontroller.
* XTOS operating system.
* 128kbytes memory.
* 4MBytes storage.
* Voltage:3.3V.
* Integrated TCP/IP protocol stack.
* Flash memory attachable: 16MB max (512K normal).
* Processor: Tensilica L106 32-bit.
* Processor speed: 80~160MHz.
* 16 General purpose input/output pins.
* Serial Peripheral Interface Bus.
* GPIOs: 17 (multiplexed with other functions).
* Analog to Digital: 1 input with 1024 step resolution.
* +19.5dBm output power in 802.11b mode
* 802.11 support: b/g/n.
* Maximum concurrent TCP connections: 5.

  
Figure 2.1 NodeMCU

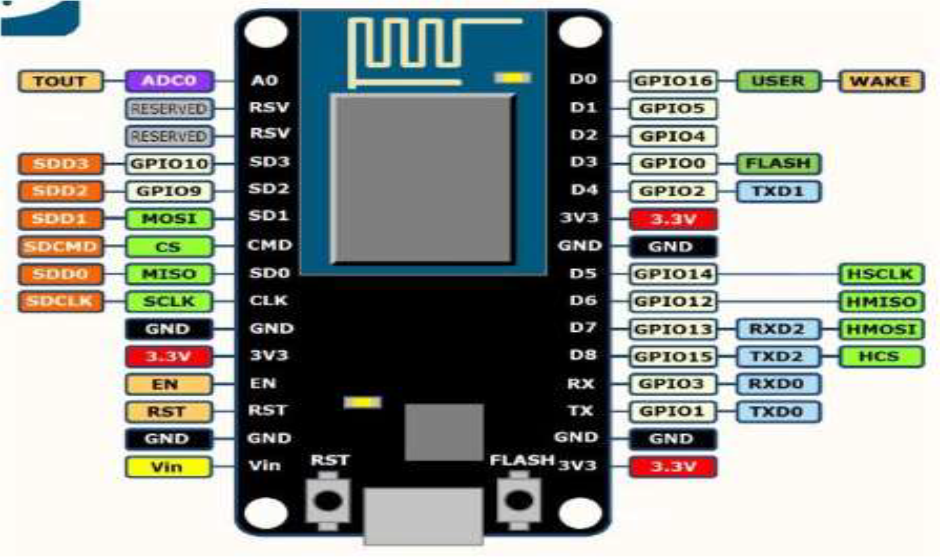


Figure 2.2 Pin diagram of NodeMCU

**2.1.1 Digital I/O pins**

Just like a normal Arduino, the ESP8266 has digital input/output pins (I/O or GPIO, General Purpose Input/output pins). They can be used as digital inputs to read a digital voltage, or as digital outputs to output either 0V (sink current) or 3.3V (source current).

* Voltage and current restrictions

The ESP8266 is a 3.3V microcontroller, so its I/O operates at 3.3V. The pins are not 5V tolerant, applying more than 3.6V on any pin will kill the chip. The maximum current that can be drawn from a single GPIO pin is 12mA.

* Usable pins

The ESP8266 has 17 GPIO pins (0-16) only 11 of them can be used. Because 6 pins (GPIO 6 - 11) are used to connect the flash memory chip. This is the small 8-legged chip right next to the ESP8266. If we try to use one of these pins, you might crash your program. GPIO 1 and 3 are used as TX and RX of the hardware Serial port (UART). So in most cases, we can’t use them as normal I/O while sending/receiving serial data.

* BOOT modes

Some I/O pins have a special function during boot.

Table 2.1 Bootable modes

|  |  |  |  |
| --- | --- | --- | --- |
| **GPIO 15** | **GPIO0** | **GPIO2** | **MODE** |
| 0V | 0V | 3.3V | Uart bootloader |
| 0v | 3.3v | 3.3v | Boot sketch (SPI Flash) |
| 3.3V | X | X | SDIO mode (not used for Arduino) |

GPIO15 is always pulled low, so you can’t use the internal pull-up resistor. GPIO0 is pulled high during normal operation. GPIO2 can’t be low at boot, so you can’t connect a switch to it.

* Internal pull-up/-down resistors

GPIO 0-15 all have a built-in pull-up resistor, just like in an Arduino. GPIO16 has a built-in pull-down resistor.

* PWM

ESP8266 doesn’t support hardware PWM. However, software PWM is supported on all digital pins. The default PWM range is 10-bits @ 1kHz, but this can be changed (up to >14-bit@1kHz).

* Serial Communication

The ESP8266 has two hardware UARTS (Serial ports). UART0 on pins 1 and 3 (TX0 and RX0 resp.), and UART1 on pins 2 and 8 (TX1 and RX1 resp.). However, GPIO8 is used to connect the flash chip. This means that UART1 can only transmit data. UART0 also has hardware flow control on pins 15 and 13 (RTS0 and CTS0 resp.). These two pins can also be used as alternative TX0 and RX0 pin. By default, the I²C library uses pin 4 as SDA and pin 5 as SCL. (The data sheet specifies GPIO2 as SDA and GPIO14 as SCL.) The maximum speed is approximately 450kHz. The ESP8266 has one SPI connection available to the user, referred to as HSPI. It uses GPIO14 as CLK, 12 as MISO, 13 as MOSI and 15 as Slave Select (SS).

**2.1.2 GPIO Overview**

Table 2.2 Overview of GPIO pins

|  |  |  |  |
| --- | --- | --- | --- |
| **GPIO** | **FUNCTION** | **STATE** | **RESTRICTIONS** |
| 0 | Boot mode select | 3.3V | No Hi-Z |
| 1 | TX0 | - | Not usable during Serial transmission |
| 2 | Boot mode select TX1 | 3.3V (boot only) | Don’t connect to ground at boot time Sends debug data at boot time |
| 3 | RX0 | - | Not usable during Serial transmission |
| 4 | SDA (I²C) | - | - |
| 5 | SCL ( I²C) | - | - |
| 6-11 | Flash connection | X | Not usable, and not broken out |
| 12 | MISO (SPI) | - | - |
| 13 | MOSI (SPI) | - | - |
| 14 | SCK (SPI) | - | - |
| 15 | SS (SPI) | 0V | Pull-up resistor not usable |
| 16 | Wake up from sleep | - | No pull-up resistor, but pull-down instead Should be connected to RST to wake up |

**2.1.3 The ESP8266 as a microcontroller – Software**

Most of the microcontroller functionality of the ESP uses exactly the same syntax as a normal Arduino.

* Digital I/O

We can set the function of a pin using pinMode(pin, mode). where pin is the GPIO number\*. And mode can be either INPUT,OUTPUT , or INPUT\_PULLUP to enable the built-in pull-up resistors for GPIO 0-15. To enable the pull-down resistor for GPIO16, you have to use INPUT\_PULLDOWN\_16. To address a NodeMCU pin, e.g. pin 5, use D5: for instance: pinMode(D5, OUTPUT). To set an output pin high (3.3V) or low (0V), use digitalWrite(pin, value); where pin is the digital pin, and value either 1 or 0 (or HIGH and LOW). To read an input, use digitalRead(pin).To enable PWM on a certain pin, use

analogWrite(pin, value); where pin is the digital pin, and value a number between 0 and 1023.

The frequency can be changed by using analogWriteFreq(new frequency);. new frequency should be between 100 and 1000Hz.

* Analog input

Just like on an Arduino, you can use analogRead(A0) to get the analog voltage on the analog input. (0 = 0V, 1023 = 1.0V). The ESP can also use the ADC to measure the supply voltage (VCC). To do this, include ADC\_MODE(ADC\_VCC); at the top of your sketch, and use ESP.getVcc(); to actually get the voltage.

**2.1.4 NodeMCU GPIO for Lua**

All the pins of ESP8266 accessed using the command GPIO. All the access is based on the I/O index number on the NodeMCU dev kits, not the internal GPIO pin. For example, the pin ‘D7’ on the NodeMCU dev kit is mapped to the internal GPIO pin 13. If you want to turn ‘High’ or ‘Low’ that particular pin you need to called the pin number ‘7’, not the internal GPIO of the pin. If you are using NodeMCU devkit, it has come prepared for working with Lua interpreter which can easily program by looking the pin names associated on the Lua board.

Table below can be used to know which IO index is associated to the internal GPIO of ESP8266.

Table 2.3 IO index for internal GPIO

|  |  |  |  |
| --- | --- | --- | --- |
| **NodeMCU dev kit** | **ESP8266 Pin** | **NodeMCU dev kit** | **ESP826 Pin** |
| D0 | GPIO16 | D7 | GPIO13 |
| D1 | GPIO5 | D8 | GPIO15 |
| D2 | GPIO4 | D9 | GPIO3 |
| D3 | GPIO0 | D10 | GPIO1 |
| D4 | GPIO2 | D11 | GPIO9 |
| D5 | GPIO14 | D12 | GPIO10 |
| D6 | GPIO12 |  |  |

**2.1.5 Blink test using NodeMCU**

Most of the programs in NodeMCU is done by Arduino IDE Tool. Here is an example, simple blink test using NodeMCU. First download the Arduino IDE tool from official website. Connect the LED to the board. Enter this into the sketch window.

1. void setup () {
2. pinMode(5, OUTPUT); // GPIO05, Digital Pin D1
3. }
4. void loop() {
5. digitalWire(5, HIGH);
6. delay(900);
7. digitalWrite(5, LOW);
8. delay(500);
9. }

Figure 2.3 Program for Blynk test

Now we need to put the board into boot load mode. We have to do this before each upload.

• Hold down the ‘Flash’ button.

• While holding down ‘Flash’, press the ‘RST’ button.

• Release ‘RST’, then release ‘Flash’.

When we release the ‘RST’ button, the blue indication will blink once, this means its ready to boot load.

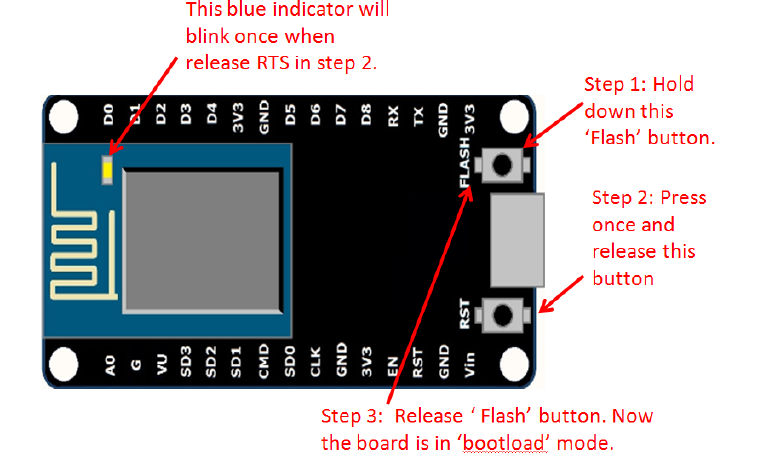


Figure 2.4 Putting board into boot load mode

Once the ESP board is in boot load mode, upload the sketch via the IDE.

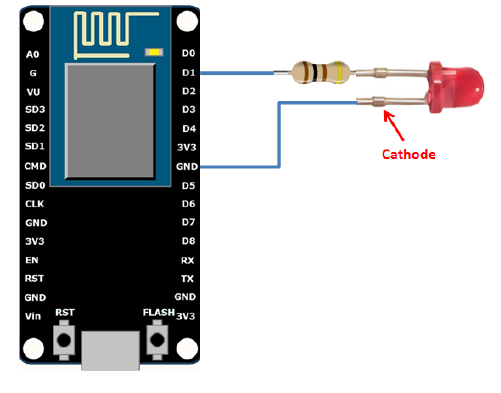


Figure 2.5 Connecting LED to the board

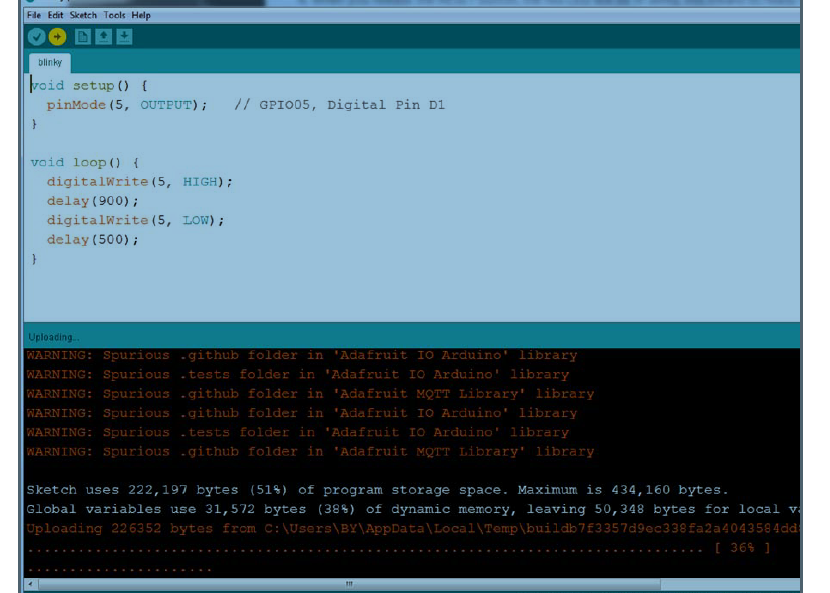


Figure 2.6 Indication of LED blinking

The sketch will start immediately and we can see the LED blinking.

**2.2 RELAY BOARD**

A relay is an electrically operated switch. Many relays use an electromagnet to, mechanically operates a switch, but other operating principles are also used, such as solid state relay. Relay are used where it is necessary to control a circuit by a separate low power signal, or where several circuits must be controlled by one signal. Relay were used extensively in telephone exchange and early computers to perform logical operation.

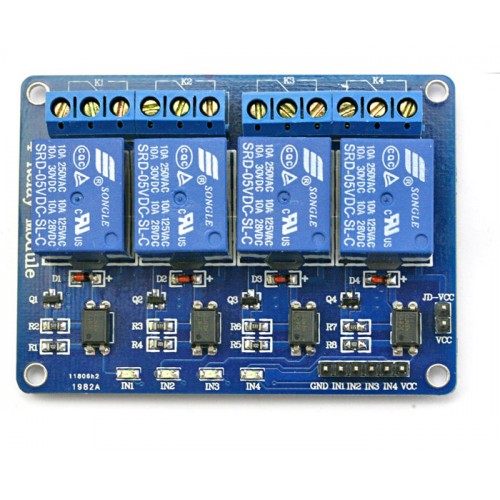


Figure 2.7 Relay board

A type of relay that can handle the high power required to directly control an electronic motor or other loads is called a contactor. Solid state relay controls power circuit with no moving parts, instead using a semiconductor device to perform switching. Relay calibrated operating characteristics and sometimes multiple operating coil are used to protect electrical circuits from overload or faults; in modern electric power system these functions are performed by digital instruments still called “protective relay”.

**2.2.1 Four channel Relay Board**

Four Channel Relay Board is a simple and convenient way to interface Four relays for switching application in your project. Input voltage level support TTL as well as CMOS. Easy interface with Microcontrollers based projects and analog circuits.

**Specifications**

* Input supply 12 VDC @ 336 Ma
* Output eight SPDT relay
* Relay specification 5 A @ 230 VAC
* Trigger level 2 ~ 15 VDC
* Header connector for connecting power and trigger voltage
* LED on each channel indicates relay status
* Screw terminal connector for easy relay output and aux power connection

**2.3 SERVO SG90**

It is tiny and lightweight with high output power. This servo can rotate approximately 180 degrees (90 in each direction), and works just like the standard kinds but smaller. You can use any servo code, hardware or library to control these servos.It comes with a 3 horns (arms) and hardware.

**Specifications**

* Operating voltage: 4.8 V (~5V)
* Operating speed: 0.1 s/60 degree
* Stall torque: 1.8 kgf•cm
* Dead band width: 10 µs
* Temperature range: 0 ºC – 55 ºC

****

Figure 2.8 Servo Sg90

**2.3 DHT 11**

This DHT11 Temperature and Humidity Sensor features a calibrated digital signal output with the temperature and humidity sensor capability. It is integrated with a high-performance 8-bit microcontroller. Its technology ensures the high reliability and excellent long-term stability. This sensor includes a resistive element and a sensor for wet NTC temperature measuring devices. It has excellent quality, fast response, anti-interference ability and high performance.

**Specifications**

* Supply Voltage: +5 V
* Temperature range :0-50 °C error of ± 2 °C
* Humidity :20-90% RH ± 5% RH error
* Interface: Digital

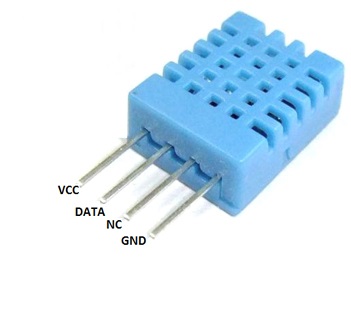


Figure 2.9 DHT 11

**CHAPTER-3**

**SOFTWARE SPECIFICATIONS**

The software of the system proposed consists of mainly the Blynk Application, Google Assistant, IFTTT application and the Arduino IDE.

**3.1 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.

It's a digital dashboard where you can build a graphic interface for your project by simply dragging and dropping widgets. It's really simple to set everything up and you'll start tinkering in less than 5 minutes. Blynk is not tied to some specific board or shield. Instead, it's supporting hardware of your choice. Whether your Arduino or Raspberry Pi is linked to the Internet over Wi-Fi, Ethernet or this new ESP8266 chip, Blynk will get you online and ready for the **Internet of Your Things**.

There are three major components in the platform: Blynk App, Blynk server & Blynk Library.

* Blynk App - allows to you create amazing interfaces for your projects using various widgets we provide.
* Blynk Server - responsible for all the communications between the smartphone and hardware. You can use our Blynk Cloud or run your private Blynk server locally. It’s open-source, could easily handle thousands of devices and can even be launched on a Raspberry Pi.
* Blynk Libraries - for all the popular hardware platforms - enable communication with the server and process all the incoming and out coming commands.

Now imagine: every time you press a Button in the Blynk app, the message travels to the Blynk Cloud, where it magically finds its way to your hardware. It works the same in the opposite direction and everything happens in a Blynk of an eye.

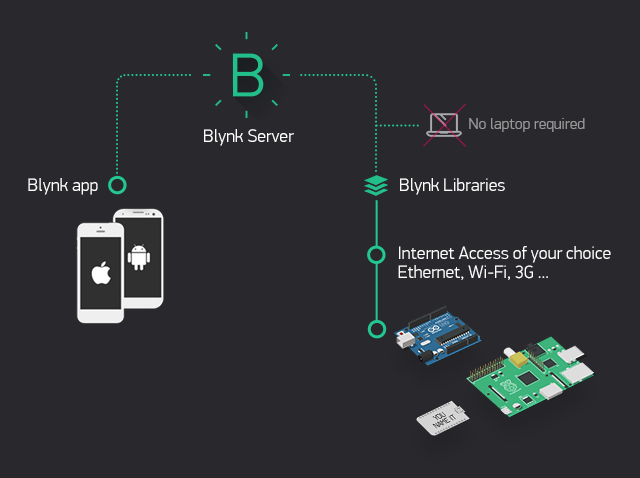


Figure 3.1 Functioning of the Blynk Application.

**Blynk works over the Internet.** This means that the hardware you choose should be able to connect to the internet. Some of the boards, like Arduino Uno will need an Ethernet or Wi-Fi Shield to communicate, others are already Internet-enabled: like the ESP8266, Raspberry Pi with Wi-Fi dongle, Particle Photon or SparkFun Blynk Board. But even if you don’t have a shield, you can connect it over USB to your laptop or desktop (it’s a bit more complicated for newbies, but we got you covered).

**3.2 GOOGLE ASSISTANT**

The Google Assistant is a software which allows its users to control all the apps in their device to be controlled directly through it. It allows the users to control and command most of the apps in their devices using voice commands. This provides more convenience to the people as they only have to command the google assistant thorough voice command.

The Google Assistant is an artificial intelligence-powered virtual assistant developed by Google that is primarily available on mobile and smart home devices. Unlike the company's previous virtual assistant, Google Now, the Google Assistant can engage in two-way conversations.

Assistant initially debuted in May 2016 as part of Google's messaging app Allo, and its voice-activated speaker Google Home. After a period of exclusivity on the Pixel and Pixel XL smartphones, it began to be deployed on other Android devices in February 2017, including third-party smartphones and Android Wear (now Wear OS), and was released as a standalone app on the iOS operating system in May. Alongside the announcement of a software development kit in April 2017, the Assistant has been, and is being, further extended to support a large variety of devices, including cars and smart home appliances. The functionality of the Assistant can also be enhanced by third-party developers.

Users primarily interact with the Google Assistant through natural voice, though keyboard input is also supported. In the same nature and manner as Google Now, the Assistant is able to search the Internet, schedule events and alarms, adjust hardware settings on the user's device, and show information from the user's Google account. Google has also announced that the Assistant will be able to identify objects and gather visual information through the device's camera, and support purchasing products and sending money, as well as identifying songs. The Google Assistant, in the nature and manner of Google Now, can search the Internet, schedule events and alarms, adjust hardware settings on the user's device, and show information from the user's Google account. Unlike Google Now, however, the Assistant can engage in a two-way conversation, using Google's natural language processing algorithm. Search results are presented in a card format that users can tap to open the page.

In May 2017, Google announced that the Assistant would support a keyboard for typed input and visual responses, support identifying objects and gather visual information through the device's camera and support purchasing products and sending money. Through the use of the keyboard, users can see a history of queries made to the Google Assistant, and edit or delete previous inputs. The Assistant warns against deleting, however, due to its use of previous inputs to generate better answers in the future. In November 2017, it became possible to identify songs currently playing by asking the Assistant.

Google Assistant allows users to activate and modify vocal shortcut commands in order to perform actions on their device -both Android and iPad/iPhone- or configuring it as a hub for the home automation. This feature of the speech recognition is available in English, among other languages. In July 2018, the Google Home version of Assistant gained support for multiple actions triggered by a single vocal shortcut command.

At the annual I/O developers conference on May 8, 2018, Google's SEO announced the addition of six new voice options for Google Assistant, one of which being John Legend's. This was made possible by WaveNet, a voice synthesizer developed by DeepMind, which significantly reduced the amount of audio samples that a voice actor was required to produce for creating a voice model.

In August 2018, Google added bilingual capabilities to Google Assistant for existing supported languages on devices. Recent reports say that it may support multilingual support by setting a third default language on Android Phone. As a default option, Google Assistant doesn't support two common features of the speech recognition on the transcripted texts, like punctuation and spelling. However, a Beta feature of Speech-to-text enables only en-Us language users to ask "to detect and insert punctuation in transcription results. Speech-to-Text can recognize commas, question marks, and periods in transcription requests."

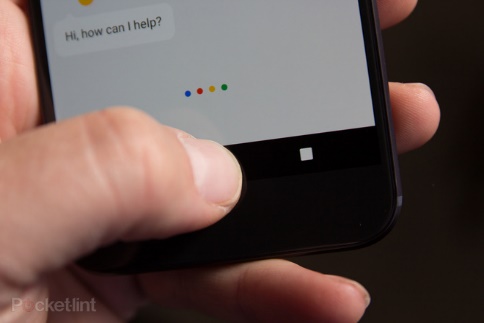


Figure 3.2 Google Assistant in smartphone

**3.3 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 favorite apps and websites to app-enabled accessories and smart devices. What the company provides is a software platform that connects apps, devices and services from different developers in order to trigger one or more automations involving those apps, devices and services. IFTTT application is used to bridge the gap between the Google Assistant commands and the Blynk app.

IFTTT can connect all your “services” together so that tasks are automatically completed. There are numerous ways you can connect all your services-and the resulting combination are called “Applets”. Applets essentially automate your daily workflow, whether it’s managing smart home devices or app and website. Once added or created, applet creates a chain reaction between at least two of your apps.

**3.4 ARDUINO IDE**

Arduino Software (**IDE**) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them.

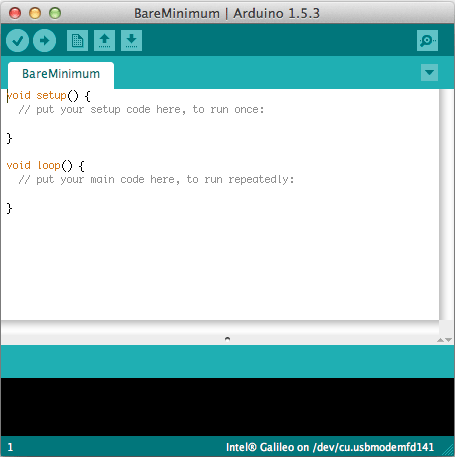


Figure 3.3 Programming in Arduino IDE

Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them.

Programs written using Arduino Software (IDE) are called **sketches**. These sketches are written in the text editor and are saved with the file extension ino. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom right hand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

**CHAPTER-4**

**CIRCUIT SETUP**

**4.1 CONFIGURING BLYNK APPLICATION**

We cannot connect the Google Assistant to the NodeMCU directly, so we are using the Blynk application (app). Blynk app can directly connect to the NodeMCU and send data to it. We can download Blynk app from playstore. Open Play store and download and install the Blynk App. We Login to it using a dedicated Email which is created only for this purpose. After logging in, We created a new project by clicking **‘New Project’**. And named the project as Home Automation. Selected the hardware device as NodeMCU, selected the connection type as WIFI since we are using hotspot in this system. Finally created the project. At this point Blynk will send an ***Auth token*** to our email id. We will use this ‘***Auth token***’ later to link our app with the NodeMCU.



Figure 4.1 Creating new project

We used four channel relay so we can create four buttons in Blynk to which four same or different appliances can be connected. We also connected DTH 11 sensor to measure humidity and temperature. And by using a slider button we can control our fan regulator through a Servo motor. Another point is that for connecting DHT-11 and Servo motor we used virtual pins.

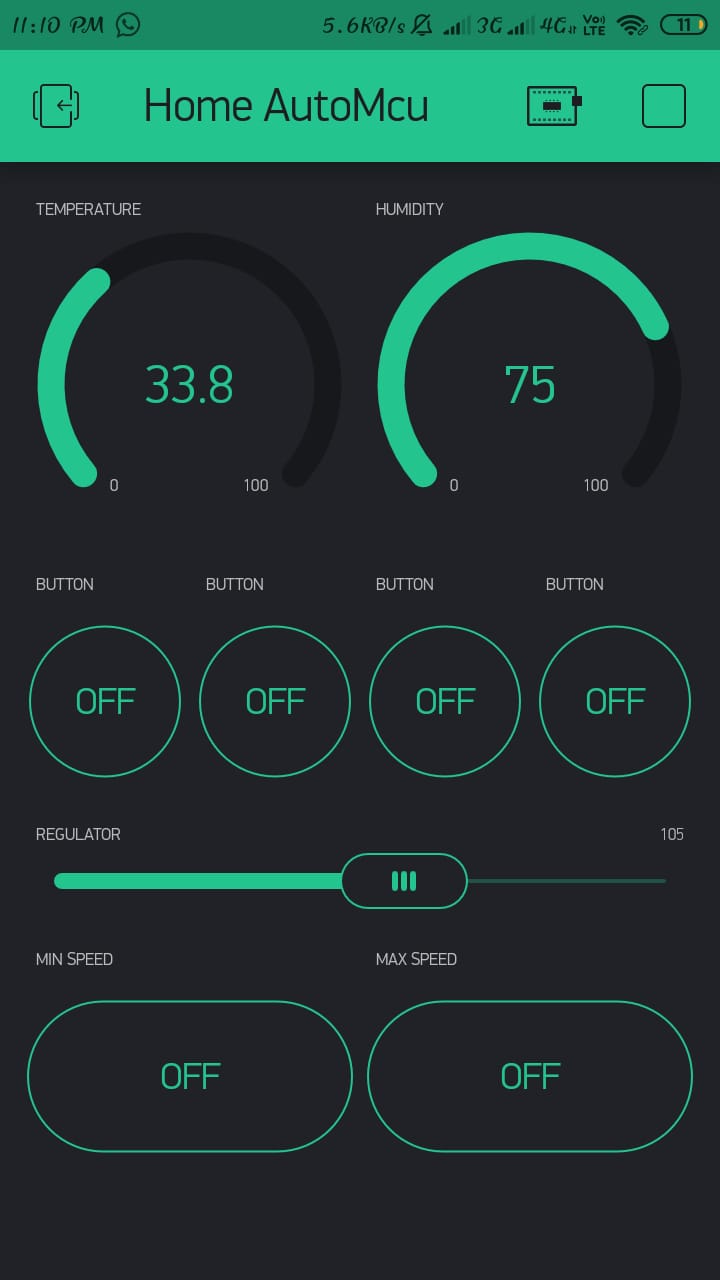


Figure 4.2 Creating buttons

**4.2 UPLOADING THE CODE TO NODEMCU**

We are using Arduino IDE software for programming. It cannot have Blynk libraries in it. So we download the Blynk libraries from its official site and copied to the Arduino IDE’s directory.

Then we connected the NodeMCU to your PC using a USB cable. After that open up the Arduino IDE. Go to *Tools* > *Port* and make sure an appropriate port is selected (eg. COM X), this is the USB port in which the NodeMCU is connected. Then selected ‘*NodeMCU 1.0 (ESP-12E Module)*’ as the board. After that we have to write the code.

1. #define BLYNK\_PRINT Serial
2. #include <ESP8266WiFi.h>
3. #include <BlynkSimpleEsp8266.h>
4. #include <DHT.h>
5. #include <Servo.h>
6. Servo servo;
8. // You should get Auth Token in the Blynk App.
9. // Go to the Project Settings (nut icon).
10. char auth[] = "929a265008254f2bbaaf5dc7bf18b453";
12. // Your WiFi credentials.
13. // Set password to "" for open networks.
14. char ssid[] = "homeautosensor";
15. char pass[] = "homeautosensor";
16. #define DHTPIN 0 // D3
17. // Uncomment whatever type you're using!
18. #define DHTTYPE DHT11 // DHT 11
19. //#define DHTTYPE DHT22 // DHT 22, AM2302, AM2321
20. //#define DHTTYPE DHT21 // DHT 21, AM2301
21. DHT dht(DHTPIN, DHTTYPE);
22. BlynkTimer timer;
23. // This function sends Arduino's up time every second to Virtual Pin (5).
24. // In the app, Widget's reading frequency should be set to PUSH. This means
25. // that you define how often to send data to Blynk App.
26. void sendSensor()
27. {
28. float h = dht.readHumidity();
29. float t = dht.readTemperature(); // or dht.readTemperature(true) for Fahrenheit
30. if (isnan(h) || isnan(t)) {
31. Serial.println("Failed to read from DHT sensor!");
32. return;
33. }
34. // You can send any value at any time.
35. // Please don't send more that 10 values per second.
36. Blynk.virtualWrite(V5, t);
37. Blynk.virtualWrite(V6, h);
38. }
39. void setup()
40. {
41. // Debug console
42. Serial.begin(9600);
43. Blynk.begin(auth, ssid, pass);
44. // You can also specify server:
45. //Blynk.begin(auth, ssid, pass, "blynk-cloud.com", 8442);
46. //Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,100), 8442);
47. dht.begin();
48. // Setup a function to be called every second
49. timer.setInterval(1000L, sendSensor);
50. servo.attach(15);
51. }
52. void loop()
53. {
54. Blynk.run();
55. timer.run();
56. }
57. BLYNK\_WRITE(V1)
58. {
59. servo.write(param.asInt());
60. }
61. BLYNK\_WRITE(V2)
62. {
63. servo.write(90);
64. }
65. BLYNK\_WRITE(V3)
66. {
67. servo.write(180);
68. }
69. BLYNK\_WRITE(V4)
70. {
71. servo.write(0);
72. }

Figure 4.3 Programme for connecting Blynk and NodeMCU

Now the code is ready to upload to the NodeMCU. So directly hit upload button. The code will be uploaded to the NodeMCU and the next time we power it on, it will automatically connect to the specified WIFI network.

**4.3 HARDWARE ASSEMBLY**

We’ll have to connect the NodeMCU with the Relay board, we can choose to do it with a bread board or without. Here we used a breadboard. Then we Connected the D3 pin of NodeMCU with Pin 1 of Relay. Similarly connect D4 pin of NodeMCU with Relay pin 2, D5 with Relay 3 and D6 with Relay 4 and connected Ground Pin of Relay with Ground Pin of NodeMCU. Here we connected DHT-11 to D2 pin. And servo motor is connected to pin D8 of NodeMCU.

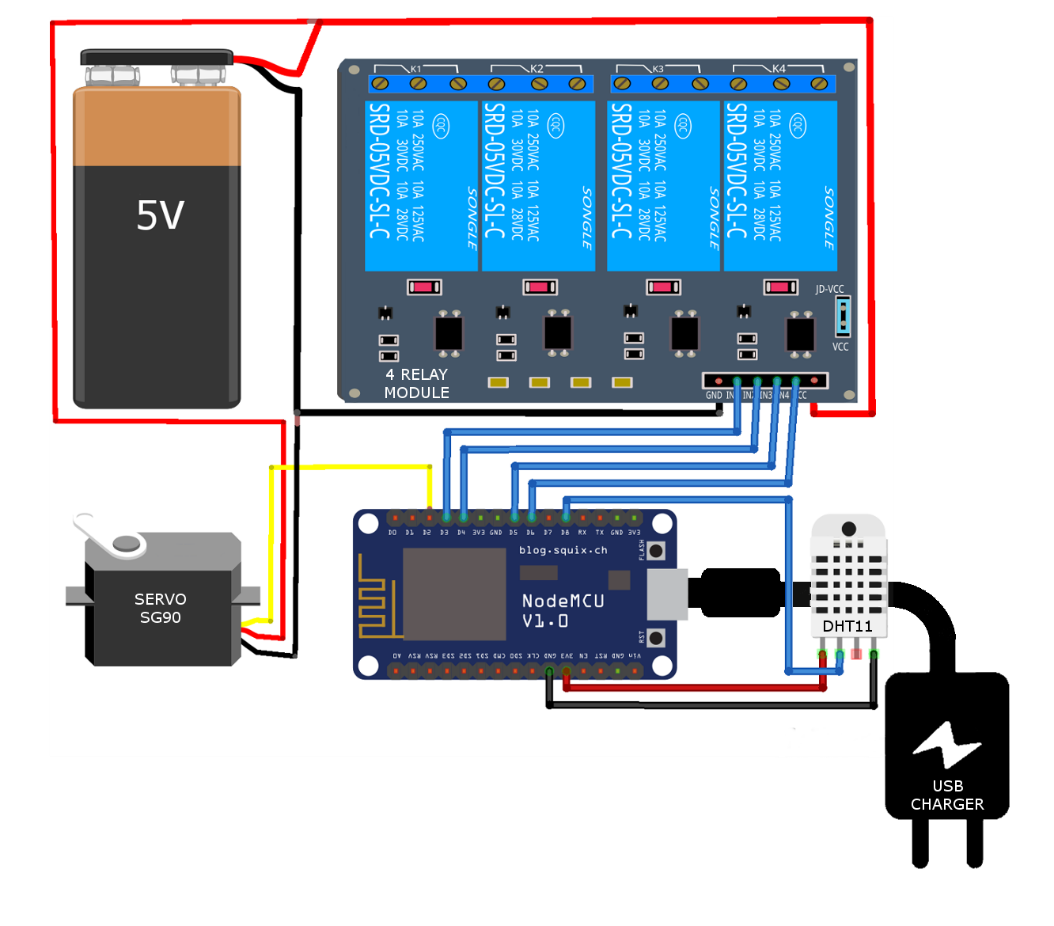


Figure 4.4 Hardware assembly

To power up the NodeMCU we can use a normal phone charger, just make sure its voltage is not too high. And to power up the Relay board, we can use a battery or a separate [breadboard power supplier](https://amzn.to/2BVKa4m). As we are using a four-channel relay you can connect at most 4 electronic appliances to the Relay and control them over the internet. At this point, we have a fully functional connection between the NodeMCU, Blynk app and our electrical appliances. So, we can directly run your Blynk project from our phone and turn the electrical appliances on or off using the buttons that we created in the app.

**4.4 CONNECTING GOOGLE ASSISTANT (USING IFTTT)**

 We cannot connect the Google Assistant to the NodeMCU directly, and that is the only reason we are using the Blynk app. Blynk app can directly connect to the NodeMCU and send data to it. So, we can send the voice commands interpreted by google assistant directly to the Blynk app, the Blynk app can then forward those commands to the NodeMCU. But the problem is Google Assistant cannot directly understand foreign commands like “turn on the fan” or “turn on light” etc. on its own. When we say a command like “turn on fan” it just search on the topic and open a page which give information. So, to solve this we use another intermediate app/website called ‘[*IFTTT*](https://ifttt.com/)’.

Simply, to control our home appliances over the internet we are using NodeMCU and to connect NodeMCU with the home appliances we use a relay board. Now to send on or off signals to the NodeMCU we use our smartphone, and we do this using the Blynk app. But we want to send the on or off signals using voice commands.  To do this we use google assistant in our smartphone and an app called IFTTT.

So, in the end what will happen is, when we say a voice command like “ok google turn on the light” to the Google Assistant, Google Assistant sends this foreign command to IFTTT. IFTTT interprets this command and sends an On or Off signal to the Blynk app via the Blynk Server. Blynk will then send this signal to the NodeMCU and then to our electrical appliances.

So we need to open IFTTT’s website and sign up to it using our Email which is also used to login Blynk app. After Signing in click on *My Applets* from the header and select *New Applet*. Click on ‘*this*’ and *Google Assistant*. The we select the card that says “Say a simple phrase”. Next, for the first textbox type the phrase that we want to say to Google Assistant. It can be anything such as “Turn on the T.V”, “Turn on the fan” or anything we like.

For the next two text boxes, we write some other ways to say the first command. For example, if in the first textbox we wrote “Turn on the T.V”, then in the second and third textboxes we can write something like “Turn the T.V On” or “Please Turn on the T.V” or “Turn the Idiot Box On”. In the fourth textbox type the reply that Google Assistant should respond with. For example, “Okay, Turning on the T.V”. Finally create trigger.

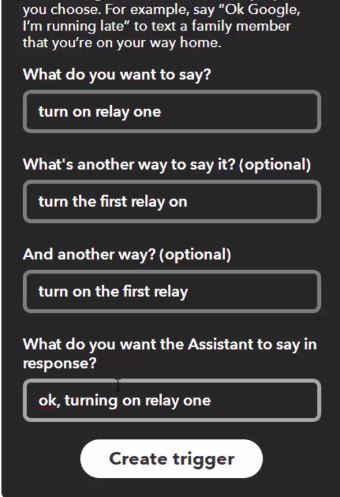


Figure 4.5 Creating trigger in IFTTT for Google assistant

Now, click on that and select webhooks. Webhooks will allow us to send commands to the Blynk Server. Now, in the URL field type this URL:

*http://188.166.206.43/ AuthTokenHere / update / DigitalPinToBeUpdateHere*

This is the URL of Blynk Server of India, but it should work for other places as well. Replace the “*AuthToken*” part with our Blynk ***Auth token*** that we received in the mail. And “DigitalPinToBeUpdateHere” part with the Digital pin of NodeMCU that is to be updated. Suppose we assigned the Digital Pin D3 of NodeMCU to relay one we must write D3 in place of “DigitalPinToBeUpdateHere”. But we cannot write D3 there, because when Blynk Server receives this command from IFTTT it assumes as if the command it received was to be sent to an ‘Arduino Uno’ board, but in our case, we are sending it to NodeMCU. To solve this, we must type the Digital pin of Arduino which corresponds with the NodeMCU. We can find the mapping in the table 2.3.

As we can see in the table 2.3, Digital Pin D3 of NodeMCU corresponds to Pin D0 of Arduino. So instead of *D3,* we write D0 as the pin.

In the end the URL should look something like this, *http://188.166.206.43/ d5846384ef7140cfg54699bbc97f2ad8 / update / D0*. Then select the ‘Method’ field as PUT. Select ‘Content type’ as *Application/JSON.*

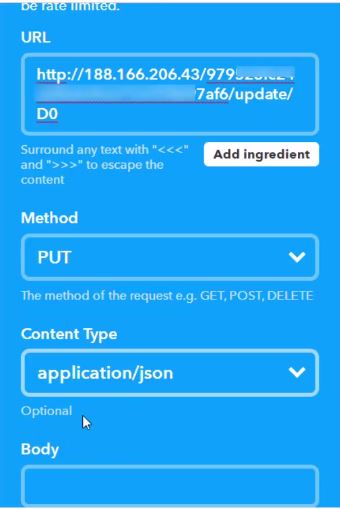


Figure 4.6 Setting in IFTTT

For the ‘Body’ type this [“0”]. Here ‘0’ means to turn on, so we are basically saying Blynk to turn on relay that is connected to pin. Now Create Action. Similarly, we create another applet to turn off the relay. Instead of writing “Turn on the T.V”, type “Turn off the T.V”. Instead of [“0”], type [“1”]. So now we have successfully created two triggers to turn on and off . So Similarly, we create triggers. Just change the phrase and Digital pin. All the other steps will remain the same. So, in the end for 4 appliances, we should have 8 triggers to turn each of them on or off.

  So now our appliances are voice controlled. We can also control virtual pins same as digital pins. Virtual pins can be used to control servo motor which is used to control regulator. That is we can change the state of motor from minimum to maximum and vice versa using voice commands

When this is all done, we will be able to say the voice commands to our Google Assistant.

**CHAPTER-5**

**CIRCUIT DIAGRAM & WORKING**

Here we are using NodeMCU. It consists of a microcontroller and a Wi-Fi module. Microcontroller is used to send signals to relay board. Wi-Fi module establish connection between smartphone and NodeMCU. We cannot connect the Google Assistant to the NodeMCU directly, so we are using Blynk app.

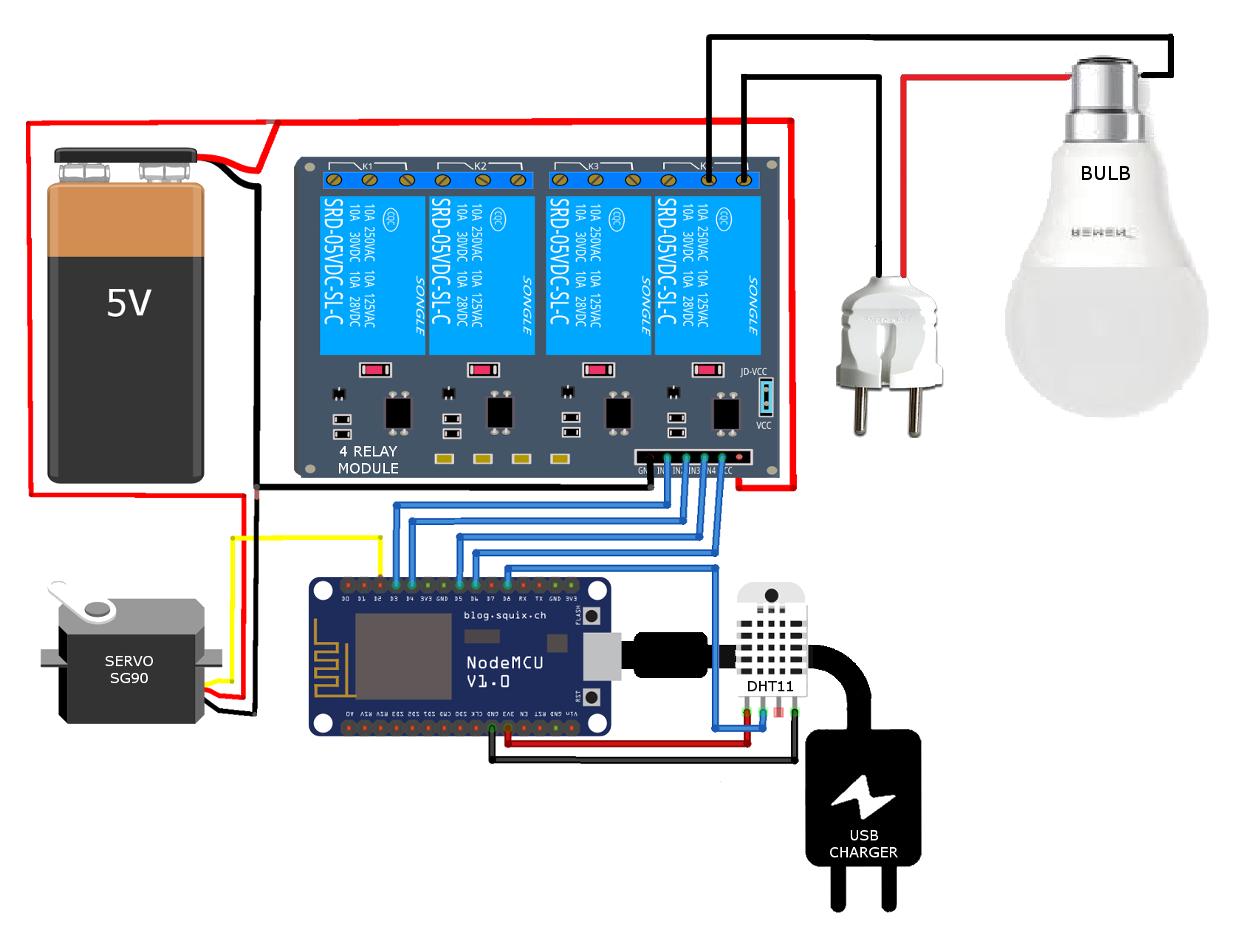
****

Figure 5.1 Circuit diagram of home automation system using google assistant

When we send voice commands to Google Assistant and IFTTT interprets and reaches Blynk app. Blynk app will then forward these commands to the NodeMCU pins which we selected. So, the ON or OFF signal according to the command reaches the NodeMCU microcontroller and it will send a signal to relay module. Relay module operate as a switch to which appliances are connected. By this way appliances are controlled using Google Assistant. We can see the temperature using Blynk app. By using Servo motor and slider we can control the regulator both by speech commands and Blynk app.

**CHAPTER-6**

**EXPERIMENTAL SETUP WITH RESULTS**

****

Figure 7.1 Green bulb gets turned ON by saying command “ Turn on green bulb ” in

Google Assistant

****

Figure 7.2 Green and white bulbs gets turned ON by saying command in

Google Assistant

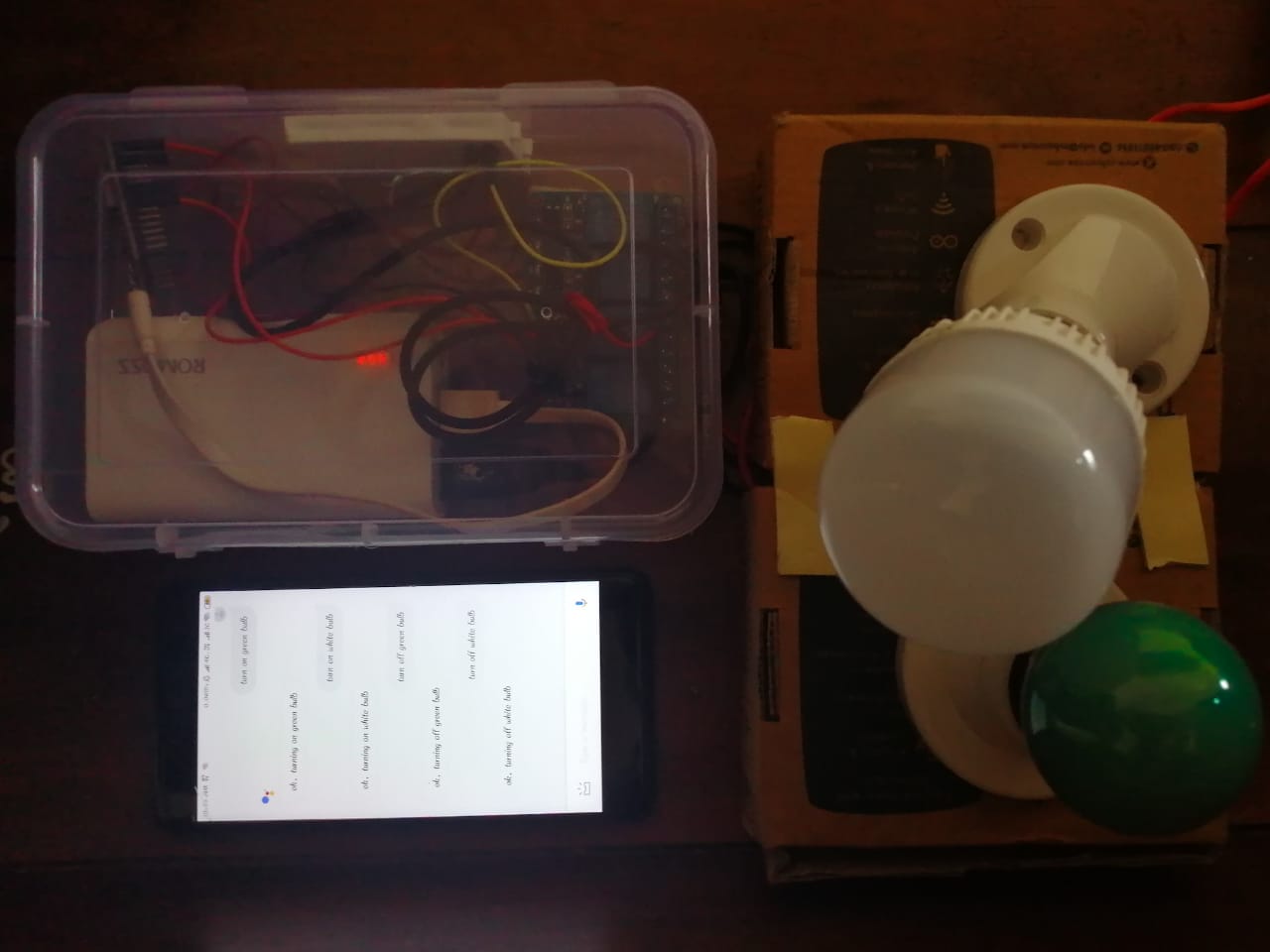
****

Figure 7.3 Both Green and white bulbs gets turned OFF by saying command in

Google Assistant

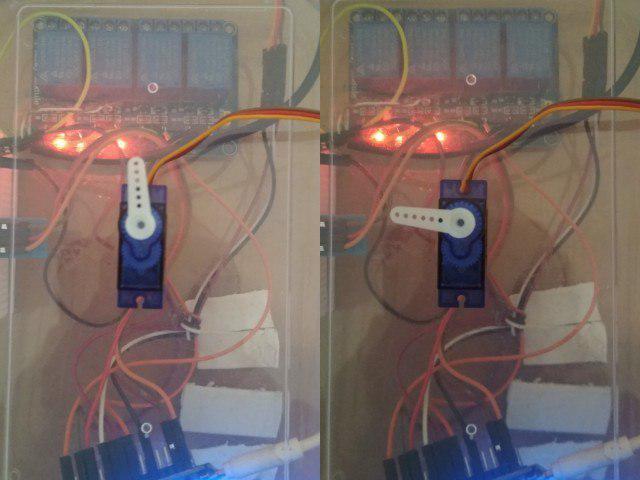


Figure 7.4 Controlling servo motor to control regulator

From the above result, we say that we can control our home appliances from anywhere by using a smartphone with Google assistant.

**CHAPTER-7**

**COST AND EXPENDITURE**

Table 7.1 Cost and expenditure

|  |  |  |
| --- | --- | --- |
| **COMPONENT** | **QUANTITY** | **PRICE (Rs)** |
| Male to Female Jumper wire (40 pcs) | 1 | 85 |
| Relay Module 4CH 5Volt | 1 | 249 |
| ESP8266 NodeMCU | 2 | 700 |
| WeMos D1 Mini | 1 | 415 |
| Dht 11 | 1 | 99 |
| Servo Sg90 | 1 | 180 |
| **Total cost** | | 1728 |

**CHAPTER-8**

**CONCLUSION**

A novel and simple Home Automation System using Google Assistant is proposed. We can ensure that the energy conservation can be done. By help of this system we can increase the efficiency of the appliances. We can have the complete control over the home appliances from a long distance. This will Increase the comfortability of human being and it will reduce the Human efforts.

**CHAPTER-9**

**FUTURE SCOPE**

The system can be available in very less cost to rural area peoples and schools. It reduces more complexity of things and easily available to people. It also increasing security of the system. It can be expanded for energy monitoring, or weather stations. This kind of a system with respective changes can be implemented in the hospitals for disable people or in industries where human invasion is impossible or dangerous, and it can also be implemented for environmental monitoring.

**REFERENCE**

(Journal paper)

**1. Aayush Agarwal, Anshul Sharma, Asim Saket Samad, S Babeetha,** *“UJALA- Home Automation System Using Google Assistant”* International Journal of Advance Research and Innovative Ideas in Education -ISSN(O)-2395-4396 Vol-4 Issue-2 2018

**2. Rudrendu Mahindar, Madhav Prakash, Sananda Ghosh, Sumani Mukherjee and Dr. Rabindranath Ghosh,** *“IoT-based Home Appliances Control System Using NodeMCU and Blynk Server*” International Advanced Research Journal in Science, Engineering and Technology Vol. 5, Issue 6, June 2018

**3. Mohammad miraj shekh, Asha S. R, Hariprakash, Harshitha,** “*IoT Based Home Automation using NODE MCU”,*International Journal of Engineering Science and Computing, Volume 8 Issue No.5 May 2018

**4. E. Yavuz, B. Hasan, I. Serkan and K. Duygu,** *"Safe and Secure PIC Based Remote Control Application for Intelligent Home”*, a. Volume 7, No. 5, May-2007

**5. Raj Sharma, Chirag, Pranjalkatara, Vishnu Shankar,** *“Proceedings of IEEE TechSym 2014 Satellite Conference VIT University, Paper on Advanced Low-Cost Security system using sensors, Arduino and GSM communication module”.*

**6. Vinay sagar K N, Kusuma S M,** *“Home Automation Using Internet of Things”* July 1999

**7. Niharika Shrotriya, Anjali Kulkarni, Priti Gadhave,** International Journal of Science, Engineering and Technology Research (IJSETR), “*Smart Home Using WI-FI*” December 1996

**8.** **Atzori, L., Iera, A., and Morabito G,** *“The internet of things: A survey.”* Computer networks, 2010 54(15), 2787-2805