

## Introduction to Arduino Platform and Programming

**Aim :**

To Perform Interfacing of LED with ARDUINO and evaluate the response of variations.

**APPARATUS REQUIRED :**

1. Arduino Kit
2. USB Cable
3. Arduino SDK Software tool
4. Patch cards

**THEORY:**

LEDs are the most efficient way to turn an electric current into illumination. When a current flows through a diode in the forward direction, it consists of surplus electrons moving in one direction in the lattice and “holes” (voids in the lattice) moving in the other. Occasionally, electrons can recombine with holes. When they do, the process releases energy in the form of photons.

This is true of all semiconductor junctions, but LEDs use materials that maximize the effect. The color of the light emitted (corresponding to the energy of the photon) is determined by the semiconductor materials that form the diode junction.

The latest high-brightness (HB) white LEDs are made possible by the discovery of semiconductor materials that produce blue or ultraviolet photons. In addition to the diode, an HB package contains “yellow” phosphors on the inside of its lens. Some “blue” photons escape, but others excite the phosphors, which then give off “yellow” photons. The result can be tuned in manufacturing to produce “white” light.



Figure. LED symbol

A great deal of LED engineering relates to controlling the quality of this light. From a circuit standpoint, there are a number of ways to interconnect multiple LEDs to increase and manage light output. The general approach is to drive series strings with a constant current.

**PROCEDURE :**

**Step-1:** There are 4 LEDs in the kit namely D4, D5, D6 and D7. They are to be connected to the D4, D5, D6 and D7 pins of Arduino board. The power supply +5V and Gnd pins of Arduino board also are to be connected in this section.

**Step-2:** Connect the USB connector to the USB of Arduino board and the computer system.

**Step-3:** Using this program, the first LED (left most LED) is switched on for 0.5 sec and then it is

switched off.

**Step-4:** After 0.5 sec., the second LED is switched on for 0.5 sec. and then it is switched off. In this way, all the four LEDs are switched on and off. Then the cycle repeats continuously.

**Step-5:** In computer, open the sketch software and write the program **LED blinking** and execute the program in sketch and check for the proper result.

### Program :

```
#include <Arduino.h>

/*RGB LED pins (Common Cathode)*/
#define Red_Pin 9
#define Green_Pin 10
#define Blue_Pin 11
/*Delay (in milliseconds)*/
#define COLOR_CHANGE_DELAY 1000

void setup() {
  /*Initialize the RGB LED pins as outputs*/
  pinMode(Red_Pin, OUTPUT);
  pinMode(Green_Pin, OUTPUT);
  pinMode(Blue_Pin, OUTPUT);
}

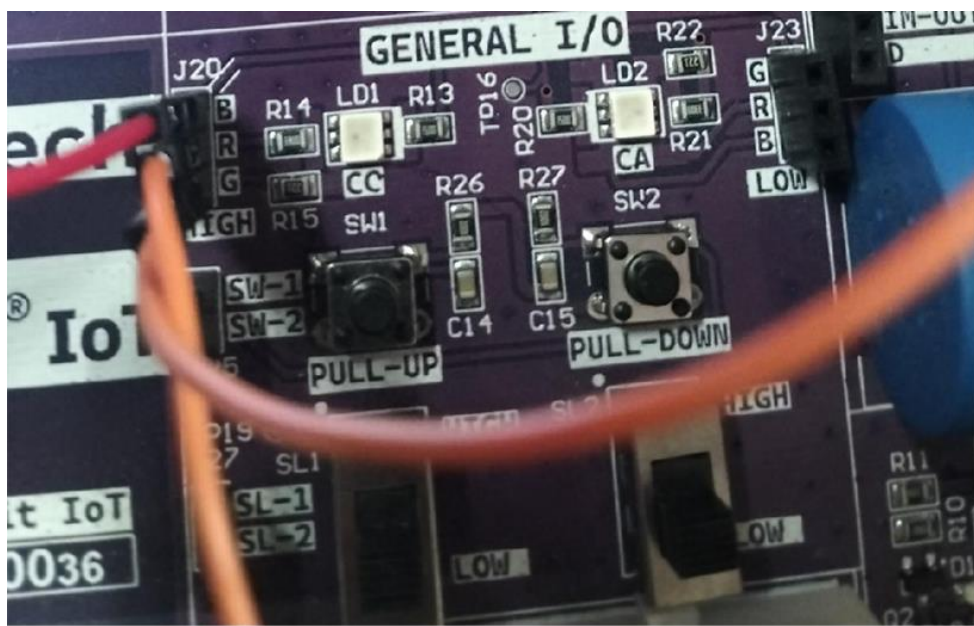
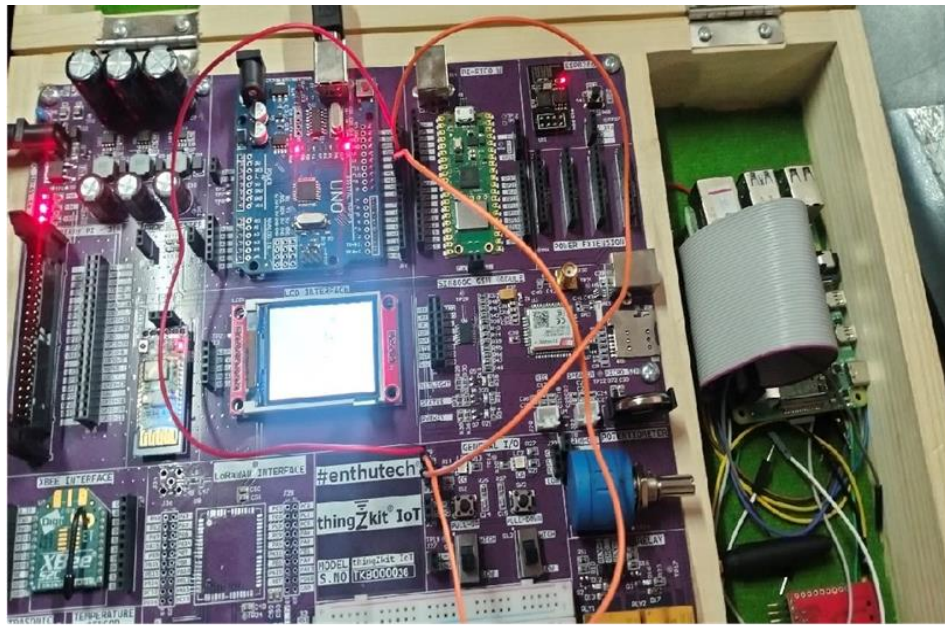
void loop() {
  /*Turn on Red, turn off Green and Blue*/
  digitalWrite(Red_Pin, HIGH);
  digitalWrite(Green_Pin, LOW);
  digitalWrite(Blue_Pin, LOW);
  delay(COLOR_CHANGE_DELAY);

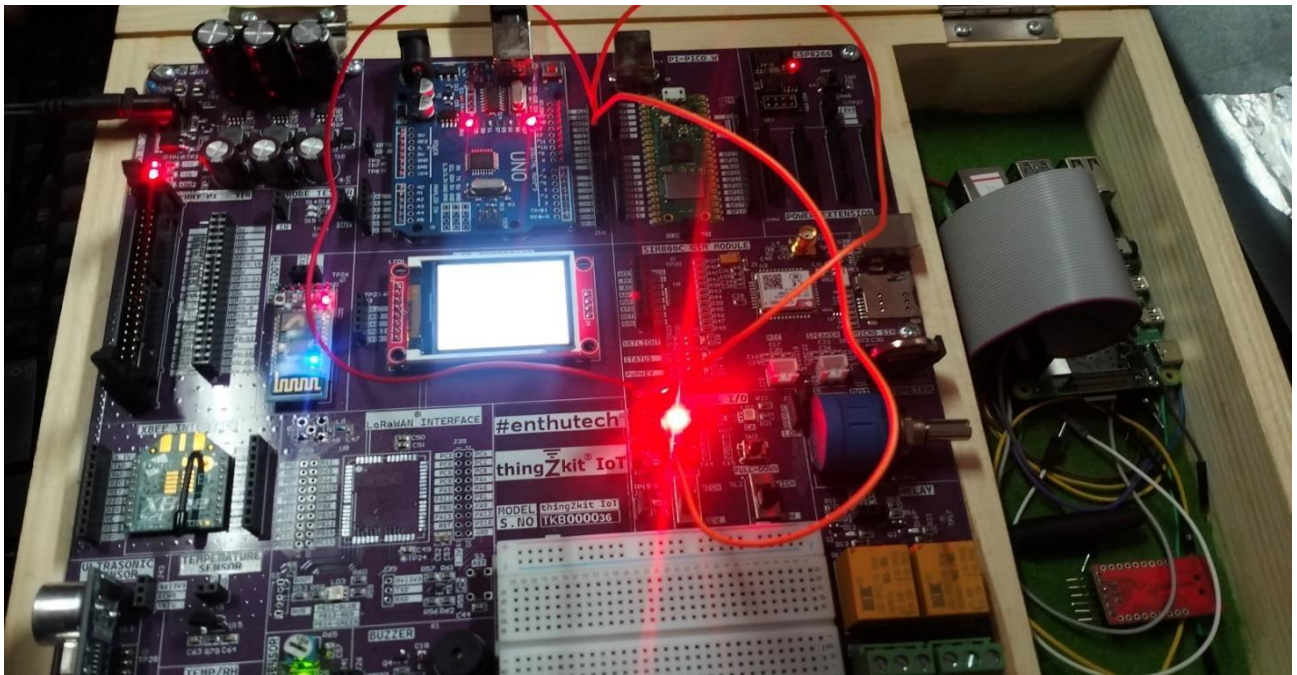
  /*Turn off Red, turn on Green, turn off Blue*/
  digitalWrite(Red_Pin, LOW);
  digitalWrite(Green_Pin, HIGH);
  digitalWrite(Blue_Pin, LOW);
  delay(COLOR_CHANGE_DELAY);

  /*Turn off Red, turn off Green, turn on Blue*/
  digitalWrite(Red_Pin, LOW);
  digitalWrite(Green_Pin, LOW);
  digitalWrite(Blue_Pin, HIGH);
  delay(COLOR_CHANGE_DELAY);
```

```
/*Turn ON Red,Green,Blue*/  
digitalWrite(Red_Pin, HIGH);  
digitalWrite(Green_Pin, HIGH);  
digitalWrite(Blue_Pin, HIGH);  
delay(COLOR_CHANGE_DELAY);  
}
```

Output :





## Result:

Thus the Interfacing of LED with ARDUINO was performed successfully.

Ex No: 6	Explore different communication methods with IoT devices ( Zigbee, GSM, Bluetooth )
Date:	

AIM:

To write a python program to explore different communication methods with IoT devices ( Zigbee, GSM, Bluetooth )

APPARATUS REQUIRED:

Arduino UNO kit, Arduino SDK Software

PROCEDURE:

1. Include SoftwareSerial library and define LED\_pin constant.
2. Initialize SoftwareSerial object 'bluetooth', float variable 'data', and character array 'charArray'.
3. Setup:
  - Begin serial communication at 9600 baud rate.
  - Begin Bluetooth communication at 9600 baud rate (connect Bluetooth module's RX pin to Arduino pin 2, TX pin to Arduino pin 3).
  - Set LED\_pin as an output (connect LED's anode to Arduino pin 13, cathode to ground).
4. Loop:

If Bluetooth data available:

  - Read received character.
  - Print received character.
  - If character is '1', turn on LED (connect LED\_pin to digital pin HIGH).
  - If character is '0', turn off LED (connect LED\_pin to digital pin LOW).
  - If character is '2':
  - Convert 'data' to charArray.
  - Send charArray over Bluetooth.
  - Delay for transmission.

## PROGRAM:

```
#include <SoftwareSerial.h>

#define LED_pin 13

SoftwareSerial bluetooth(2, 3); // RX, TX

float data=25.98;

char charArray[10];

void setup() {
    Serial.begin(9600); // Initialize the serial monitor
    bluetooth.begin(9600); // Initialize the Bluetooth communication
    pinMode(LED_pin,OUTPUT);
}

void loop() {
    // Check if data is available from Bluetooth module
    if (bluetooth.available()) {
        char receivedChar = bluetooth.read(); // Read the character received via Bluetooth
        Serial.print("Received: ");
        Serial.println(receivedChar);
        // Example: Send a response back to the Bluetooth module
        if (receivedChar == '1') {
            digitalWrite(LED_pin, HIGH);
            Serial.print("LED_ON");
        }
        else if(receivedChar == '0')
        {
            digitalWrite(LED_pin, LOW);
            Serial.print("LED_OFF");
        }
        else if(receivedChar == '2'){
            dtostrf(data, 6, 2, charArray); // float to char conversion
            bluetooth.write(charArray);
            delay(100);
        }
    }
}
```



}

}

## Output:



## RESULT:

Thus the python program to explore different communication methods with IoT devices ( Zigbee, GSM, Bluetooth ) was written and executed successfully.



**Ex No:7**

**Date:**

## **Introduction to Raspberry PI Platform and Python Programming**

### **Aim:**

The purpose of this experiment is to study about Raspberry PI platform and Python programming.

### **Introduction to Raspberry PI:**

The Raspberry Pi is a small, affordable, single-board computer developed by the Raspberry Pi Foundation, with the primary goal of promoting computer science education and fostering DIY projects. It was first introduced in 2012 and has since gained immense popularity as a versatile and accessible computing platform.

### **Key Features of Raspberry Pi:**

#### **1. Affordability:**

Raspberry Pi boards are cost-effective, making them accessible to a wide range of users.

#### **2. Compact Size:**

Raspberry Pi is credit card-sized, portable, and suitable for embedded systems.

#### **3. Hardware Specifications:**

Different models offer varying hardware specs, typically including a CPU, RAM, USB ports, HDMI output, audio output, Ethernet (on some models), and GPIO pins.

#### **4. Operating System:**

Raspberry Pi runs on a Linux-based OS, often Raspbian (now known as Raspberry Pi OS), but supports other OS options.

#### **5. GPIO Pins:**

General Purpose Input/Output (GPIO) pins allow interaction with the physical world, making it ideal for IoT and electronics projects.

#### **6. Community and Support:**

Raspberry Pi has a large and active user community, providing online resources, tutorials, forums, and project ideas.

### **Common Raspberry Pi Models:**

#### **1. Raspberry Pi 4:**

Features a quad-core ARM Cortex-A72 CPU, up to 8GB of RAM, dual HDMI outputs, USB 3.0 ports, and Gigabit Ethernet (as of my last update in January 2022).

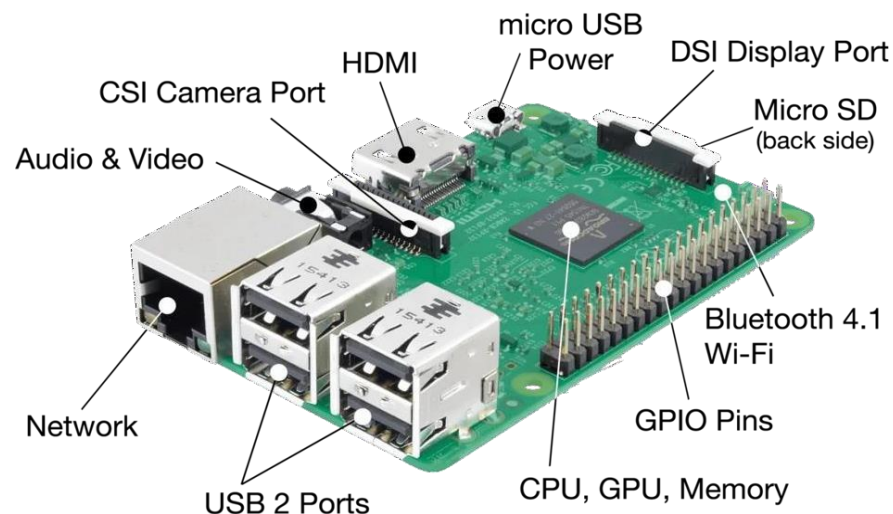
## 2. Raspberry Pi 3:

Equipped with a quad-core ARM Cortex-A53 CPU, 1GB of RAM, Wi-Fi, and Bluetooth.

## 3. Raspberry Pi Zero:

An ultra-compact, low-cost model with a single-core CPU, 512MB of RAM, and a mini HDMI port.

## Parts in Raspberry Pi:



## Raspberry Pi Hardware Specifications:

The raspberry pi board comprises a program memory (RAM), processor and graphics chip, CPU, GPU, Ethernet port, GPIO pins, Xbee socket, UART, power source connector. And various interfaces for other external devices. It also requires mass storage, for that we use an SD flash memory card. So that raspberry pi board will boot from this SD card similarly as a PC boots up into windows from its hard disk.

Essential hardware specifications of raspberry pi board mainly include SD card containing Linux OS, US keyboard, monitor, power supply and video cable. Optional hardware specifications include USB mouse, powered USB hub, case, internet connection, the Model A or B: USB WiFi adaptor is used and internet connection to Model B is LAN cable.

## Memory

The raspberry pi model A board is designed with 256MB of SDRAM and model B is designed with 512MB. Raspberry pi is a small size PC compare with other PCs. The normal PCs RAM memory is available in gigabytes. But in raspberry pi board, the RAM memory is available more than 256MB or 512MB

## CPU (Central Processing Unit):

The Central processing unit is the brain of the raspberry pi board and that is responsible for carrying out the instructions of the computer through logical and mathematical operations. The raspberry pi uses ARM11 series processor, which has joined the ranks of the Samsung galaxy phone.

## GPU (Graphics Processing Unit):

The GPU is a specialized chip in the raspberry pi board and that is designed to speed up the operation of image calculations. This board designed with a Broadcom video core IV and it supports OpenGL

## Ethernet Port:

The Ethernet port of the raspberry pi is the main gateway for communicating with additional devices. The raspberry pi Ethernet port is used to plug your home router to access the internet.

## GPIO Pins:

The general purpose input & output pins are used in the raspberry pi to associate with the other electronic boards. These pins can accept input & output commands based on programming raspberry pi. The raspberry pi affords digital GPIO pins. These pins are used to connect other electronic components. For example, you can connect it to the temperature sensor to transmit digital data.

## XBee Socket:

The XBee socket is used in raspberry pi board for the wireless communication purpose.

## Power Source Connector:

The power source cable is a small switch, which is placed on side of the shield. The main purpose of the power source connector is to enable an external power source.

## UART:

The Universal Asynchronous Receiver/ Transmitter is a serial input & output port. That can be used to transfer the serial data in the form of text and it is useful for converting the debugging code.

## Display

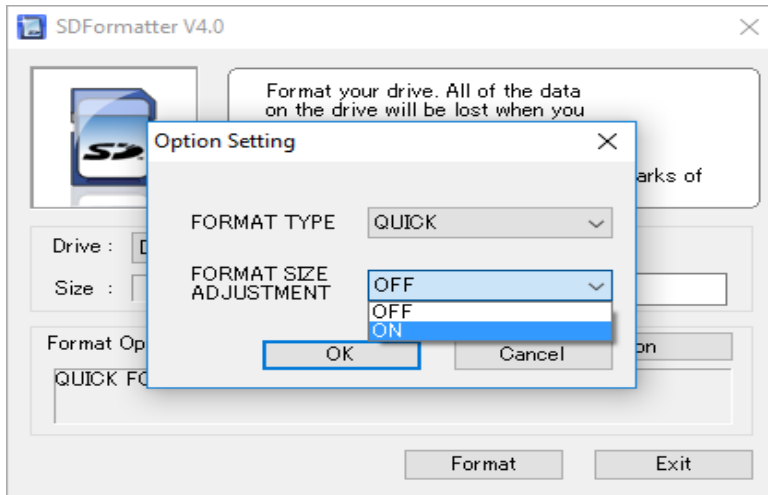
The connection options of the raspberry pi board are two types such as HDMI and Composite. Many LCD and HD TV monitors can be attached using an HDMI male cable and with a low-cost adaptor. The versions of HDMI are 1.3 and 1.4 are supported and 1.4 version cable is recommended. The O/Ps of the Raspberry Pi audio and video through HDMI, but does not support HDMI I/p. Older TVs can be connected using composite video. When using a composite video connection, audio is available from the 3.5mm jack socket and can be sent to your TV. To send audio to your TV, you need a cable which adjusts from 3.5mm to double RCA connectors.

## Steps to install:

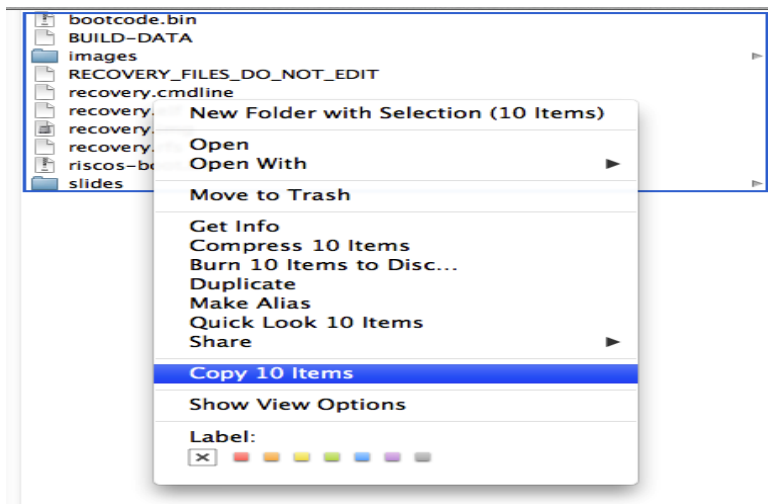
**Step 1:** Download NOOBS and extract it.



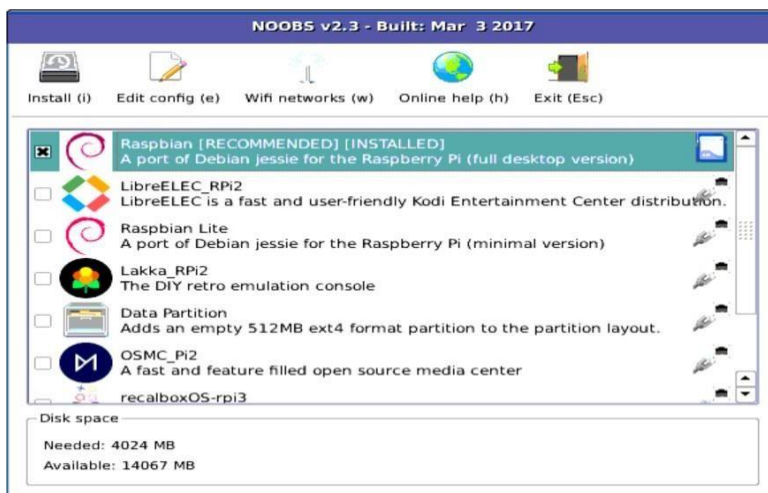
## Step 2: Format an SD card.



## Step 3: Put the NOOBS files on the SD card.



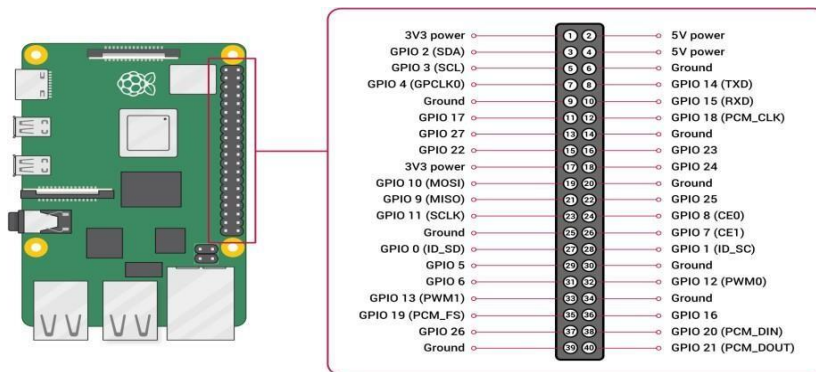
## Step 4: Put your SD card into your Raspberry Pi and boot it up.



Boot up Rainbow screen.



## Raspberry pi GPIO PINOUT



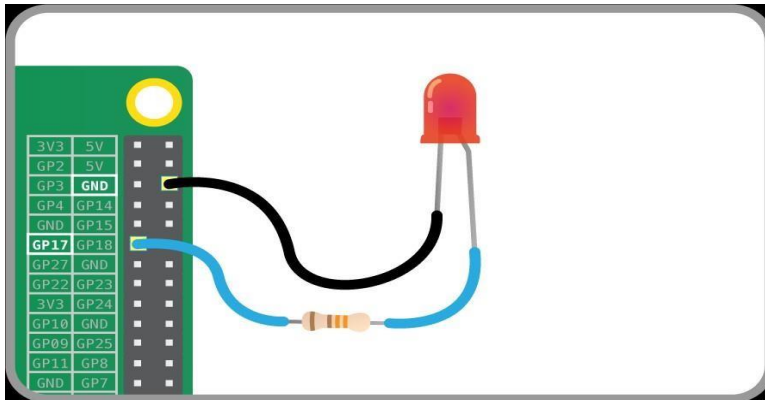
## Python Programing in Raspberry Pi:

Python programming on the Raspberry Pi is a powerful combination that has revolutionized the world of embedded computing and DIY electronics. The Raspberry Pi, a credit-card-sized single-board computer, is particularly well-suited for Python due to its simplicity, versatility, and a vast array of libraries and frameworks.

Python's clear syntax and readability make it an ideal language for beginners, allowing enthusiasts to quickly grasp the basics of programming while experimenting with hardware interfaces. The Raspberry Pi's GPIO (General Purpose Input/Output) pins enable users to interact with the physical world by connecting sensors, actuators, and other devices, all of which can be easily controlled through Python scripts. This seamless integration has sparked a surge in innovative projects ranging from home automation and robotics to IoT applications, fostering a vibrant community of developers and makers exploring the limitless possibilities of Python programming on the Raspberry Pi platform.

## LED Blinking Using RaspberryPI:

Blinking is done by connecting an LED to one of GPIO pins of PI and turning it ON and OFF.



## PROGRAM:

```
import machine as Gpio
import utime as TM

# Define the RGB LED pins (common cathode)
Red_Pin = Gpio.Pin(15, Gpio.Pin.OUT)
Green_Pin = Gpio.Pin(14, Gpio.Pin.OUT)
Blue_Pin = Gpio.Pin(13, Gpio.Pin.OUT)

# Helper function to set RGB color
def set_rgb_color(red, green, blue):
    Red_Pin.value(red)
    Green_Pin.value(green)
    Blue_Pin.value(blue)

# Main loop
while True:
    # Red
    set_rgb_color(1, 0, 0)
    TM.sleep(1)

    # Green
    set_rgb_color(0, 1, 0)
    TM.sleep(1)

    # Blue
    set_rgb_color(0, 0, 1)
    TM.sleep(1)
```



**Result:**

Thus the Raspberry Pi platform and python programing was studied successfully.

<b>Ex No: 8</b>	<b>Interfacing sensors with Raspberry PI</b>
<b>Date:</b>	

**AIM:**

To write a python program to interface sensors with Raspberry PI.

**APPARATUS REQUIRED:**

Raspberry PI kit, IR sensors, connecting wires, python IDE.

**PROCEDURE:**

1. Setup and connect the Raspberry PI with the monitor and I/O devices.
2. Now connect the Ultrasonic sensor with Raspberry PI.
3. Type the python program to measure the distance with ultrasonic sensor.
4. Now save and run the program.

**PROGRAM:**

```
import requests

import json

import time

import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)

GPIO.setwarnings(False)

GPIO_TRIGGER = 24 #RPi 18
GPIO_ECHO  = 25 #RPi 22
print ("Ultrasonic Measurement")

# Set pins as output and input
GPIO.setup(GPIO_TRIGGER,GPIO.OUT) # Trigger
GPIO.setup(GPIO_ECHO,GPIO.IN)    # Echo

# Set trigger to False (Low)
GPIO.output(GPIO_TRIGGER, False)

# Allow module to settle
time.sleep(0.5)

while True:

    try:

        # Send 10us pulse to trigger
        GPIO.output(GPIO_TRIGGER, True)

        time.sleep(0.00001)

        GPIO.output(GPIO_TRIGGER, False)

        while GPIO.input(GPIO_ECHO)= =0:

            start = time.time()

            while GPIO.input(GPIO_ECHO)= =1:

                stop = time.time()

            # Calculate pulse length
```

```
elapsed = stop-start
# Distance pulse travelled in that time is time
# multiplied by the speed of sound (cm/s)
distance = elapsed * 34300
# That was the distance there and back so halve the value
distance = distance / 2 #in centimeter
inch=distance/2.5 #in Inch
print ("Distance (cm): %.1f" % distance)
print ("Distance (inch): %.1f" % inch)
time.sleep(20)
```

**RESULT:**

Thus the python program to interface sensors with Raspberry PI was written and executed successfully.

**Ex No: 09**

**Communicate between Arduino and Raspberry PI  
using any wireless medium.**

**Date:**

**AIM:**

To Perform and communicate between Arduino and Raspberry Pi using any Wireless medium and evaluate the response.

**APPARATUS REQUIRED:**

- Arduino IOT Kit
- USB Cable
- Arduino SDK Software tool
- VGA to mini HDMI Converter cable
- Power Adapter
- Short USB power cable
- Patch cards

**PROCEDURE:**

**Arduino:**

- ❖ Write the arduino program and upload it
- ❖ As per connection bluetooth to arduino TX-RX , RX-TX

**Raspberry:**

edit the DOS Prompt

sudo bluetoothctl                      ## Bluetooth path

agent on                                  ## agent registered

scan on                                  ## our bluetooth will display copy the mac address

sudo rfcomm --help                      ## rfcomm command will display

sudo rfcomm bind 7 --mac address-----      ## enter command paste your mac ad

Connect the bluetooth on raspberry pi ,right corner click bluetooth icon and pair

- ❖ Run the program both side arduino & raspberry
- ❖ Enter data 1 led will go on state
- ❖ enter data 0 led will go off state
- ❖ finally disconnect the raspberry bluetooth -sudo rfcomm release 7 ----mac address---

**Arduino Program:**

```
int led=13;

void setup()
{
  pinMode(led, OUTPUT);
  Serial.begin(9600); //default baud rate for bt 38400
```

```

}

void loop()
{
  if(Serial.available())
  {
    int a=Serial.parseInt();
    Serial.println(a);
    if (a==1)
    {
      digitalWrite(led, HIGH);

    }
    if (a == 0)
    {
      digitalWrite(led, LOW);

    }
  }
}

```

### **Raspberry Pi Program:**

```

import serial
import time
bluetooth=serial.Serial("/dev/rfcomm7",9600)
while True:
    a=input("enter:-")
    string='X{0}'.format(a)
    bluetooth.write(string.encode("utf-8"))

```



**RESULT:**

Thus the communication between Arduino and Raspberry Pi using any Wireless medium was done successfully.

**Ex No: 10**

## Setup a cloud platform to log the data.

**Date:**

### AIM:

To setup a cloud platform to log the data.

### CLOUD PLATFORM:

ThingSpeak

### ThingSpeak:

ThingSpeak is an IoT analytics platform service that allows you to aggregate, visualize, and analyze live data streams in the cloud. You can send data to ThingSpeak from your devices, create instant visualization of live data, and send alerts.

Here we are going to setup ThingSpeak and log the data of Ultrasonic Sensor.

### PROCEDURE:

#### 1. Create a ThingSpeak Account:

If you don't have an account, sign up on the ThingSpeak website.

thingspeak.com/login?skipSSOCheck=true

ThingSpeak™ Channels Apps Support Commercial Use How to Buy

To use ThingSpeak, you must sign in with your existing MathWorks account or create a new one.

Non-commercial users may use ThingSpeak for free. Free accounts offer limits on certain functionality. Commercial users are eligible for a time-limited free evaluation. To get full access to the MATLAB analysis features on ThingSpeak, log in to ThingSpeak using the email address associated with your university or organization.

To send data faster to ThingSpeak or to send more data from more devices, consider the [paid license options](#) for commercial, academic, home and student usage.

Create MathWorks Account

Email Address

**i** To access your organization's MATLAB license, use your school or work email.

Location

United States

First Name

Last Name

Continue

Cancel

DATA AGGREGATION AND ANALYTICS ThingSpeak

SMART CONNECTED DEVICES

MATLAB

ALGORITHM DEVELOPMENT SENSOR ANALYTICS

This website uses cookies to improve your user experience, personalize content and ads, and analyze website traffic. By continuing to use this website, you consent to our use of cookies. Please see our [Privacy Policy](#) to learn more about cookies and how to change your settings.

## 2. Log in to ThingSpeak:

Once you're logged in, go to your ThingSpeak channel.

The screenshot shows the 'My Channels' page on the ThingSpeak website. The page has a blue header with the ThingSpeak logo and navigation links: Channels, Apps, Devices, and Support. On the right side of the header are links for Commercial Use, How to Buy, and a user profile icon. The main content area is titled 'My Channels' and features a 'New Channel' button, a search bar, and a table of existing channels. The table has columns for Name, Created, and Updated. Two channels are listed: 'Smart Home' and 'Ultrasonic'. Below the table are links for Private, Public, Settings, Sharing, API Keys, and Data Import / Export. To the right of the table is a 'Help' section with instructions on how to collect data, create a new channel, and use the column headers. Below the help section are 'Examples' (Arduino, Arduino MKR1000, ESP8266, Raspberry Pi, Netduino Plus) and an 'Upgrade' section. At the bottom of the page is a cookie consent banner.

Name	Created	Updated
Smart Home	2023-04-21	2023-04-21 14:53
Ultrasonic	2023-10-31	2023-10-31 04:37

## 3. Create a Channel:

- Click on "Channels" and then "New Channel."
- Fill in the required information, like Name, Field labels (e.g., Distance), and any other relevant details.

The screenshot shows the 'New Channel' page on the ThingSpeak website. The page has a blue header with the ThingSpeak logo and navigation links: Channels, Apps, Devices, and Support. On the right side of the header are links for Commercial Use, How to Buy, and a user profile icon. The main content area is titled 'New Channel' and features a form for creating a new channel. The form has fields for Name, Description, and Metadata. Below these fields are eight 'Field' labels (Field 1 to Field 8) with checkboxes. Field 1 is checked. To the right of the form is a 'Help' section with instructions on how to create a channel, including a list of 'Channel Settings' (Percentage complete, Channel Name, Description, Field#, Metadata, Tags, Link to External Site, Show Channel Location). At the bottom of the page is a cookie consent banner.

#### 4. Get Your Write API Key:

- Go to the "API Keys" tab to find your Write API Key.
- This key is essential for updating data on your channel.

The screenshot shows the ThingSpeak website interface for a specific channel. The browser address bar shows 'thingspeak.com/channels/2325265/api\_keys'. The page header includes the ThingSpeak logo and navigation links for Channels, Apps, Devices, and Support. The channel name 'Ultrasonic' is displayed, along with its ID (2325265), author (mwa000029640674), and access status (Private). A tabbed interface at the top allows switching between Private View, Public View, Channel Settings, Sharing, API Keys (the active tab), and Data Import / Export. The 'Write API Key' section shows a key '5R6R0X5SN6BR7SXK' and a 'Generate New Write API Key' button. The 'Read API Keys' section shows a key 'ABF4X6FWN9AJKL5N' and a 'Generate New Read API Key' button. A 'Help' section explains the purpose of API keys and provides instructions on how to use them. A 'Note' section provides additional information about channel read keys. A 'Write a Channel Feed' link is also visible. A footer notice states that the website uses cookies to improve user experience and that users consent to their use by continuing to use the site.

#### 5. Configure Your Ultrasonic Sensor:

- Set up your ultrasonic sensor to measure distance.
- Connect your sensor to a microcontroller or single-board computer (like Arduino or Raspberry Pi).

#### 6. Program Your Device:

- Use the ThingSpeak API to send data to your channel.
- Include your Write API Key in the HTTP POST request.

#### 7. Test Your Setup:

- Upload your code to your device.
- Check your ThingSpeak channel to ensure data is being logged correctly.

**RESULT:**

Thus a cloud platform to log the data was setup successfully.

<b>Ex No: 11</b>	<b>Log Data using Raspberry PI and upload to the cloud platform.</b>
<b>Date:</b>	

**AIM:**

To write a python program to log data using Raspberry PI and upload to the cloud platform.

**APPARATUS REQUIRED:**

Raspberry PI kit, Ultrasonic sensors, connecting wires, python IDE, ThingSpeak.

**PROCEDURE:**

1. Setup and connect the Raspberry PI with the monitor and I/O devices.
2. Connect the Ultrasonic Sensor with Raspberry PI.
3. Setup the cloud platform ( ThingSpeak ) to upload the data.
4. Type the python program to upload the data from Ultrasonic Sensor to ThingSpeak.
5. Copy and paste your ThingSpeak channel write API key in the program.
6. Now save and run the program.



## PROGRAM:

```
import machine
import utime
import urequests
import network

IR_SENSOR_PIN = 2 # GPIO pin for the IR sensor
API_KEY = "BYRGZSN129A5E7P9"
THINGSPEAK_URL = "https://api.thingspeak.com/update?api_key={}".format(API_KEY)
WIFI_SSID = "SHRUTI L G"
WIFI_PASSWORD = "12345678"

ir_sensor = machine.Pin(IR_SENSOR_PIN, machine.Pin.IN)

wlan = network.WLAN(network.STA_IF)

def connect_to_wifi():
    wlan.active(True)
    if not wlan.isconnected():
        print("Connecting to WiFi...")
        wlan.connect(WIFI_SSID, WIFI_PASSWORD)
        while not wlan.isconnected():
            pass
        print("Connected to WiFi")

def read_ir_sensor():
    return ir_sensor.value()

def send_to_thingspeak(data):
    # Send data to ThingSpeak
    url = "{}&field1={}".format(THINGSPEAK_URL, data)
    response = urequests.get(url)
    print("ThingSpeak response:", response.text)

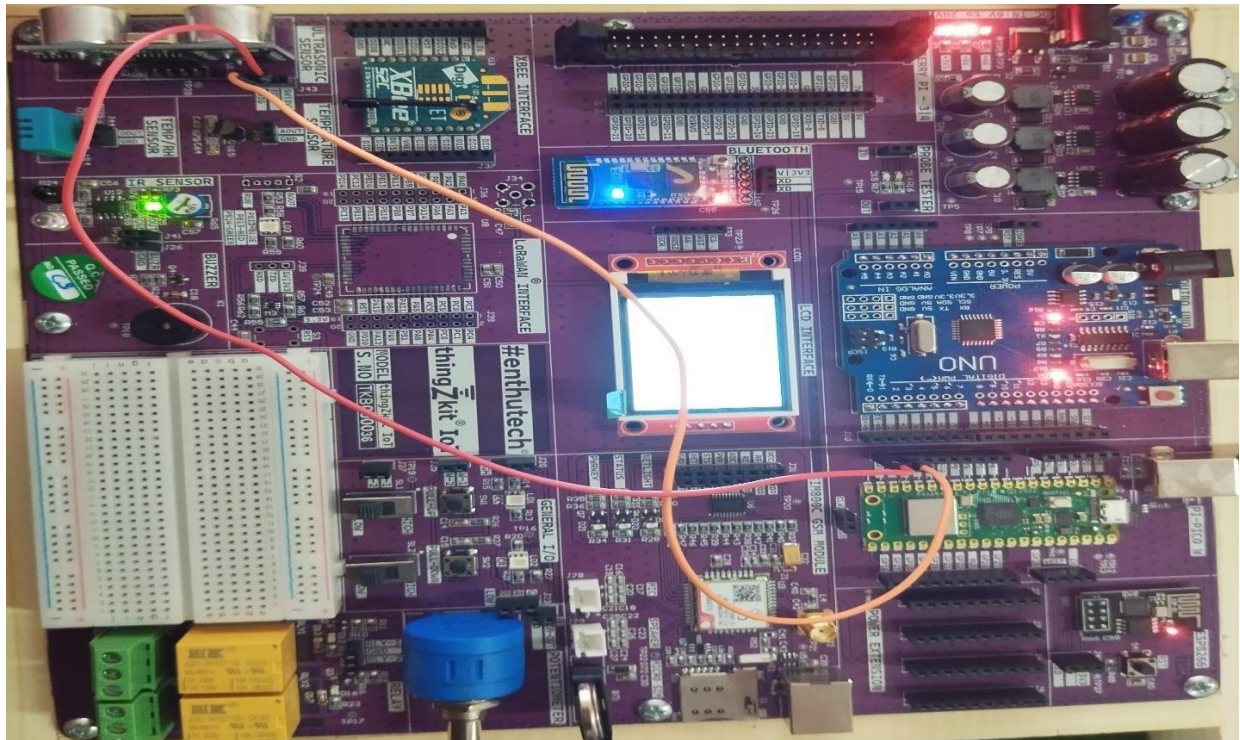
# Connect to WiFi
connect_to_wifi()

while True:
    ir_data = read_ir_sensor()
    print("IR Sensor Data:", ir_data)

    # Send data to ThingSpeak via HTTP
    send_to_thingspeak(ir_data)

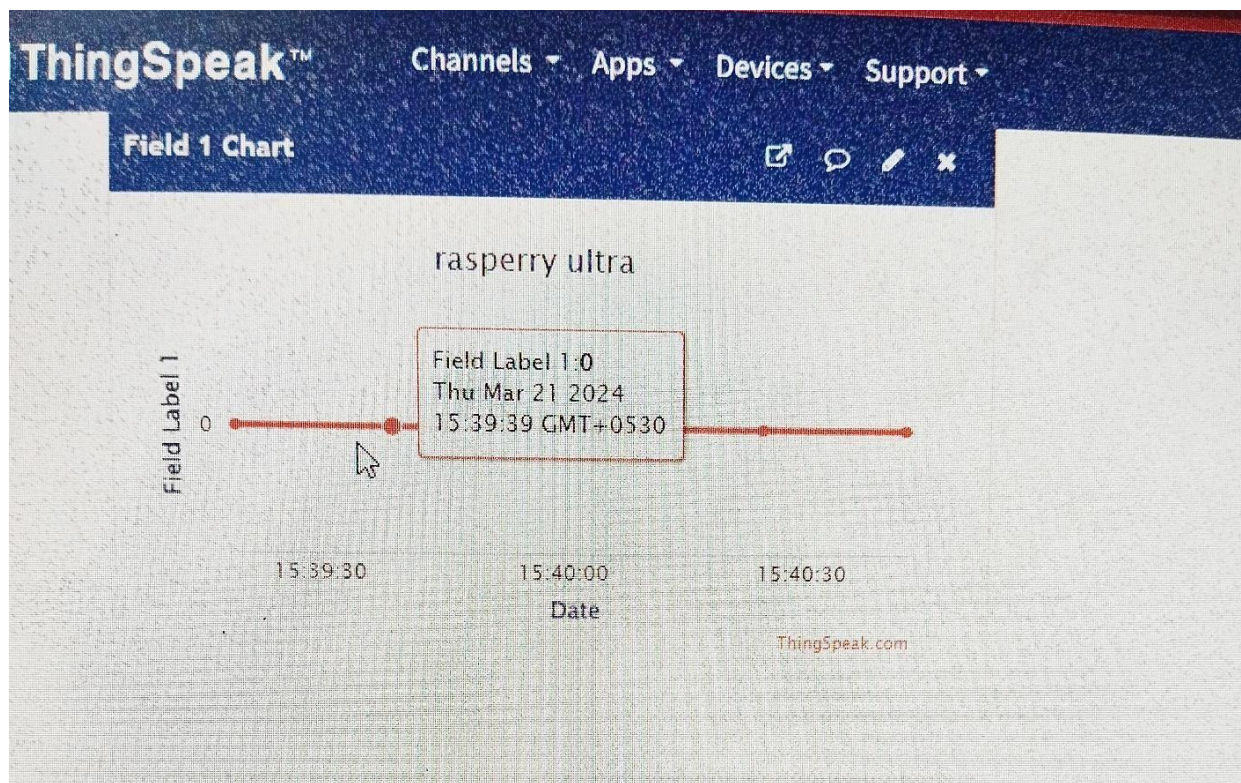
    utime.sleep(15) # Wait for 15 seconds before the next reading
```

OUTPUT:



```
Shell x
Distance: 20.33 cm
Distance: 20.34 cm
Distance: 20.69 cm
Distance: 15.34 cm
Distance: 20.43 cm
Distance: 15.93 cm
Distance: 19.95 cm
```





## RESULT:

Thus the python program to log data using Raspberry PI and upload to the cloud platform was written and executed successfully.

<b>Ex No: 12</b>	<b>Design an IOT based system.</b>
<b>Date:</b>	

**AIM:**

To design an IOT based System with Node MCU ESP8266 and BLYNK App and evaluate the response of variations.

**APPARATUS REQUIRED:**

1. Embedded IOT Kit
- 2.USB Cable
- 3.Software tool
- 4.Patch cards
- 5.Power Adapter

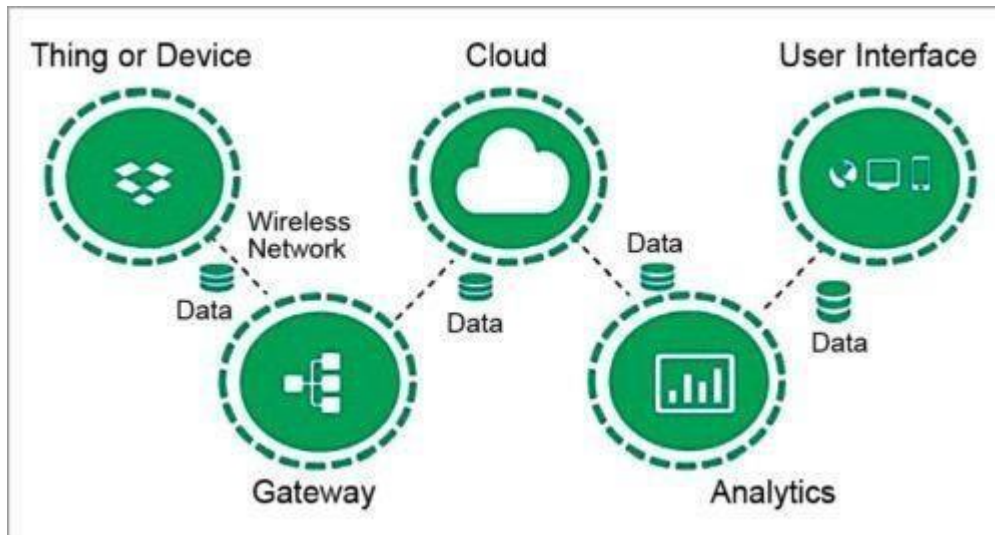
**THEORY:**

A quick guide to designing a perfect Internet of Things (IoT) system taking into account performance, connectivity, power consumption and security issues

The Internet of Things (IoT) is no longer a technology of the future. Smart cities, connected industries and smart households have indeed ushered in an era where machines can communicate. The beauty of this technology lies in the fact that the complex backend structure of systems is represented to the end-user in the simplest possible form. This requires profound design know-how.

The IoT can be designed at different scales for different uses. It can start from our homes with simple lighting or appliance control, and expand into the realm of factories and industries with automated machines, smart security systems and central management systems—called connected factories. It has scaled up to entire cities with smart parking, smart metering, waste management, fire control, traffic management and any similar functions involved. However, irrespective of the scale of application, the main IoT backbone remains similar.





The IoT architecture is multi-layered with delicate components intricately connected to each other. It starts with sensors, which are the source of data being collected. Sensors pass data onto an adjacent edge device, which converts data into readable digital values and stores these temporarily. When the edge senses a suitable wireless network or the Internet, it pushes the locally stored data to a cloud server involved in the application. The data is processed, analysed, stored and forwarded to the end-user device, represented by an application software. All the design fundamentals and challenges revolve around these layers.

### Designing the connectivity module:

The ESP8266 is a system on a chip (SOC) Wi-Fi microchip for Internet of Things (IoT) applications produced by Espressif Systems.

Given its low cost, small size and adaptability with embedded devices, the ESP8266 is now used extensively across IoT devices. Although it's now been succeeded by the newer generation [ESP32 microcontroller chip](#), the ESP8266 is still a popular choice for IoT developers and manufacturers.

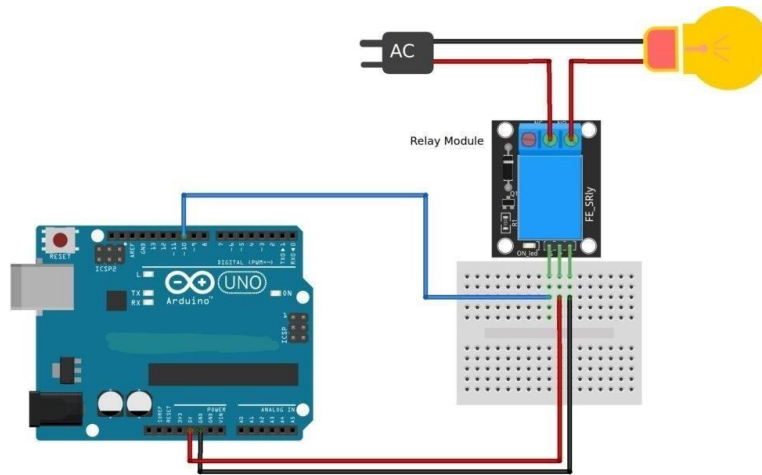
In this article, we'll explain the main features of ESP8266 modules and development boards and their application within the world of IoT.

The ESP8266 module enables microcontrollers to connect to 2.4 GHz Wi-Fi, using IEEE 802.11 bgn. It can be used with ESP-AT firmware to provide Wi-Fi connectivity to external host MCUs, or it can be used as a self-sufficient MCU by running an [RTOS](#)-based SDK. The module has a full TCP/IP stack and provides the ability for data processing, reads and controls of GPIOs.



## Espressif NodeMCU module V1.0:

This board has the ESP-12E module and comes with 4 Mbits of flash and features a row of pins on each side of the breadboard. The board comes with four communication interfaces: SPI, I2C, UART, and I2S, with 16 GPIO and one ADC. The RAM is 160KB, divided into 64KB for instruction and 96KB for data.



## PROGRAM:

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

// Your Wi-Fi credentials
char auth[ ] = " ";
char ssid[ ] = " ";
char pass[ ] = " ";

void setup( )
{
  // Initialize Serial Monitor
  Serial.begin(9600);

  // Connect to Wi-Fi
  Blynk.begin(auth, ssid, pass);

  // Set GPIO pins for relays as OUTPUT

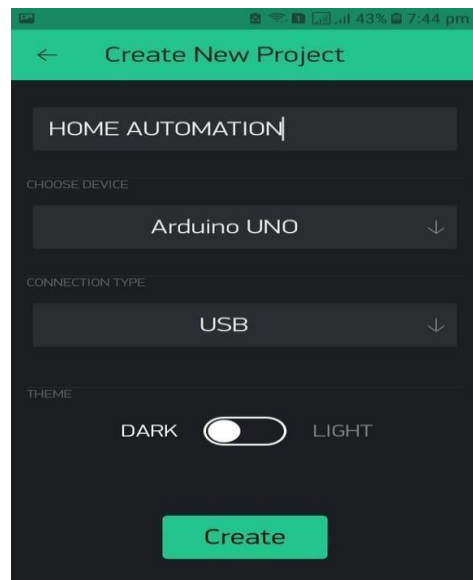
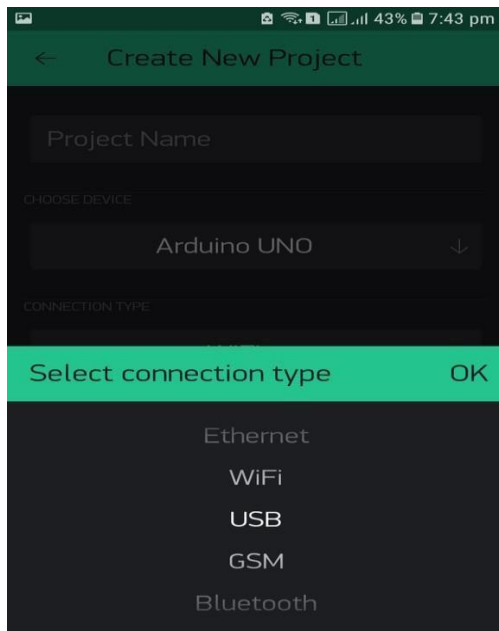
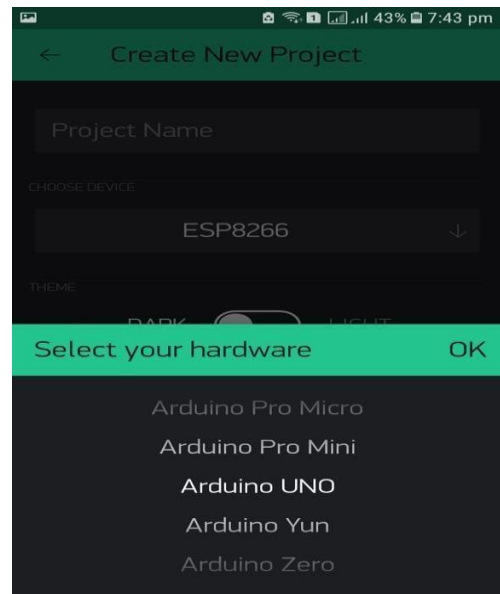
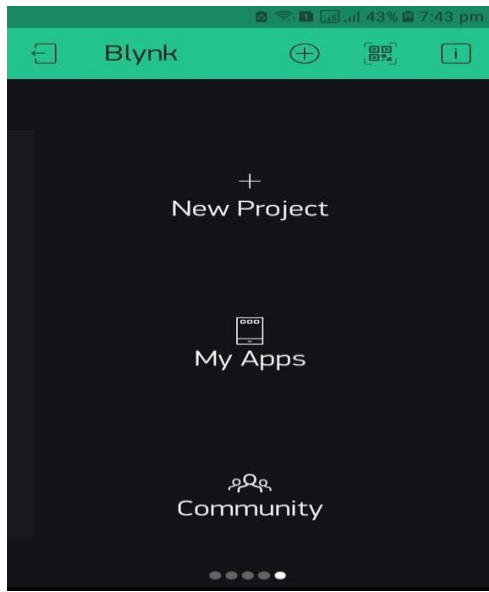
  pinMode(D1, OUTPUT); // Relay 1
}

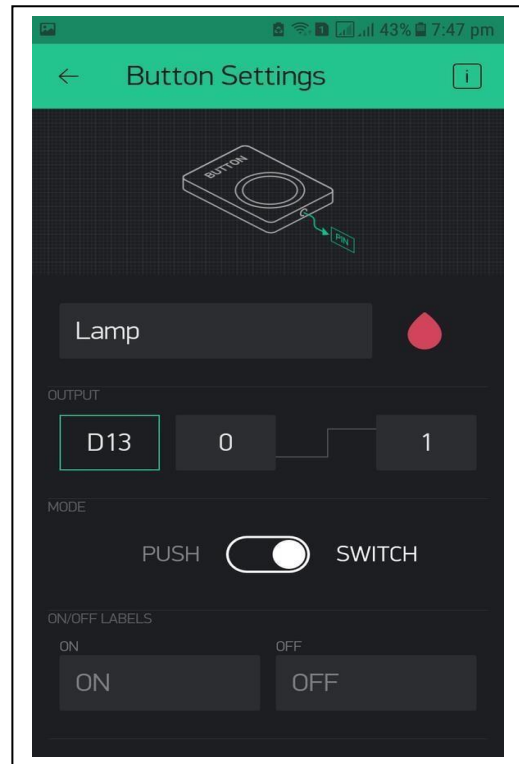
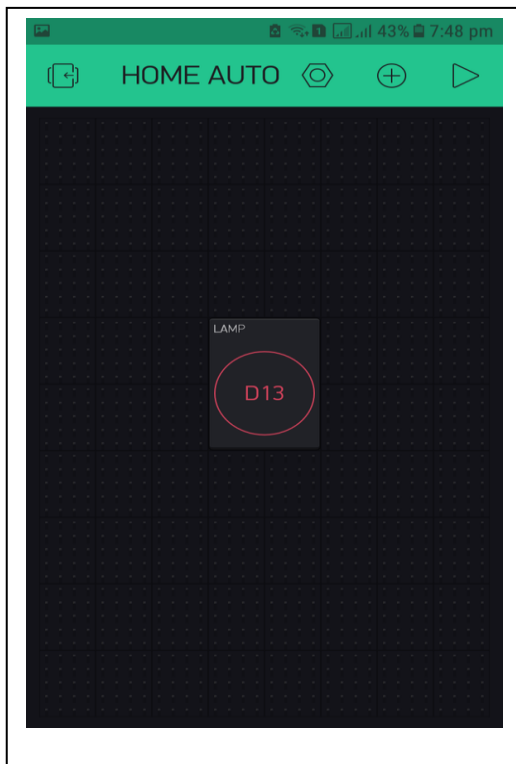
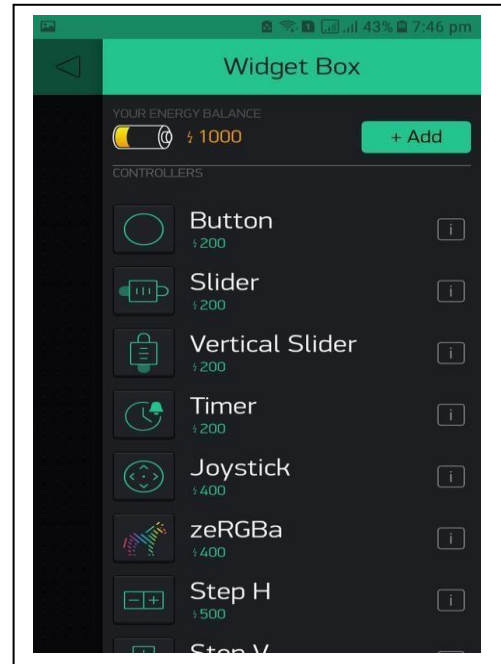
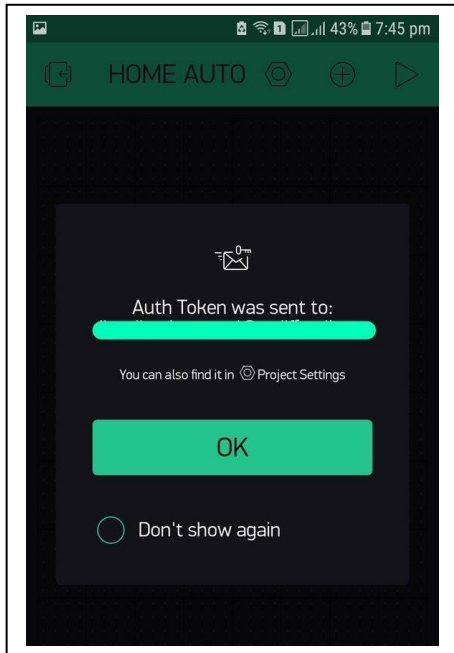
void loop( )
{
  Blynk.run( );
}
```

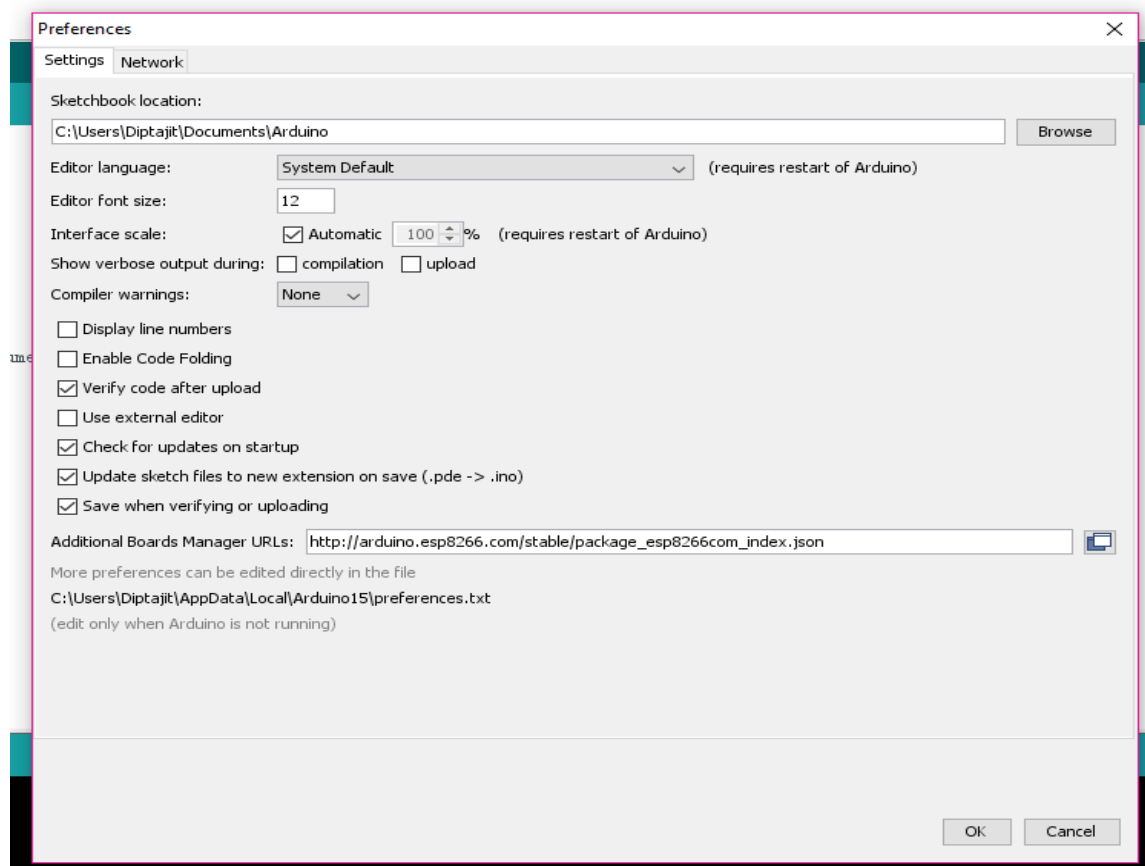
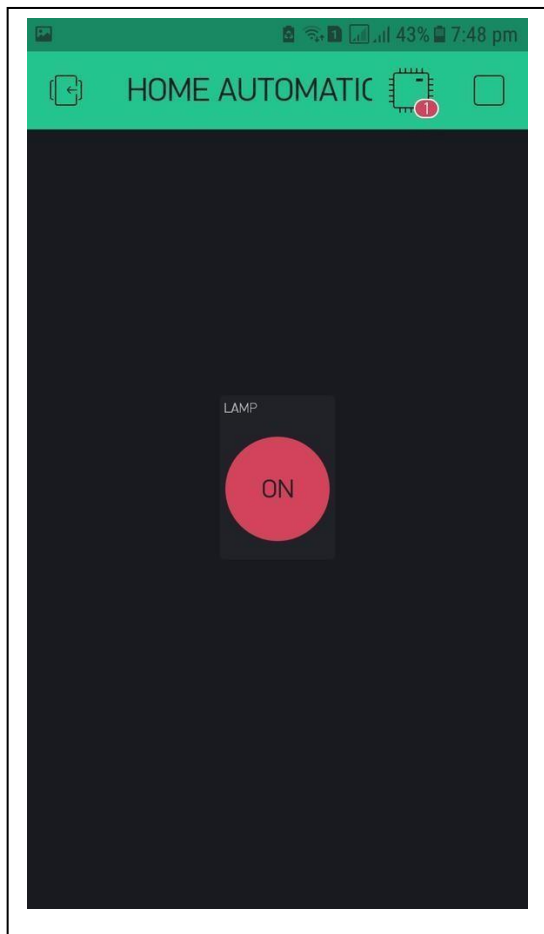


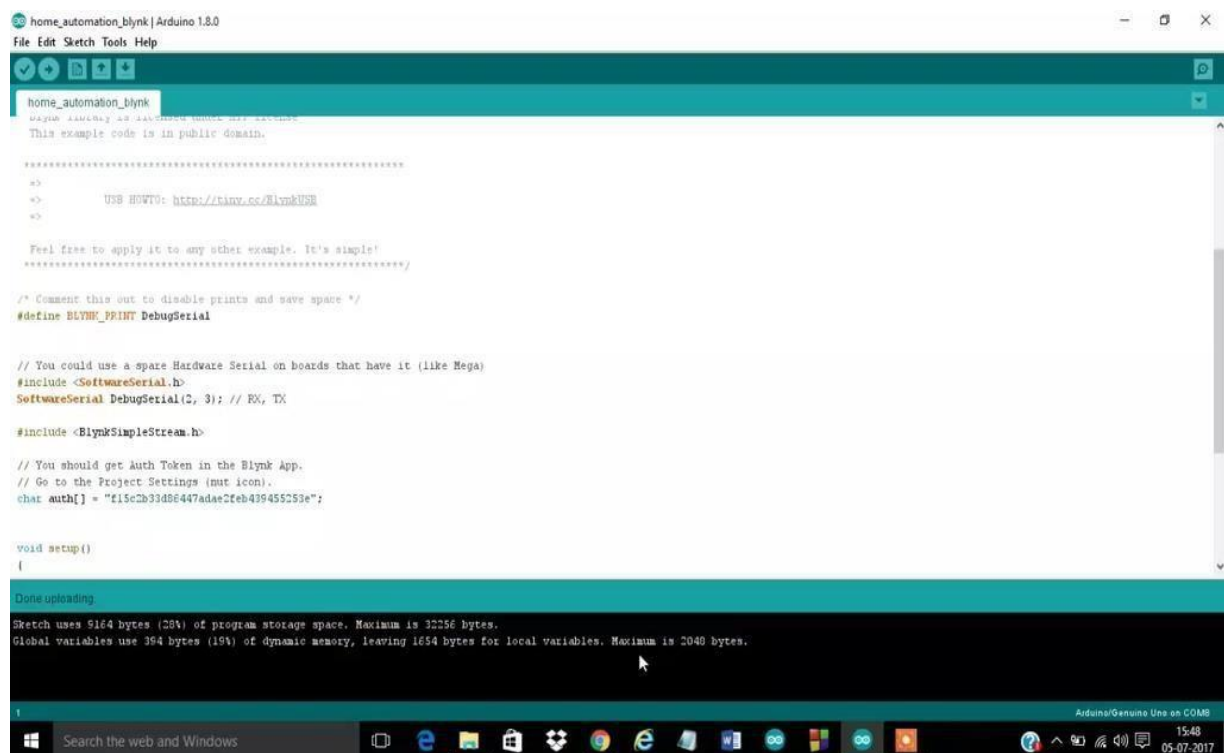
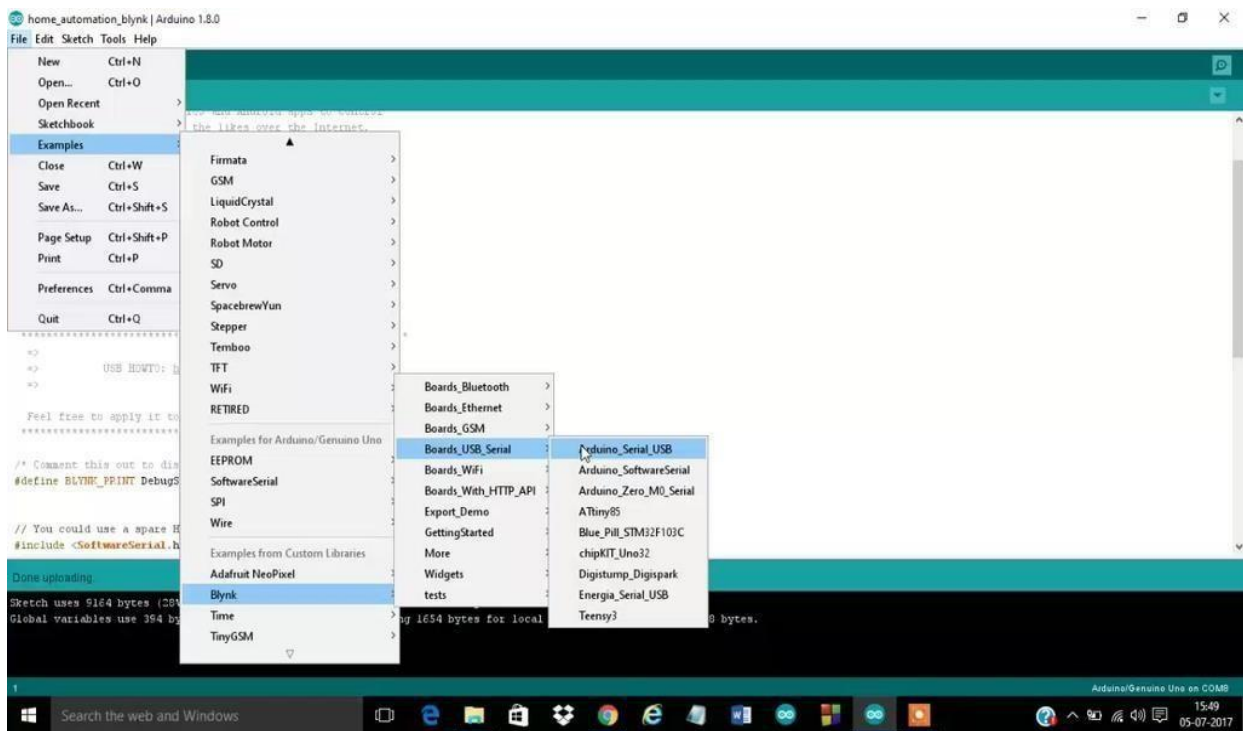
```
// Blynk virtual pins for the 8 relays (V1)
BLYNK_WRITE(V1)
{
  int value = param.asInt(); // Get the value from the app
  digitalWrite(D1, value);
}
```

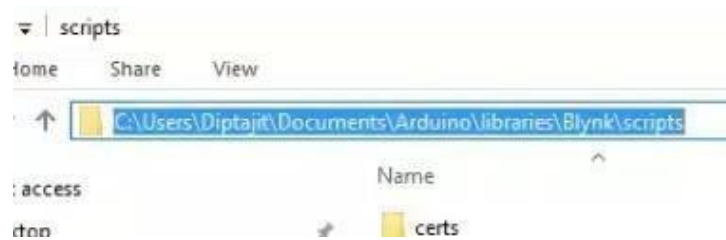
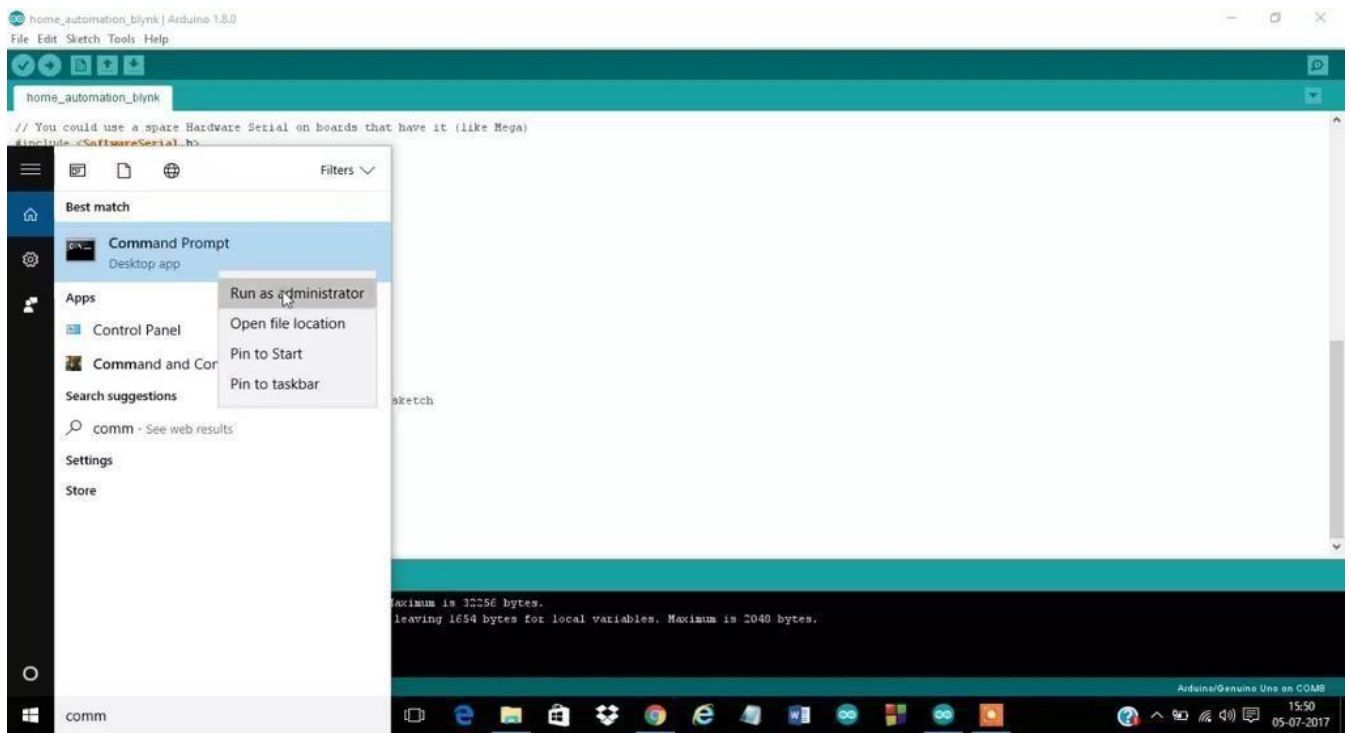
### WORKING PRINCIPLE:

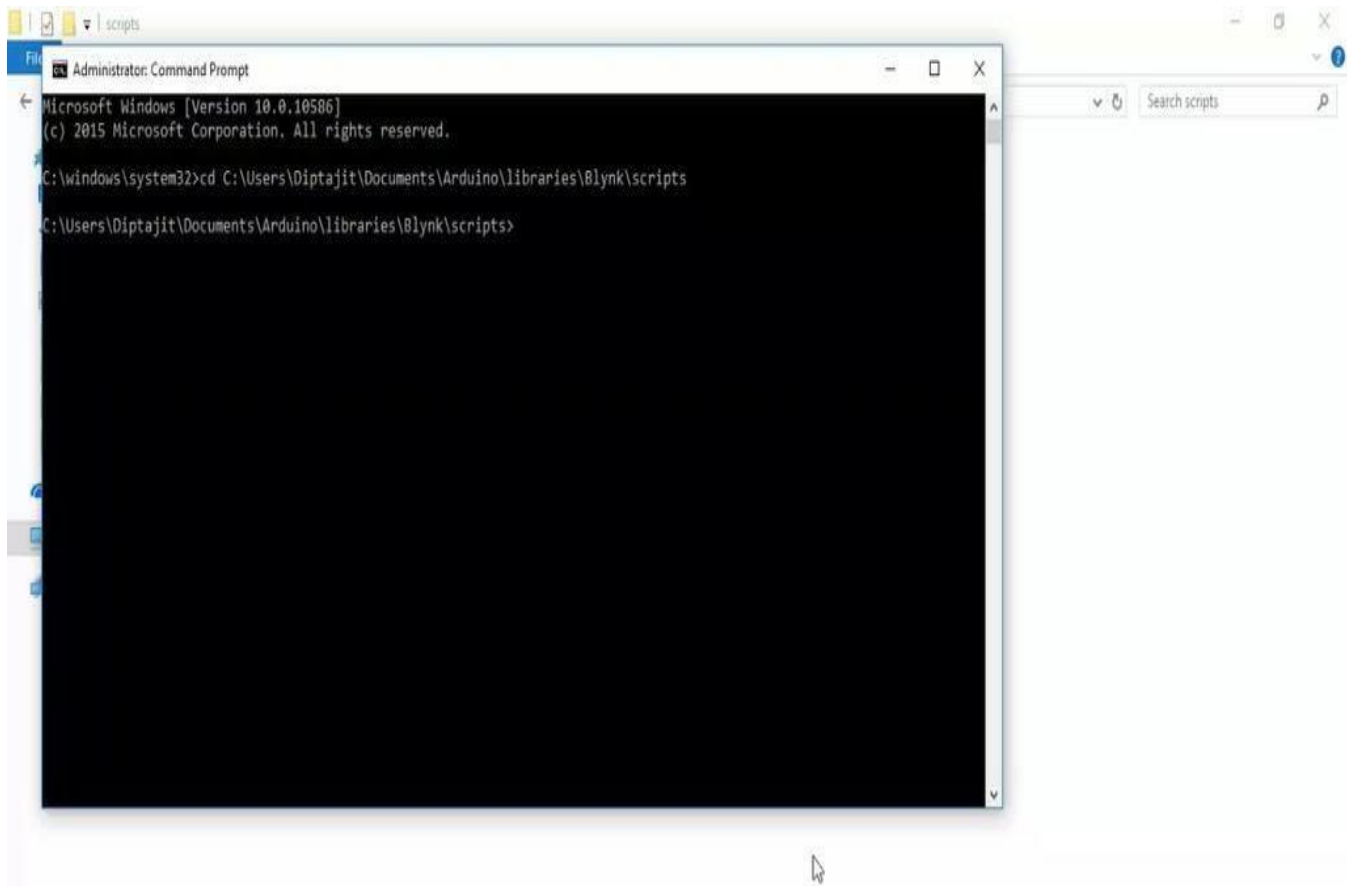










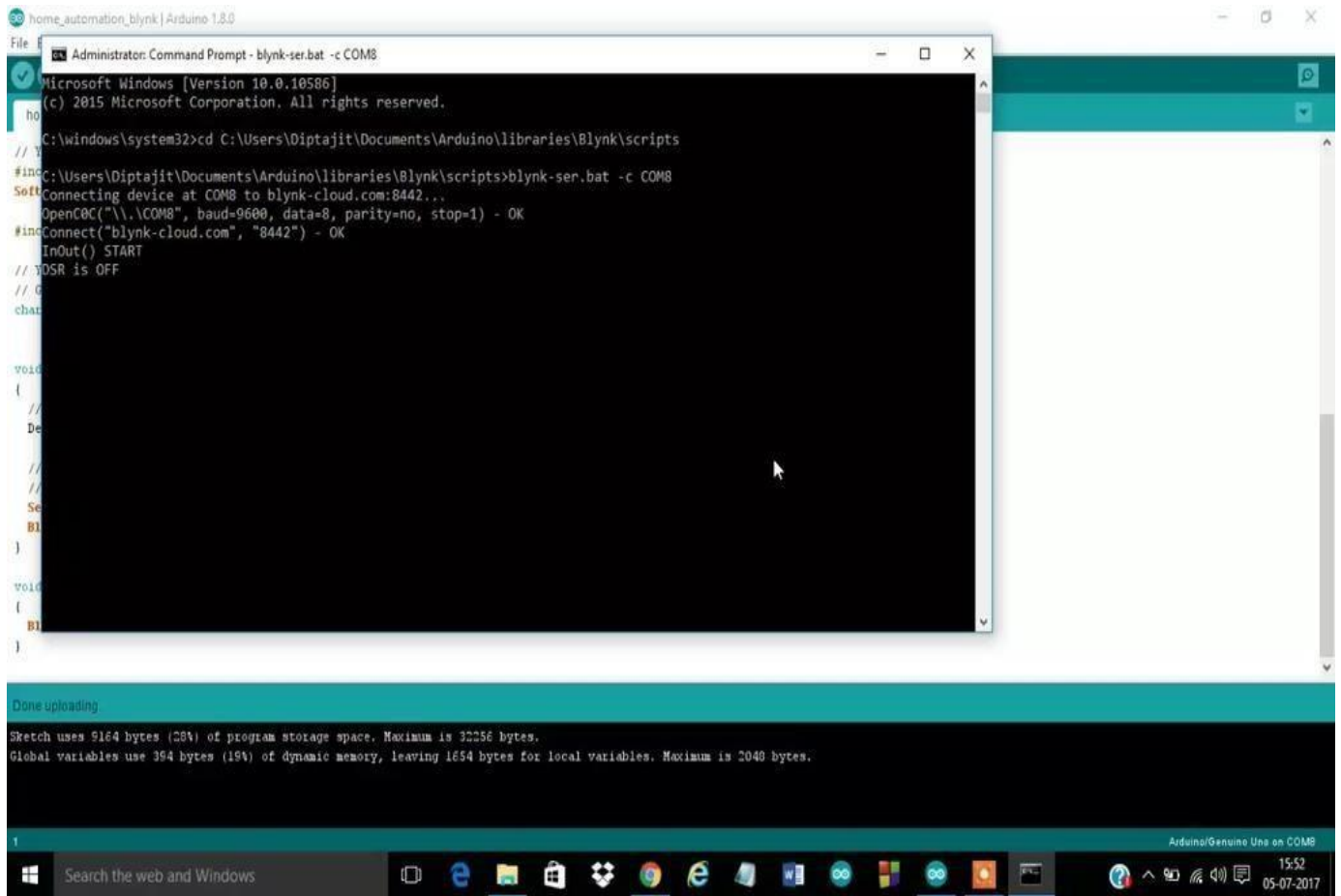


Write your path to blynk-ser.bat folder. For example:

```
cd C:\blynk-library-0.3.1\blynk-library-0.3.1\scripts
```

Run `blynk-ser.bat` file. For example : `blynk-ser.bat -c COM4` (where COM4 is port with your Arduino)

And press "Enter", press "Enter" and press "Enter"



### Working of Relay module:

According to the diagram we can see that there is switch like thing inside the relay module whose one end is connected to COM i.e. Pin 4 and the other end is either connected between NO i.e. Pin 5 or NC i.e. Pin 6. When we are applying 0 V to the signal pin i.e. Pin 3 then the switch remains in NO position (normally open). When we apply +5 V to signal pin the switch drips from NO to NC (normally connected).

### Creating the project in BLYNK App:

Download the BLYNK App from Google Playstore (link has been already given). Open it and you have to make an account there. After that click on "New Project". Now you have to click "CHOOSE DEVICE" and you will be asked to select required hardware, you will choose "Arduino UNO" and in "CONNECTION TYPE" you have to select "USB". You have to give a project name also. Then you click on "Create". Your project is now created and BLYNK will send an authorization token to your mail which you have to put in the arduino code. Then you will get a free space where you have to add buttons, graphs etc. You will get all these from the widget box. In this project as we are operating only one appliance so we will add only one button. After clicking on "Button" the icon will be added in the free space. You can place the button anywhere on the screen. Then you have to click on the button to customize it. You have to give a name there and you have to select whether you are using digital or analog as virtual pin. You also have to mention the pin no. As in this project we are using D13 i.e. Digital pin 13. Now select the mode whether "Push" or "Slide", it depends upon you. After that return to the main screen, you will see a play button on the right corner of the screen, you have to click on that to activate the project. If your system is ready and connected to internet then on mobile after clicking the play button it will show "Online" otherwise "Offline".

### 3. Code analysis and final connection :

First of all you have to add the following link in "additional boards manager URL" in preferences in the Arduino IDE. **Link :** [http://arduino.esp8266.com/stable/package\\_esp8266c...](http://arduino.esp8266.com/stable/package_esp8266c...)

You have to go to the following link : <https://github.com/blynkkk/blynk-library/releases/...> and download the blynk library. After downloading the zip file you have to unzip it and copy the contents of the files (libraries and folders) to the sketchbook-folder of the Arduino IDE. To check whether the blynk library has been added or not restart the Arduino IDE and check in the library section, if you see "**Blynk**" it means that blynk library has been successfully added.

Just copy the code (already provided) or you can get the code from **Examples-->Blynk-->Boards\_USB\_Serials-->Arduino\_Serial\_USB**. In both cases the only change you have to make is that copy the authorization code sent to your mail to Arduino code. Don't upload the code now. Now open "Command Prompt" and run it as administration. A black screen will appear on the screen. Then you have to copy the path of "**scripts**" folder. In my case it is "My Documents\Arduino\libraries\Blynk\scripts" and paste it on the black screen and place enter. Then you have to copy and paste the .bat file in the black screen. The file is "blynk-ser.bat -c COM4". You have to change the COM port number. In my case it was COM8. Now upload the arduino code. Now come back to the command prompt part and press "**enter**" thrice. This will connect you to Blynk Server.

### 4. Control with Blynk App :

Now open blynk app from your mobile and open the project you have created. If your system is connected to Blynk server then you will see '**Online**' in your mobile otherwise you will see '**Offline**'. Now click on the button to On or Off the appliance. If it is not working then check whether the system is connected to the blynk server.



**RESULT:**

Thus an IOT based system was designed successfully.