

Abstract

Home automation has gained a lot of admiration and marketability in recent years due to the rapid growth of cutting-edge technology. Life is getting simpler as almost everything has become automatic and digitalized. In this project, we designed a cheap and wireless home automation system that can control different electronic components via the internet. A renowned open-source firmware and IoT platform, Node MCU has been used for executing the automation process. The different transmission modes are utilized to communicate the control of the device by the user through Node MCU to the electrical appliance. To provide remote access from mobile phones, the main control system implements wireless technology. We have employed a cloud server-model for communication which adds to the practicality of the project and enables unrestrained access of appliances to the user regardless of the distance. The system has been designed with the intention to control appliances with a relatively user-friendly GUI, low cost, and ease of installation for the elderly and disabled people.

Table of Contents

List of Figures/Diagrams

Table 1. Allocation of work.....	18
Table 2. Project Timetable (Schedule).....	19
Figure 4.1 system block diagram	20
Figure 4.2 Schematic diagram	22
Figure 4.3 System Flowchart.....	20
Figure 5.1.1 DHT11 module	23
Figure 5.1.2 Single Channel Relay Module (LEDs).....	26
Figure 5.1.3 Single Channel Relay Module (Output Terminal Block)	27
Figure 5.1.4 Single Channel Relay Module (Control Pins)	27
Table 3: Pin Configuration of Node MCU Development Board	28
Figure 5.1.5 GPIO Pins	30
Figure 5.1.6 SPI Pins.....	31
Figure 5.1.7 UART Pins.....	32
Figure 5.1.8 ADC Pin	34
Figure 5.1.9 PWM Pins	35
Figure 5.1.10 ESP-12E module on NodeMCU.....	36
Figure 5.1.11 Power module on NodeMCU.....	37
Figure 5.1.12 Multiplexed GPIO pins on NodeMCU.....	38
Figure 5.1.13 Onboard switches and LED indicators on NodeMCU.....	38
Figure 5.1.14 CP2120 on NodeMCU.....	39
Figure 5.1.10 Blynk Application	40
Table 4. Temperature and Humidity data from DHT11.....	47
Table 5. Status of fan with regards to threshold temperature.....	48
Figure 6.2.1 Temperature-Time graph.....	50
Figure 6.2.2 Humidity-Time graph.....	

chapter I

Introduction

The Internet of Things, or IoT, is a system of interrelated computing devices, which are assigned a unique IP address to be identifiable on the internet. These systems are capable of transferring data over a network without the need of human-to-human or human-to-computer interaction. IoT systems let appliances be connected and remotely accessed across the Internet. Over the last few years, the IoT concept has had a robust evolution and expansion, and is presently utilized in various disciplines such as smart homes, telemedicine, industrial environments etc.

A global interconnection of smart appliances with cutting-edge functionalities is facilitated by wireless sensor network technologies incorporated into the IoT. The fundamental technology to making intelligent homes is a wireless home automation network, comprised of sensors and actuators that share resources and are interconnected to each other.

A series of benefits are brought about by home automation systems. Through appliance and lighting control safety is provided, through automatic door locks homes are guarded, awareness is improved through security cameras. These save valuable time, give authority, and save money. The effortless management and control using various devices, including smartphones, laptops, tablets, smart watches, or voice assistants is one of the most prominent advantages of home automation systems.

1.1 Background of the project

Home Automation Systems (HASs) are composed of a centralized control and remote status monitoring of lighting, security system, and other appliances within a house. HASs promote energy efficiency, enhance security systems, and undoubtedly the comfort and ease of users. According to Statista, revenue in the Smart Home market is envisioned to reach US\$20million in 2022 in Oman. Despite the enormous growth in marketability every year, HASs come with their challenges, however. It can be irksome and sometimes quite disruptive when our smart devices quit communicating concurrently because of a network issue. There are more than 3 billion mobile users globally and nearly 8 billion IoT devices creating a very extensive network of devices as reported by [hdlautomation](#). But the enormity of the network imposes security risks. Elderly and disabled, aren't seen to accept the system due to the sophistication and cost factor even though aided to a great extent.

1.2 Statement of the problem

The project aims to answer the following questions:

1. Does the project benefit society?
2. Why is home automation required?
3. Where is the project most applicable and beneficial?
4. What are the challenges concerning IoT in home automation?

1.3 Objectives of the project

1. control home appliances remotely (Wireless)
2. Secure links between application and Node MCU. The communication between the devices and the application is secure as protocols such as SSL (Secure Socket Layer) are used to secure the communication channel so that intrusive devices are not allowed to control the devices
3. Save power consumption and reduce electricity bills.
4. Extendable platform for future improvements.

5. **1.4 Scope and Limitation of the project**

The following shows the scope of the project:

1. Effortless management of several home appliances with just a single app from anywhere across the globe.
2. Comfortable, benefited, elegant, yet simple living style
3. Implementing the system in homes, small offices, malls etc. and being in charge of control of appliances
4. Developing a technology friendly environment where the system incorporates the use of technology and makes home automation possible. By the use of widgets, we can operate common home appliances from a different perspective.

The limitations of the project are the following:

1. Our IoT project won't execute without internet and internet connectivity and its reliability is unpredictable, to this day.
2. As a matter of course, there will be fewer requirements of human resources, mainly workers and less educated staff with daily activities getting automated. This may create unemployment issues in society.
3. With all home appliances, personal gadgets, public sector services like transport, water and electricity supply, industrial machinery and equipment connected to the internet, a lot of information is available and obtainable on it. This information is prone to attack by hackers.
4. Node MCU doesn't only rely on the internet but also a free server issued by a third party. If the server doesn't function, then the circuit will also cease to operate.

1.5 Significance of the project

We decided to set up an electrical circuit to control the home's electrical appliances through the internet application, where the appliances in the home are converted into smart appliances, and then they are controlled from any part of the home or world even. An application is downloaded into the phone in order to control the household appliances, and the Internet must be provided in order for it to be easily controlled.

1.6 Definition of terms

A. NodeMCU

is an open-source firmware for which open-source prototyping board designs are available. The firmware utilizes the Lua scripting language. The firmware is based on the eLua project and is built on the Espressif Non-OS SDK for ESP8266. Due to resource limitations, users need to select the modules relevant to their projects and build firmware customized to their needs. Support for the 32-bit ESP32 has also been implemented.

B. Single channel Relay Module

The Single Channel Relay Module is a board used to control high voltage, and high current loads such as motor, solenoid valves, lamps, and AC loads. This board is developed to interface with a microcontroller such as Arduino, PIC, NodeMCU, etc. The relays terminal (COM, NO, and NC) is being fetched out into a screw terminal. This module also includes 2 LEDs to reveal the status of the relay.

C. Resistor

Resistance is the opposition to the current flow. The unit of Resistance is Ohms Ω . All conductors exhibit a particular amount of resistance since no conductor is 100% efficient. To contain the electron flow predictably, we use resistors. A resistor can be divided into two groups, fixed & adjustable (variable) resistors. In fixed resistors, the value is already fixed & cannot be altered. In variable resistors, we can change the resistance value by scrolling an adjuster knob. Resistors can also be split into (a) Carbon composition (b) Wire wound (c) Special type. The most typical resistor used in our projects is the carbon type.

D. Blynk Application

Blynk is an IoT platform with designated Android and IOS apps that can handle many hardware modules like Arduino, NodeMCU. To connect the hardware module device to the internet, many choices of connectivity are available for example Wi-Fi, Ethernet, Cellular, USB, and Bluetooth. The GUI (Graphical User Interface) of Blynk App is remarkably User-friendly. Users can add widgets simply by dragging and dropping them and controlling many appliances with these widgets.

E. LED

LED is a light-emitting diode mainly used in electronic devices such as watches, light bulbs, mobile phones, monitors and many more. They allow current to flow only in one direction. Because they have a longer life span and are more energy efficient than regular bulbs, they have often replaced them, especially in home environment

Chapter II

Review of Related Literature

“Voice Controlled Home Automation System using Natural Language Processing and Internet of Things”, by Mrs. Paul Jasmin Rani, Jason Bakthakumar, Praveen Kumaar.B, Praveen Kumaar.U, Santhosh Kumar.

The paper concentrates on the building of a voice-based home automation system that employs Internet of Things, Artificial Intelligence, and Natural Language Processing (NLP) to provide an economical, efficient way to concurrently work with home appliances using different technologies such as GSM, NFC, etc. it implements a seamless integration of all the appliances to a central console, i.e. the mobile device. The prototype uses Arduino MK1000. User is given the freedom to interact with the home appliances with their voice and language instead of complex computer commands using the NLP feature on this project. The appliances are linked to the mobile via an Arduino Board that demonstrates the idea of Internet of Things. The Arduino Boards are interfaced with the appliances and coded in a way that they react to mobile inputs.

“Smart Energy Efficient Home Automation System Using IoT”, by Babita Kumari, Arun Kumar Mishra, Satyendra K. Vishwakarma, Prashant Upadhyaya.

This paper delivers a step-by-step methodology of a smart home automation controller. An energy efficient system that uses voice and a web-based service (Adafruit IO) to control home appliance. By implementing IoT, home appliances are converted to smart and intelligent devices which are controlled from any corner of the world. The proposed system utilizes Node MCU, an open-source firmware that provides flexibility to build many IoT based application, Adafruit, a library that supports MQTT and acts as an MQTT broker, IFTT, an interface that comes with web services allow devices to connect to mobile apps. This is a multimodal system as it uses google assistant along with a web-based service to direct the smart home. The main controller unit, Node MCU is connected to the Wi-Fi that must be available 24/7. The microcontroller unit can be programmed to establish an automatic connection to available networks and connected to auto power backup to ensure the Wi-Fi connection do not switch off.

“IoT Based Smart Security and Home Automation”, by Shardha Somani, Parikshit Solanki, Shaunak Oke, Parth Medhi, Prof. P. P. Laturkar.

This paper converges on a system that provides features of Home Automation relying on IoT to operate efficiently, along with a camera module, and offers home security. If movement is sensed at the entrance of the house via motion sensors; a notification is sent to the owner through the app, which contains a photo of the house entrance in real-time. The owner can ring an alarm in case of any intrusion, or they can toggle the appliances like unlocking the door if the person is a visitor. Raspberry Pi, a small-sized computer that acts as a server is used by the system. The smart home consists of two modules. Home automation composing fanlight and door controller, and a security module that contains a smoke sensor, motion sensor, and a camera module.

Chapter III

Project Plan

3.1 Identification of Activities

week 1: After first meeting with the project coordinator, we will choose a supervisor for our project work.

week 2: Choose a project, its title, and innovate it by adding extended features.

week 3: surf the internet for similar projects and read research articles on the chosen topic.

week 4: draw analyze circuit diagram, draw schematic, and block diagram, and write chapter 1 and poster

week 5: Testing the circuit in lab after collecting all the required components and write down the chapter 2 and 3.

week 6: write chapter 3, chapter 4 and chapter 5

week 7: continue to write chapter5 ,6 and 7

3.2 Allocation of work

Activities	BAYAN	MAYSA	NISHA	ZAINAB
Choosing a project topic and title				
Surfing the internet for similar projects and reading research articles.				
Adding new features and drawing the circuit diagram				
Chapter 1				
Chaprer2				
Chapter 3				
Chapter 4				
Chapter 5				
Chapter 6				
Logbook				

3.2 Project Timetable (Schedule)

Activities	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7
Choose a supervisor							
Select a project, it's title and innovate it							
Read research articles on the chosen topic							
Draw and analyze circuit diagram, draw schematic, block diagram, and write chapter 1 and design poster							
Test the circuit in lab and write chapter 3,4, and 5							
Continue writing chapter 5,6, &7							

Chapter IV

Project Design

4.1 System/Block diagram

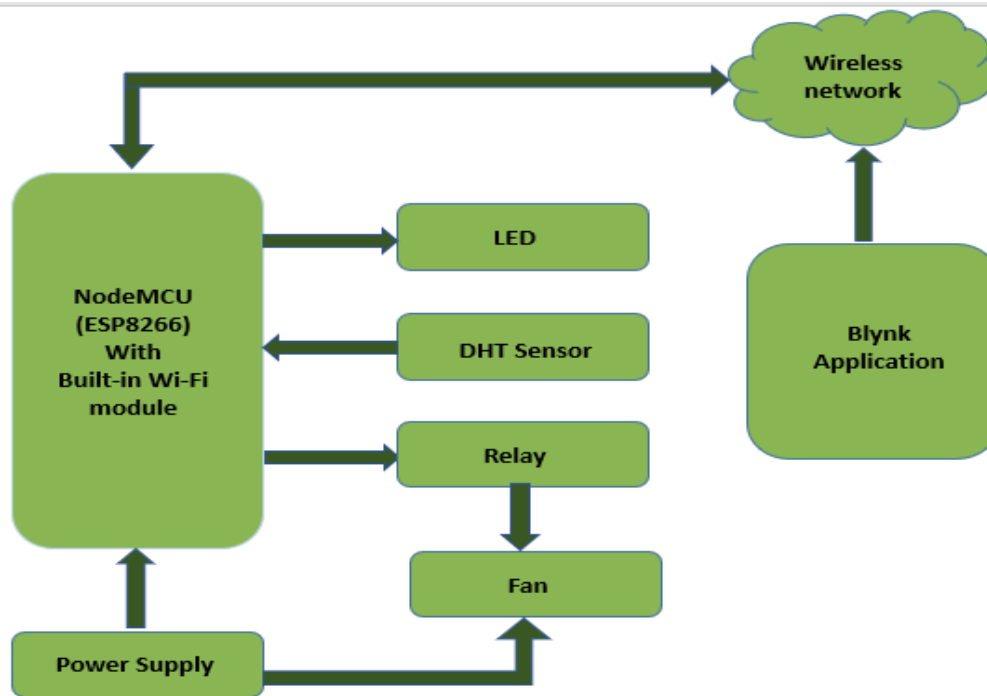


Figure 4.1 system block diagram

The block diagram gives the function of the overall project. The NodeMCU is the major controlling unit of the system. The user uses the mobile application in setting commands for the functioning of the appliances. The mobile application interprets the command from the user, then transmits a signal to the Node MCU unit, over a wireless network established by Wi-Fi communication. Hence the Wi-Fi module (inbuilt into Node MCU), enables the microcontroller to establish Wi-Fi communication with a device and carry commands from an application over a wireless network. The Node MCU further receives the signal and then turns on/off the appliance(fan) with the assistance of a relay or controls the brightness of the LED. The Node MCU, relay, and LED are physically linked. There is a power supply unit to power the NodeMCU and the fan.

4.2 Schematic diagram

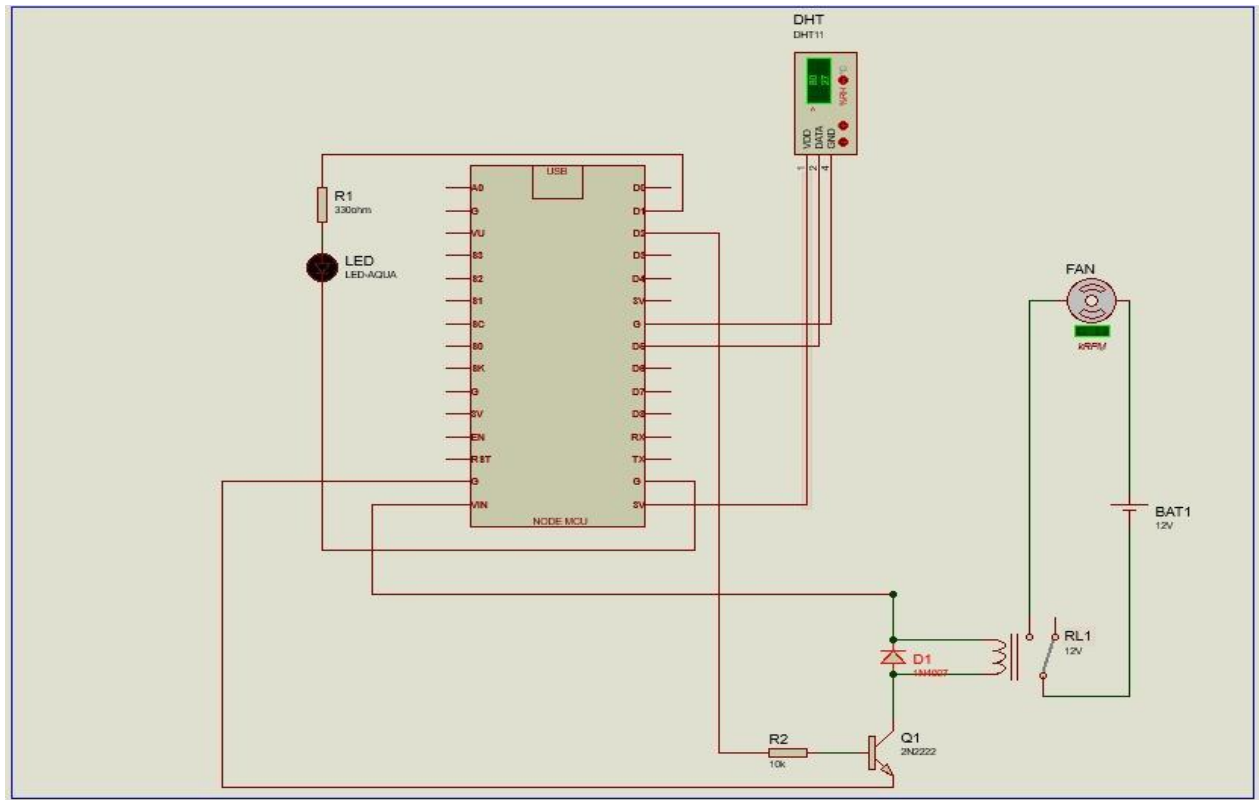


Figure 4.2 Schematic diagram

4.3 System Flowchart

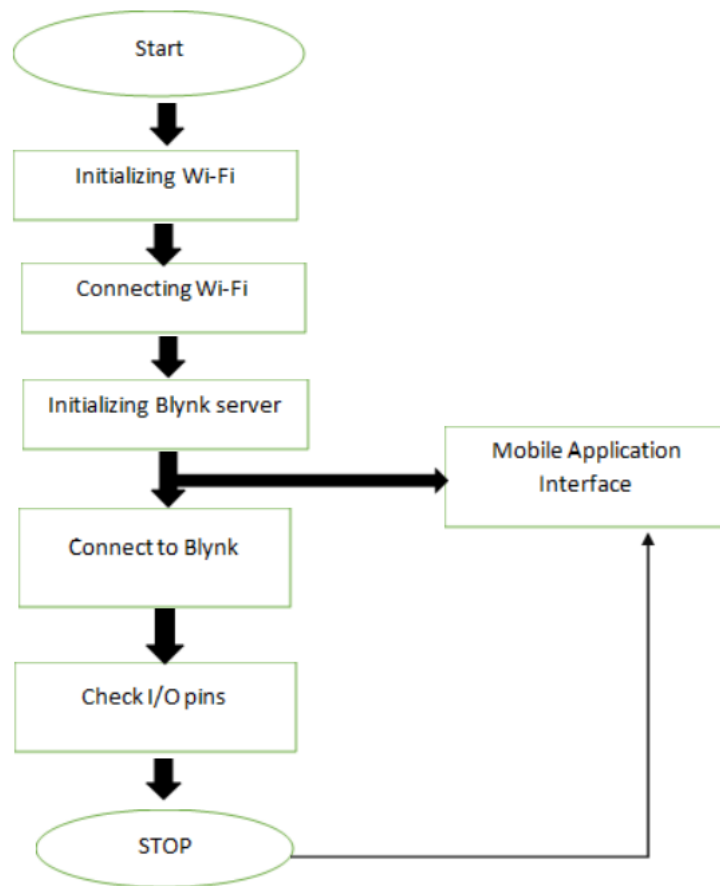


Figure 4.3 System Flowchart

This flow chart shows the functioning of the project. The operation starts by initializing the Wi-Fi. The android device is linked to Node MCU over Wi-Fi. The Blynk server is set up and a connection is made, the device is identified in the Blynk server using the template ID and device name included in the source code. The command for controlling the load is given to the blynk application by the user, and this command, over the Wi-Fi network, is sent to the Node MCU which then transmits a signal to the appropriate pins.

Chapter V

Simulation and Testing

5.1 Functions of Major Components

A. DHT11 Module

DHT11 module is a cheap, small-sized & easy-to-operate embedded sensor. It is used to measure temperature, relative humidity and constructs a scaled digital output. DHT11 module Pinout consists of 3 Pins in total, as listed below:

- a. **Vcc**: Supply +5V at this pinout.
- b. **Data**: This the digital output pin, that delivers either 0V or 5V.
- c. **GND**: supply Ground at this pin.

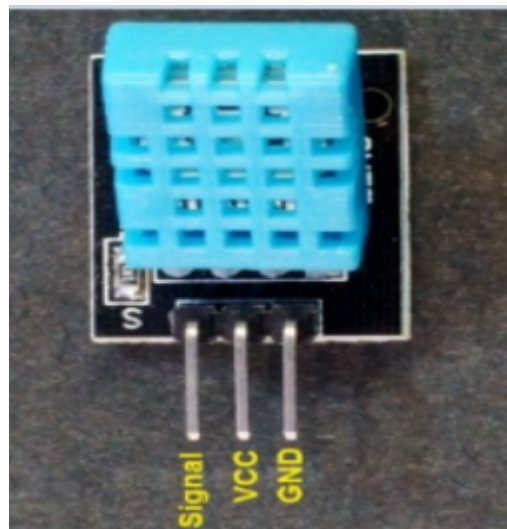


Figure 5.1.1 DHT11 module

DHT11 sensor contains a capacitive humidity sensing element for measuring humidity. The humidity sensing capacitor includes two electrodes with a moisture-holding substrate that serve as a dielectric between them. As moisture content changes in surroundings, these substrates get accumulated on the substrate material. This varies the resistance between electrodes. The IC

measures and processes these changed resistance values and transforms them into digital form. For measuring temperature, DHT11 sensor uses a Negative Temperature coefficient thermistor, whose resistance decreases with an increase in temperature. Even for the slightest temperature change we get a larger resistance because NTC is usually made from ceramics or polymers.

The sampling rate of this sensor is 1Hz.i.e., it can read temperature and humidity every second. operating voltage of DHT is from 3 to 5 volts. The maximum current drawn whilst measuring is 2.5mA. The response time 6-15s and has a resolution of 8-bit.

A single-wire, two-way Serial Protocols used by DHT sensor for communicating with devices like microcontrollers. Because DHT11 receives commands from the microcontroller and responds back with a pulse for acknowledgment it is also known as 2-way communication protocol. Data sent by the DHT11 sensor is 40bits and it transmits Higher Data Bits first. Both Integral and decimal values of temperature and relative humidity along with a checksum value for error detection are included in that data

DHT11 transmits the 40Bit serial data in the below configuration:

- a.8-Bit Humidity (Integral)
- b.8-Bit Humidity (Decimal)
- c.8-Bit Temperature (Integral)
- d.8-Bit Temperature (Decimal)
- e.8-Bit Checksum

The data pin of DHT11 remains at the HIGH voltage level and the sensor remains in low power consumption mode. To start the communication process, a start pulse is sent to the sensor from the microcontroller which is achieved when the data pin is pulled down to GND for at least 18 μ s. DHT11 would then transmits out the 40-Bit calibrated output value serially when the Data Pin gets HIGH again by the microcontroller. After transmitting the data, DHT11 goes back to low power consumption mode and awaits next instruction from the microcontroller.


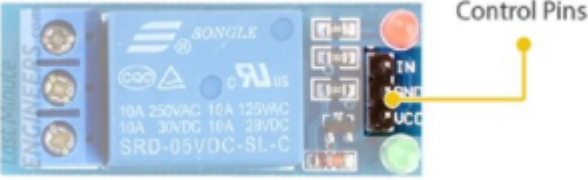
For acquiring a response from the DHT11 sensor the microcontroller has to wait 20-40 μ s.

B. Single Channel Relay Module

RELAY A relay is an electrically driven switch. When current flows through the coil of the relay, it produces a magnetic field that draws a lever and alters the switch contacts. Common (COM), normally closed (NC), and normally open (NO) are the usual labels on a relay. When there is no magnetic field in the coil, COM and NC are coupled. Once the relay coil gets magnetized, it pulls the lever so COM and NO will be connected now.

we are using a 5V 1 channel relay module in our project. This module is devised for switching only a single high-powered device from our NodeMCU. Therefore, it contains additional components such as LEDs, an output terminal block, and control pins.

Components	Description
<p>LEDs</p>	<p>There are two LEDs on the relay module indicating the position of the relay. When the module is powered, the Power LED will light up. When the relay is activated Status LED will illuminate.</p> <div data-bbox="656 562 1269 1054" data-label="Image"> </div> <p>Figure 5.1.2 Single Channel Relay Module (LEDs)</p>
<p>Output Terminal Block</p>	<p>We have three channels of the relay coming out into blue screw pin terminals. These channels are labeled for their part: common (COM), normally closed (NC), and normally open (NO)</p> <p>The labels define the state of the channel in relation to the switch at rest.</p> <p>COM (Common): This is the pin we connected to the negative terminal of the battery</p> <p>NC (Normally Closed): This configuration is used is when we want to turn off the relay by default. Here, the com is always connected NC and remains like this until we transmit a signal from the node MCU to the relay module to open the circuit.</p>

	<p>NO (Normally Open): this works the other way round. The relay is always open until we send a signal from the nodemcu to the relay module to complete the circuit.</p>  <p>Figure 5.1.3 Single Channel Relay Module (Output Terminal Block)</p>
Control Pins	<p>There are three pins on the side opposite to terminal block – a Ground pin for ground connection, a VCC pin to power the module, and an input pin IN to control the relay. The input pin is active low, implying that the relay will be triggered when we pull the pin LOW, and it will be inactive when the pin is pulled HIGH.</p>  <p>Figure 5.1.4 Single Channel Relay Module (Control Pins)</p>

C. NodeMCU

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). “NodeMCU” rigidly refers to the firmware rather than the associated development kits. The firmware utilizes the Lua scripting language. The firmware is based on the eLua project and is built on the Espressif Non-OS SDK for ESP8266. It employs many open-source projects, such as Lua JSON and SPIFFS. Due to resource limitations, users need to select the modules relevant to their projects and build firmware customized to their needs. This development kit also support the 32-bit ESP32.

Pin Configuration of Node MCU Development Board

We can access the I/O index number of Node MCU kits but not the internal GPIO pins. For instance, the D0 pin on the development kit is mapped to GPIO pin 16. The GPIO pins accesses are provided by Node MCU. The following pin mapping table

Table 3. Pin mapping in NodeMCU development board

Io index	ESP8266 pin	IO index	ESP8266 pin
0[*]	GPIO16	7	GPIO13
1	GPIO5	8	GPIO15
2	GPIO4	9	GPIO3
3	GPIO0	10	GPIO1
4	GPIO2	11	GPIO9
5	GPIO14	12	GPIO10
6	GPIO12		

[*] D0 (GPIO16) can only be used for GPIO read/write. It won't support open-drain/interrupt/PWM/I²C or 1-Wire. The ESP8266 Node MCU has a total of 30 pins that interface it to the outside world. The pins are categorized by their functionality as

Pin Category	Description
Power	<p>Micro-USB: NodeMCU can be powered through the USB port.</p> <p>3.3V: The three 3.3V pins are used to supply power to outer components.</p> <p>GND: is a ground pin of ESP8266 Node MCU development board.</p> <p>Vin: Power is supplied to ESP8266 and its peripherals directly by the VIN pin through a regulated 5V voltage source</p>
Control	<p>These pins are used to control ESP8266. These pins contain a Enable pin (EN), Reset pin (RST), and WAKE pin.</p> <p>EN pin – The ESP8266 chip is enabled when the EN pin is pulled HIGH. The chip operates at the lowest power when pulled LOW.</p> <p>RST pin – RST pin resets the ESP8266 chip</p> <p>WAKE pin – A WAKE pin awakens the chip from deep-sleep. To awaken the ESP8266 from a deep sleep GPIO16 should be coupled to the RST pin.</p>
Analog	Used to measure analog voltage in the range of 0-3.3V
	<p>General-purpose input/output (GPIO) is a pin on an IC (Integrated Circuit). It can be either an input pin or output pin, whose actions can be controlled when the program is executed.</p> <p>Access to these GPIOs of ESP8266 is provided by the NodeMCU Development kit. However, number on the NodeMCU Devkit pins different than internal GPIO notations of ESP8266.</p>

GPIO

For instance, the D0 pin on the NodeMCU Devkit is mapped internally to GPIO pin 16 of ESP8266.

The GPIO's shown in the blue box (1, 3, 9, 10) are not generally used for GPIO purposes on Dev Kit. The processor includes 17 GPIO lines, but some of them are used internally to interface with other components of the SoC, e.g., flash memory.

This means we have about 11 GPIO pins remaining for GPIO purposes.

2 pins out of 11 are commonly reserved for RX and TX to communicate with host PC from which compiled object code is downloaded. Consequently, only 9 general-purpose I/O pins remain for users i.e., **D0 to D8**.

RX, TX, SD2, SD3 pins are not typically used as GPIOs since they are used for other internal processes.

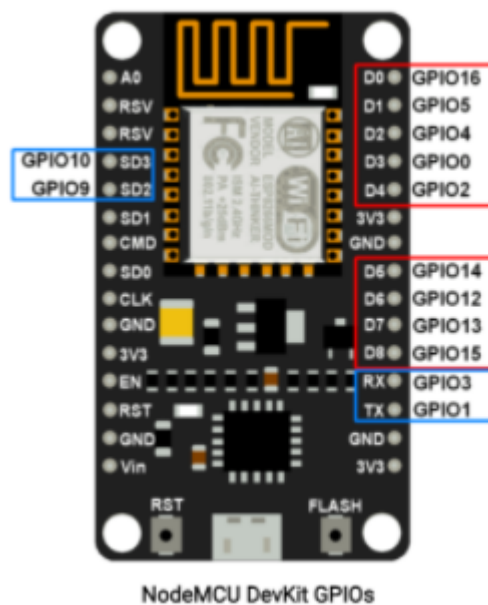


Figure 5.1.5 GPIO

SPI

The Serial Peripheral Interface (SPI) is a bus interface connection protocol. It is a full-duplex master-slave communication protocol and uses four wires for communication. SPI-enabled devices operate in two basic modes i.e., SPI Master Mode and SPI Slave Mode. Any SPI-enabled devices can be connected to the SPI interface to allow communication between the two. The master Device initiates the communication, where it generates a Serial Clock for synchronous data transfer. These SPI pins (SD1, CMD, SD0, CLK) are exclusively used for Quad-SPI communication with flash memory on ESP-12E, hence, they can't be used for SPI applications. for user-end applications the Hardware SPI interface can be used. NodeMCU therefore has four pins(D5-D8) available for SPI communication.

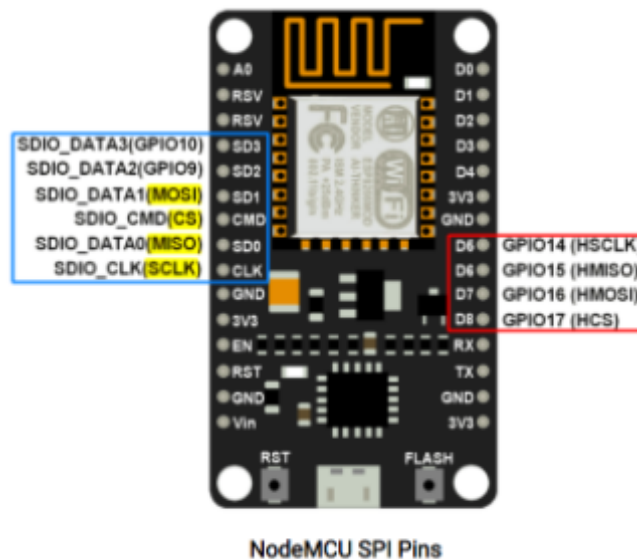


Figure 5.1.6 SPI Pins

UART (Universal Asynchronous Receiver/Transmitter) is a serial communication protocol in which data is transmitted serially bit by bit at a time. For byte-oriented transmission, Asynchronous serial communication is widely used. In this protocol, a byte of data is transferred at a time.

UART

UART serial communication protocol has a specified frame structure for their data bytes. A START bit (1 bit) followed by a data byte (8 bits) and then a STOP bit (1 bit), which constructs a 10-bit frame in asynchronous serial communication generally. The frame can also consist of 2 STOP bits instead of a single bit, and there can even be a PARITY bit after the STOP bit.

NodeMCU based ESP8266 owns two UART interfaces, UART0 (RXD0 & TXD0) and UART1(RXD1 & TXD1). The ESP8266 data transfer rate through UART interfaces can attain 4.5 Mbps. For the oscillator of 40MHz, the UART0 baud rate is 115200 by default. It can be changed to a user-defined value in line with the application requirements. UART1 is used to upload the program.

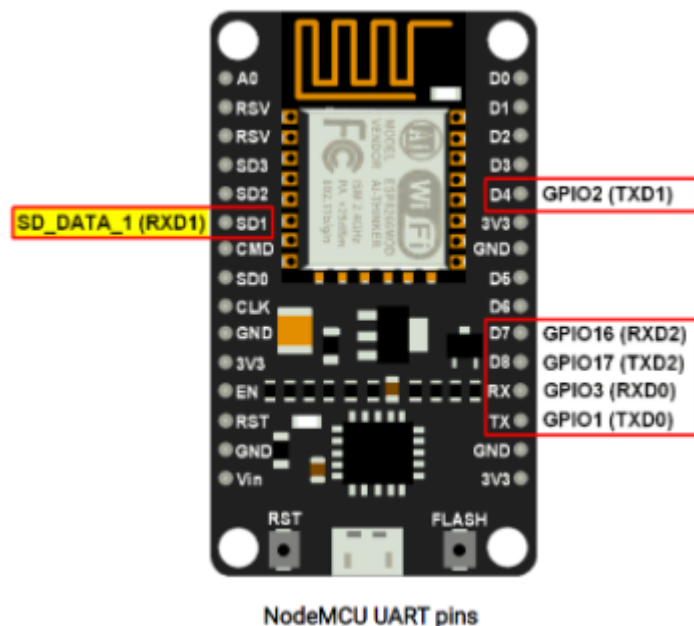


Figure 5.1.7 UART

I2C (Inter-Integrated Circuit) is a serial bus interface connection protocol. It uses only two wires for communication, SDA (serial data) and SCL (serial clock). To ensure data is received by the receiver successfully, the receiver sends an acknowledgment to the transmitter. SDA (serial data) wire is used for data exchange between master and slave devices. SCL (serial clock) wire carries the clock signal generated by the master device

I2C

synchronizes data transfer between master and slave devices. It needs a slave device address to begin a conversation with a slave device. The slave device then responds to the master device when it is addressed by a master device.

NodeMCU has I2C protocol features on its GPIO pins. Due to internal functions on ESP-12E, all its GPIO pins cannot be used for I2C functionality. So, we need to check up before using any GPIO for I2C applications. All types of I²C sensors and peripherals can be connected. Both I²C Master and I²C Slave are supported. I²C interface functions can be realized programmatically, and the maximum clock frequency is 100 kHz.

ADC

Analog to Digital Converter (ADC) converts the analog signal into digital form. The, main processor, ESP8266EX SoC in all the ESP8266 Boards, has only one pin for ADC input. The 10-bit resolution means the output values will be in the range of 0 to 1023.

The ADC can primarily be used for two types of measurements. They are:

- Measure the Power Supply Voltage at Pin 3 and Pin 4 i.e., VDD3P3.
- Measure the input voltage of Pin 6 i.e., TOUT Pin.

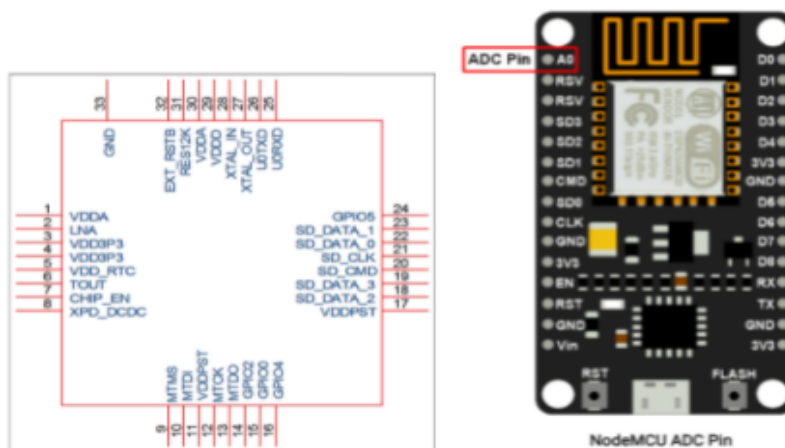


Figure 5.1.8 ADC pin

PWM

Pulse Width Modulation (PWM) is a method by which the width of a pulse is varied while maintaining the frequency of the wave constant.

A period of a pulse consists of an **ON** cycle (VCC) and an **OFF** cycle (GND). The proportion for which the signal is ON over a period is called the duty cycle. we can control the power delivered to the load by using the ON-OFF signal and duty cycle via the PWM technique.

NodeMCU based ESP8266 has the functionality of PWM interfaces through software programming. It is realized with the timer interruption method. The PWM frequency range for ESP8266 is adjustable from 100Hz to 1KHz. Controlling the speed of motors or varying the intensity of LED are 2 common applications of PWM

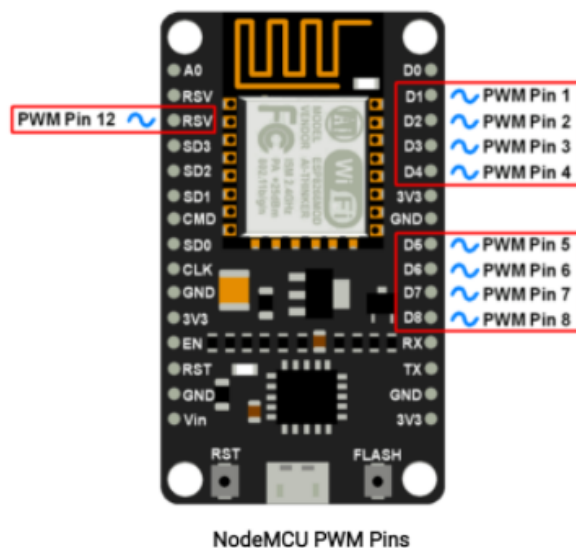


Figure 5.1.9 PWM pins

Node MCU development board parts

A. ESP12E module

An ESP12E module is furnished on the development board of NodeMCU. The ESP8266 chip is equipped with a Tensilica Xtensa® 32-bit LX106RISC microprocessor that operates at an adjustable clock frequency from 80 to 160 MHz and supports an RTOS. There's also 128KB of RAM and 4MB of flash memory (for programs and data storage), enough to handle web pages, JSON / XML data, and the large strings that make up everything that's used in IoT-These days The ESP8266 integrates an 802.11b / g / n HT40 Wi-Fi transceiver, so you can not only connect to your Wi-Fi network and interact with the Internet, but also build your own network to allow other devices to connect directly. This makes the ESP8266 node MCU even more flexible compared to other microcontrollers.

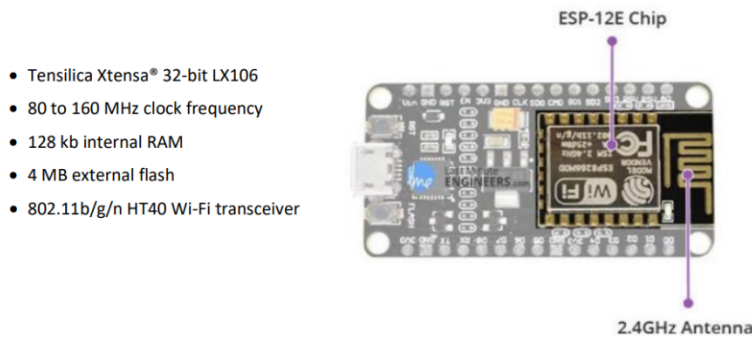


Figure 5.1.10 ESP-12E module on NodeMCU

B. Power requirements

The NodeMCU board has an LDO voltage regulator to maintain voltage constant at 3.3 V as the operating voltage range of ESP8266 is 3V to 3.6V. ESP8266 pulls about 80mA and It can reliably deliver up to 600mA which is adequate. The labeled 3v3 pin on the board is the output of the regulator broken out to this pin. This pin can be used to power external components. Onboard

Micro B USB connector can also be used to supply power to NodeMCU. Alternatively, if we have a regulated 5V voltage source, the VIN pin can be utilized to directly supply the ESP8266 and its peripherals.

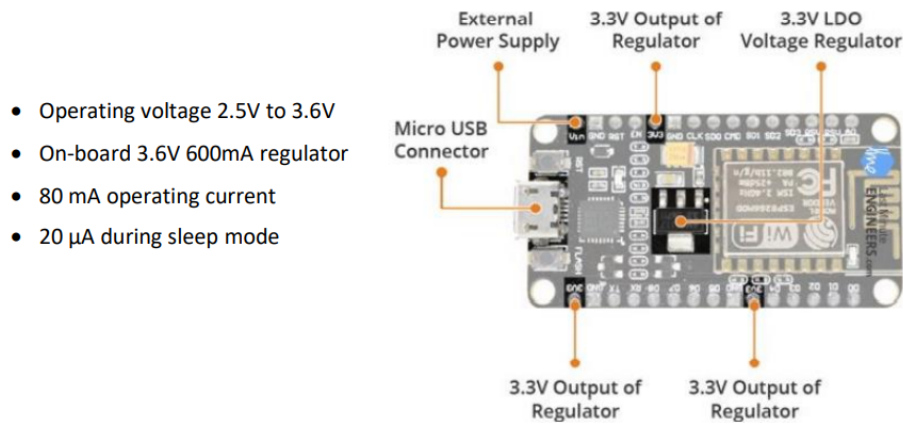


Figure 5.1.11 Power module on NodeMCU

C. Peripheral I/O

A total of 17 GPIO pins has been broken out to the pin headers on both sides of the development board. These pins are designated to all sorts of peripheral functions, including

- **ADC channel** – A 10-bit ADC channel that converts analog voltage to a digital form.
- **UART interface** – UART interface is utilized to load code serially.
- **PWM outputs** – PWM pins are utilized for dimming LEDs or controlling motors.
- **SPI, I2C** – SPI, and I2C interfaces allow communication between sensors and v peripherals.

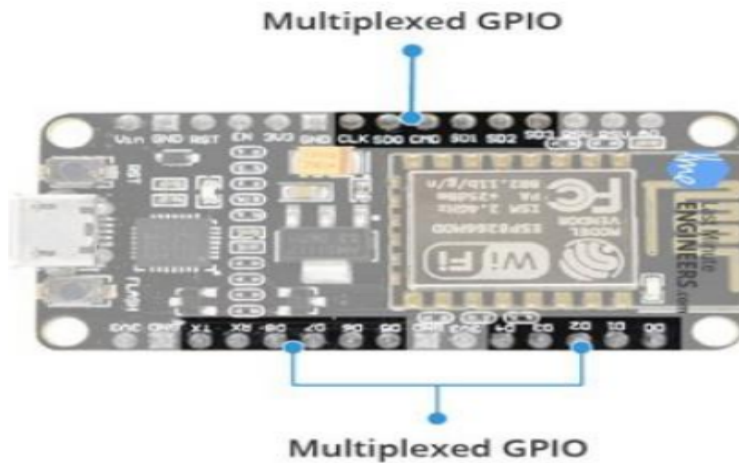


Figure 5.1.12 Multiplexed GPIO pins on NodeMCU

D. Onboard switch and LED indicator

The ESP8266 Node MCU contains two buttons. One labeled as RST; a reset button located on the top left corner that is used to reset the ESP8266 chip. On the bottom left corner is the FLASH button, which is the download button used while upgrading firmware. For testing purposes, the board also comes up with an LED indicator. It is programmable and is linked to the D0 pin of the board.

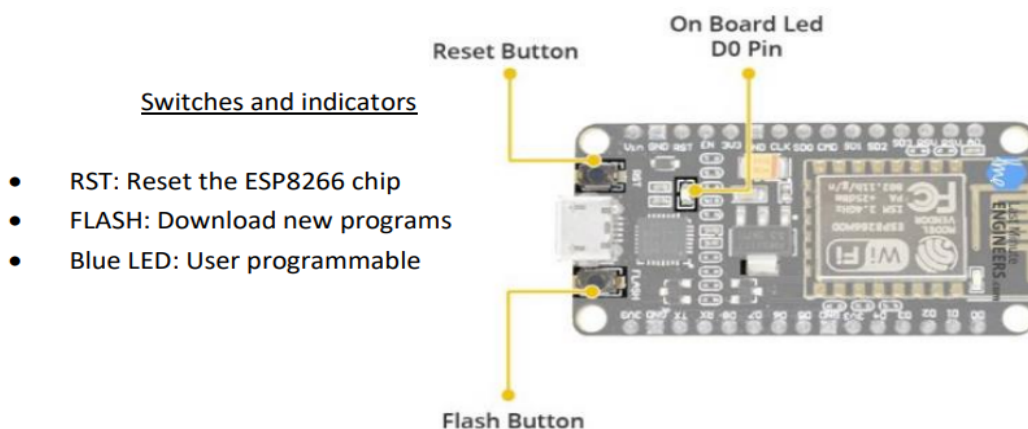


Figure 5.1.13 Onboard switches and LED indicators on NodeMCU

E. Serial Communication

To transform a USB signal to serial and enable our computer to program and communicate with the ESP8266 chip, the board comes up with a CP2102 USB to UART Bridge controller.

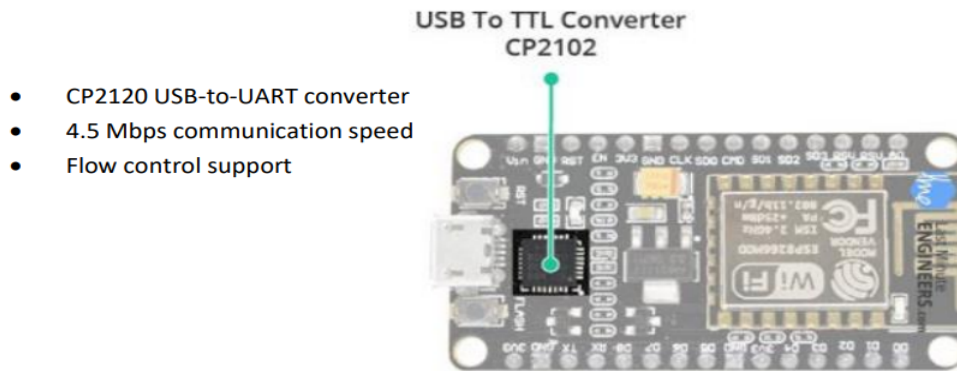


Figure 5.1.14 CP2102 on NodeMCU

D. Blynk Application

The Blynk application was designed for the primary goal of the Internet of Things. Blynk provides a platform with iOS and Android apps to manage Arduino, Raspberry Pi, node MCU, and the likes over the Internet. It's a digital dashboard where a graphic interface for a prototype can be created by merely dragging and dropping widgets. It can control hardware remotely. For example, it can display sensor data, accumulate, store, and visualize data, and include a lot more features. There are three primary elements in the platform:

- ❖ **Blynk Application:** this allows us to build impressive interfaces for our projects using various widgets provided.
- ❖ **Blynk Server:** This plays a role in all the transmissions hence, communication between the smartphone and hardware. We can use Blynk Cloud or operate our private Blynk

server locally. Blynk server is an open source and could effortlessly manage hundreds of devices.

- ❖ **Blynk Libraries:** This enables communication with the server and processes all the incoming and outgoing commands for all the prevalent hardware platforms. Every time a user accesses a radio button in Blynk application, the message is carried to the Blynk Cloud, where it finds the specific hardware by the unique generated template ID and Device name. It operates in the same manner in the opposite direction.

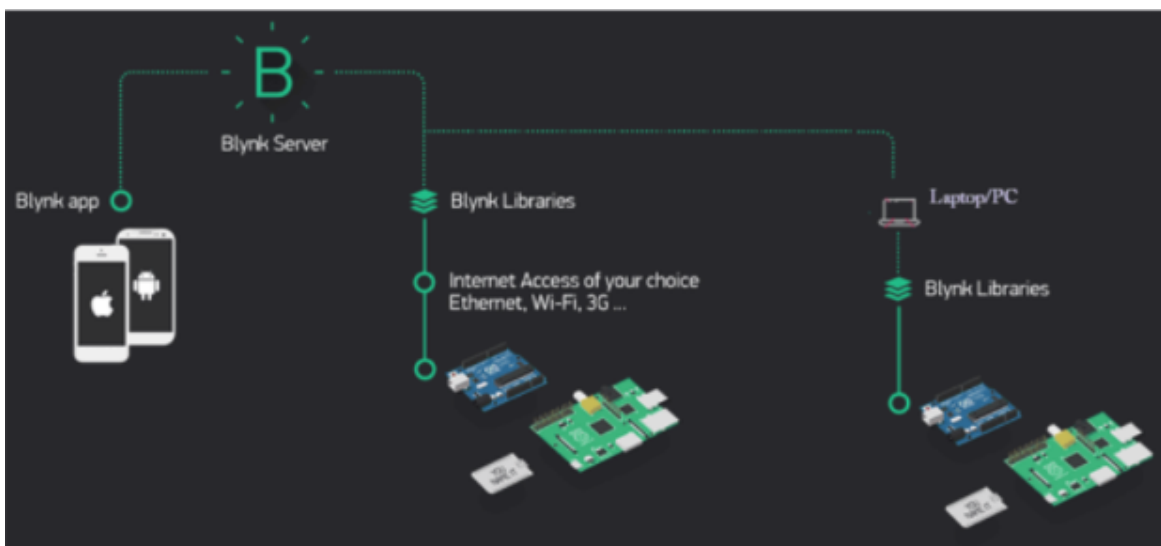


Figure 5.1.15 Blynk Application Working Principle.

code for the project

```
#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[] = "2ro_L9gmugpg6FfbtZ4Il5QMxJqlHY34";
// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "AndroidAP3DEC";
char pass[] = "123456";
void setup()
{
  // Debug console
  Serial.begin(9600);
  Blynk.begin(auth, ssid, pass);
  // You can also specify server:
  //Blynk.begin(auth, ssid, pass, "blynk-cloud.com", 80);
  //Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,100), 8080);
}
void loop()
{
  Blynk.run();
}
BLYNK_WRITE(V0)
{
  int pinValue = param.asInt(); // assigning incoming value from pin V1 to a variable
  // You can also use:
  // String i = param.asStr();
  // double d = param.asDouble();
```

```
analogWrite(D0, pinValue);  
Blynk.virtualWrite(V1, pinValue);  
Serial.print("V0 Slider value is: ");  
Serial.println(pinValue);  
}
```

5.2 Functional Simulation

We couldn't find any software that could simulate our project.

5.3 Actual Functional Test

We experimented the functioning of our project in the project lab. It performed exactly as we anticipated it to be. The fan only turned ON when the temperature was greater than the set threshold temperature and we could control the brightness of the LED as we drove the blynk slider and the gauge displayed the value of light intensity in lux. Whilst Node MCU was powered on, we collected temperature and humidity values for 30 mins at 2 mins intervals. The threshold temperature had been incremented or decremented to various values of temperatures and it was observed that the fan turned OFF whenever the threshold value was greater than the room temperature and vice versa.

5.4 Tabulation of actual outputs

Table 3. Temperature and Humidity data from DHT11

Time /min	Temperature /°C	Humidity / %
0	25	50
2	25.1	50
4	25.0	51
6	25.0	50
8	25.3	51
10	25.3	51
12	25.3	51
14	25.7	51
16	25.7	51
18	25.4	52
20	25.5	52
22	25.7	52
24	25.7	52
26	25.7	52
28	25.4	53
30	26.0	53

Table 4. Status of fan with regards to threshold temperature

Temperature /°C	Threshold temperature	Status of fan ON/OFF
25.7	15	ON
25.3	20	ON
25.0	25	ON
25.0	30	OFF
25.5	35	OFF

Chapter VI

Results and Discussion

6.1 Findings

Our project was successful as the results came out as expected. It took us a significant amount of time to interface the new blynk 2.0 app with Arduino IDE and resolve Wi-Fi connectivity issues because it is essential to implement wireless technology for our project operation. The Arduino IDE couldn't execute the program until we installed the node MCU board, blynk, DHT sensor libraries. While experimenting, when we turned on the gas lighter near the DHT sensor, it bore 6-8 seconds on average to measure and display the new temperature in the widget, but the fan turned on immediately once the threshold temperature has been reached. We commenced by first creating a QuickStart template on the blynk website, bought a router, and connected that with the HCT Switch Board to link NodeMCU to a Wi-Fi network, but the outcome was futile, so we had figure out an alternative.

6.2 Graphical representation of results

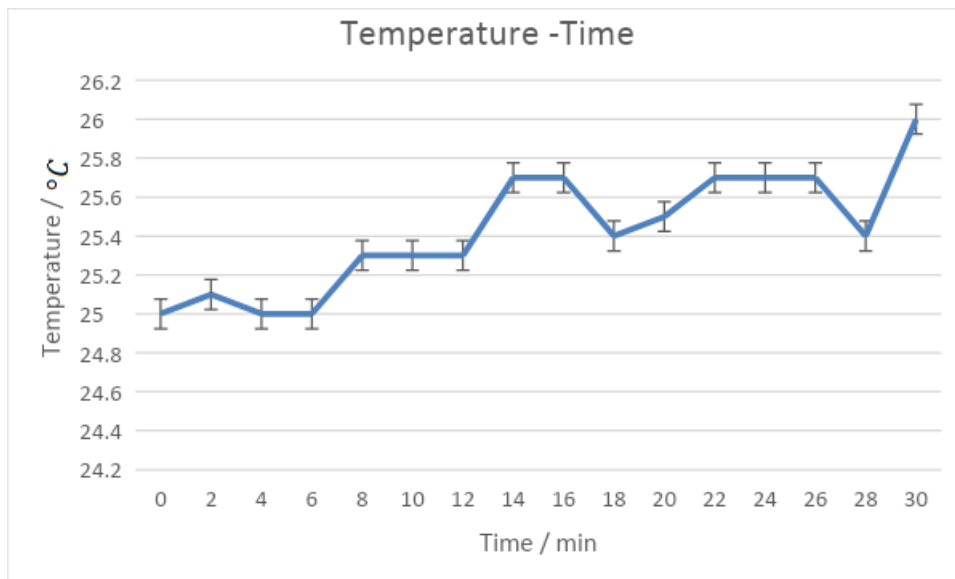


Figure 6.2.1 Temperature -Time Graph

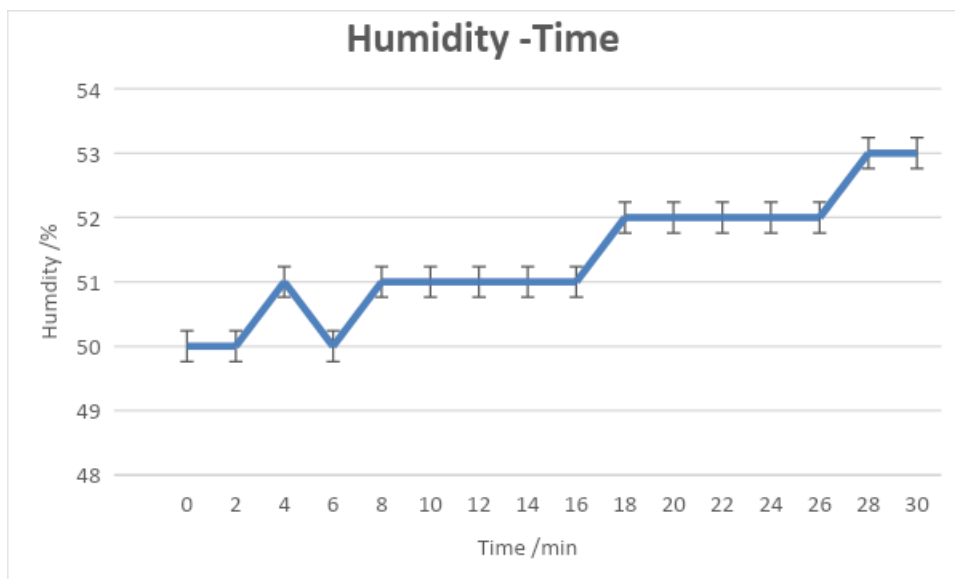


Figure 6.2.2 Humidity-Time Graph

6.3 Analysis of the results and discussion of findings

The height of error bars on the temperature and humidity graph in figures 6.2.1 and 6.2.2 are short which means values are not very spread out from the mean so our results for temperature and humidity measurements are reliable and hence the DHT sensor. The fluctuations are a result of random errors which are impossible to eliminate but their effect can reduce to a good extent if measurements are repeated and that's what we did. The resolution of DHT for temperature and humidity is 1°C and 1% respectively. The range for temperature and humidity is 0 to 50 °C and 20% to 80% respectively, so a suitable range was selected for their respective DataStreams. Resolution is the smallest scale division of an instrument. So, 1°C resolution means DHT sensor can measure temperature correct to 1°C, precisely. The datasheet of DHT specifies its response time to be between 6 to 30 seconds which confirms our experimental results.

We opted for WI-FI over other wireless technologies like Bluetooth, UWB (Ultra-Wideband), Zigbee as it has many advantages. For instance, the Bandwidth of WI-FI can up to be 150Mbps, it provides more secure communication than Bluetooth and has a greater range (100m) than all other wireless technologies. We used a mobile hotspot as a Wi-Fi network source instead, as college administration policies have blocked all these activities. For this reason, we couldn't test the Wi-Fi coverage area for node MCU. But the blynk app worked with different Wi-Fi networks so we could control our components from various locations.

Chapter VII

Conclusion and Recommendation

7.1 Conclusion

By the end of the successful completion of this work, we conclude that the IoT is a new design of wireless communication devices. It is the development of present internet facilities to handle everything which exists in the world or exists in the future. The IoT also can be considered as a global network that offers communication between things to things, things to human, and human to human. Depending on the Application, these day's devices are fitted with various sensors. Sensors are communicating with each other using diverse topologies in IoT. Data travels locally or remotely from or to each sensor node. For building a smart home, combining a sensor network with the internet is an essential task. IoT is Integration of these sensors, actuators, smart objects, devices, and network.

This project can be implemented in kitchen to automate the exhaust fan, in bedroom to control brightness of LED which is more convenient than traditional filament bulbs as they are more energy efficient. The system was experimented several times and we are self-assured and optimistic that the proposed system can be used to control different home appliances employed in the lightning system, air conditioning system, Refrigerator, Home entertainment systems etc. Therefore, this system is expandable, reliable, and flexible.

7.2 Recommendation

- Local blynk server instead of blynk cloud can be installed if privacy protection is insufficient to the user.
- DHT22 instead of DHT11 can be used if users want to utilize the system at greater range of temperature and measure temperature more accurately.
- An additional button widget that serves as a switch, can be added and programmed on Arduino IDE if user wants to turn ON/OFF the fan irrespective of the threshold temperature.