DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

LAB MANUAL

15CSE480 Internet of Things

AMRITA SCHOOL OF ENGINEERING

AMRITA VISHWA VIDYAPEETHAM

AMRITA NAGAR

COIMBATORE 641 112

Syllabus

15CSE480 **Internet of Things** **3 0 0 3**

Unit - 1

Introduction to loT - loT definition - Characteristics - Things in loT - loT Complete Architectural Stack - loT enabling Technologies - loT Challenges - loT Levels - A Case Study to realise the stack.

Sensors and Hardware for loT - Accelerometer, Proximity Sensor, lR sensor, Gas Sensor,

Temperature Sensor, Chemical Sensor, Motion Detection Sensor. Hardware Kits - Arduino,

Raspberry Pi, Node MCU. A Case study with any one of the boards and data acquisition from sensors (Lab Component)

Unit - 2

Protocols for loT - infrastructure protocol IPV4/V6|RPL), Identification (URLs), Transport (Wi-Fi, Li-Fi, BLE), Discovery, Data Protocols, Device Management Protocols. - A Case Study with MQTT/CoAP usage. (Lab Component)

Cloud and Data analytics- Types of Cloud - loT with cloud challenges - Selection of cloud for loT applications - Fog computing for loT - Edge computing for loT - Cloud security aspects for loT applications - RFM for Data Analytics - Case study with AWS / AZURE / Adafruit / IBM Bluemix (Lab Component).

Unit - 3

Case studies with architectural analysis:

loT applications - Smart City - Smart Water - Smart Agriculture - Smart Energy - Smart Healthcare - Smart Transportation - Smart Retail -Smart waste management . (Lab Component - As a project)

Course Outcomes

	Course Outcome
CO01	Understand the key techniques and theory behind Internet of Things
CO02	Apply effectively the various enabling technologies (both hardware and software) for IoT
CO03	Understand the integration of Cloud and IoT , Edge and Fog Computing
CO04	Apply various techniques for Data Accumulation, Storage and Analytics
CO05	Design and build IoT system for any one interesting Use case

List of Experiments

S.No	List of Experiments
1	Led and Switch Interface [CO02]
2	Analog Sensor Interface [CO02]
3	Serial Communication [CO02]
4	Local Display of Sensor data using LCD [CO02]
5	Local Storage of Sensor Data using SD card [CO04]
6	Display of Sensor values in Mobile handset using Bluetooth [CO04]
7	Remote control of Electrical appliances using Mobile handset and Bluetooth [CO05]
8	Location based service using RFID / BLE [CO05]
9	Remote control of Electrical appliances using Mobile handset and Wi-Fi using mobile/ Web Application [CO05]
10	Local Webserver using NodeMCU and displaying Sensor values [CO03] [CO04].
11	Data Accumulation and Storage in cloud using NodeMCU and Free cloud Platform like Thingspeak, Adafruit IO Platform. [CO03] [CO04]
12	Remote monitoring and control of actuators using NodeMCU and Android Application[CO05]

CO-PO Mapping

PO/PSO / CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	1	2	2										3	2
CO2	1	2	2										3	2
CO3	1	2	2					1					3	2
CO4	1	2	3		3			3	2				3	2

TABLE OF CONTENT

S.No	Topic	Page No
1	Introduction to Internet of Things	6
2	Introduction to NodeMCU	9
3	Input Output Programming	13
4	ADC Programming	18
5	Display Programming	24
6	Communication Protocol Programming	28
7	Web and Cloud Programming	34
8	Appendix A - Procedure to visualize through Thingspeak	48
9	Appendix B - Procedure to control using Voice Assistance	53
10	Appendix C - List of Case Studies	64
11	References	65

1. Introduction to Internet of Things

Definition of IoT

The **Internet of things** (**IoT**) is the interconnection of existing physical devices and everyday objects through Internet connectivity. IoT is an integration of embedded technology, Sensor Technology and communication technology to interact with objects in the environment and they can be remotely monitored and controlled through visualization.

Technologies concerned in IoT

- Real time Analytics
- Machine learning
- Artificial Intelligence
- Embedded systems
- Wireless sensor networks
- Control systems
- Automation in devices

IoT technology is most synonymous with products pertaining to the concept of the "smart home", covering devices and appliances (such as lighting fixtures, thermostats, home security systems and cameras, and other home appliances) that support one or more common ecosystems, and can be controlled via devices associated with that ecosystem, such as smart phones and smart speakers with privacy and security issues.

There are many technologies that facilitate the IoT such as Addressability, Short range wireless (BLE,ZigBee,NFC,RFID), Medium range wireless(LTE-Advanced), Long Range wireless(LPWAN,VSAT) and Wired (Ethernet and Power Line Communication).

Research Issues in IOT

- Platform fragmentation
- Privacy, autonomy, and control
- Data storage
- Security and Safety
- Environmental sustainability impact
- Intentional obsolescence of devices
- Lack of interoperability and unclear value propositions
- Business planning and models

IOT Prototype design

IOT prototype is the process of building IoT hardware and devices enhanced with smart sensors and embedded systems using many off-the-shelf components like sensors, circuit boards, and microcontrollers. A lot of these off-the-shelf solutions are readily available to end consumers. Take an NodeMCU/Arduino board, for illustration. You can order it online and have it delivered within 24 hours. Also, a prototype is by no means a market-ready product. It is just a trial version of your connected solution and acts as proof that your innovative idea will work the way

IOT will seep into every facet of our daily lives, managing our homes, tending our gardens, and even monitoring our mailboxes. In future IOT products will be as omnipresent as mobile devices you use in your daily life, so now it is a great time to get involved and practicing in this area to fit you in the present job market. Your IOT prototype is used to understand the pinch points and frame out the necessary parameters of your IOT product deployment. The prototype must be end-to-end, including a thin thread connecting the sensor through the device, network, cloud, end-user interface, and enterprise integration. However, building an Internet of Things (IoT) prototype is rewarding and also a frustratingly challenging engineering process.

Application of IOT

- Smart Parking System
- Smart Street Lighting
- Smart Water Management
- Smart Homes & Building
- Smart Waste Management
- Smart Transportation
- Smart Citizen Safety
- Smart Security
- Smart Appliances
- Smart Health Monitoring
- Smart Retail
- Smart Energy Management
- Smart Grids
- Smart Environment
- Smart Manufacturing
- Smart Industries
- Smart Roads & Infrastructure
- Smart Agriculture
- Smart Public Information systems
- Smart Asset Management

IOT Engineer will have expertise in any one of the IOT devices, gateways, IOT Platforms, Cloud services and application development.

Smart Device/Gateway	**Communication**	**Cloud Platform**	**Application**
Arduino UNO + Ethernet Shield	Bluetooth EDR Bluetooth Low Energy	Thingspeak	PHP + MySQL
Arduino YUN	Wi-Fi	Amazon Web Services	Android Application
Arduino UNO + GSM shield	Ethernet	IBM Watson IOT	Arduino IDE
Arduino + Lora Shield	GSM /GPRS	Microsoft Azure	Angular js + Node js
ESP8266 – ESP12E	RF	Google Firebase	MIT App inventor
ESP8266 – ESP32	LoRa, LoRaWAN	Thingworx	Ionic Framework
Raspberry Pi 3	RFID, NFC	API.ai	Android Things
Raspberry Pi Zero	HTTP, HTTPS	Clarify	Python + Flask
Intel Edison	MQTT	--	Python + Django

2. Introduction to NodeMCU

The IoT boards can be broken down into two types, microcontroller boards and single computer boards (SBC). A microcontroller board is a system on a chip (SoC) that has data processing and storage Containing processing cores, RAM and EPROM for the storage of custom programs that are executed on the microcontroller, these boards are PCBs with added circuitry that support the microcontroller. This makes it more convenient when using the board to prototype and program. A single board computer is a step up from microcontroller boards as it allows for the attachment of computer peripheral devices while offering more processing power and memory. Just like microcontroller boards, SBC capabilities can be expanded with the addition of expansion boards or through external modules, such as motor controllers, to mitigate device limitations.

NodeMCU is an open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi transceiver module with low cost, and hardware which is based on the ESP-12 module. The term "NodeMCU" by default refers to the firmware rather than the development kits. The firmware uses the Lua scripting language. NodeMCU was created shortly after the ESP8266 came out. On December 30, 2013, Espressif Systems began production of the ESP8266. The ESP8266 is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core, widely used in IoT applications. NodeMCU started on 13 Oct 2014.The pinout diagram of NodeMCU is shown in the following Figure 1

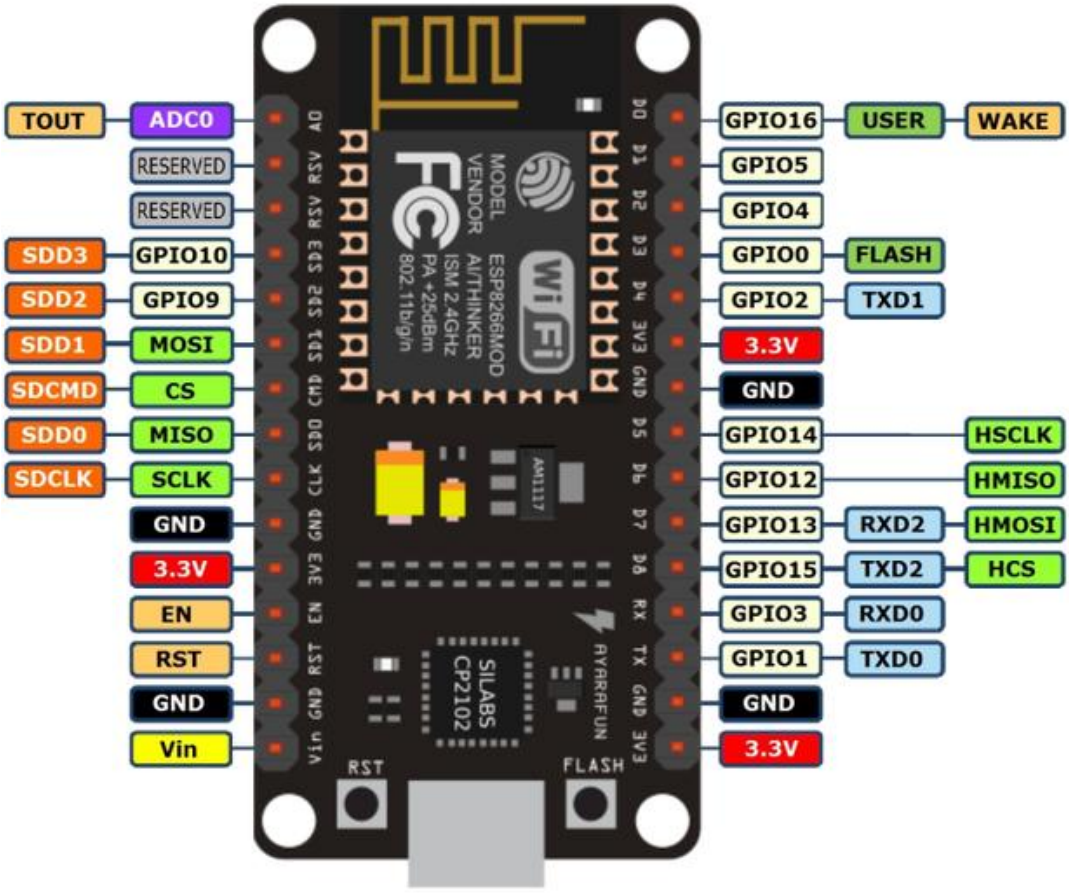


Figure1: Pin out Diagram of NodeMCU

Specification

MCU	32bit TensilicaL106
Frequency	80/160 MHZ
I/O	17XD10
ADC pin	1X10bit(1V)
Operating volt	3-3.6v
Memory	4MB
Wifi	IEEE802.11 b/g/n

With features such as Firmware LUA, Micro python Python3, Espruino JavaScript, Arduino IDE, Official Hardware ESP8266

Feature of ASC CSE IoT Node MCU ESP8266

Open source, interactive and programmable, lowcost, simple and smart, wifi enabled, power via USB, Plug and play board

Pros and Cons

Pros: Low energy consumption, integrated support for wifi network, reduced size of the board, power supply via usb, low cost

Cons: Connecting limited number of sensors, Less pin out, New Language and IDE
ESP8266 GPIOs are mapped as detailed below in NodeMCU.

NodeMCU	**ESP8266**
D0	GPIO 16
D1	GPIO 5
D2	GPIO 4
D3	GPIO 0
D4	GPIO 2
D5	GPIO 14
D6	GPIO 12
D7	GPIO 13
D8	GPIO 15
D9	GPIO 3
D10	GPIO 1

For simplicity, ASC CSE designed an IOT KIT with plug and play concepts to motivate the Engineers in the field of Automation with Node MCU as the controlling unit of ASC CSE IoT kit.

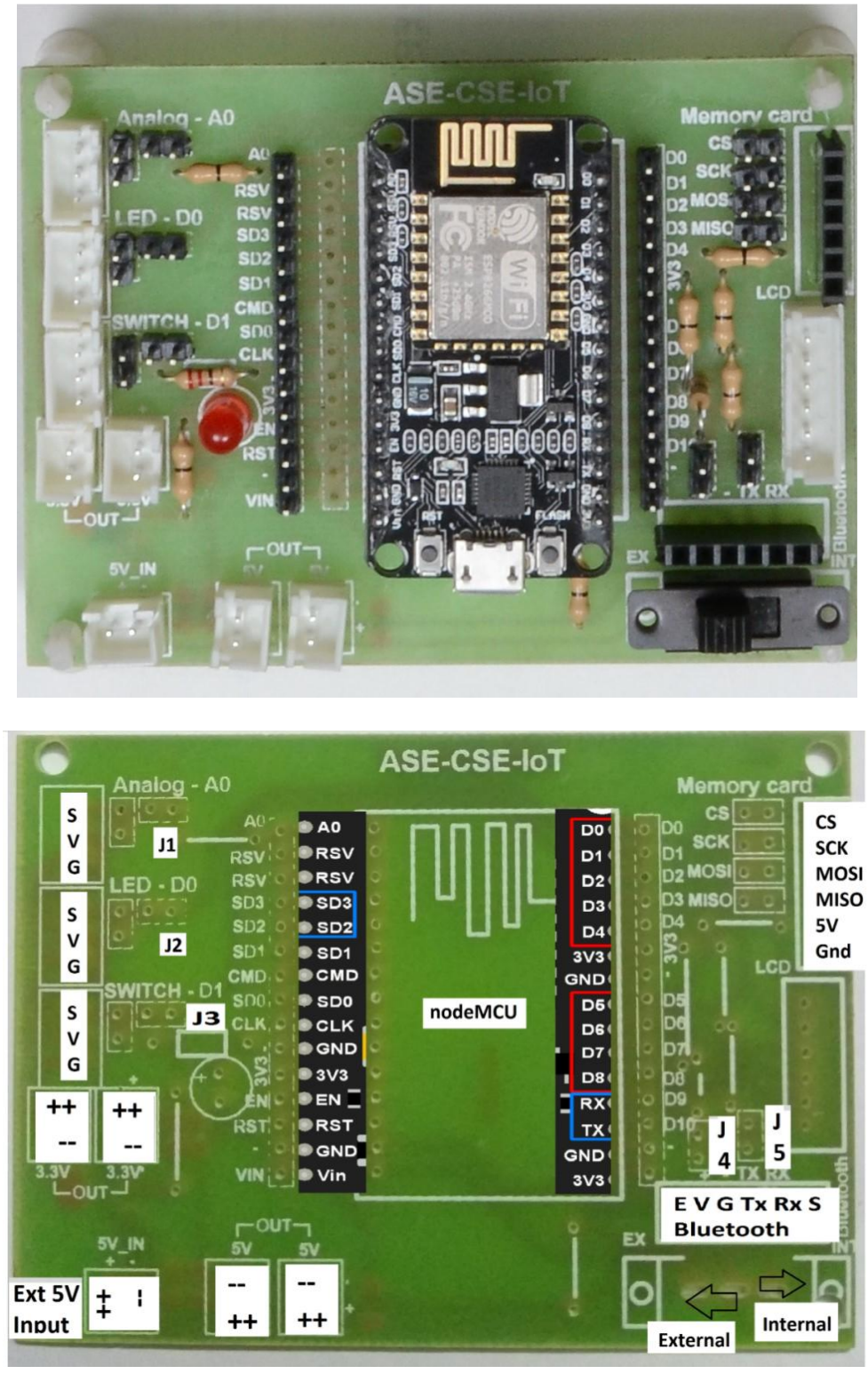


Figure 3Board level pin details of ASC CSE IoT Board

Instructions to install NodeMCU in Arduino:

Keep your PC /laptop connected to the internet while installing NodeMCU.

1. Download Arduino IDE.
2. Open you IDE and click on **"File -> Preferences".**
3. In "Aditional Boards Manager URLs" add this line and click on "OK": **"*http://arduino.esp8266.com/stable package_esp8266com_index.json*"**
4. Go to"**Tools -> Board -> Boards Manager"**, type "ESP8266" and install it.
5. Go again to **"Tools -> Board"** and select "NodeMCU 1.0 (ESP12E module)"
6. Go again to **"Tools->Port"** and select the com port to which NodeMCU is connected.

List of Prototype model developed for practice

1. I/O concept
2. ADC Concept
3. Display Concept
4. Communication Protocol Concept
5. Web & Cloud Concept

3. Input Output Programming

Exercise 3a: LED blinking

Aim: To blink an LED

Requirements: Node Mcu , LED, Connecting Wires

Procedure: Connect Single Color Red LED to port D0

Program:

```
void setup()
        {
        pinMode ( D0, OUTPUT);
        }
void loop()
         {
        digitalWrite(D0, HIGH);
        delay(1000);
        digitalWrite(D0, LOW);
        delay(1000);
        }
```

Output: Red LED connected to port D0 blinks at an interval of 1 second.

Exercise 3b: LED Fading

Aim: LED fading

Requirements: Node MCU , LED, Connecting Wires

Procedure: Connect Single Color Red LED to port D0

Program:

```
int led = D0; // the pin that the LED is attached to
int brightness = 0; // how bright the LED is
int fadeAmount = 5; // how many points to fade the LED by
// the setup routine runs once when you press reset:
void setup()
        {
        pinMode(led, OUTPUT);
        }
        // the loop routine runs over and over again forever:
        void loop()
```

```
{
// set the brightness of pin 9:
analogWrite(led, brightness);
// change the brightness for next time through the loop:
brightness = brightness + fadeAmount;
// reverse the direction of the fading at the ends of the fade:
if (brightness == 0 || brightness == 255)
 {
fadeAmount = -fadeAmount ;
}
// wait for 30 milliseconds to see the dimming effect
delay(30);
}
```

Output: LED brightness increase and decrease continuously

Exercise 3c: Bi-Color LED

Aim: To Blink Bi-Color LEDs (Alternate blinking of Red and Green LEDs.)

Requirements: Bicolor LED, nodemcu, connecting wires

Procedure: Bicolor LED has 3 pins one is cathode and Others are anode (Red/Green) Connect two digital pin D0 and D1 corresponding Bicolor LED

Program:

```
void setup()
        {
        pinMode ( D0, OUTPUT);
        pinMode (D1, OUTPUT);
        }
void loop()
        {
        digitalWrite(D0, HIGH);
        digitalWrite(D1, LOW);
        delay(1000);
        digitalWrite(D0, LOW);
        igitalWrite(D1,HIGH);
        delay(1000);
        }
```

Output: Red LED connected to port D0 is ON, Green LED connected to port D1 will be OFF and Green LED will be ON for 1 second. This process will be repeated till power is applied.

Exercise 3d: Tri-Color LED

Aim: TRI Color LED

Requirements: TRI Color LED, Nodemcu , connecting wires

Procedure: LED connect Respective pins D1,D2,D3

Program:

```
int Red =D3, Green = D2, Blue =D 1; //LED pins
void setup()
        {
        pinMode(Red, OUTPUT);//declare pin-5 to be an output
        pinMode(Green, OUTPUT);//declare pin-6 to be an output
        pinMode(Blue, OUTPUT);//declare pin-7 to be an output
        }
void loop()
         {
        digitalWrite(Red, LOW); digitalWrite(Green, HIGH);
        digitalWrite(Blue, LOW); delay(1000);
        digitalWrite(Red, HIGH); digitalWrite(Green, LOW);
        digitalWrite(Blue, LOW); delay(1000);
        digitalWrite(Red, LOW); digitalWrite(Green, LOW);
        digitalWrite(Blue, HIGH); delay(1000);
        }
```

Output: You should see your Red LED turn on, Green LED turn off and Blue LED turn off, your Red LED turn off, Green LED turn on and Blue LED turn off, your Red LED turn off, Green LED turn off and Blue LED turn on. If the required output is not seen, make sure you have assembled the circuit correctly, and verified and uploaded the code to your board.

Exercise 3e: Control LED using Button

Aim: An LED indicator by pressing a Switch (INPUT and OUTPUT Concept)

Requirements: LED, Button , Nodemcu , connecting wires

Procedure: LED is connected to D0 and Switch is connected to D1When Switch (Push Button) is pressed, LED will be switched ON and when Switch (Push Button) is released, LED will be switched OFF.

Program:

```
/* Input – Switch and Output – LED demo */
const int ledPin = D0; // the number of the LED pin
```

```
const int buttonPin = D1; // the number of the pushbutton pin
// variables will change:
int buttonState = 0; // variable for reading the pushbutton status
void setup() {
// initialize the LED pin as an output:
pinMode(ledPin, OUTPUT);
// initialize the pushbutton pin as an input:
pinMode(buttonPin, INPUT);
        }
void loop() {
        // read the state of the pushbutton value:
        buttonState = digitalRead(buttonPin);
        // check if the pushbutton is pressed. If it is, the buttonState is HIGH:
if (buttonState == HIGH)
        {
        // turn LED on:
        digitalWrite(ledPin, HIGH);
        }
        else
        {
        // turn LED off:
        digitalWrite(ledPin, LOW);
        }
}
```

Output: When Switch is pressed, LED gets switched ON and the LED will be switched OFF when the switch gets released.

Exercise 3f: Controlling Relay using Button

Aim: To design a Relay by pressing a Switch (INPUT and OUTPUT Concept)

Requirements: Relay Module,Lamp Holder, AC Power supply,LED,Button , Nodemcu , connecting wires

Procedure : Relay is connected to D0 and Switch is connected to D1 When Switch (Push Button) is pressed, Relay will be ON trigger the AC unit and when Switch (Push Button) is released, Relay will be switched OFF.

Program:

```
/* Input – Switch and Output – Relay demo */
const int relayPin = D0; // the number of the LED pin
const int buttonPin = D1; // the number of the pushbutton pin
// variables will change:
int buttonState = 0; // variable for reading the pushbutton status
```

```
void setup()
        {
        // initialize the LED pin as an output:
        pinMode(relayPin, OUTPUT);
        // initialize the pushbutton pin as an input:
        pinMode(buttonPin, INPUT);
        }
void loop()
        {
        // read the state of the pushbutton value:
        buttonState = digitalRead(buttonPin);
        // check if the pushbutton is pressed. If it is, the buttonState is HIGH:
        if (buttonState == HIGH)
        {
        // turn LED on:
        digitalWrite(relayPin, HIGH);
        }
        Else
        {
        // turn LED off:
        digitalWrite(relayPin, LOW);
        }
}
```

Output: When Switch is pressed, Relay gets switched ON and the Relay will be switched OFF when the switch gets released.

4. ADC Programming

Exercise 4a: POT Sensor Interface

Aim: To Interface a POT Sensor to ADC Channel and sending equivalent digital data to COM port

Requirements: POT, Nodemcu , connecting wires

Procedure : POT is connected to A0 and Serial port is configured to 9600, N81.When POT sensor is rotated, analog value applied to Analog Channel A0 will be varied from 0V to 5V and its equivalent digital value (in decimal form) will be displayed in the serial monitor.

Program:

```
// the setup routine runs once when you press reset:
void setup()
        {
        // initialize serial communication at 9600 bits per second:
        Serial.begin(9600);
        }
        // the loop routine runs over and over again forever:
void loop()
        {
        // read the input on analog pin 0:
        int sensorValue = analogRead(A0);
        // print out the value you read:
        Serial.println(sensorValue);
        delay(100); // delay in between reads for stability
        }
```

The Table shows Analog, Binary and Decimal equivalent reading

S.No	Analog Input in Volts	Binary Output	Decimal Output
1	0.0V	0b 00 0000 0000	0
2	1.25 V	0b 00 1111 1111	255
3	2.5V	0b 01 1111 1111	511
4	3.75 V	0b 10 1111 1111	767
5	5.0V	0b 11 1111 1111	1023

Output: Present value of the serial Monitor

Exercise 4b: LED Control using POT Threshold Value

Aim: To Check for POT threshold Voltage indication through LED

Requirements: POT, LED, Nodemcu , connecting wires

Procedure: POT connected A0 pin,LED connect D0 pin

Program:

```
void setup()
        {
        Serial.begin(9600);
        pinMode(D0,OUTPUT);
        }
        // the loop routine runs over and over again forever:
void loop()
        {
        // read the input on analog pin 0:
        int sensorValue = analogRead(A0);
        // print out the value you read:
        Serial.println(sensorValue);
        delay(1000); // delay in between reads for stability
        if(sensorValue >580)
        {
        digitalWrite(D0,HIGH);
        }
        else
        {
        digitalWrite(D0,LOW);
        }
}
```

Output: POT can be connected A0 pin.so, NodeMcu reads analog value certain condition met LED on else LED off.

Exercise4c: Gas Sensor

Aim: Check for Gas Leakage through Gas Sensor

Requirements: Gas sensor (MQ-2),Connecting wires,Nodemcu

Procedure: Gas sensor connected to A0 Pin of Nodemcu.Open serial monitor

Program:

```
void setup()
        {
        Serial.begin(9600);
        }
void loop()
        {
        // read the input on analog pin 0:
        int sensorValue = analogRead(A0);
        //(default state is below 350)
        if(sensorValue>350) //read Gas sensor
        Serial.println("gas Detected ");
        else
        Serial.println("No gas Detected ");
        delay(1000);// delay in between reads for stability
}
```

Output: You will see Gas Detected on Serial Monitor when the values will exceeds 350 values corresponding to the voltage at pin A0. If those values are below 350 then you will see No Gas Detected on Serial Monitor Frequently of every 1000 milli seconds.

Exercise 4d: Temperature Sensor (LM35)

Aim: To monitor room temperature using Temperature Sensor

Requirements: Temperature Sensor (LM35),Nodemcu,Connecting Wires

Procedure: Temperature sensor Connected to A0 pin of Nodemcu

Program:

```
void setup()
         {
        Serial.begin(9600);
```

```
}
// the loop routine runs over and over again forever:
void loop()
{
// read the input on analog pin 0:
float sensorValue = analogRead(A0);
sensorValue=(sensorValue*5000)/10230 ;
// print out the value you read:
Serial.println(sensorValue);
delay(1000);// delay in between reads for stability
}
```

Output: You will see the temperature display on the serial port monitor which is updated every second.

Exercise 4e: Light Sensor using LDR

Aim: To monitor light intensity level using LDR(Light Dependent Resistor)

Requirements: LDR,Nodemcu,connecting wires

Procedure: LDR connected to A0 Pin of Nodemcu.Open Serial monitor

Program:

```
void setup()
{
//initialize serial communication
//at 9600 bits per second:
Serial.begin(9600);
}
void loop()
{
// read the input on analog pin 0:
int sensorValue = analogRead(A5);
// print out the value you read:
Serial.println(sensorValue);
delay(1000);//delay in between reads for stability
}
```

Output:

The sensor value is much higher. The numbers you see will vary. This depends on how much light is in the room, and how much gets through to the sensor even when your hand is covering it.

Exercise 4f: DHT sensor

Aim: To measure temperature/Humidity using DHT sensor

Requirements: DHT11, Nodemcu , connecting wires

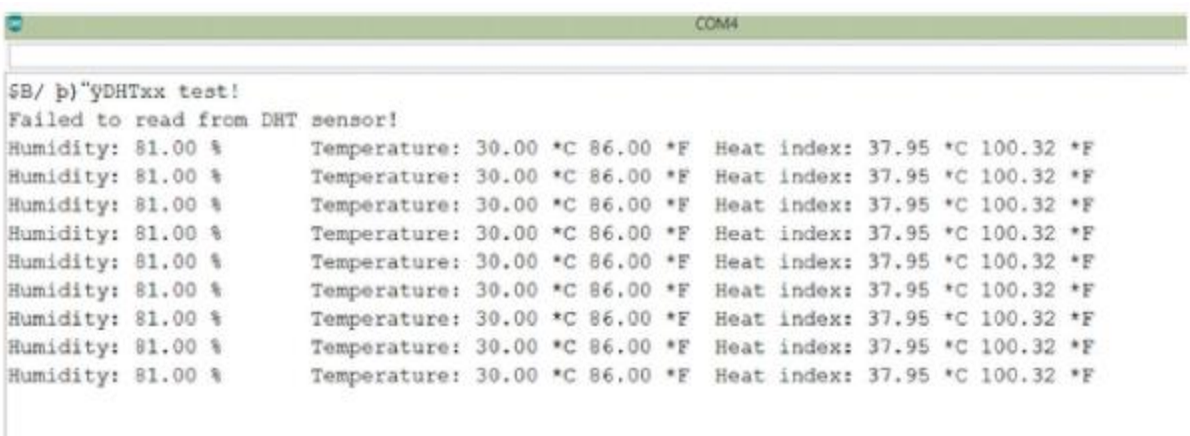
Procedure: Connect pin 1 (on the left) of the sensor to +5V/3.3V
Connect pin 4 (on the right) of the sensor to GROUND
Connect pin 2 of the sensor to whatever your Digital Pin DHTPIN

Program:

```
// Example testing sketch for various DHT humidity/temperature sensors
#include "DHT.h"
#define DHTPIN D2 // what digital pin we're connected to
// Uncomment whatever type you're using!
#define DHTTYPE DHT11 // DHT 11
//#define DHTTYPE DHT22 // DHT 22 (AM2302), AM2321
//#define DHTTYPE DHT21 // DHT 21 (AM2301)
// Connect pin 1 (on the left) of the sensor to +5V
// NOTE: If using a board with 3.3V logic like an Arduino Due connect pin 1
// to 3.3V instead of 5V!
// Connect pin 2 of the sensor to whatever your DHTPIN is
// Connect pin 4 (on the right) of the sensor to GROUND
// Connect a 10K resistor from pin 2 (data) to pin 1 (power) of the sensor
// Initialize DHT sensor.
// Note that older versions of this library took an optional third parameter to
// tweak the timings for faster processors. This parameter is no longer needed
// as the current DHT reading algorithm adjusts itself to work on faster procs.
DHT dht(DHTPIN, DHTTYPE);
void setup()
        {
        Serial.begin(9600);
        Serial.println("DHTxx test!");
        dht.begin();
        }
void loop()
        {
        // Wait a few seconds between measurements.
        delay(2000);
        // Reading temperature or humidity takes about 250 milliseconds!
        // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
        float h = dht.readHumidity();
        // Read temperature as Celsius (the default)
        float t = dht.readTemperature();
        // Read temperature as Fahrenheit (isFahrenheit = true)
```

```
float f = dht.readTemperature(true);
// Check if any reads failed and exit early (to try again).
if (isnan(h) || isnan(t) || isnan(f)) {
Serial.println("Failed to read from DHT sensor!");
return;
}
// Compute heat index in Fahrenheit (the default)
float hif = dht.computeHeatIndex(f, h);
// Compute heat index in Celsius (isFahreheit = false)
float hic = dht.computeHeatIndex(t, h, false);
Serial.print("Humidity: ");
Serial.print(h);
Serial.print(" %\t");
Serial.print("Temperature: ");
Serial.print(t);
Serial.print(" *C ");
Serial.print(f);
Serial.print(" *F\t");
Serial.print("Heat index: ");
Serial.print(hic);
Serial.print(" *C ");
Serial.print(hif);
Serial.println(" *F");
}
```

Output:



```
COM4

SB/ þ)˝ÿDHTxx test!
Failed to read from DHT sensor!
Humidity: 81.00 %	Temperature: 30.00 *C 86.00 *F	Heat index: 37.95 *C 100.32 *F
Humidity: 81.00 %	Temperature: 30.00 *C 86.00 *F	Heat index: 37.95 *C 100.32 *F
Humidity: 81.00 %	Temperature: 30.00 *C 86.00 *F	Heat index: 37.95 *C 100.32 *F
Humidity: 81.00 %	Temperature: 30.00 *C 86.00 *F	Heat index: 37.95 *C 100.32 *F
Humidity: 81.00 %	Temperature: 30.00 *C 86.00 *F	Heat index: 37.95 *C 100.32 *F
Humidity: 81.00 %	Temperature: 30.00 *C 86.00 *F	Heat index: 37.95 *C 100.32 *F
Humidity: 81.00 %	Temperature: 30.00 *C 86.00 *F	Heat index: 37.95 *C 100.32 *F
Humidity: 81.00 %	Temperature: 30.00 *C 86.00 *F	Heat index: 37.95 *C 100.32 *F
Humidity: 81.00 %	Temperature: 30.00 *C 86.00 *F	Heat index: 37.95 *C 100.32 *F
```

5. Display Programming

Exercise 5a: LCD interfacing

Aim: To demonstrate the functioning of Liquid crystal Display:4 bit Mode LCD

Requirements: 16*2 LCD, Nodemcu , connecting wires

Procedure: Display **Hello World to** Demonstrates the use a 16x2 LCD display.

Procedure: The LiquidCrystal library works with all LCD displays that are compatible with the Hitachi HD44780 driver.
There are many of them out there, and you can usually tell them by the 16-pin interface. This sketch prints "Hello World!" to the LCD and shows the time.

LCD Pins	NodeMCU Pins
RS	D0
EN	D1
Data 4	D2
Data 5	D5
Data 6	D6
Data 7	D7

ASC CSE circuit board details:

- LCD RS pin to digital pin 12
- LCD Enable pin to digital pin 11
- LCD D4 pin to digital pin 5
- LCD D5 pin to digital pin 4
- LCD D6 pin to digital pin 3
- LCD D7 pin to digital pin 2
- LCD R/W pin to ground
- LCD VSS pin to ground
- LCD VCC pin to 5V
- 10K resistor:
- ends to +5V and ground
- wiper to LCD VO pin (pin 3)

Program:

```
#include <LiquidCrystal.h>
// initialize the library by associating any needed LCD interface pin
// with the arduino pin number it is connected to
/*const int rs = D0, en = D1, d4 = D2, d5 = D5, d6 = D6, d7 = D7;
```

```
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);*/ //This above line also correct
LiquidCrystal lcd(D0,D1,D2,D5,D6,D7); // here directly mention PIN name
void setup()
        {
        // set up the LCD's number of columns and rows:
        lcd.begin(16, 2);
        // Print a message to the LCD.
        lcd.print("hello, world!");
        }
void loop()
        {
        // set the cursor to column 0, line 1
        // (note: line 1 is the second row, since counting begins with 0):
        lcd.setCursor(0, 1);
        // print the number of seconds since reset:
        lcd.print(millis() / 1000);
}
```

Output: The following String will be displayed in the LCD.**Hello World**

Exercise 5b: Sensor value in LCD

Aim: To display sensor/ POT value to LCD interface

Requirements: POT,Nodemcu, connecting wires

Procedure: Potentiometer POT connected to Analog Channel A0 and LCD wiring as like

Program:

```
#include <LiquidCrystal.h>
// initialize the library by associating any needed LCD interface pin
// with the arduino pin number it is connected to
/*const int rs = D0, en = D1, d4 = D2, d5 = D5, d6 = D6, d7 = D7;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);*/ //This above line also correct
LiquidCrystal lcd(D0,D1,D2,D5,D6,D7); // here directly mention PIN name
void setup()
        {
        // set up the LCD's number of columns and rows:
        lcd.begin(16, 2);
        Serial.begin(9600);
        // Print a message to the LCD.
        lcd.print("hello, world!");
        delay(2000);
        lcd.clear();
        }
```

```
void loop()
        {
        int sensorvalue=(analogRead(A0));
        lcd.setCursor(0, 0);
        lcd.print("sensor value");
        // set the cursor to column 0, line 1
        // (note: line 1 is the second row, since counting begins with 0):
        lcd.setCursor(0, 1);
        // print the number of seconds since reset:
        lcd.print(sensorvalue);
        delay(1000);
        Serial.println(sensorvalue);
        delay(1000);
}
```

Output: The following ADC POT value will be displayed in the LCD.

Exercise 5c: Digital Thermometer

Aim: To design a Digital Thermometer

Requirements: Temperature sensor LM35, Nodemcu, connecting wires,LCD

Procedure: Temperature Sensor LM35 is connected to Analog Channel A0 and LCD wiring as like

Program:

```
#include <LiquidCrystal.h>
// initialize the library by associating any needed LCD interface pin
// with the arduino pin number it is connected to
const int rs = D0, en = D1, d4 = D2, d5 = D5, d6 = D6, d7 = D7;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
void setup()
        {
        // set up the LCD's number of columns and rows:
        lcd.begin(16, 2);
        // Print a message to the LCD.
        Serial.begin(9600);
        }
void loop()
        {
        // set the cursor to column 0, line 1
        // (note: line 1 is the second row, since counting begins with 0):
        int Temperature = analogRead(A0);
        float TempinC = ( Temperature * 3300)/ 10230.0 ;
```

```
lcd.setCursor(0,0);
lcd.print("Digital Thermometer!");
lcd.setCursor(0, 1);
lcd.print(" ");
lcd.setCursor(0, 1);
// print the number of seconds since reset:
lcd.print(TempinC);
Serial.println(TempinC);
delay(1000);
}
```

Output: Temperature sensor display the Digital value of the room temperature

6. Communication Protocol Programming

Exercise 6a: UART Protocol wired

Aim: LED ON/OFF using Serial monitor input condition

Requirements: USB cable, LED, Nodemcu, connecting wires

Procedure: LED will be connect D0 pin,

Program:

```
const int led1=D0;
char incomingByte = 0; // for incoming serial data
void setup()
        {
        Serial.begin(9600); // opens serial port, sets data rate to 9600 bps
        pinMode(led1,OUTPUT);
        Serial.println(" .");
        Serial.println("you have entered 1 LED ON , 0 LED OFF");
        }
void loop()
        {
        // send data only when you receive data:
        if (Serial.available())
        {
        // read the incoming byte:
        incomingByte = Serial.read();
        Serial.println(incomingByte);
        if((incomingByte=='1'))
        {
        Serial.println("you have entered 1 LED ON ");
        digitalWrite(led1,HIGH);
        }
        if((incomingByte=='0'))
        {
        Serial.println("you have entered 0 LED OFF ");
        digitalWrite(led1,LOW);
        }
        }
        }
```

Output: LED will be ON when 1 enter in serial Monitor, LED will be OFF when 0 enter in serial Monitor

Exercise 6b: UART Protocol wireless

Aim: LED ON/OFF using Bluetooth Module input condition

Requirements: Bluetooth, Bluetooth Terminal app,USB cable, LED,Nodemcu, connecting wires

Procedure: Blue Tooth Module HC-05 is interfaced to NodeMCU

BlueTooth Rx -> Tx of NodeMCU
BlueTooth Tx -> Rx of NodeMCU
Bluetooth vcc ->(3.3v) of NodeMcu
Bluetooth gnd ->(gnd) of NodeMcu
Finally , All set up finish. Open Bluetooth terminal app .Scan and Connect Bluetooth.
Send Command in ASIC format

Program:

```
const int led1=D0;
char incomingByte = 0; // for incoming serial data
void setup()
        {
        Serial.begin(9600); // opens serial port, sets data rate to 9600 bps
        pinMode(led1,OUTPUT);
        Serial.println(" .");
        Serial.println("you have entered 1 LED ON , 0 LED OFF");
        }
void loop()
        {
        // send data only when you receive data:
        if (Serial.available())
        {
        // read the incoming byte:
        incomingByte = Serial.read();
        Serial.println(incomingByte);
        if((incomingByte=='1'))
        {
        Serial.println("you have entered 1 LED ON ");
        digitalWrite(led1,HIGH);
        }
        if((incomingByte=='0'))
        {
        Serial.println("you have entered 0 LED OFF ");
        digitalWrite(led1,LOW);
        }
        }
        }
```

Output: LED will be ON when 1 enter in Bluetooth app, LED will be OFF when 0 enter in Bluetooth app.

Exercise 6c: Hard serial port

Aim: Display your name using Serial monitor through hard serial port

Requirements: NodeMCU and Connecting wires

Procedure: Run following Program. Open Serial Monitor

Program:

```
void setup()
{
Serial.begin(9600);
delay(20);
}
void loop()
{
Serial.println(“ AMRITA “);
delay(2000);
}
```

Output: AMRITA will be printed in Serial Monitor.

Exercise 6d: Soft serial port

Aim: Display your name using serial monitor through soft serial port

Requirements: NodeMCU and Connecting Wires

Procedure: Connect NodeMCU Digital Pin D7 and D8 to bluetooth as Rx and Tx

Program:

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(D7, D8); // RX, TX
void setup()
{
mySerial.begin(9600);
delay(20);
}
void loop()
{
```

```
        mySerial.println("AMRITA ");
        delay(200);
        }
```

Output: AMRITA will be printed in Serial Monitor.

Exercise 6e: Bi-directional Communication

Aim: Bidirectional data Communication using BTM (Bluetooth Module)

Requirements: Bluetooth, Bluetooth Termonal app,LED, Nodemcu, connecting wires

Procedure: Blue Tooth Module HC-05 is interfaced to NodeMCU through Soft Serial.

BlueTooth Rx -> D7 of NodeMCU
BlueTooth Tx -> D8 of NodeMCU
POT connected to A0 and LED connected to D0
Analog Value at A0 will be sent to mobile phone through Blue Tooth.
LED connected to Digital Output DO will be controlled (ON/OFF) by sending a Character (A to ON and a to OFF) from Mobile phone through Blue Tooth

Program:

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(D7, D8); // RX, TX
char c;
#define LED D0
void setup()
        {
        // Open serial communications and wait for port to open:
        Serial.begin(9600);
        while (!Serial)
        {
        ; // wait for serial port to connect. Needed for native USB port only
        }
        pinMode(LED,OUTPUT);
        Serial.println("Goodnight moon!");
        // set the data rate for the SoftwareSerial port
        mySerial.begin(9600);
        mySerial.println("Hello, world?");
        }
void loop()
        {
        // run over and over
        //Serial.println("HI...");
        int a =analogRead(A0);
        mySerial.println(a);
```

```
delay(200);
if (mySerial.available())
{
c = mySerial.read();
if(c == ‘A’) //ASCII mode in app
digitalWrite(LED,HIGH);
if(c == ‘a’)
digitalWrite(LED,LOW);
}
}
```

Output:

In your Android Mobile phone, install BlueTooth terminal HC- 05 available in Google play store.
Select the paired device ASCBT1. (BlueTooth module HC-05 is already renamed as RiyassaBT1 and it is paired with our mobile phone – default pairing code is 1234)
Run the BlueTooth terminal HC-05 App and select RiyassaBT1 as the device, you will be able to see the value of Analog POT in The terminal of BT HC-05 App.
By sending character A, LED at port D0 of NodeMCU will be switched ON.
By sending character a, LED at port D0 of NodeMCU will be switched OFF.
Both Transmission and reception through Bluetooth interface of NodeMCU can be understood through this program.

Exercise 6f: SPI Protocol

Aim: To connect ADC channel using MCP3008 Analog IC through SPI protocol

Requirements: Any 1 Sensor, MCP3008 Analog IC, Nodemcu, connecting wires.

Procedure: NodeMCU has one Analog channel. So ,To increase the Analog Channel connect MCP3008 Analog IC to NodeMCU

Download Zip file: https://github.com/adafruit/Adafruit_MCP3008 Add .zip fle in Arduino IDE

Connect following SPI Pins MCP3008 to NodeMCU

MCP3008	**NodeMCU**
16 vcc	3.3V
14 gnd	gnd
13 clk	D5
12 Dout	D6
11 Din	D7
10 CS	D8

Program:

```
#include <MCP3008.h>
#include <SPI.h>
// put pins inside MCP3008 constructor
//MCP3008 adc(CLOCK_PIN, MOSI_PIN, MISO_PIN, CS_PIN);
MCP3008 adc(D5,D7,D6,D8);
void setup()
        {
        // open serial port
        Serial.begin(9600);
        }
void loop()
        {
        /*int val = adc.readADC(0); // read Chanel 0 from MCP3008 ADC
        Serial.println(val);
        */
        Serial.println("value 0 :" + String(adc.readADC(0)));
        Serial.println("value 1 :" + String(adc.readADC(1)));
        Serial.println("value 2:" + String(adc.readADC(2)));
        Serial.println("value 3:" + String(adc.readADC(3)));
        Serial.println("value 4:" + String(adc.readADC(4)));
        Serial.println("value 5:" + String(adc.readADC(5)));
        Serial.println("value :" + String(adc.readADC(6)));
        delay(4000);
        }
```

Output: Using SPI Protocol read Analog data from MCP3008 IC to NodeMCU

7. Web and Cloud Programming

Exercise 7a: WiFi scan

Aim: To Scan the available WiFi Network through NodeMCU

Requirements: Nodemcu, connecting wires,WiFi Network

Procedure: Run following Program. Open Serial Monitor

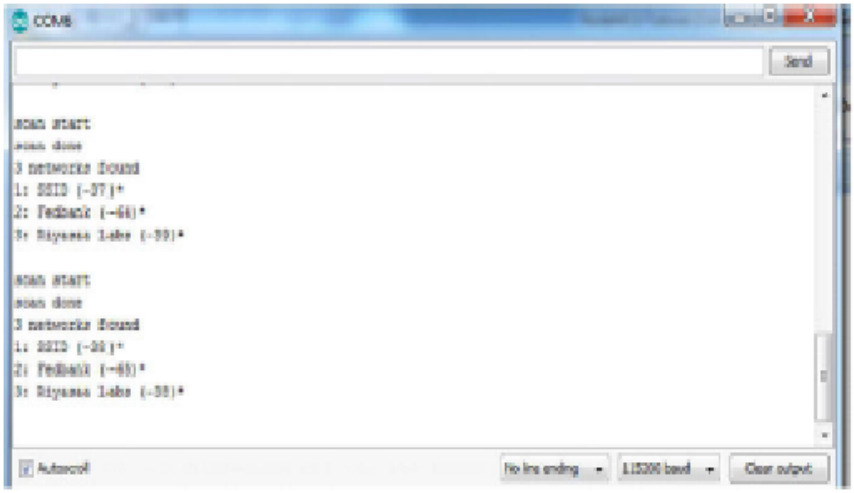
Program:

```
#include “ESP8266WiFi.h”
void setup()
        {
        Serial.begin(115200);
        // Set WiFi to station mode and disconnect from an AP if it was previously connected
        WiFi.mode(WIFI_STA);
        WiFi.disconnect();
        delay(100);
        Serial.println(“Setup done”);
        }
void loop()
        {
        Serial.println(“scan start”);
        // WiFi.scanNetworks will return the number of networks found
        int n = WiFi.scanNetworks();
        Serial.println(“scan done”);
        if (n == 0) { Serial.println(“no networks found”);
        }
        else
        {
         Serial.print(n);
        Serial.println(“ networks found”);
        for (int i = 0; i < n; ++i)
        {
        // Print SSID and RSSI for each network found
        Serial.print(i + 1);
        Serial.print(“: “);
        Serial.print(WiFi.SSID(i));
        Serial.print(“ (“);
        Serial.print(WiFi.RSSI(i));
        Serial.print(“)”);
        Serial.println((WiFi.encryptionType(i) == ENC_TYPE_NONE) ? “ “ : “*”);
        delay(10);
        }
```

```
}
Serial.println("");
// Wait a bit before scanning again
delay(5000);
}
```

(Note that COM Port is configured to work at baud rate of 115200)

Output: Open Serial Monitor,WiFiScan Network display.



Exercise 7b: WiFi scan and Connect

Aim: To Scan and connect the specific WiFi Network

Requirements: Nodemcu, connecting wires,WiFi Network

Procedure: Run following Program. Before that replace SSID and PASSWORD for your Network in Program
Open Serial Monitor

Program:

```
#include<ESP8266WiFi.h>
/************** WiFi Access Point ************************/
#define WLAN_SSID "Riyasaa labs"
#define WLAN_PASS "riyasaa54321"
WiFiServer server( 80);
void setup()
        {
        Serial.begin(9600);
        delay(10);
        Serial.println();
```

```
        Serial.println();
        Serial.print("Connecting to ");
        Serial.println(WLAN_SSID);
        delay(500);
        WiFi.begin(WLAN_SSID, WLAN_PASS);
        while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
        }
        Serial.println();
        server.begin();
        Serial.println("WiFi connected");
        Serial.println("IP address: ");
        Serial.println(WiFi.localIP());
        }
void loop()
        {
        Serial.print("IP address: ");
        Serial.println(WiFi.localIP());
        delay(5000);
        }
```

Output:

(Note that COM Port is configured to work at baud rate of 9600)

Exercise 7c: Control AC units using Webpage

Aim: To Control AC units through Webpage input

Requirements: LED, Nodemcu, connecting wires,WiFi Network

Procedure: Run following Program.. Before that replace SSID and PASSWORD for your Network in Program.
Open Serial MonitorLocal IP address display.Open
Browser Type IP address. Webpage Will be appeared.
In that To control Your LED

Program:

```
#include<ESP8266WiFi.h>
/************************* WiFi Access Point *********************************/
#define WLAN_SSID “Iot labs”
#define WLAN_PASS “Iot54321”
#define light “off”
WiFiServer server( 80);
void setup()
        {
        Serial.begin(115200);
        delay(10);
        pinMode( D2, OUTPUT); // D2 in nodemcu LIGHT 1
        pinMode( D7, OUTPUT); // D7 in nodemcu LIGHT 2
        digitalWrite( D2, LOW);
        digitalWrite( D7, LOW);
        Serial.println(); Serial.println();
        Serial.print(“Connecting to “);
        Serial.println(WLAN_SSID);
        WiFi.begin(WLAN_SSID, WLAN_PASS);
        while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(“.”);
        }
        Serial.println();
        server.begin();
        Serial.println(“WiFi connected”);
        Serial.println(“IP address: “); Serial.println(WiFi.localIP());
        }
void loop()
        {
        WiFiClient client = server.available();
        if (! client) { return;
        }
```

```
Serial.println(“ new client”);
while(! client.available())
{
 delay( 1);
}
// Read the first line of the request
String request = client.readStringUntil(‘\ r’);
Serial.println( request);
client.flush(); // Match the request
if (request.indexOf(“l1on”) > 0)
{
digitalWrite( D2, HIGH);
Serial.println(“ Light1 on”);
}
if (request.indexOf(“l1off”) > 0)
{
digitalWrite( D2, LOW);
Serial.println(“Light 1 off”);
}
if (request.indexOf(“l2on”) > 0)
{
digitalWrite( D7, HIGH);
Serial.println(“LIGHT 2 on”);
}
if (request.indexOf(“l2off”) > 0)
{
digitalWrite( D7, LOW);
Serial.println(“Light 2 off”);
}
client.println(“ HTTP/ 1.1 200 OK”);
client.println(“ Content-Type: text/ html”);
client.println(“”); // do not forget this one
client.println(“ <!DOCTYPE HTML>”);
client.println(“ <html>”);
client.println(“ <head>”);
client.println(“ </head>”);
//client.println(“ <body bgcolor = \”#f7e6ec\”>”);
client.println(“ <hr/><hr>”);
client.println(“ <h4><center> RIYASAA LABS </center></h4>”);
client.println(“ <hr/><hr>”);
client.println(“ <br><br>”);
client.println(“ <br><br>”);
client.println(“ <center>”);
client.println(“ Light1”);
//client.println(“<button onclick=\”funt1()\”> one</button> “);
//client.println(“<script> funtion funt1()”);
```

```
if (digitalRead(D2))// read digital pin 5
{
client.println(“ <a href =\”/ l1on\”\”><button style =\”backgroundcolor:green\”> Turn On
</button></a>”);
client.println(“ <a href =\”/ l1off\”\”><button> Turn Off </button></ a><br/>”);
}
else
{
client.println(“ <a href =\”/ l1on\”\”><button > Turn On </button></a>”);
client.println(“ <a href =\”/ l1off\”\”><button style =\”backgroundcolor: red\”> Turn Off
</button></a><br/>”);
}
client.println(“ </center>”);
client.println(“ <br><br>”);
client.println(“ <center>”);
client.println(“ Light 2”);
if (digitalRead(D7))// read digital pin 4
{
client.println(“ <a href =\”/ l2on\”\”><button style=\”backgroundcolor:green\”> Turn On
</button></a>”);
client.println(“ <a href =\”/ l2off\”\”><button> Turn Off </button></a><br/>”);
}
else
{
client.println(“ <a href =\”/ l2on\”\”><button > Turn On </button></a>”);
client.println(“ <a href =\”/ l2off\”\”><button style=\”backgroundcolor:red\”> Turn Off
</button></a><br/>”);
}
client.println(“ </center>”);
client.println(“ <br><br>”);
client.println(“ <center>”);
client.println(“ <table border =\” 5\”>”);
client.println(“ <tr>”);
if (digitalRead(D2))// read digital pin 5
{
        client.print(“ <td> Light 1 is ON </td>”); }
        else
{
         client.print(“ <td> Light 1 is OFF </td>”); }
        if (digitalRead(D7))// read digital pin 4
{
        client.print(“ <td> Light 2 is ON </td>”); }
        else
{
client.print(“ <td> Light 2 is OFF </td>”); }
client.println(“ </tr>”);
```

```
client.println(" </table>");
client.println(" </center>");
client.println(" </html>");
delay( 1);
Serial.println(" Client disonnected");
Serial.println("");
}
```

Output: Webpage Will be displayed in that control your LED through ON/OFF

Exercise 7d: Environmental Data in Webpage

Aim: Displaying Environmental data in Webpage

Requirements: POT, Nodemcu, connecting wires, WiFi Network

Procedure: Run following Program,. Before that replace SSID and PASSWORD for your Network in Program.
Open Serial Monitor
Local IP address display. Open Browser Type IP address. Webpage Will be appeared.
In that page your sensor data update periodically.

Program:

```
#include <ESP8266WiFi.h>
const char* ssid = "Riyasaa labs";
const char* password = "riyasaa54321";
WiFiServer server(80);
void setup()
{
Serial.begin(115200);
Serial.println();
Serial.printf("Connecting to %s ", ssid);
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED)
{
delay(500);
Serial.print(".");
}
Serial.println(" connected");
server.begin();
Serial.printf("Web server started, open %s in a web browser\n",
WiFi.localIP().toString().c_str());
}
// prepare a web page to be send to a client (web browser)
String prepareHtmlPage()
```

```
        {
        String htmlPage =
        String("HTTP/1.1 200 OK\r\n") +
        "Content-Type: text/html\r\n" +
        "Connection: close\r\n" + // the connection will be closed after completion of the
        response
        "Refresh: 5\r\n" + // refresh the page automatically every 5 sec
        "\r\n" +
        "<!DOCTYPE HTML>" +
        "<html>" +
        "<h1>Welcome to Riyasaa Labs 1</h1>"+
        "Analog input POT Value : " + String(analogRead(A0)) +
        "</html>" +
        "\r\n";
        return htmlPage;
        }
void loop()
        {
        WiFiClient client = server.available();
        // wait for a client (web browser) to connect
        if (client)
        {
        Serial.println("\n[Client connected]");
        while (client.connected())
        {
        // read line by line what the client (web browser) is requesting
        if (client.available())
        {
        String line = client.readStringUntil('\r');
        Serial.print(line);
        // wait for end of client's request, that is marked with an empty line
        if (line.length() == 1 && line[0] == '\n')
        {
        client.println(prepareHtmlPage());
        break;
        }
        }
        }
        delay(1); // give the web browser time to receive the data
        // close the connection:
        client.stop();
        Serial.println("[Client disonnected]");
        }
        }
```

Output: Webpage Will be displayed in that control your LED through ON/OFF

Exercise 7e: Thingspeak Cloud

Aim: Uploading Environmental Data to cloud and Visualized through Thingspeak

Requirements: POT, Nodemcu, connecting wires,WiFi Network, Thingspeak account

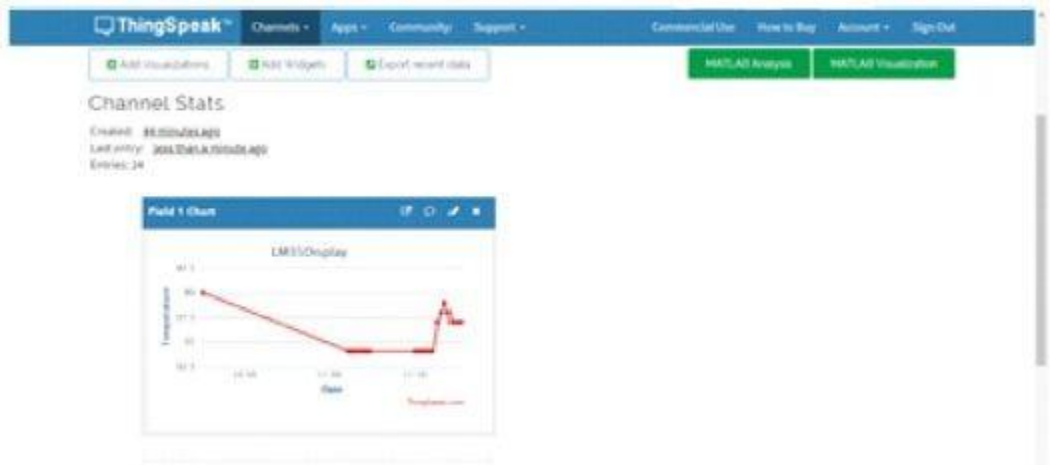
Procedure: Login Thingspeak account, Create channel and choose field, copy Write API Key, Paste it in your program.
Run following Program.. Before that replace SSID and PASSWORD for your Network in Program,Open Serial Monitor.

Program:

```
#include <ESP8266WiFi.h>
String apiKey = "W7WZLXU0GVYVPM9K"; // Enter your Write API key from ThingSpeak
const char *ssid = "Riyasaa labs"; // replace with your wifi ssid and wpa2 key
const char *pass = "riyasaa54321";
const char* server = "api.thingspeak.com";
WiFiClient client;
void setup()
        {
        Serial.begin(115200);
        delay(10);
        Serial.println("Connecting to ");
        Serial.println(ssid);
        WiFi.begin(ssid, pass);
        while (WiFi.status() != WL_CONNECTED)
        {
        delay(500);
        Serial.print(".");
        }
        Serial.println("");
        Serial.println("WiFi connected");
        }
```

```
void loop()
        {
        float t = analogRead(A0);
        if (client.connect(server,80)) // “184.106.153.149” or api.thingspeak.com
        {
        String postStr = apiKey;
        postStr +=”&field1=”;
        postStr += String(t);
        //postStr +=”&field2=”;
        // postStr += String(h);
        postStr += “\r\n\r\n”;
        client.print(“POST /update HTTP/1.1\n”);
        client.print(“Host: api.thingspeak.com\n”);
        client.print(“Connection: close\n”);
        client.print(“X-THINGSPEAKAPIKEY: “+apiKey+”\n”);
        client.print(“Content-Type: application/x-www-form-urlencoded\n”);
        client.print(“Content-Length: “);
        client.print(postStr.length());
        client.print(“\n\n”);
        client.print(postStr);
        Serial.print(“Temperature: “);
        Serial.print(t);
        //Serial.print(“ degrees Celcius, Humidity: “);
        // Serial.print(h);
        Serial.println(“%. Send to Thingspeak.”);
        }
        client.stop();
        Serial.println(“Waiting...”);
        delay(10000);
        }
```

Output: POT /sensor data stored in cloud Platform



Exercise 7f: Home Automation using Voice Control (GA)

Aim: To Develop Smart home automation Control using Google Assistance

Requirements: Relay,LED,Nodemcu, connecting wires,WiFi Network,IFTTT accounthttps://ifttt.com/,Ada fruit Accounthttps://io.adafruit.com/

Procedure : Login IFTTT account, Create trigger and Action,and Login Ada fruit account,create action, copy Write API Key, Paste it in Your program
Run following Program. Before that replace SSID and
PASSWORD for your Network in Program, Open Serial Monitor.
Then Open google Assistance.
Talk to Google Assistance.

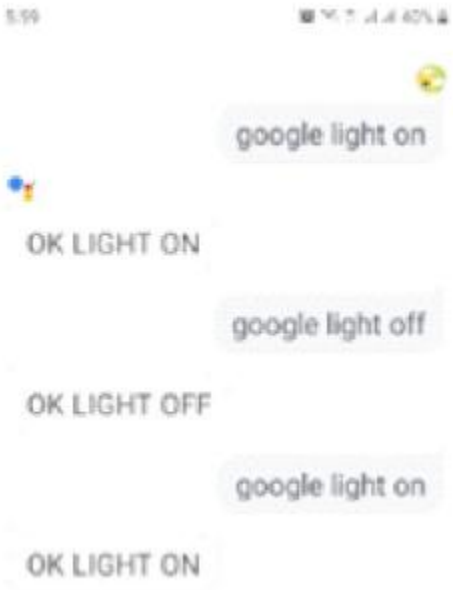
Program:

```
#include <ESP8266WiFi.h>
#include “Adafruit_MQTT.h”
#include “Adafruit_MQTT_Client.h”
#define Relay1 D1
#define Relay2 D5
#define Relay3 D2
#define Relay4 D6
#define WLAN_SSID “Riyasaa labs” // Your SSID
#define WLAN_PASS “riyasaa54321” // Your password
/**************** Adafruit.io Setup *********************/
#define AIO_SERVER “io.adafruit.com”
#define AIO_SERVERPORT 1883 // use 8883 for SSL
#define AIO_USERNAME “ARUNV_25” // Replace it with your username
#define AIO_KEY “8500637f6fe4480397314e1c2acd650a” / / Replace with your Project Auth Key
/************ Global State (you don’t need to change this!) ******************/
// Create an ESP8266 WiFiClient class to connect to the MQTT server.
WiFiClient client;
// or... use WiFiFlientSecure for SSL
//WiFiClientSecure client;
// Setup the MQTT client class by passing in the WiFi client and MQTT
server and login details.
Adafruit_MQTT_Client mqtt(&client, AIO_SERVER,
AIO_SERVERPORT, AIO_USERNAME, AIO_KEY);
/********************** Feeds ***************************/
// Setup a feed called ‘onoff’ for subscribing to changes.
Adafruit_MQTT_Subscribe Light1 = Adafruit_MQTT_
Subscribe(&mqtt, AIO_USERNAME”/feeds/Relay1"); // FeedName
Adafruit_MQTT_Subscribe Light2 = Adafruit_MQTT_ Subscribe
(&mqtt, AIO_USERNAME “/feeds/Relay2”);
```

```
Adafruit_MQTT_Subscribe Light3 = Adafruit_MQTT_ Subscribe
(&mqtt, AIO_USERNAME "/feeds/Relay3");
Adafruit_MQTT_Subscribe Light4 = Adafruit_MQTT_ Subscribe
(&mqtt, AIO_USERNAME "/feeds/Relay4");
void MQTT_connect();
void setup()
        {
        Serial.begin(115200);
        pinMode(Relay1, OUTPUT);
        pinMode(Relay2, OUTPUT);
        pinMode(Relay3, OUTPUT);
        pinMode(Relay4, OUTPUT);
        // Connect to WiFi access point.
        Serial.println(); Serial.println();
        Serial.print("Connecting to ");
        Serial.println(WLAN_SSID);
        WiFi.begin(WLAN_SSID, WLAN_PASS);
        while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
        }
        Serial.println();
        Serial.println("WiFi connected");
        Serial.println("IP address: ");
        Serial.println(WiFi.localIP());
        // Setup MQTT subscription for onoff feed.
        mqtt.subscribe(&Light1);
        mqtt.subscribe(&Light3);
        mqtt.subscribe(&Light2);
        mqtt.subscribe(&Light4);
        }
void loop()
        {
        MQTT_connect();
        Adafruit_MQTT_Subscribe *subscription;
        while ((subscription = mqtt.readSubscription(20000)))
        {
        if (subscription == &Light1) {
        Serial.print(F("Got: "));
        Serial.println((char *)Light1.lastread);
        int Light1_State = atoi((char *)Light1.lastread);
        digitalWrite(Relay1, Light1_State);
        }
        if (subscription == &Light2)
        {
        Serial.print(F("Got: "));
```

```
        Serial.println((char *)Light2.lastread);
        int Light2_State = atoi((char *)Light2.lastread);
        digitalWrite(Relay2, Light2_State);
        }
        if (subscription == &Light3)
        {
        Serial.print(F("Got: "));
        Serial.println((char *)Light3.lastread);
        int Light3_State = atoi((char *)Light3.lastread);
        digitalWrite(Relay3, Light3_State);
        }
        if (subscription == &Light4)
        {
        Serial.print(F("Got: "));
        Serial.println((char *)Light4.lastread);
        int Light4_State = atoi((char *)Light4.lastread);
        digitalWrite(Relay4, Light4_State);
        }
        }
        }
void MQTT_connect()
        {
        int8_t ret;
        // Stop if already connected.
        if (mqtt.connected()) {
        return;
        }
        Serial.print("Connecting to MQTT... ");
        uint8_t retries = 3;
        while ((ret = mqtt.connect()) != 0) { // connect will return 0 for connected
        Serial.println(mqtt.connectErrorString(ret));
        Serial.println("Retrying MQTT connection in 5 seconds...");
        mqtt.disconnect();
        delay(5000); // wait 5 seconds
        retries—;
        if (retries == 0) {
        // basically die and wait for WDT to reset me
        while (1);
        }
        }
        Serial.println("MQTT Connected!");
        }
```

Output:

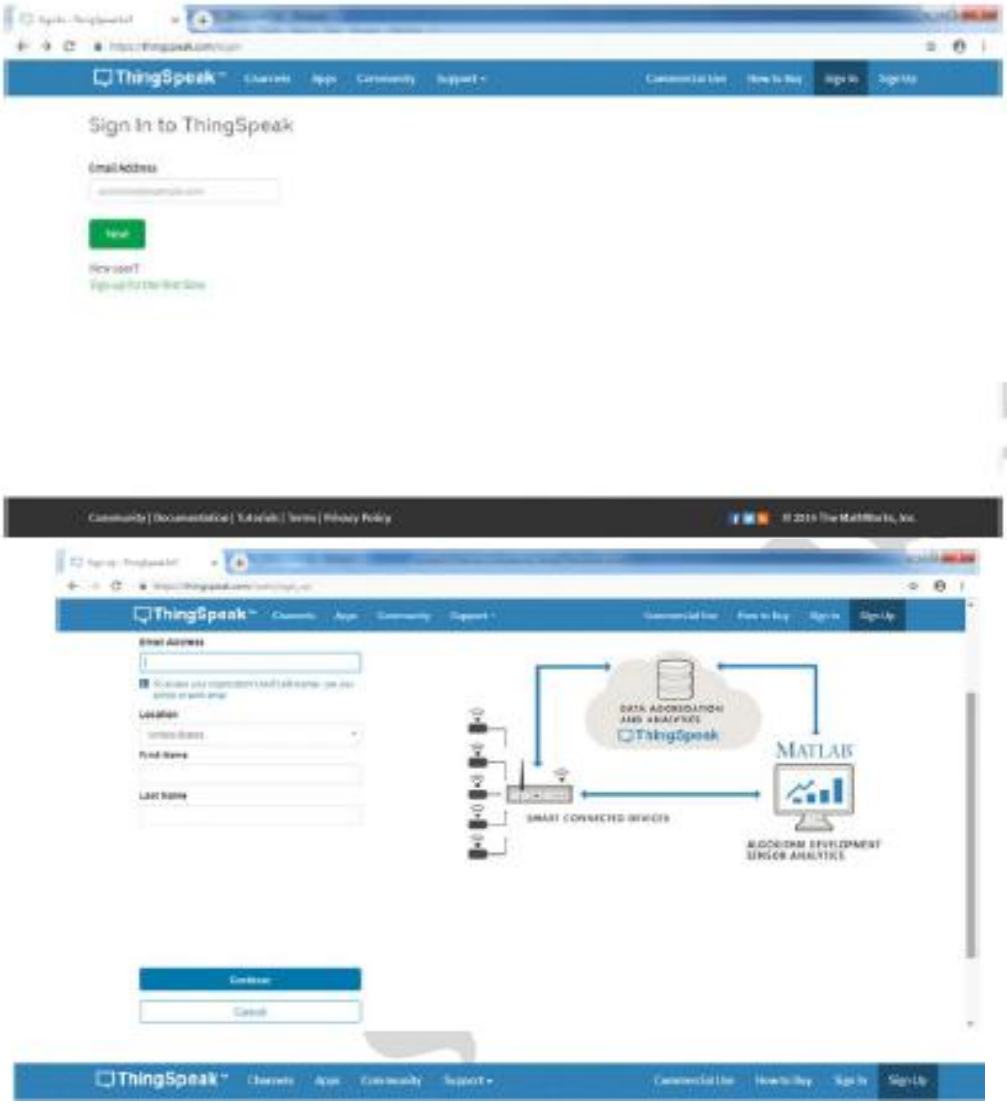

google light on
OK LIGHT ON
google light off
OK LIGHT OFF
google light on
OK LIGHT ON

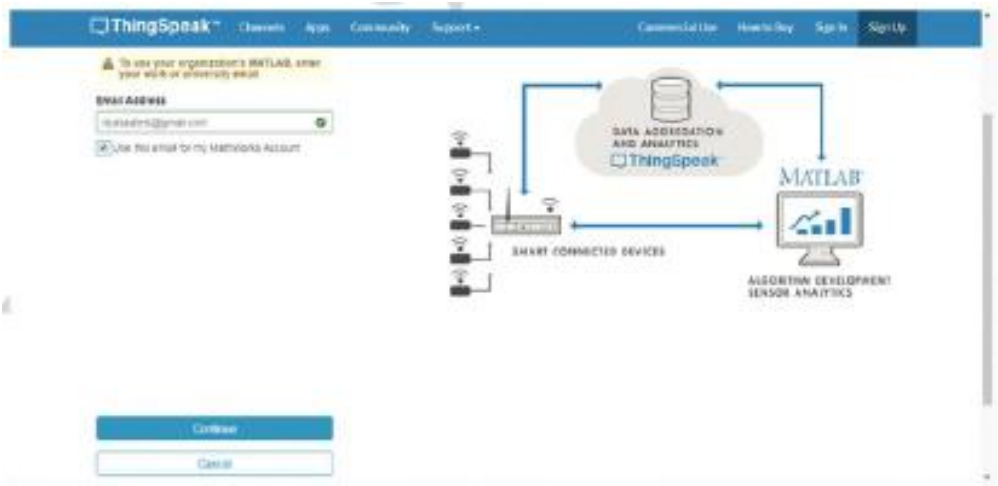
Appendix A

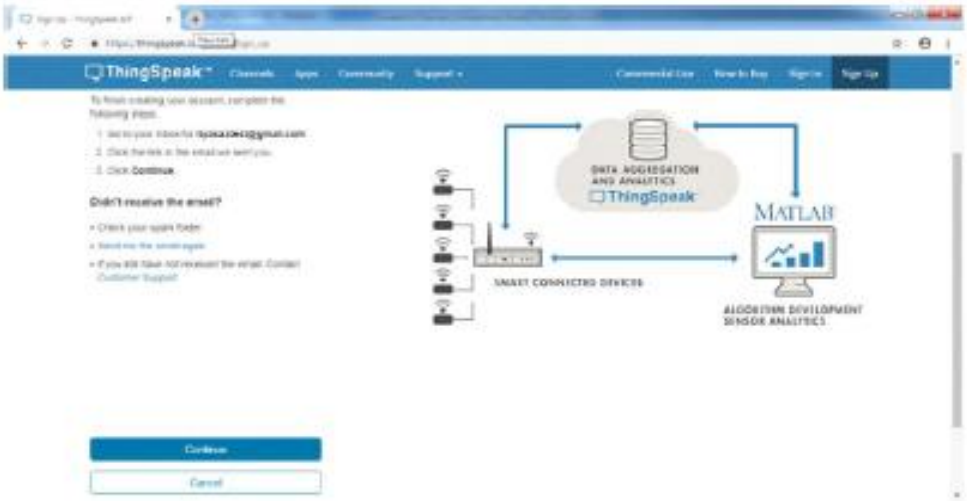
Procedure to visualize through Thingspeak

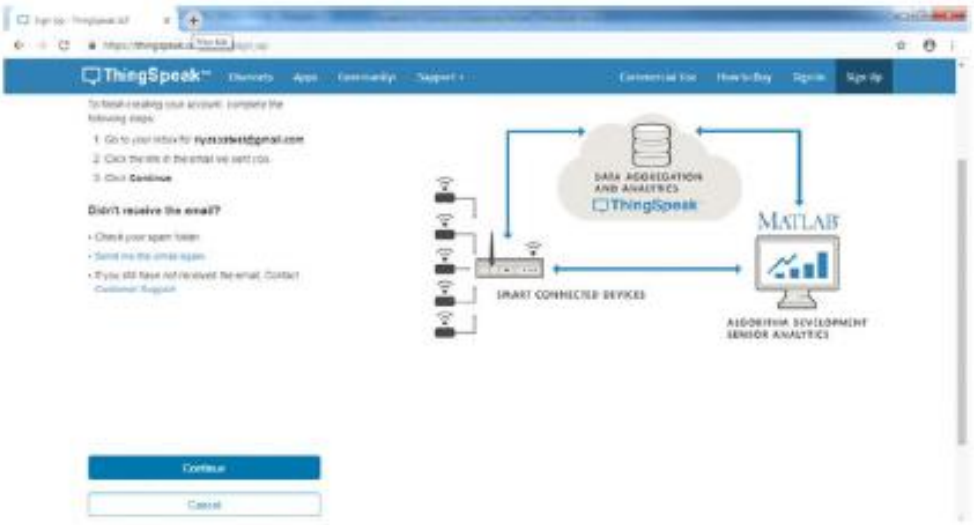
Thingspeak Procedure

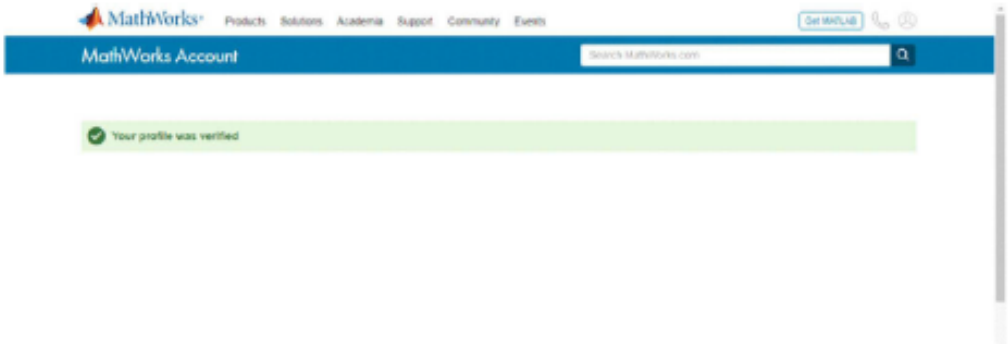
How can create Thingspeak account
Go to Thingspeak Websitehttps://thingspeak.com/
Click sign in and follow the Procedure Below

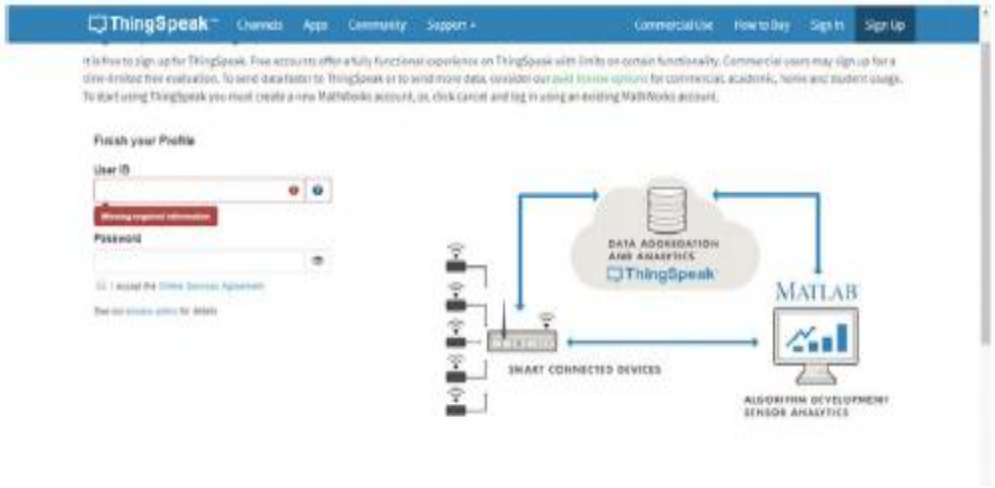


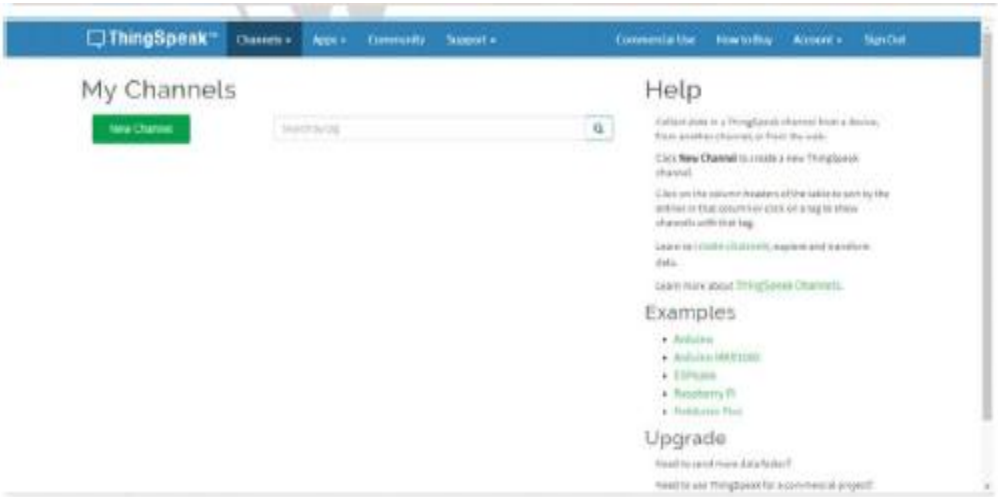

ThingSpeak
DATA AGGREGATION AND ANALYTICS
ThingSpeak
MATLAB
SMART CONNECTED DEVICES
ALGORITHM DEVELOPMENT SENSOR ANALYTICS
Continue
Cancel

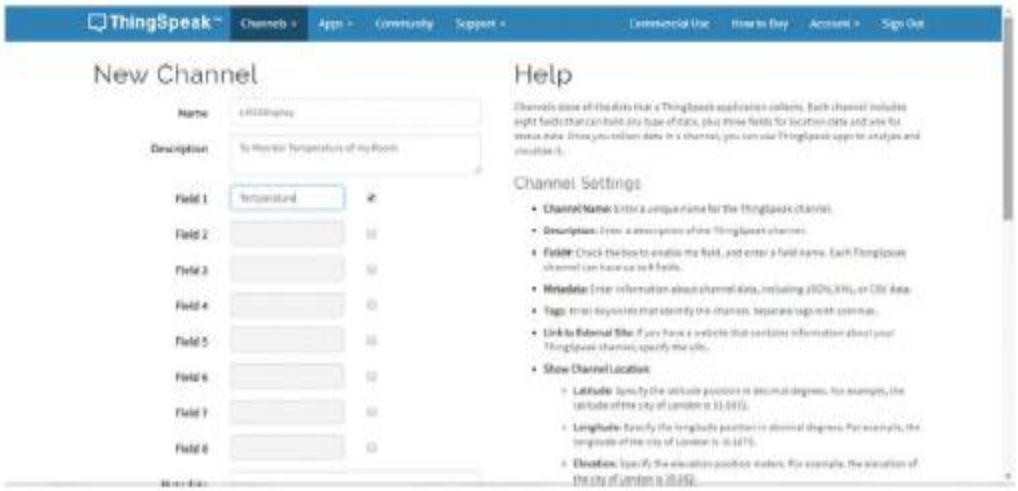

ThingSpeak
Didn't receive the email?
DATA AGGREGATION AND ANALYTICS
ThingSpeak
MATLAB
SMART CONNECTED DEVICES
ALGORITHM DEVELOPMENT SENSOR ANALYTICS
Continue
Cancel

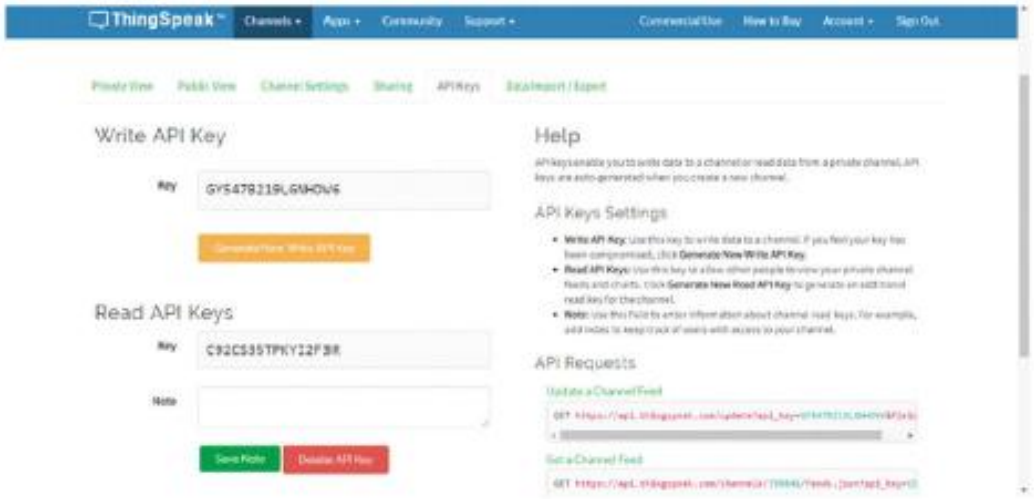

ThingSpeak
Didn't receive the email?
DATA AGGREGATION AND ANALYTICS
ThingSpeak
MATLAB
SMART CONNECTED DEVICES
ALGORITHM DEVELOPMENT SENSOR ANALYTICS
Continue
Cancel


MathWorks
Products
Solutions
Academia
Support
Community
Events
MathWorks Account
Your profile was verified


ThingSpeak
Channels
Apps
Community
Support
Commercial Use
How to Buy
Sign In
Sign Up
Finish your Profile
User ID
Password
DATA ACQUISITION AND ANALYTICS
ThingSpeak
MATLAB
SMART CONNECTED DEVICES
ALGORITHM DEVELOPMENT SENSOR ANALYTICS

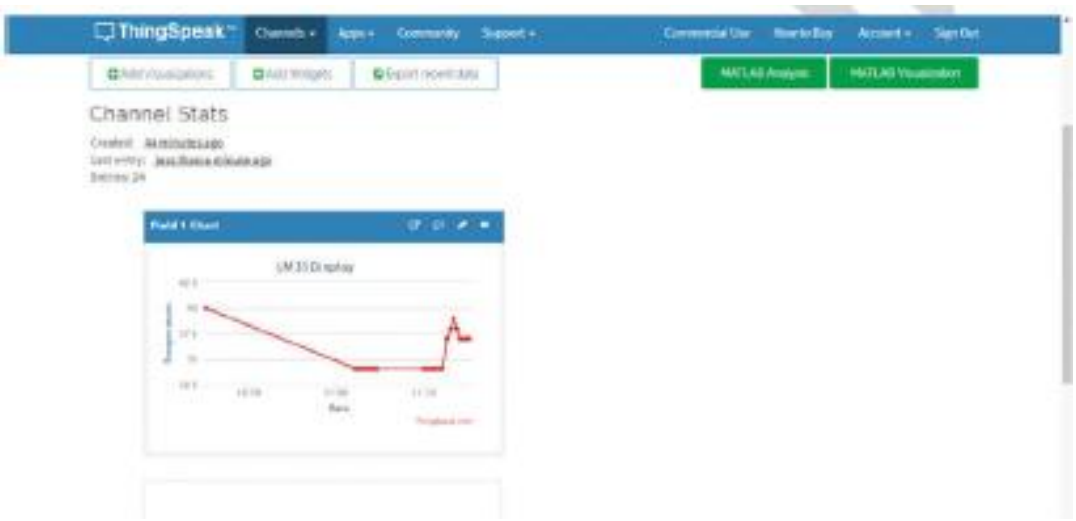

ThingSpeak
Channels
Apps
Community
Support
Commercial Use
How to Buy
Account
Sign Out
My Channels
New Channel
Help
Examples
Upgrade





Note: Copy your Channel write API Key and Read API Key

Write API Key: GY547B219LGNHOV6
Read API Key: C92CS35TPKYI2F3R


ThingSpeak
Channels
Apps
Community
Support
Commercial Use
How to Buy
Account
Sign Out
MATLAB Analysis
MATLAB Visualization
Channel Stats
Field 1 Chart
LM35 Display

Appendix B

Procedure to control using Voice Assistance

Adafruit Procedure and IFTTT procedure

Voice-controlled home automation using Google Assistance

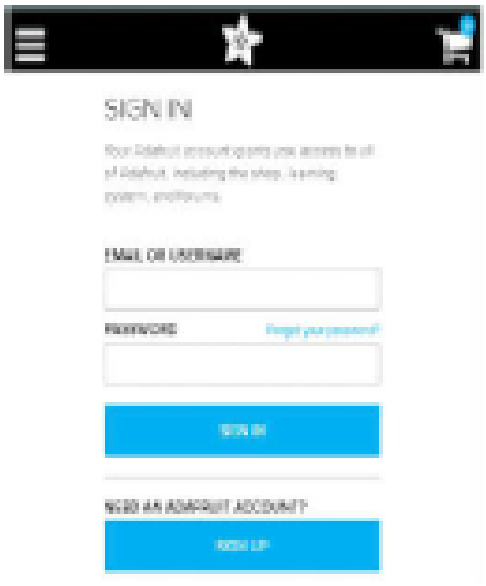
To build home automation application, I used three different platforms

- Google Assistant
- Adafruit
- IFTTT

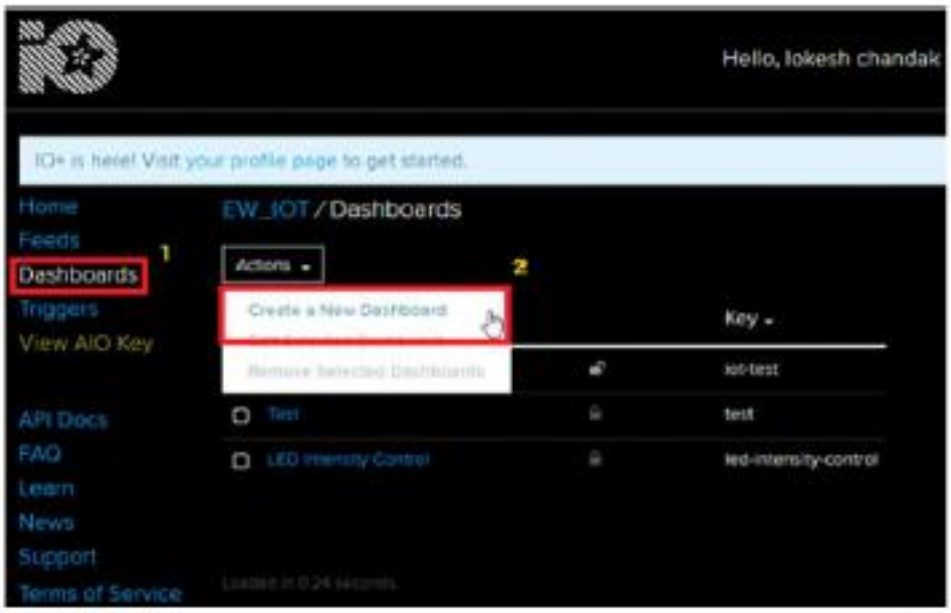
To use above services we need to configure them.

Adafruit Procedure

First, created account at www.Adafruit.io



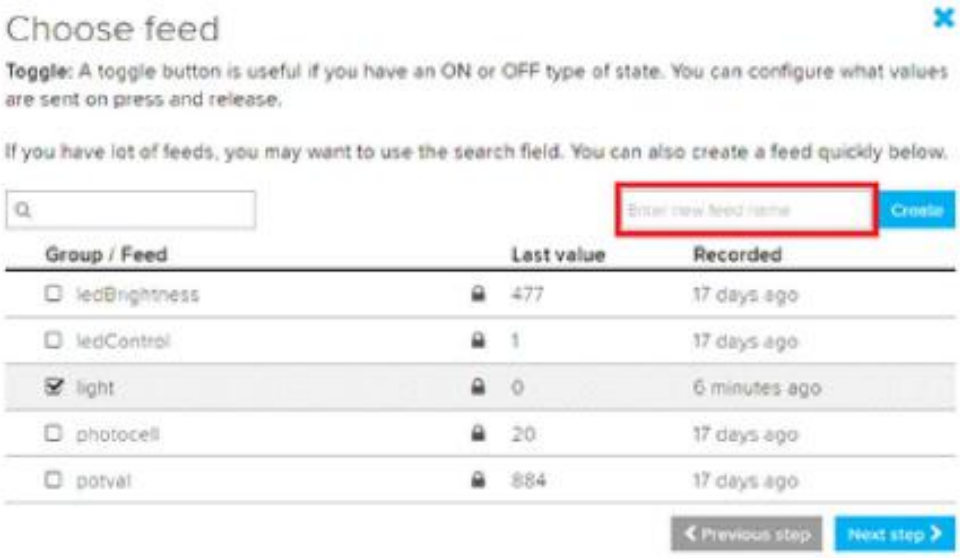
Now, create dashboard at Adafruit. This dashboard is a user interface to control things remotely.



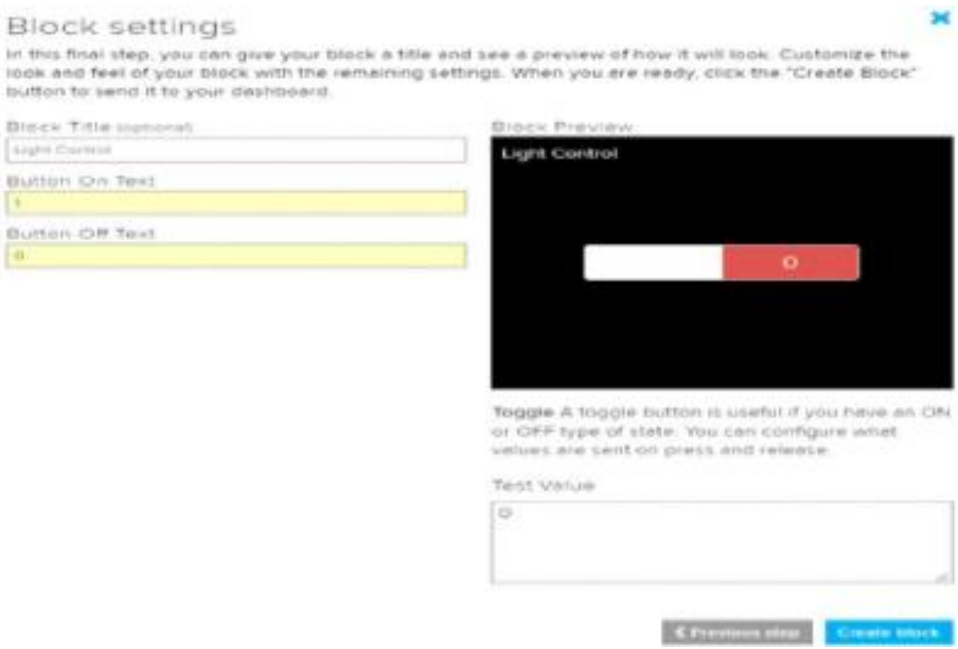
After following above steps, provide name to the dashboard and save it. We can see our dashboard as follows,

Now, create feed (user interface) to control light On-Off. To create it,just click on '+' symbol and select toggle feed shown below,

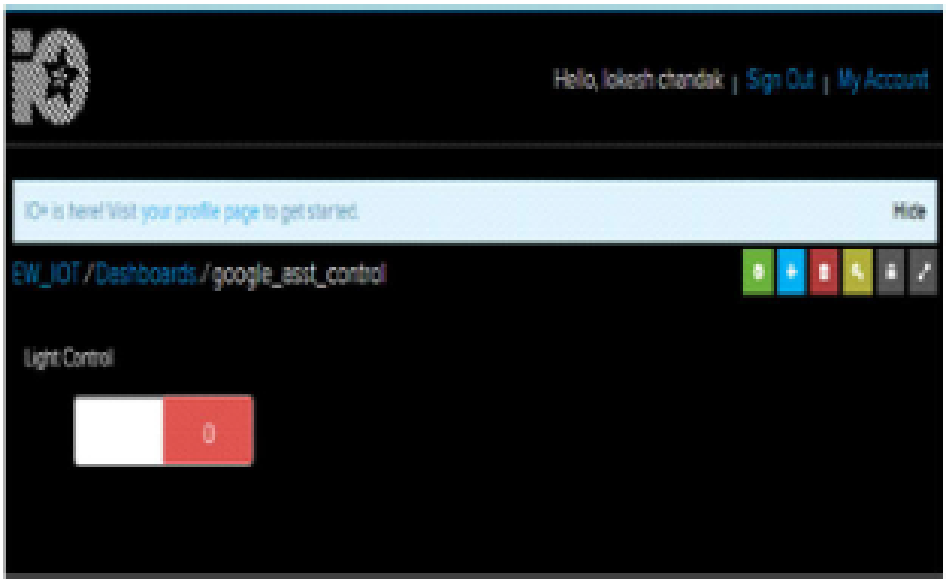
After selecting toggle feed, pop-up window appears as shown below.



Enter name of our feed (shown in red box) and create it. After creation, select the created feed (here mine is light) and then click on Next step.
In the next step configure the feed which is shown below,



Here, I used 0(OFF) and 1(ON) text for button and then click on create.
This will create toggle button on your dashboard which can be used to control things remotely.



Now, my dashboard is ready for IoT application like home automation.

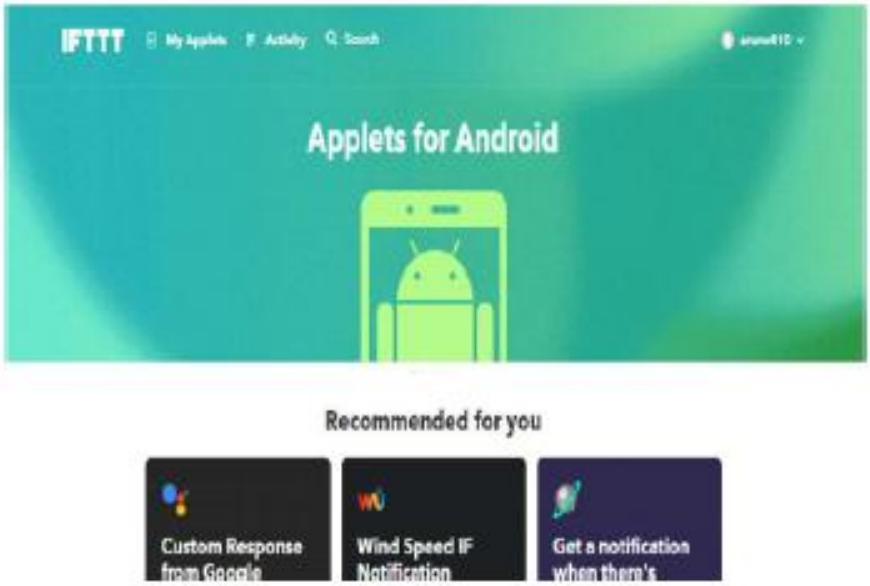
IFTTT Procedure

How can create IFTTT account?

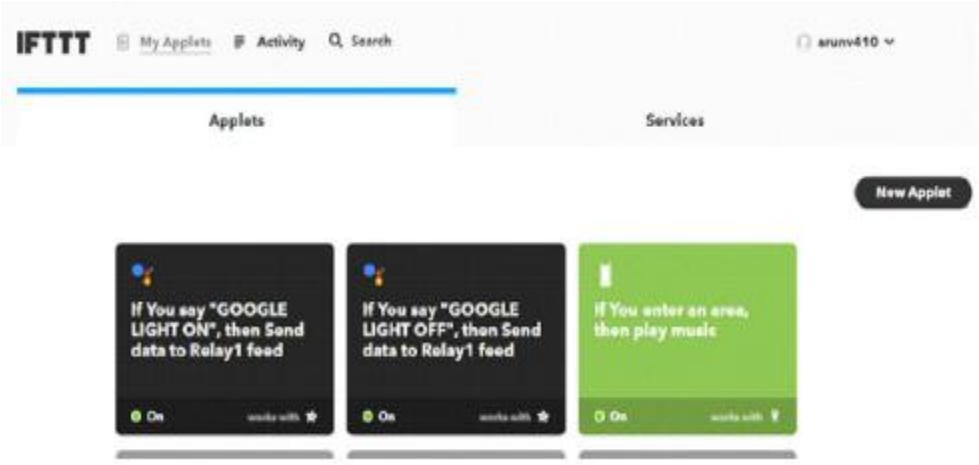
Go to website https://ifttt.com/
Click signup

Click continue with google
Select your email account
Verify and come back to IFTTT
Click sign in
Enter your Email Id and Password

Click My applets

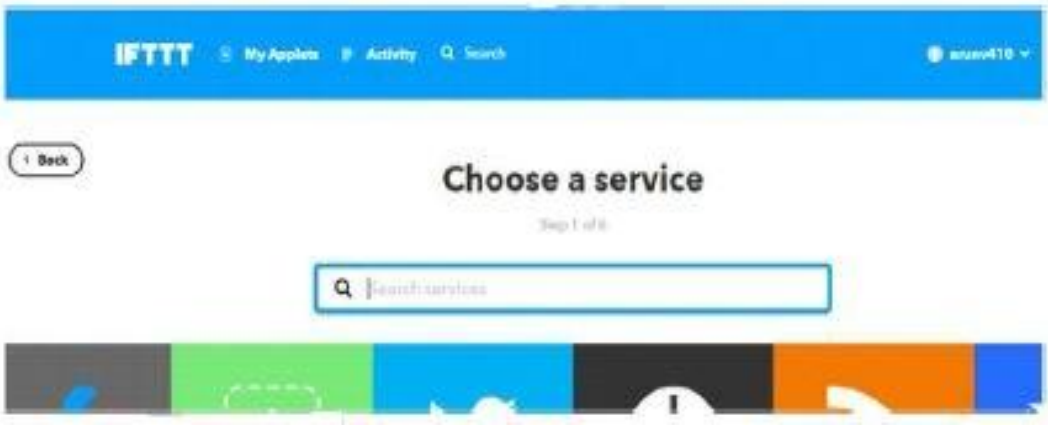


Click your New Applet

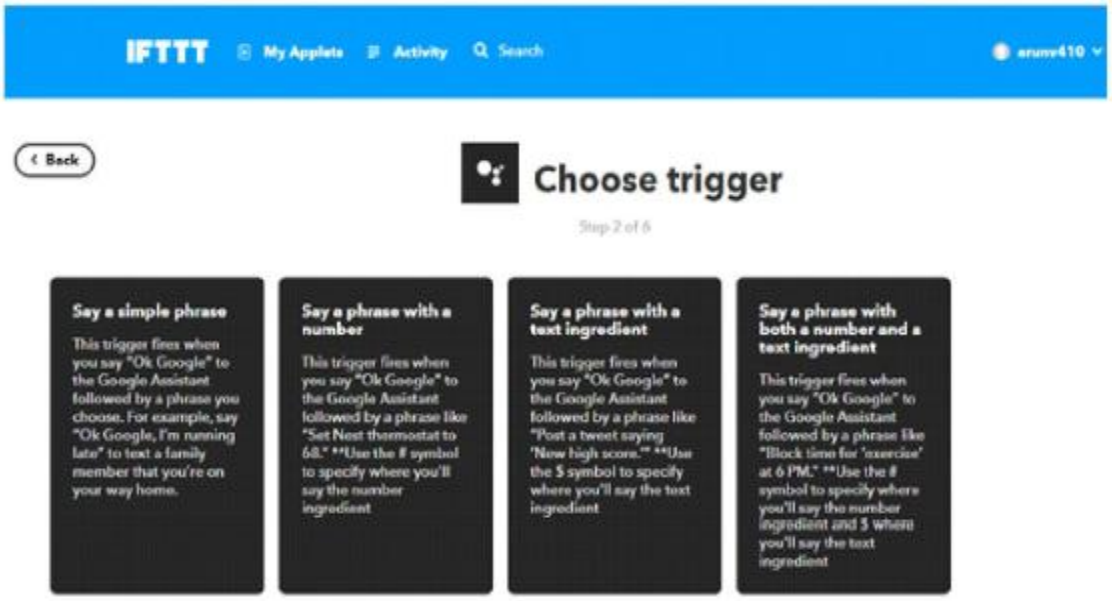


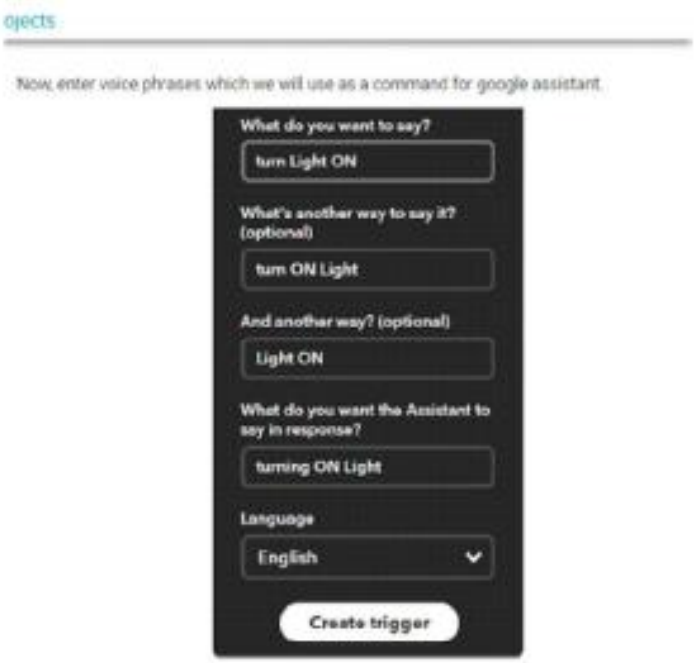
Click this

Choose service here select Google assistance



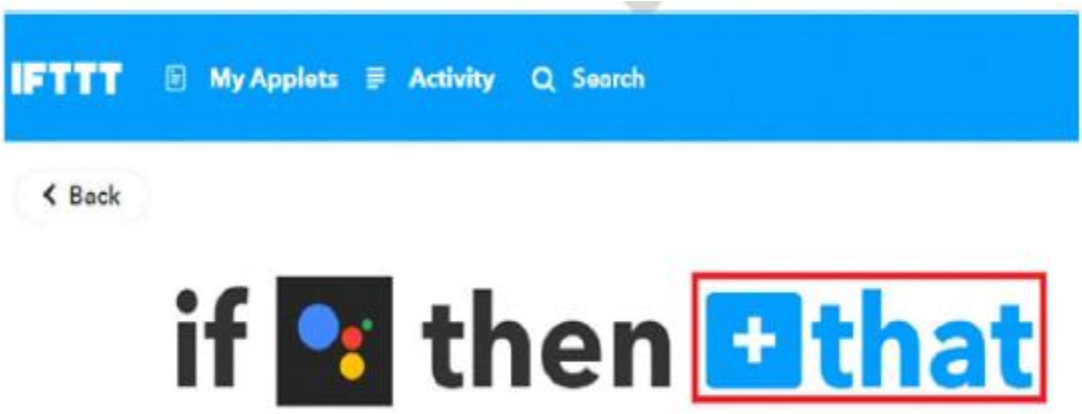
This page appear select Say a single Phrase



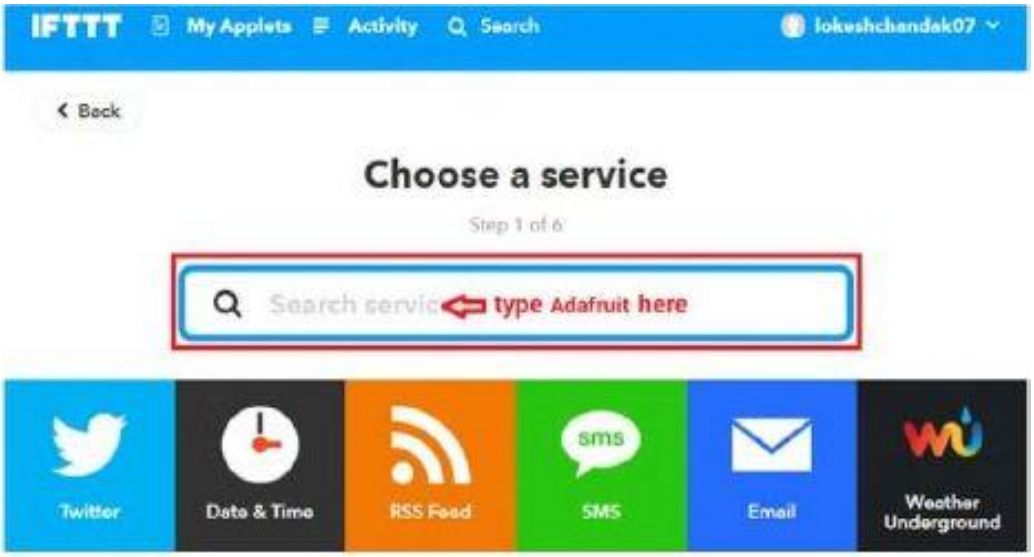


We can enter any phrase as per our application. As you can see, the phrases entered in the above fields is for making **Light ON.** For making **Light OFF**, we have to create another applet with different phrases.

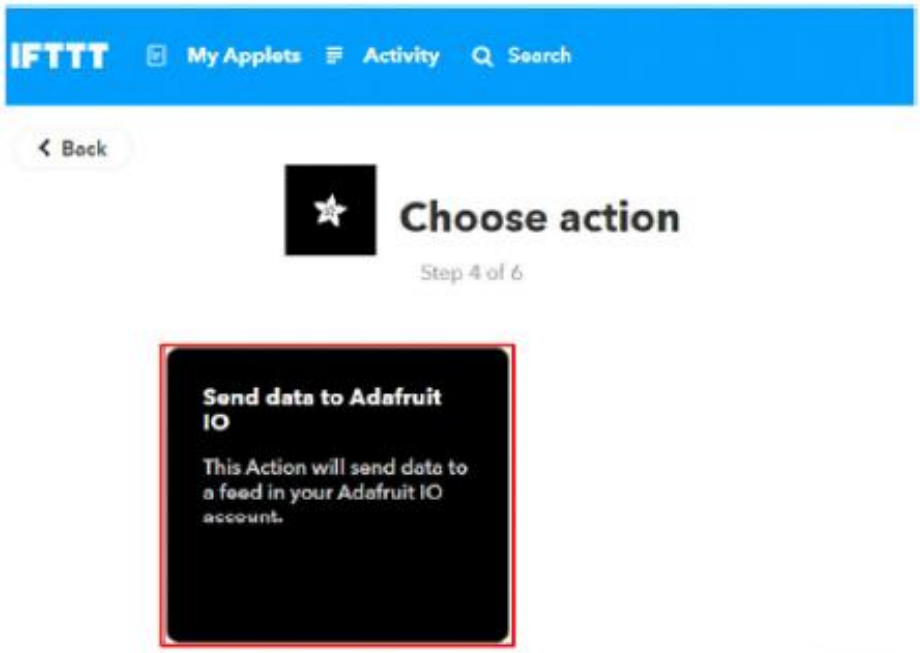
Now, we get another page on which we have to click on **that** option which is used to connect Google Assistant with Adafruit.



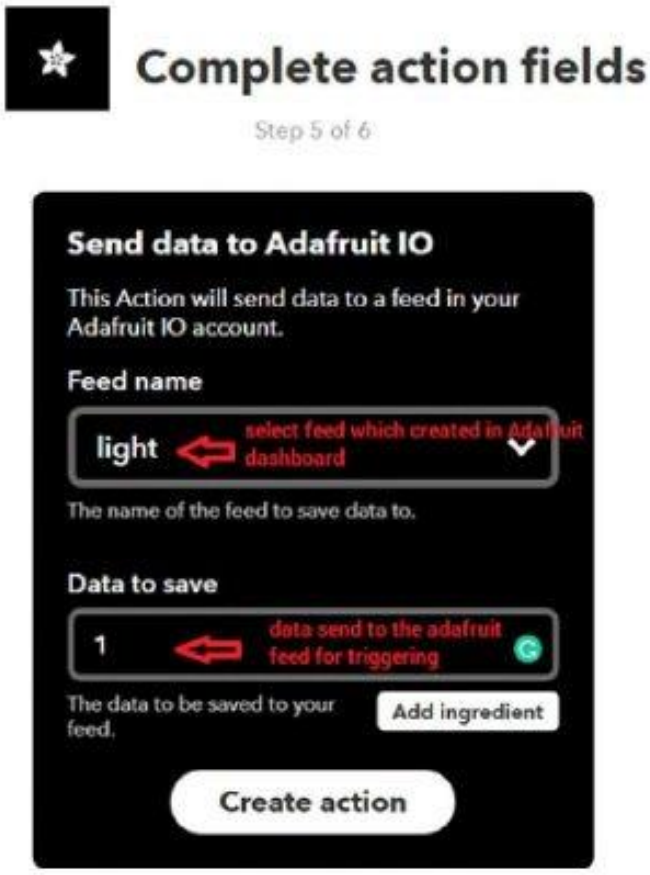
Then search for **Adafruit** and select it.



After selecting Adafruit, choose action as shown below,



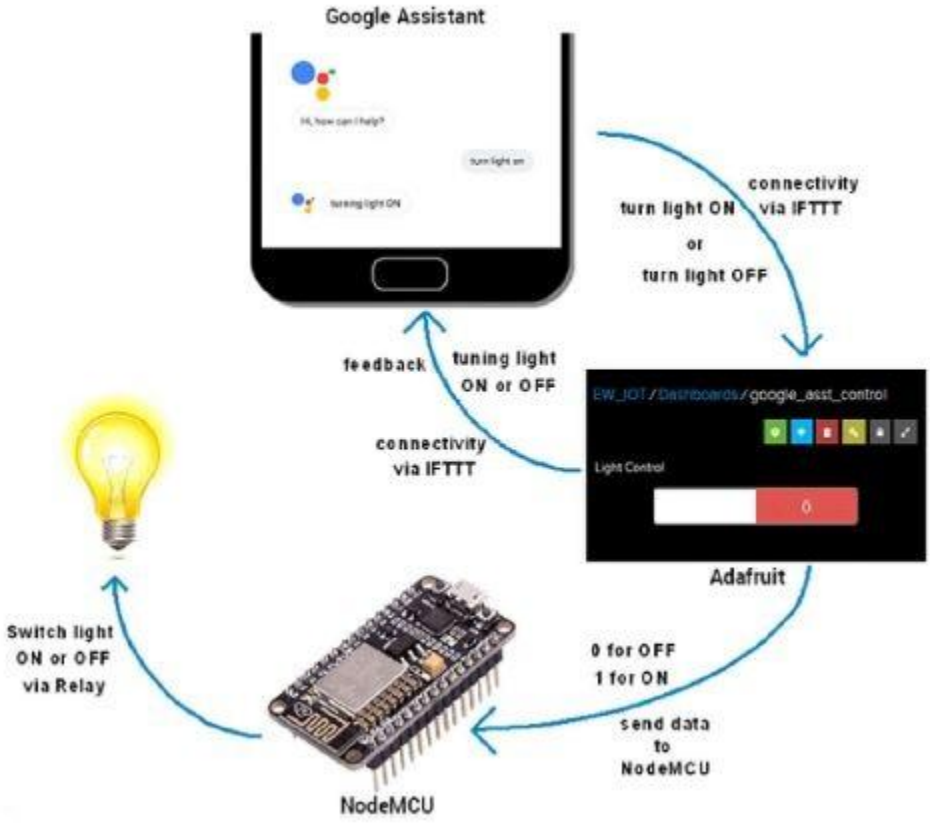
Now enter what data we need to send to which feed of Adafruit dashboard.



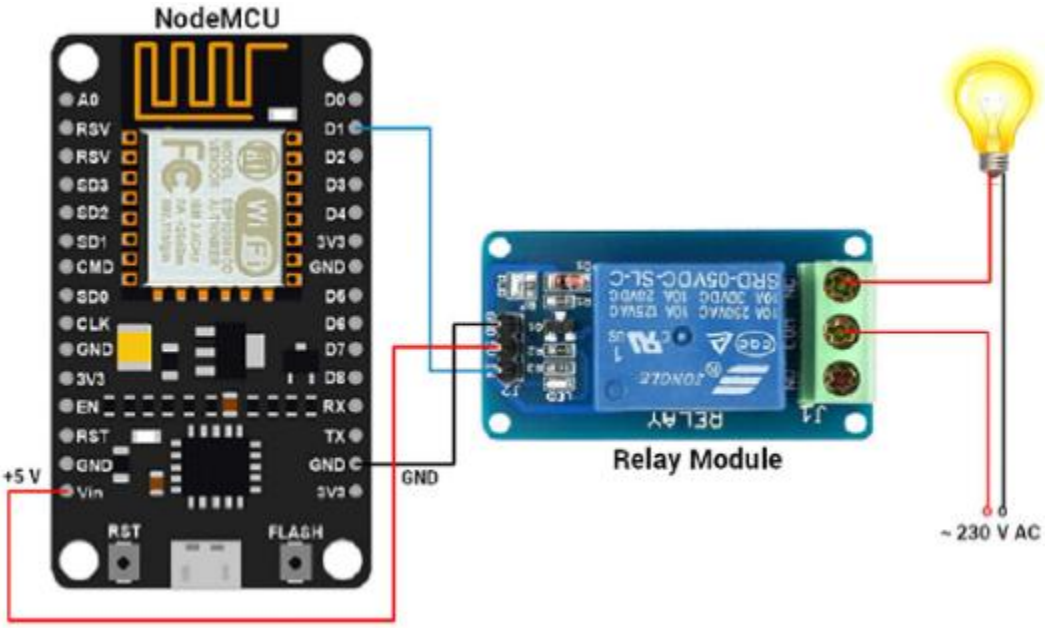
Click on **Create Action.**

So, when I use Google Assistant on my mobile and give voice command as "Ok Google, Turn LED ON", applet created in IFTTT receive this command and will send data '1' to the Adafruit feed. This will trigger the event on Adafruit dashboard which is continuously monitored by the microcontroller (here NodeMCU). This microcontroller will take action as per the data change on the Adafruit dashboard.

Overall view of Voice controlled Home automation using Google Assistance



Interfacing Diagram



https://www.youtube.com/watch?v=1goTMGq26wE

Appendix C

List of Case Studies

IoT BASED CASE STUDIESS

- IoT based Home Automation
- IoT based Agriculture System
- IoT based Patient Monitoring System
- IoT based Humidity and Temperature Monitoring System
- IoT based Weather Reporting System
- IoT based Smart Water Management System
- IoT based Garbage Monitoring System
- IoT based Smart Street Light Management System
- IoT based Industry Automation
- To control AC units through Webpage
- To Displaying Environment Data in Webpage
- Uploading Environment Data to cloud & visualized through Thingspeak
- Bluetooth Based Home automation
- Smart Home Automation control using Google Assistance(GA)
- Smart Home Automation control using Alexa
- Smart Building Project using PIR

References

1. "lnternet of Things: A Hands-on Approach", by Arshdeep Bahga and Vijay Madisetti (Universities Press)

2. Internet of Things, Authors: Shriram K Vasudevan, Abhishek S Nagarajan, RMD Sundaram

3. ISBN: 9788126578375, Wiley Publishing House, India. 332 pages, INR 479

4. lnfosys Training E Materials.

5. "The Internet of Things: Enabling Technologies, Platforms, and Use Cases',, by pethuru Raj and Anupama C. Raman (CRC press)

6. Adrian McEwen, Designing the internet of Things, Wiley (B November 2Ot3), ISBN-13:978-.11-L1,8430620,

7. NPTEL Reference : https://onlinecourses.nptel.ac.in/noc17_cs22/preview