Institute of Engineering and Technology

Devi Ahilya Vishwavidyalaya, Indore

Department of Electronics & Telecommunication

INTERNET OF THINGS (6ETRC2)

ETC-B (3RD YEAR)

Session January-April 2025

Submitted To:

Dr. Raksha Upadhyay Ma'am

Dr. Uma Bhatt Ma'am

Submitted By:

(22T6113)-ANKUSH VERMA

(22T6149)-OMPRAKASH ALAWA

(21T6169)-VISHAKHA CHATSE

(22T6179)-YASH SISODIYA

SIGNATURE :-

INDEX

S.NO	TOPIC	SIGNATURE
1.	Assignment-1 INSTALLATION AND INTRODUCTION TO MATLAB AND THINGSPEAK CLOUD FOR WINDOWS	
2.	Assignment-2 Sensors Based Experiments on ARDUINO UNO Board with Details	
3.	Assignment-3 Hardware Project Description (ALL DETAILS)	

ASSIGNMENT-1 Introduction to Matlab & Thingspeak

MATLAB offline Installation Guidelines

Prerequisites.

1. Enrolling the MATLAB portal with your University Email ID.
2. Download the installer ISO file, mount it in your PC and open the installer file.

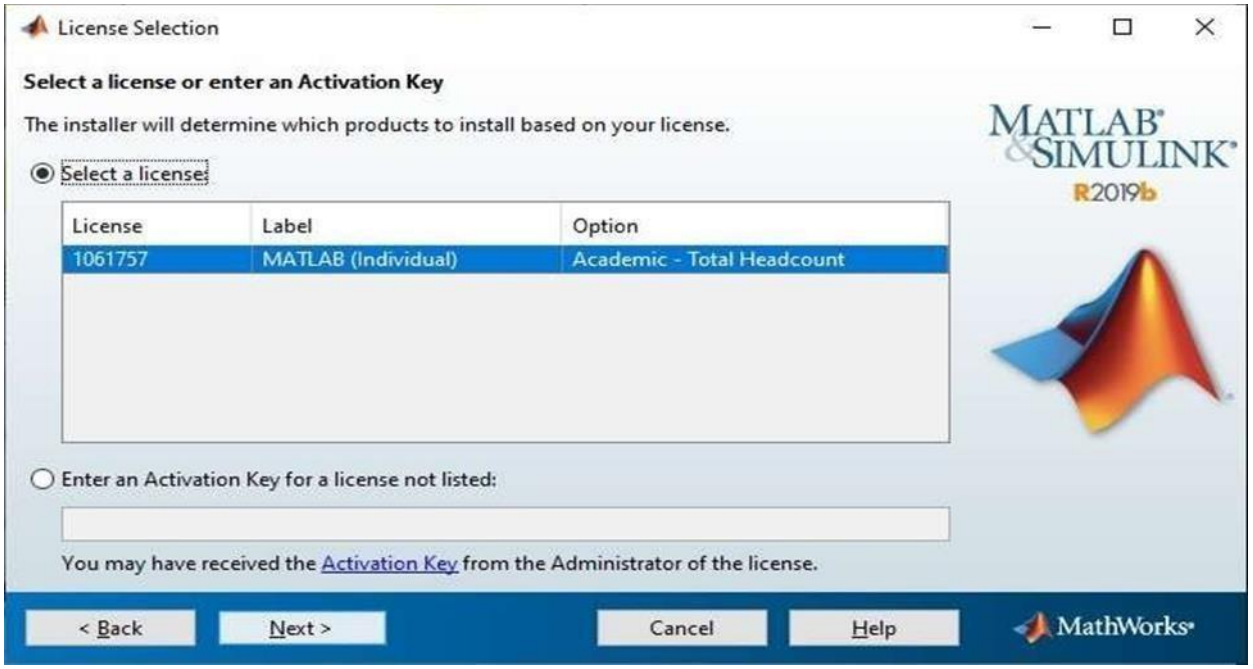
Step 1: Double click on the MATLAB icon (the binary file which we down loaded earlier). After clicking the icon, a pop-up will ask for the installer to run, click on the ***Run***. The MathWorks Installer window will pop-upon the screen.

Step 2: By default, the first option, i.e., Log in with a MathWorks Account, is selected, proceed with this option and do remember to check your internet connection for proper installation of the MATLAB environment. Click on Next in the bottom of the window.

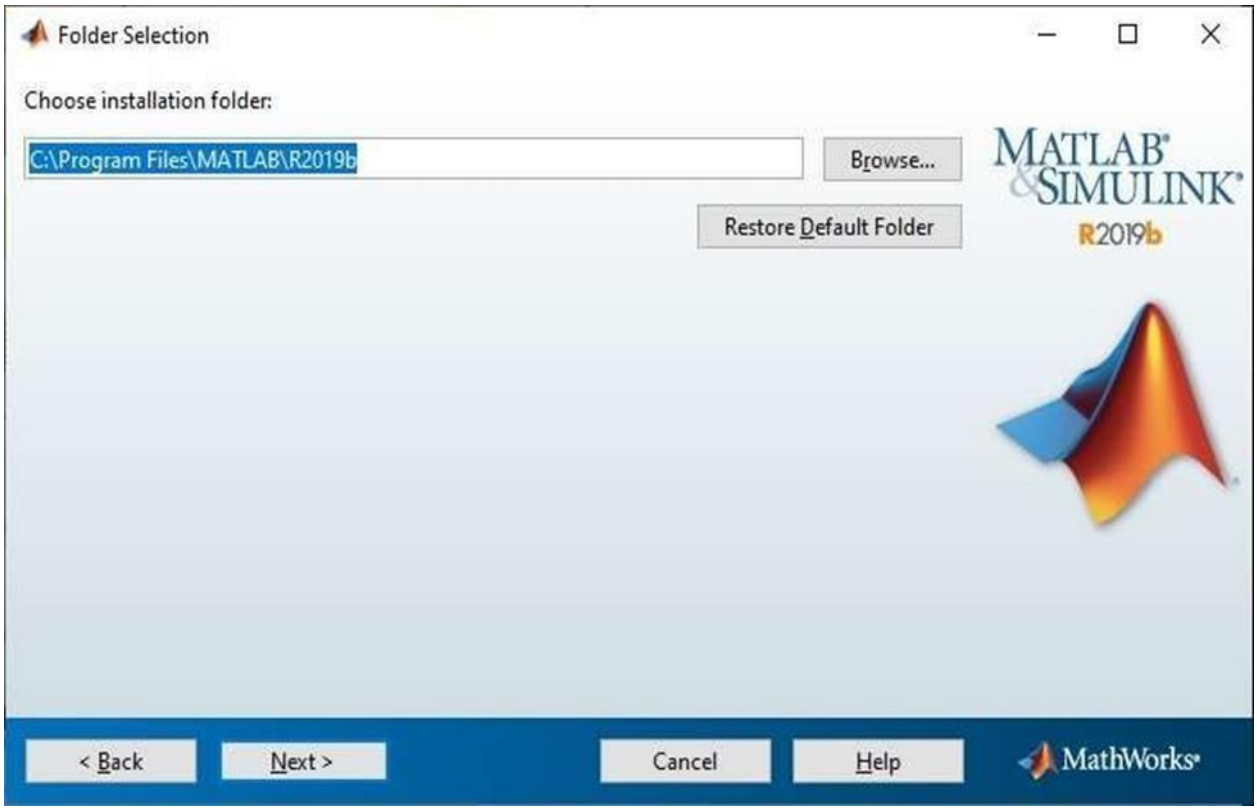
Step 3: Accept the license terms by selecting **Yes** in the next page and again click on the **Next** button.

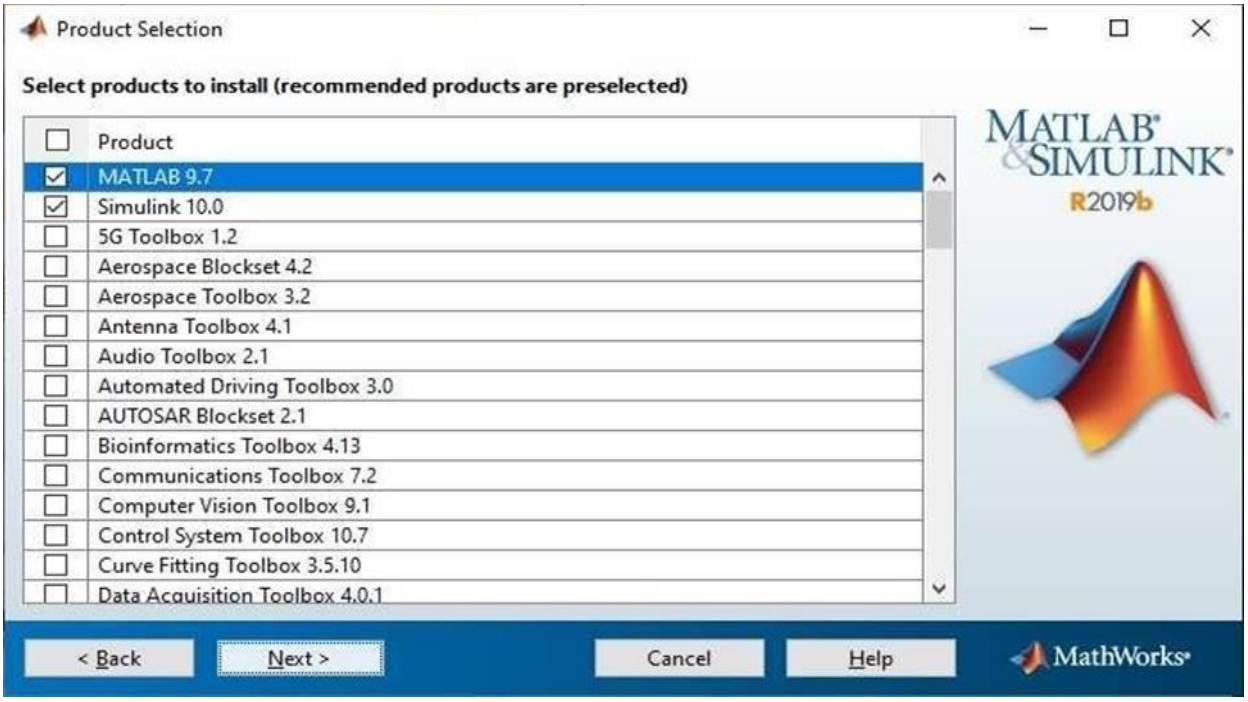
Step 4: A new page appears, by default, the first option is selected, Log in to your MathWorks Account. Enter here your **email id** and **password** that we created during the creation of our account with MathWorks. Refer to the image below and click on **next**.

Step 5: A license Selection window will appear, a preselected license id will be highlighted with a blue background. Select "Select a license:", select license 1061757 then click Next.

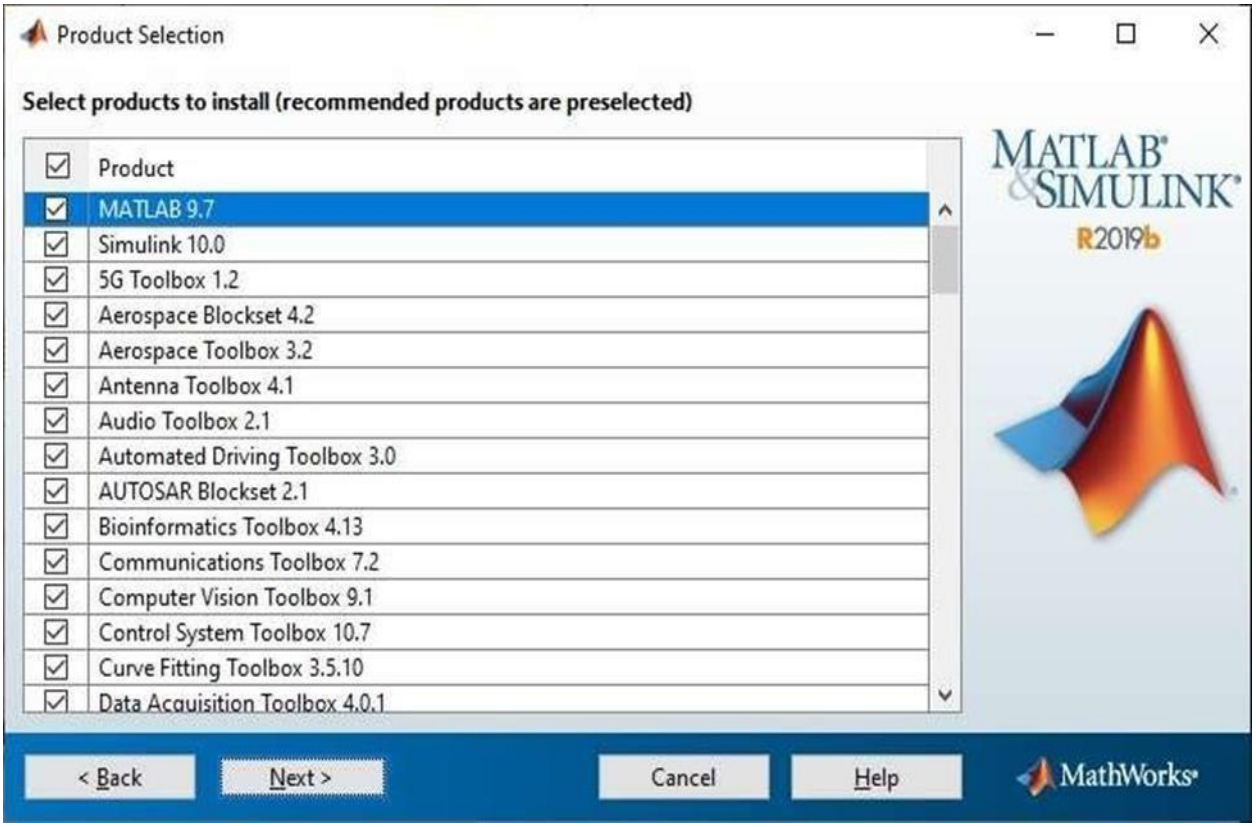


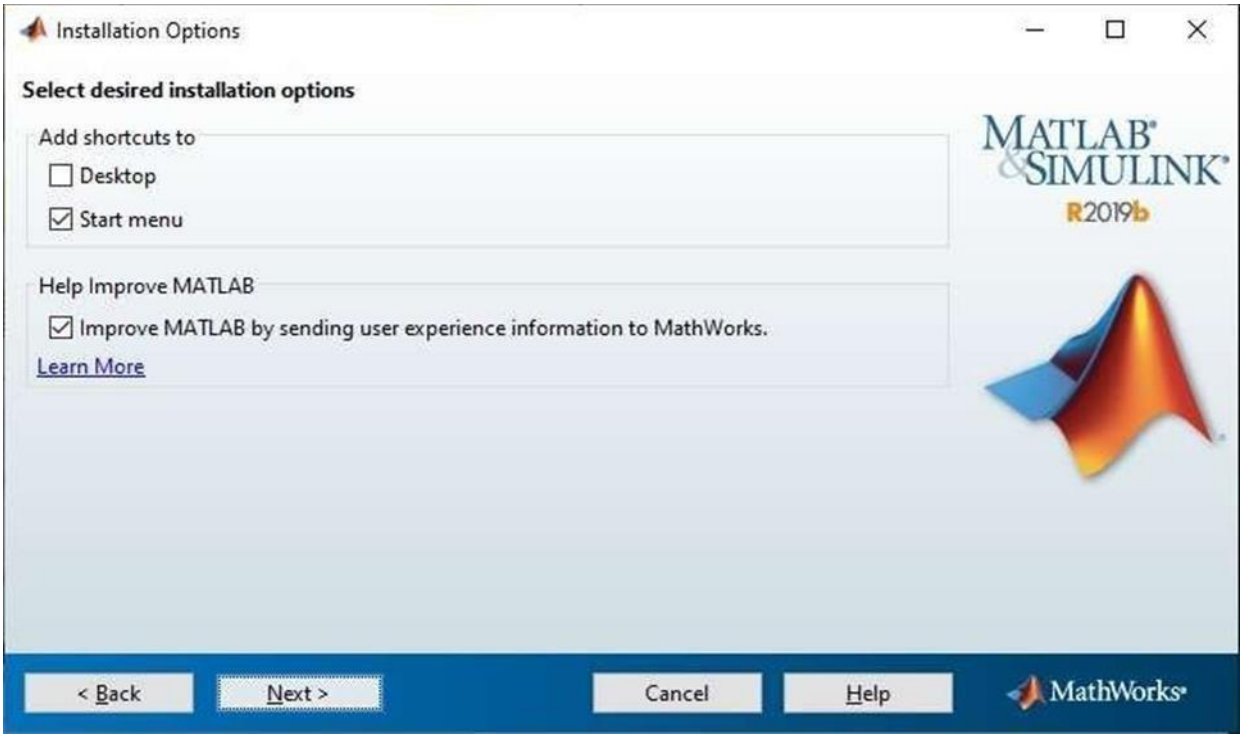
Step 6: A new Folder Selection window appears, no need to change the folder location for installation of MATLAB, click on Next.





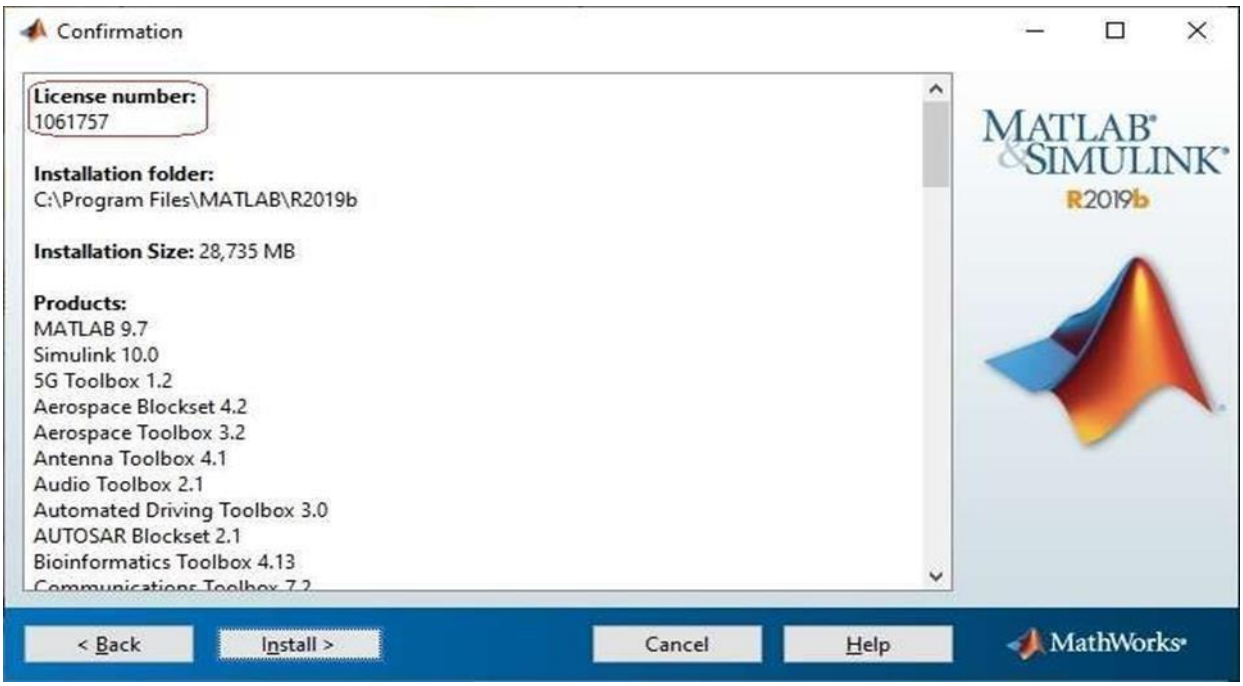
Step 7: Next is Product Selection window, the first product is MATLAB9.7, this is mandatory to select because it is the MATLAB environment, and from other products, you can choose as many of your choices and click on **Next**.





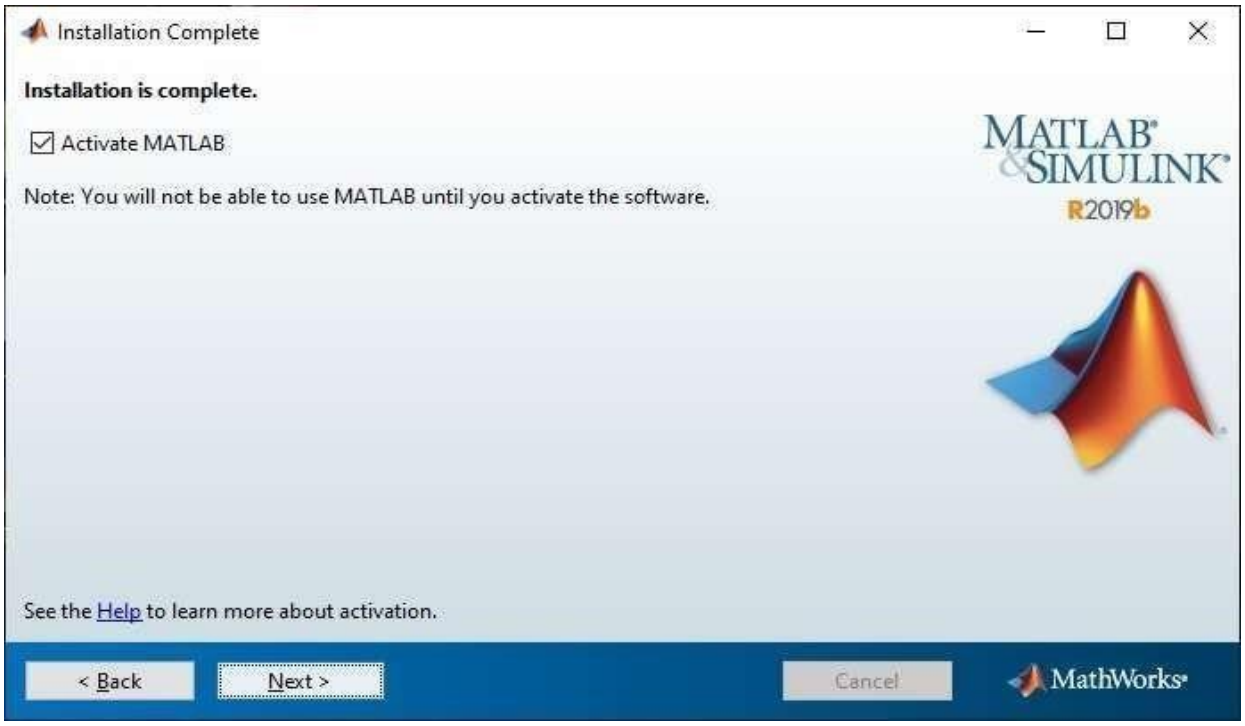
Step 8: Optional: Configure your additional shortcuts for MatLab, then click Next

Step 9: Next is the *Installation Options* window, select options as per your choice. Anytime you feel something to change, you can go back to the previous step by clicking on the Back button.

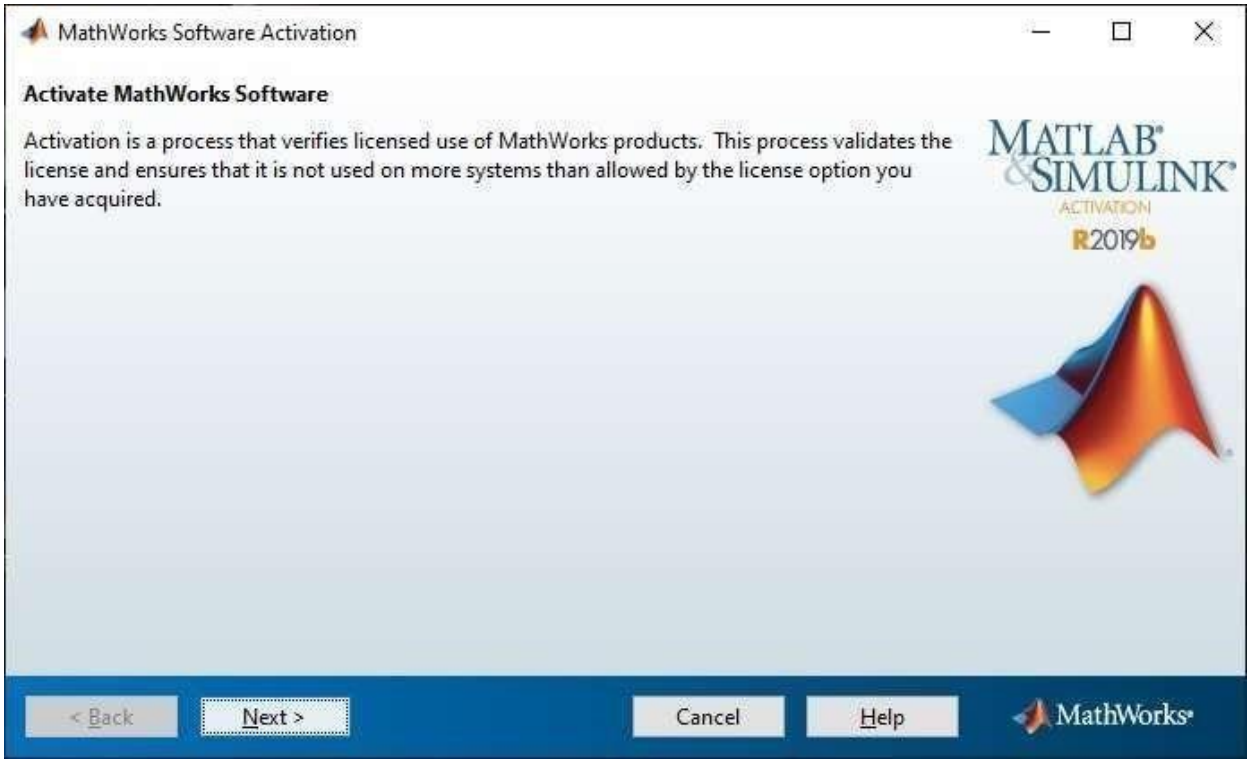


Step 10: Next is the Confirmation window, here you no need to do anything, confirm what you are going to download in the process of the installation of MATLAB, its other Add-on products, and what is the size of the downloads; and click on Install.

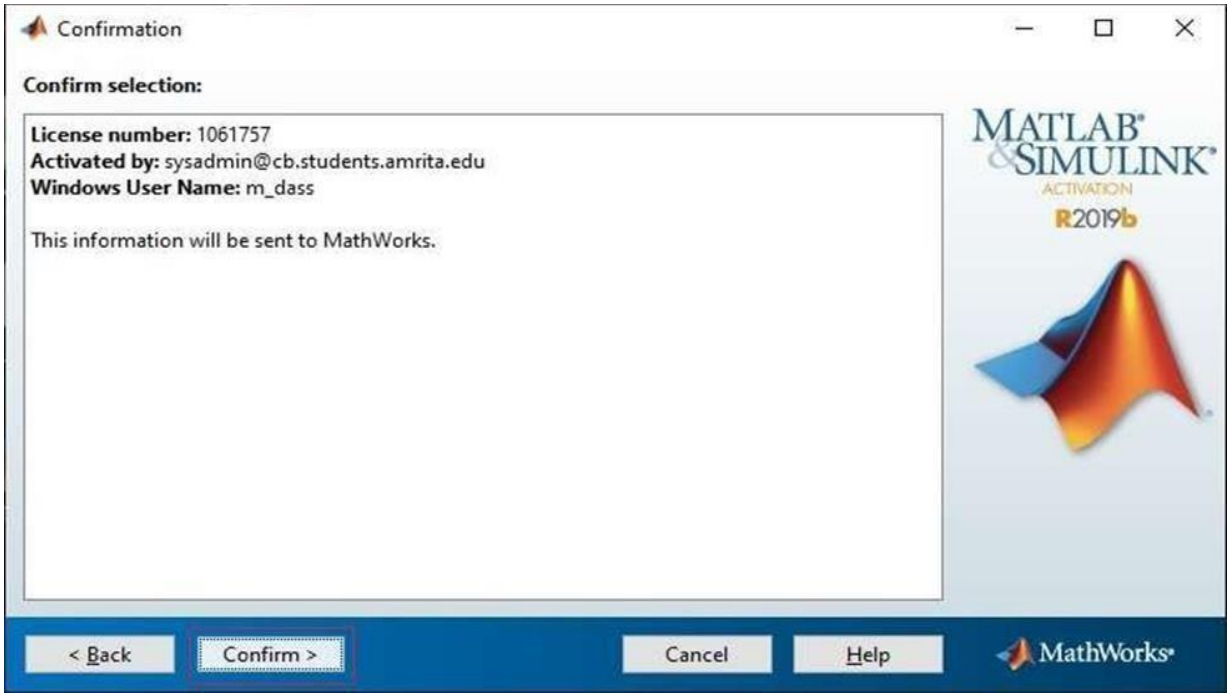
Step 11: After completion of the installation, a window appears that says to Activate the MATLAB, no need to do anything, click on the **Next** button.



Step 12: After clicking on **Next**, a new window appears that says about what is the meaning of activation. Proceed by clicking on *Next*.



Step 13: Make sure you enter the correct user name, then click Next.



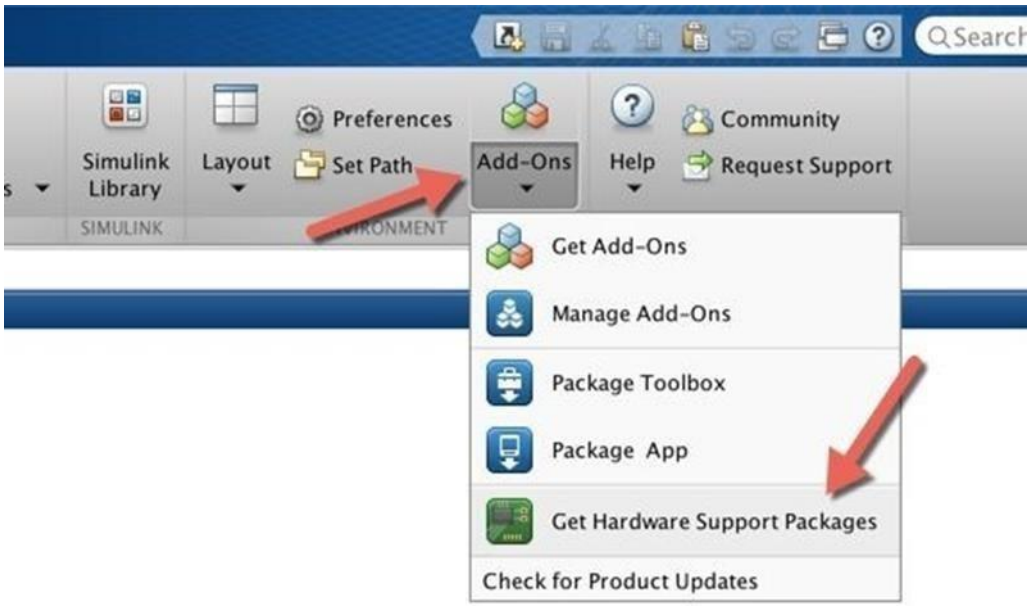
Step 14: Again, a new window appears displaying your **email id** and your products **license id**, proceed by clicking on **Confirm** *button*.

The installation process of MATLAB and other selected components have been completed successfully. Now click on the **Finish** button. Your MATLAB offline installation Is ready to use.

Setting up Arduino for MATLAB

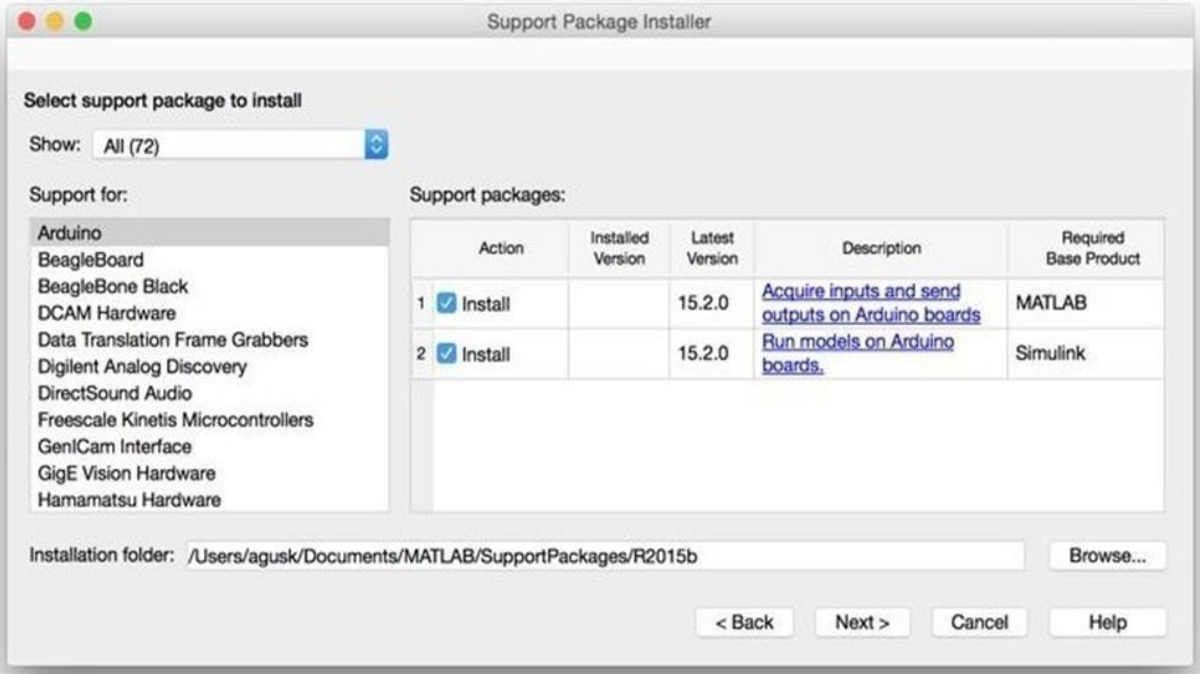
In this section, we try to set up Arduino development board for MATLAB. You can configure MATLAB Support Package for Arduino hardware using MATLAB2014a or later. We also need internet connection to download this package.

Run MATLAB application. Click **Get Hardware Support Packages** on **Add-Ons** icon on tool box.

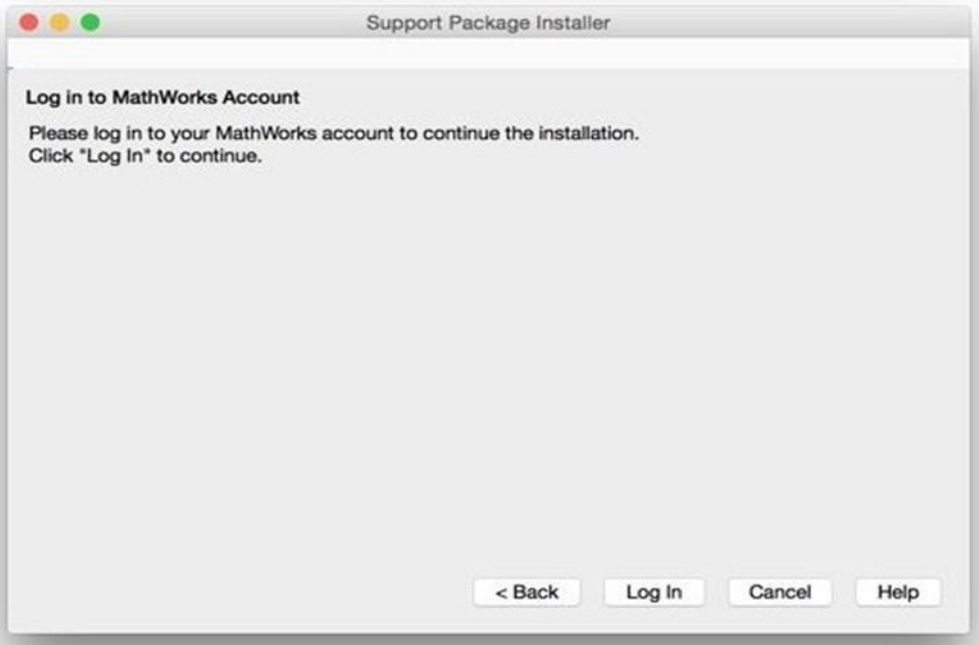


Then, you get a dialog. Select Install from Internet. If done, click **Next>**button.

Click on Arduino option and select both support packages. If done, click **Next>**button.

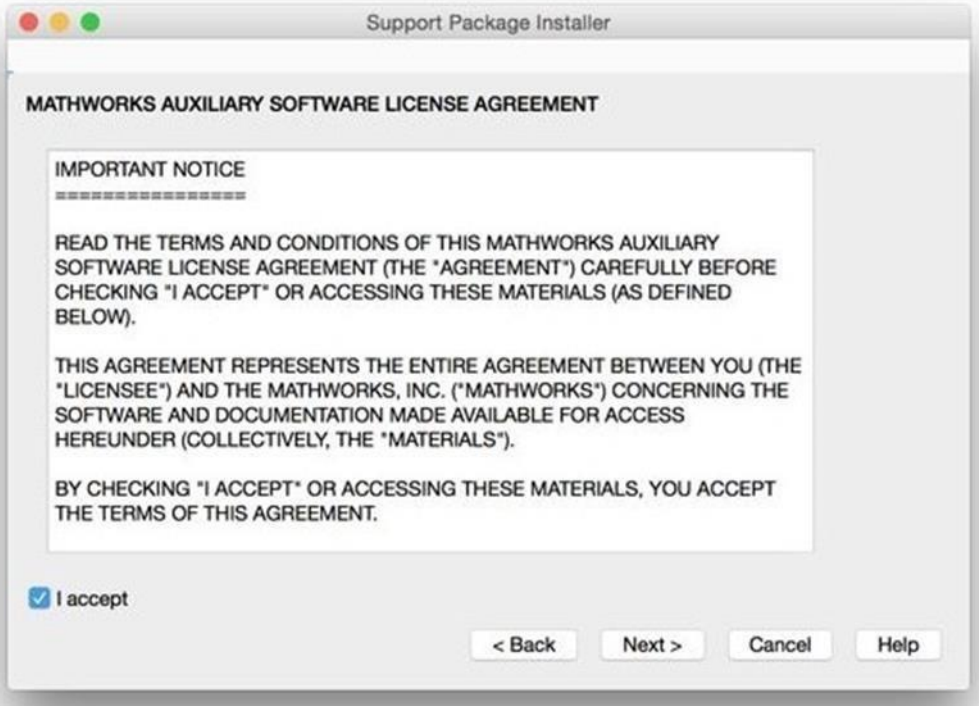


You will be asked to log on with your MATLAB account. You should have MATLAB license. Click **Log In** button.

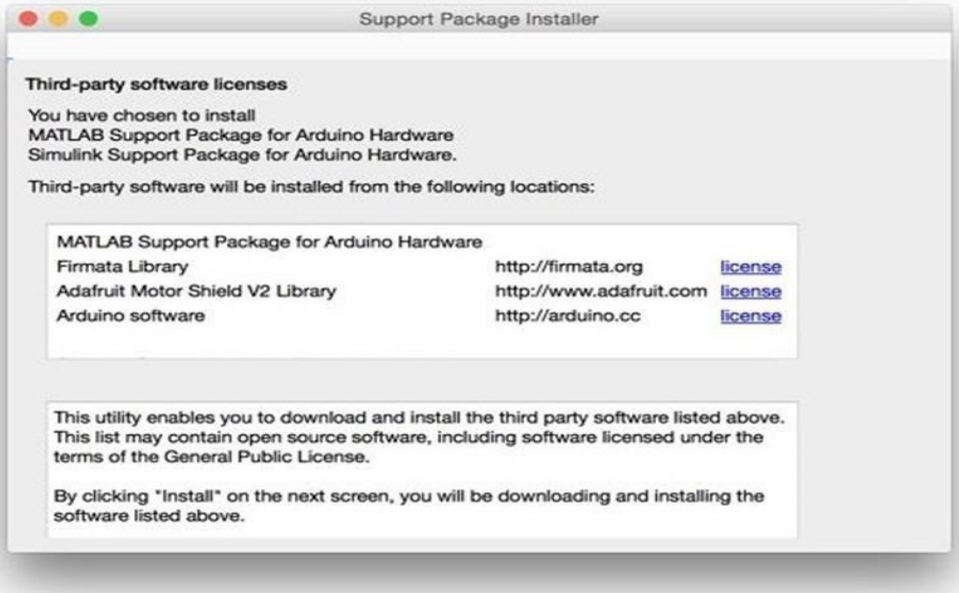


You will see the authentication dialog box. Fill in your login credentials, then click **Log In** button.

If success, you should get a software license agreement. Checkin **I accept** and then click **Next>**button.



You will get confirmation. Click **Next>**button.



Click **Install** button to start installation.

After done, you will be asked to configure Arduino board. Select Arduino and then, click **Next>**button.

Confirmation form will be shown. Click **Continue>**button.

The program will check if your platform needs Arduino driver or not. If you're working on Linux and Mac, you don't need a driver. You need to install Arduino driver if you're working on Windows platform. Click **Next >** button if done.

Installation is over. Click **Finish** button to close installation.

Connecting Arduino Board to Computer

Now you can connect Arduino board to computer. Then, verify which serial port is used for Arduino board. On Mac platform, you type this command.

```
$ls/dev/cu*
```

On Linux platform, you type this command.

```
$ls/dev/tty*
```

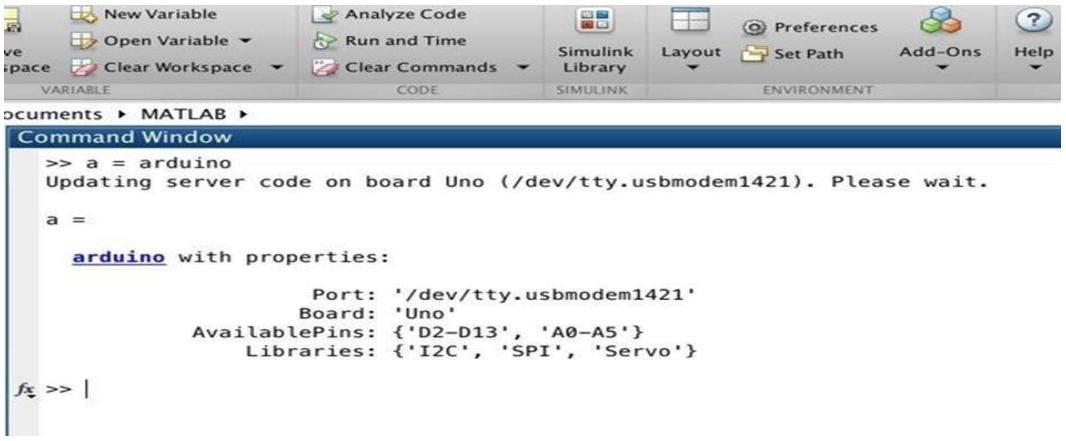
You can use Device Manager on Windows platform.

After that, you should see serial port of Arduino board which is attached on the computer. My OSX detected my Arduino board used/dev/cu.usb modem 1421 serial port.

On MATLAB command Windows, type this command

```
>>a = arduino
```

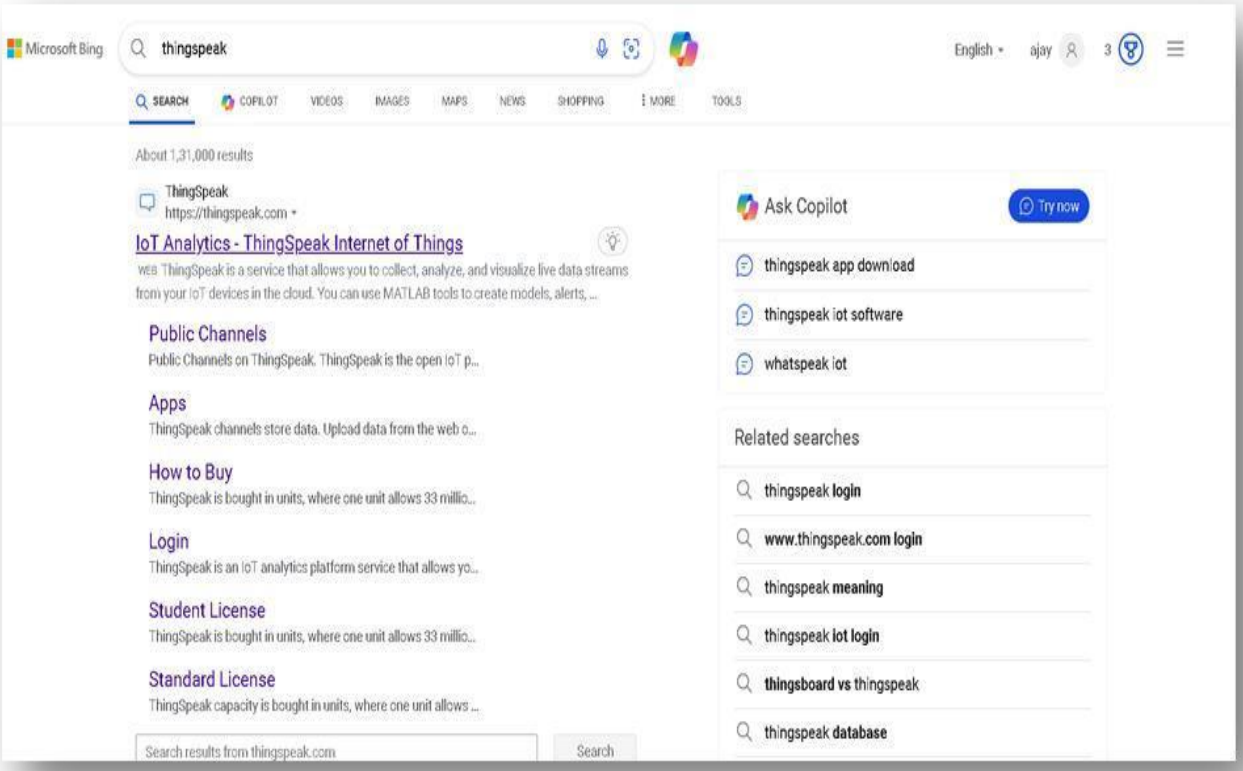
MATLAB will detect your Arduino board. You should detect Arduino board information on Matlab Command Window.



Thingspeak

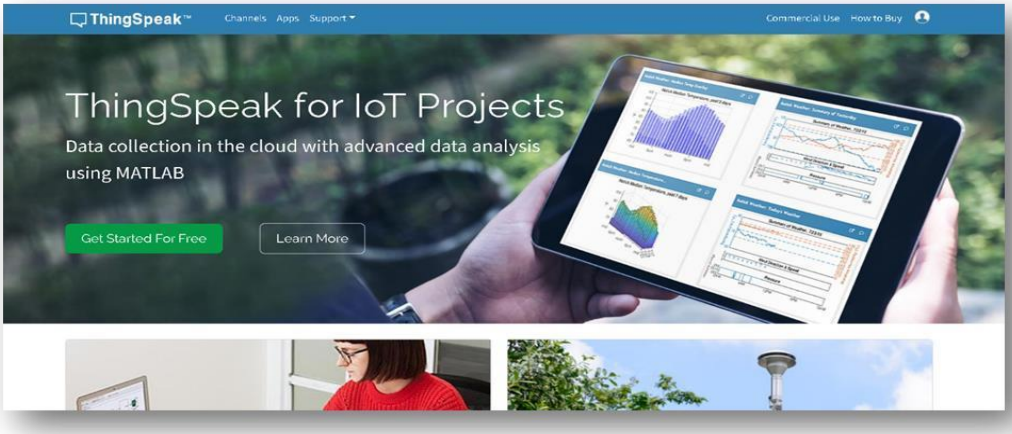
How to use Thing speak and create channels (stepwise method):

1. Open your Internet Browser and search Thingspeak.



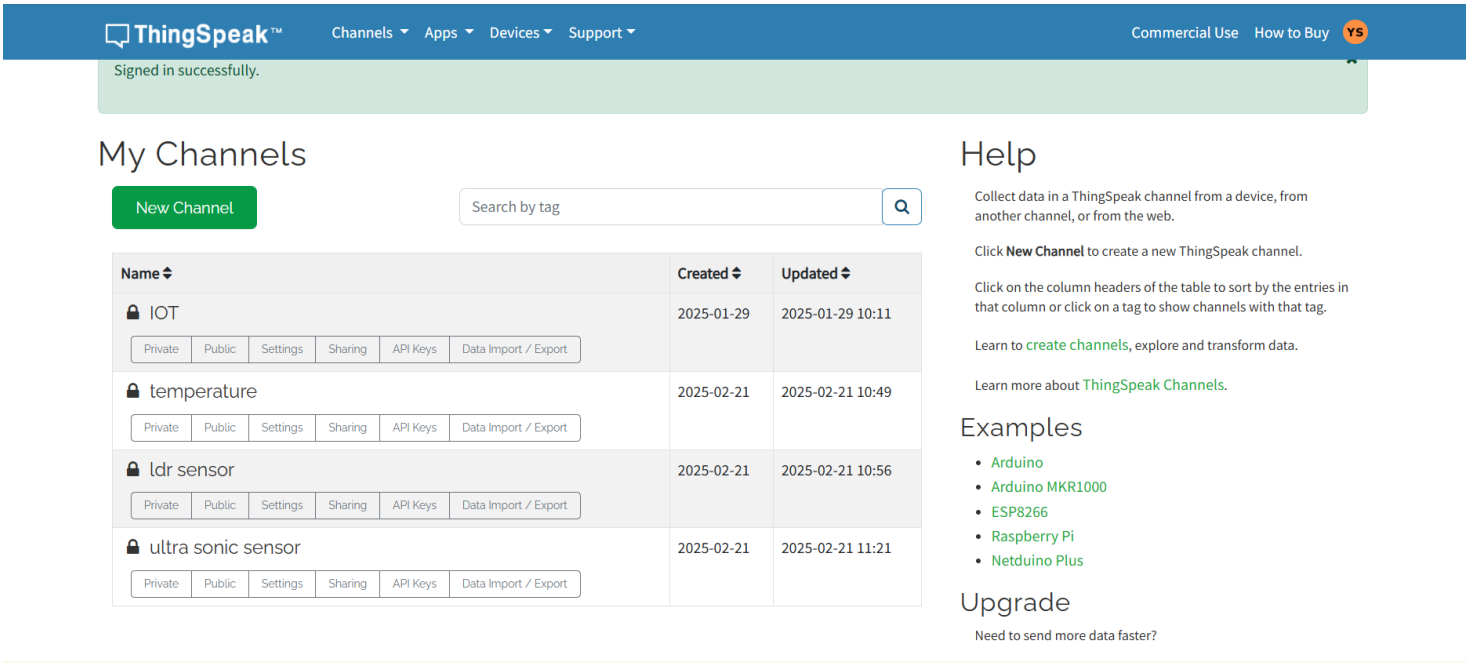
2. Click on the first link Thingspeak.

3. The user interface will look like as follows:



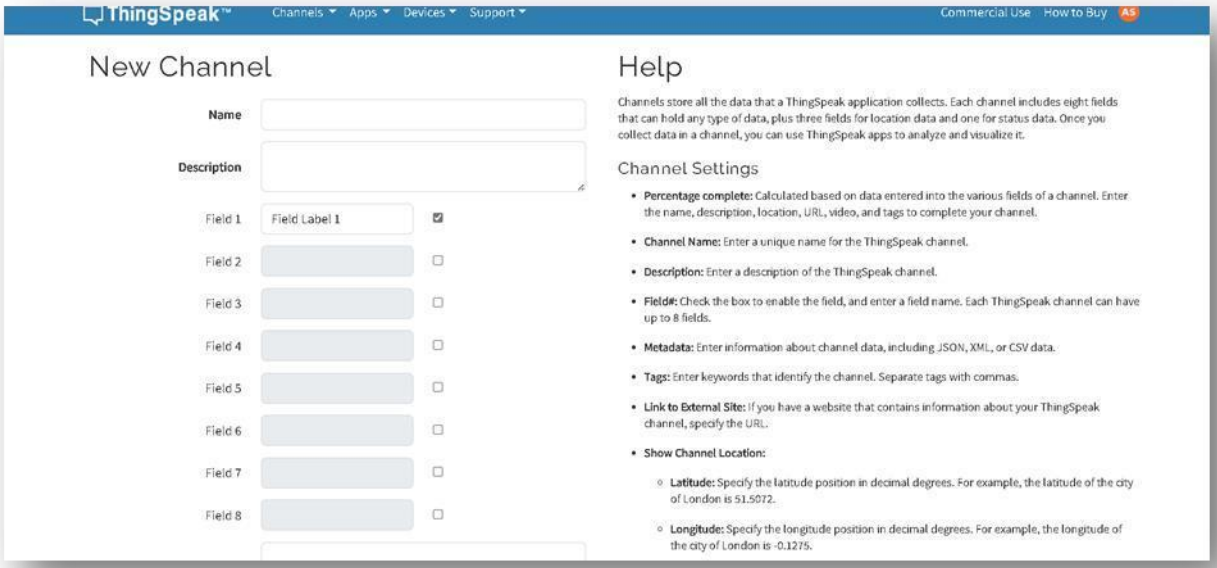
4. Log in with your college email ID.

5. After logging in the user interface will look like this:



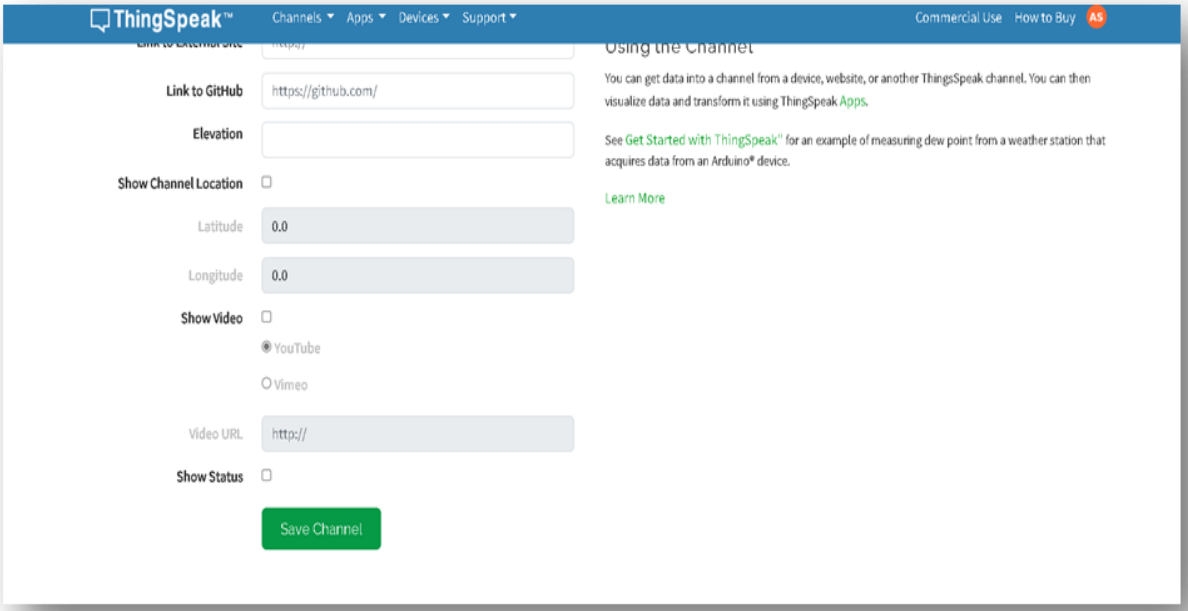
6. Click on New Channel.

7. The following window will open:



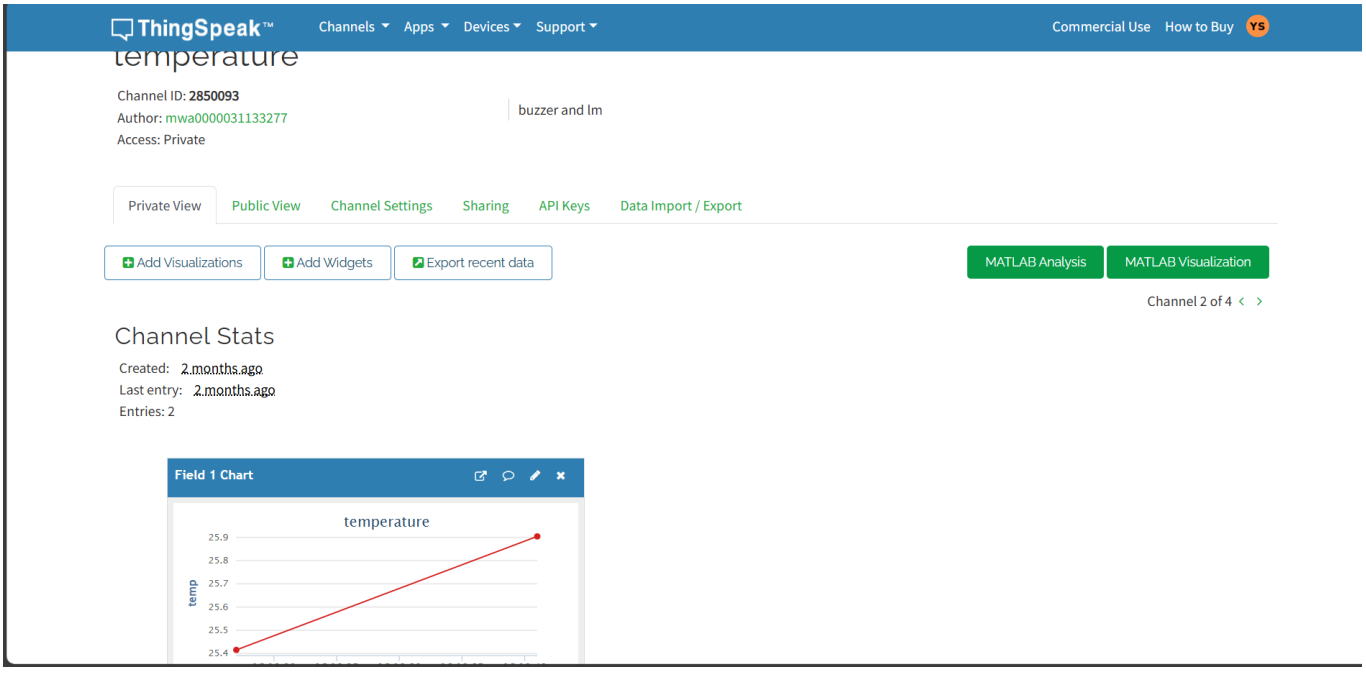
8. Now give the name of the channel, description and the fields. The field column will contain the variable which is to be measured and can contain up to 8 variables, variables may include temperature, voltage, current, distance etc.

9. Click on save channel at the bottom after filling all the details.

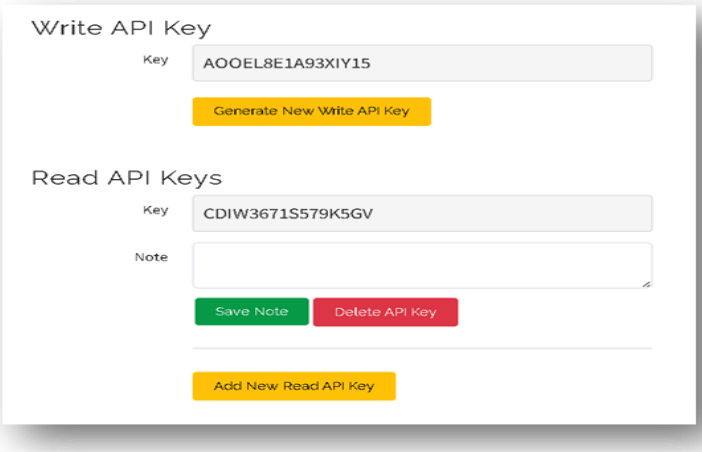


10. Now the following window will open:

11. The name of the channel and the variable which is to be measured is shown with date on the x-axis. Details of the channel is specified under the name of the channel.

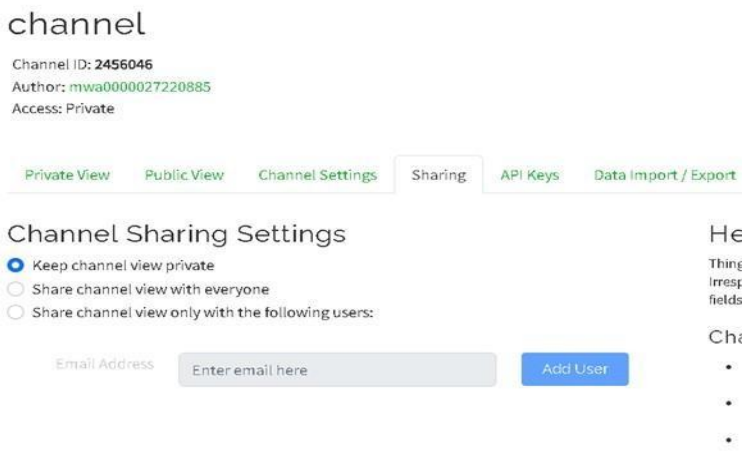


12. Now we take a look at the API keys.



13. The following keys are used to communicate with the Thingspeak to send or write data
to or from the cloud respectively.

14. In sharing options, you can choose who you want to share your channel with, either with a particular or sharing it with everyone by keeping your channel public or keeping your channel private.



Connectting Matlab with Thingspeak

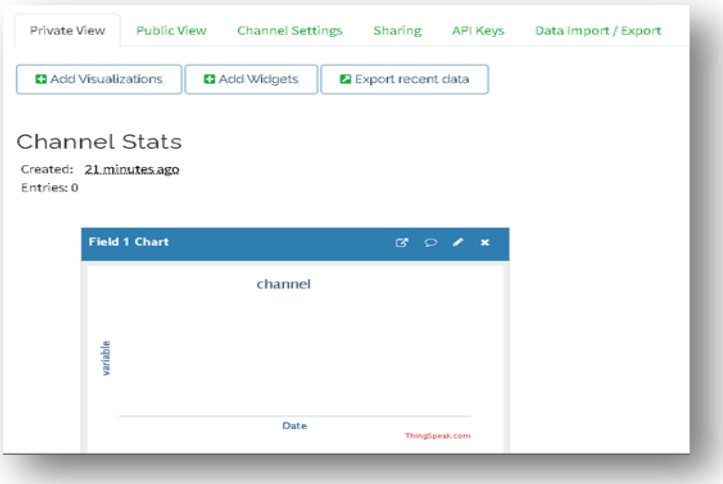
1. Open matlab command window.

2. In the command window type the following command:
   ChannelID = 2454257;
   writeAPIkey =
   'NP6QWGVKYOK1UPDI';
   readAPIkey =
   'L5MTWQ20ODNFH2RG';

3. The following are the details of the channel that you have created. writeData = thingSpeakWrite(ChannelID, variable, 'WriteKey', writeAPIKey); readData = thingSpeakRead(ChannelID, 'ReadKey', readAPIKey);

The following command will read the variable from the sensor and will publish it to your channel created in Thingspeak.

4. In Thingspeak you can view your data by clicking on private view in Thingspeak.



ASSIGNMENT-2
Sensors Based Experiments

Sensors step wise interfacing with Arduino using MATLAB

Interfacing LM35 with Arduino using MATLAB:

1. Pin Connection of LM35 with Arduino

LM35	**Arduino**
Vcc	5 volts
Ground	Ground
Aout	A0

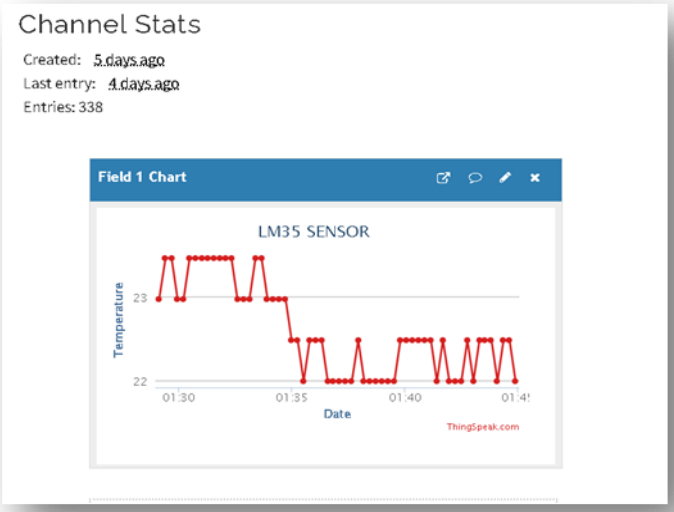
Buzzer	**Arduino**
Positive Node	D12
Negative Node	Ground

2. Code for interfacing LM35 with Arduino and displaying data on Thingspeak:

```
clear;
a = arduino();
ChannelID = 2423758;
writeAPIKey =
 'I4ZMNIS1KEZRJF8B';
 readAPIKey =
 'MJIX26NSU3OHHITE';

 while (1)
            voltage
      =
```

```
        readVoltage(a,
        A0); temp =
        voltage*100;
                writeData = thingSpeakWrite(ChannelID, temp, 'WriteKey',
writeAPIKey);
                readData = thingSpeakRead(ChannelID, 'ReadKey',
        readAPIKey);  pause(15);

        if readData >= 50
                writeDigitalPin(a, 'D12', 1);
        else
                writeDigitalPin(a, ‘D12’, 0);
        end
    end
```

3. Graph of Temperature vs date graph on thingspeak:



Interfacing Flame Sensor with Arduino and displaying the voltage on Thingspeak:

1. Pin Connections

Flame Sensor	**Arduino**
Vcc	5 V
GND	GND

A0	A0

2. Code for interfacing Flame Sensor with Arduino and displaying data on Thingspeak:

```
clear;
a = arduino();
ChannelID = 2457997;
writeAPIKey = '5XKMOND3TFW7AL1W';
readAPIKey = 'MJIX26NSU3OHHITE';

while(1)
        voltage =
readVoltage(a,'A0');
flame = (5-
voltage)*10;
                writeData = thingSpeakWrite(ChannelID, flame,
        'WriteKey', writeAPIKey);  readData =
        thingSpeakRead(ChannelID, 'ReadKey', readAPIKey);
        pause(15);

        if(readData>=20)

writeDigitalPin(a,'D9',
1);        else

writeDigitalPin(a,'D9',
0);        end
end
```

Interfacing ultrasonic sensor and a buzzer with Arduino using MATLAB and sending the data to Thingspeak:

1. Pin connections

Ultrasonic sensor	**Arduino**
Vcc	5V
GND	GND
Echo	D11
Trigger	D12

Buzzer	**Arduino**
Positive terminal	D9
Negative terminal	GND

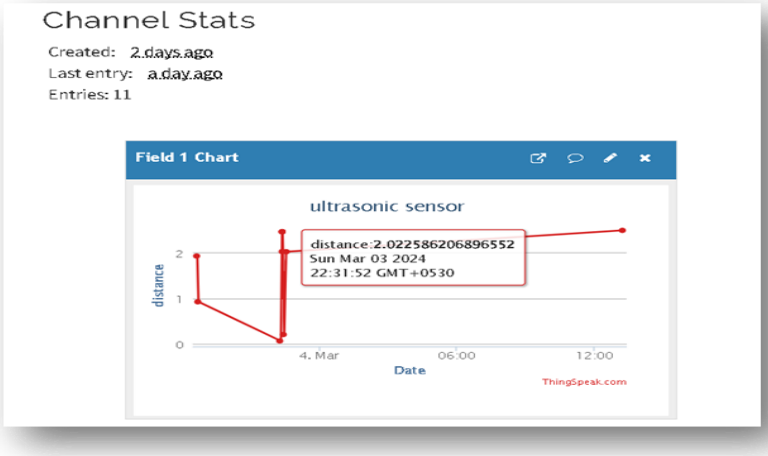
2. Code for interfacing Ultrasonic Sensor with Arduino:

Before Writing the code for ultrasonic sensor you have to go to add-on in the MATLAB window and click on manage add-ons and program the Arduino by adding ultrasonic libraries.

```
clear;
a = arduino();
ChannelID = 2454257;
writeAPIkey =
'NP6QWGVKYOK1U
PDI'; readAPIkey =
'L5MTWQ20ODNFH
2RG';
ultrasonicObj = ultrasonic(a, 'D12', 'D11');

while(1)
        distance = readDistance(ultrasonicObj);
        writeData = thingSpeakWrite(ChannelID, distance,
        'WriteKey', writeAPIKey); readData =
        thingSpeakRead(ChannelID ,'ReadKey' ,readAPIkey);
        pause(10);
```

```
        if (readData >=
                0.5)
                writeDigital
                Pin(a, 'D9',
                1);
                pause(5);
                write
        DigitalPin(a,
        'D9', 0); else
        writeDigital
        Pin(a, 'D9',
        0); end
end
```

3. Graph of distance on y-axis with date on x-axis :



Interfacing IR sensor, LDR sensor, with Arduino and displaying data on Thingspeak:

1. Pin connections

IR sensor	**Arduino**
Vcc	5 V
GND	GND
OUT	D2

LDR sensor	**Arduino**
One leg	5V
Other Leg	Junction of 10K resistor & A0 of Arduino
10kohm other leg	Ground of Arduino

2. Code for interfacing IR & LDR Sensor with Arduino:

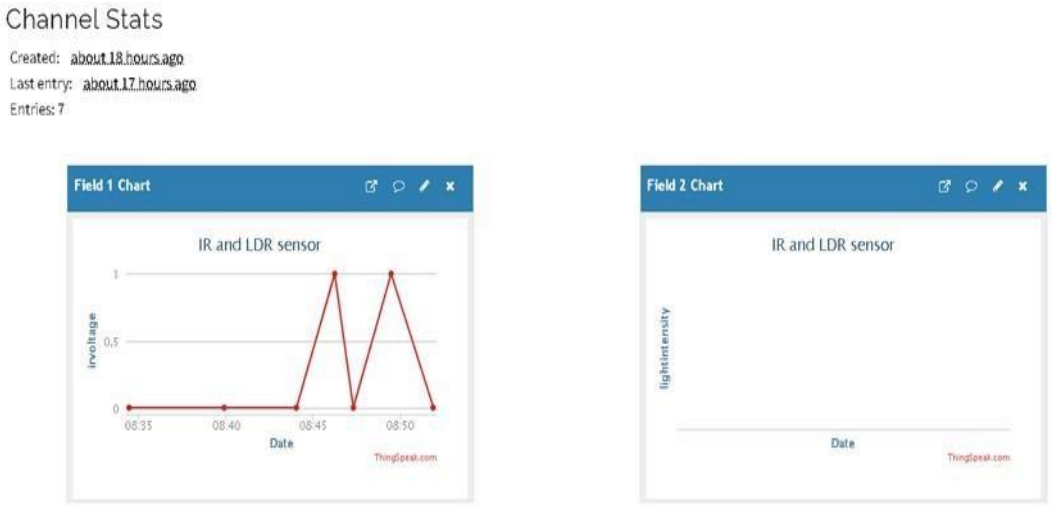
```
clear;
a=arduin
o();
ChannelI
D =
2456439
;
        writeAPIkey = 'LB0IJ3ZCZFAF5G23';
        readAPIkey = '7U436XWS4D68S9ER';

        while(1)
                IRValue = readDigitalPin(a, 'D2');              writeData =
thingSpeakWrite(ChannelID, IRValue, 'WriteKey', writeAPIKey);
readData = thingSpeakRead(ChannelID, 'ReadKey', readAPIkey);
pause(15);
        end
```

This part includes the command used to interface LDR sensor with Arduino :

```
        a = arduino();
        channelID =
        123456;
        writeAPIKey =
        'LB0IJ3ZCZFAF5
        G23';
```

```
readAPIKey = '7U436XWS4D68S9ER';

while(1)
        ldrValue = readVoltage(a,  A0);
        thingSpeakWrite(channelID, ldrValue, 'WriteKey', writeAPIKey);
        pause(15);
end
```

3. Graph of Voltage and light intensity vs date:



ASSIGNMENT -3
HARDWARE PROJECT DESCRIPTION

TOPIC - IOT based smart irrigation system

Basic Project Idea- The IoT-based Smart Irrigation System is a modern solution aimed at automating the process of watering plants or crops using Internet of Things (IoT) technology. The primary objective of this project is to develop an efficient system that conserves water, reduces manual effort, and ensures plants receive the right amount of moisture based on real-time environmental conditions. The system utilizes a NodeMCU microcontroller (ESP8266 or ESP32), which has built-in Wi-Fi capabilities, allowing it to communicate with online platforms. A soil moisture sensor is embedded in the soil to continuously monitor the moisture content. When the sensor detects that the soil moisture level drops below a predefined threshold, it sends a signal to the NodeMCU, which then activates a relay module connected to a water pump. The pump automatically irrigates the soil until it reaches an optimal moisture level, at which point the system stops the water flow.

To provide remote access and monitoring, the system is connected to an IoT platform such as Blynk, ThingSpeak, or Firebase. These platforms allow users to view real-time data such as soil moisture percentage and control the system through a smartphone or web interface. For instance, the Blynk app can be used to turn the pump on or off manually, in addition to its automated operation. Furthermore, the system can be enhanced by adding a DHT11 sensor to monitor temperature and humidity, or a rain sensor to prevent irrigation during rainfall. Notifications or alerts can also be configured to inform the user when the soil is too dry or if irrigation is taking place.

The entire setup is low-cost, energy-efficient, and can be powered via a small battery or solar panel, making it ideal for rural or remote areas. Additionally, it promotes water conservation by ensuring water is only used when necessary, and can help farmers or gardeners manage irrigation more effectively. The project includes several key components such as the NodeMCU microcontroller, soil moisture sensor, relay module, DC water pump, jumper wires, power source, and water pipes for distribution. Software like Arduino IDE is used for programming the microcontroller, and appropriate libraries are

added for Wi-Fi connectivity and sensor integration. The system logic is based on threshold values where, for example, a moisture level below 30% is considered dry, triggering the pump, while above 50% stops it.

This smart irrigation model not only demonstrates the practical application of IoT in agriculture but also highlights how technology can address real-world problems like water wastage and inefficient farming. The system is scalable, meaning more sensors and pumps can be added to cover larger fields. Future improvements could include machine learning for predictive irrigation, integration with weather forecasts, and data logging for crop analysis. Overall, this project provides a comprehensive and hands-on understanding of IoT, automation, and sustainable agriculture, making it an excellent choice for academic presentations and real-world use.

Components Required:

1. Microcontroller: ESP8266 or ESP32 (built-in WiFi)

2. Soil Moisture Sensor: To detect moisture level in the soil

3. Relay Module: To control water pump

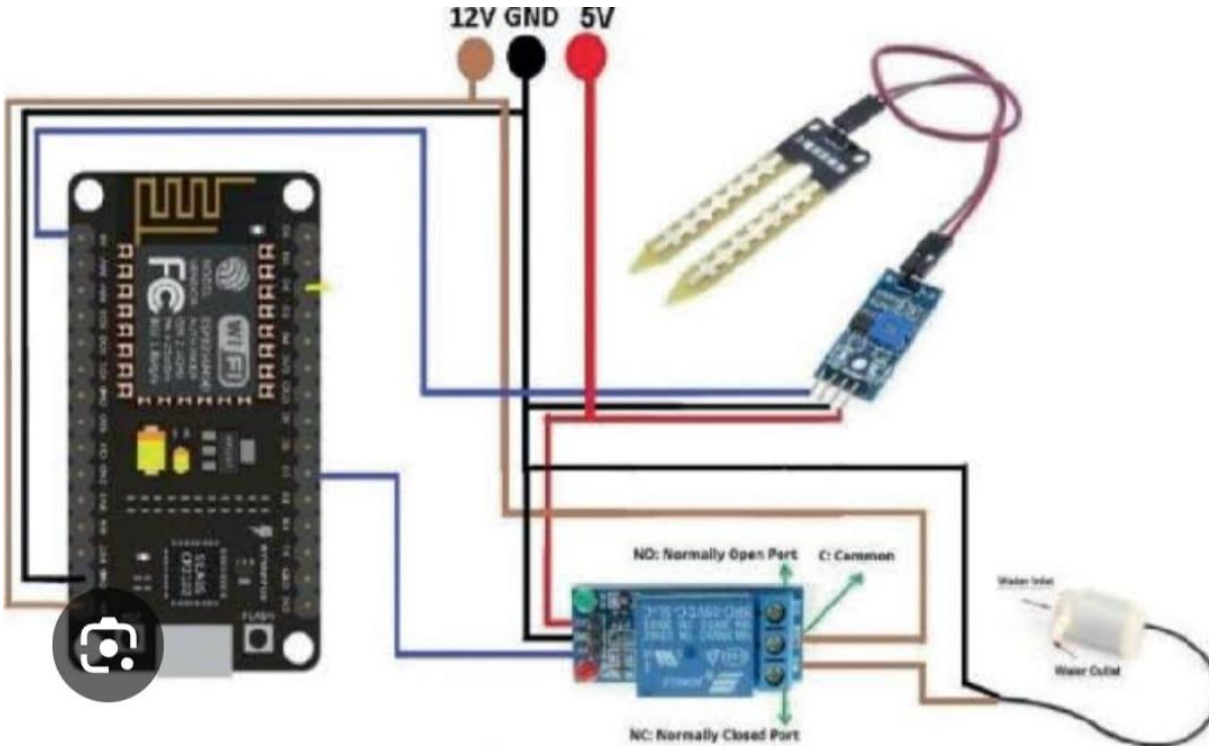
4. Water Pump: To irrigate the plants

5. Water Tank or Source

6. Jumper Wires and Breadboard

7. Power Supply

Circuit diagram



Description Of Hardware Components

1. NodeMCU(ESP8266/ESP32)-

NodeMCU is an open-source IoT development board based on either the ESP8266 or ESP32 Wi-Fi microcontroller chips made by Espressif Systems. It allows you to build wireless, internet-connected electronic projects like smart lights, home automation, weather stations, and more all easily programmable through platforms like the Arduino IDE, MicroPython, scripting language.

- NODEMCU is an open source LUA based firmware developed for ESP8266 wifi chip by exploring functionality with ESP8266 chip, NodeMCU firmware comes with ESP8266 Development board/kit i.e. NodeMCU Developmemt board. NodeMCU Dev Kit/board consist of ESP8266 wifi enabled chip.

- The ESP8266 a lowcost Wi-Fi chip developed by Express if Systems with TCP/IP protocol. For more information about ESP8266, you can refer ESP8266 WiFi Module.

2.SOIL MOISTURE SENSOR

Soil moisture sensors measure the volumetric water content in soil. Since the direct gravimetric measurement of free soil moisture requires removing, drying, and weighing of a sample, soil moisture sensors measure the volumetric water content indirectly by using some other property of the soil, such as electrical resistance, dielectric constant, or interaction with neutrons, as proxy for the moisture content.

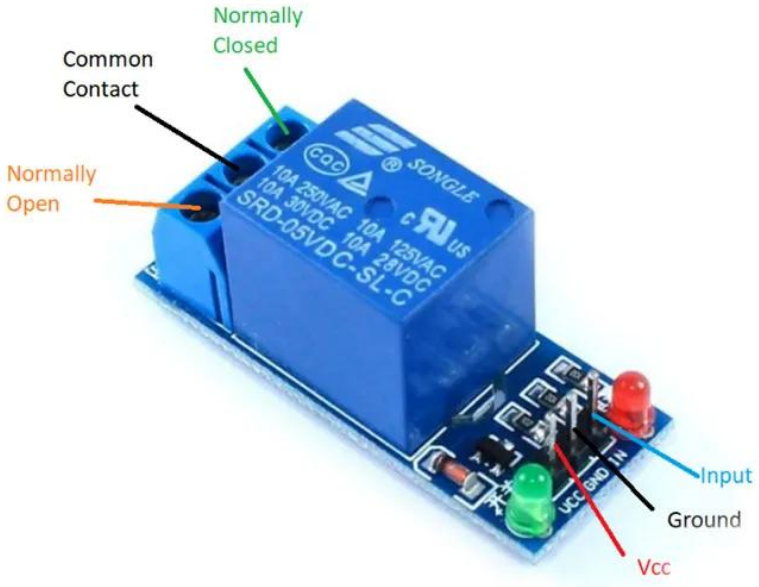
3. MOTOR PUMP

Motor pump is a rotary electrical machines that converts direct current electrical energy into mechanical energy. The most common types rely on the forces produced by magnetic fields.

A DC pump's speed can be controlled over a wide range, using either a variable supply voltage or by changing the strength of current in its field windings.

4.Relay module

A standard and generally used relay is made up of electromagnets which in general used as a switch. Dictionary says that relay means the act of passing something from one thing to another, the same meaning can be applied to this device because the signal received. from one side of the device controls the switching operation on the other side.



5.Jumper Wires-

Jumper wires are short, insulated electrical wires used to connect components together on a breadboard, or between a breadboard and other devices like microcontrollers (NodeMCU/Arduino) in electronic circuits.

They come in three main types:

Male-to-Male (pin to pin)

Male-to-Female (pin to socket)

Female-to-Female (socket to socket)

They allow for quick, flexible, and solderless connections while prototyping and testing circuits.

➤ **Software:**

-Arduino IDE

-Blynk App

-Required Libraries (ESP8266WiFi, LiquidCrystal_I2C, Blynk)

1**. Arduino IDE -**

The Arduino IDE (Integrated Development Environment) is a free, open-source software platform used to write, compile, and upload programs (called sketches) to Arduino boards, NodeMCU (ESP8266/ESP32), and other compatible microcontrollers. It features a simple, user-friendly interface where you write code in a C/C++-based language, then upload it to your hardware via a USB

connection.

2. **Blynk App -**

Blynk is a popular IoT (Internet of Things) mobile application that allows you to remotely control, monitor, and manage hardware devices like NodeMCU (ESP8266/ESP32) via the internet using your smartphone.

With Blynk, you can easily design a custom interface using virtual buttons, sliders, gauges, and displays all without needing complex coding for the mobile side.

3. Required Libraries (ESP8266WiFi, Blynk)-

ESP8266WiFi Library:

The ESP8266WiFi library allows the NodeMCU (ESP8266) to connect to Wi-Fi networks and handle network-related tasks such as creating or joining Wi-Fi connections, making HTTP requests, and more.

Use: Enables Wi-Fi connectivity and networking for ESP8266 projects.

Blynk Library:

The Blynk library connects your hardware (like NodeMCU or ESP32) to the Blynk loT platform, enabling remote control and monitoring via the Blynk mobile app. It manages virtual pins, real-time updates, and server communication.

Use: Integrates IoT projects with the Blynk app for wireless control and monitoring.

CODE FOR THE PROJECT:

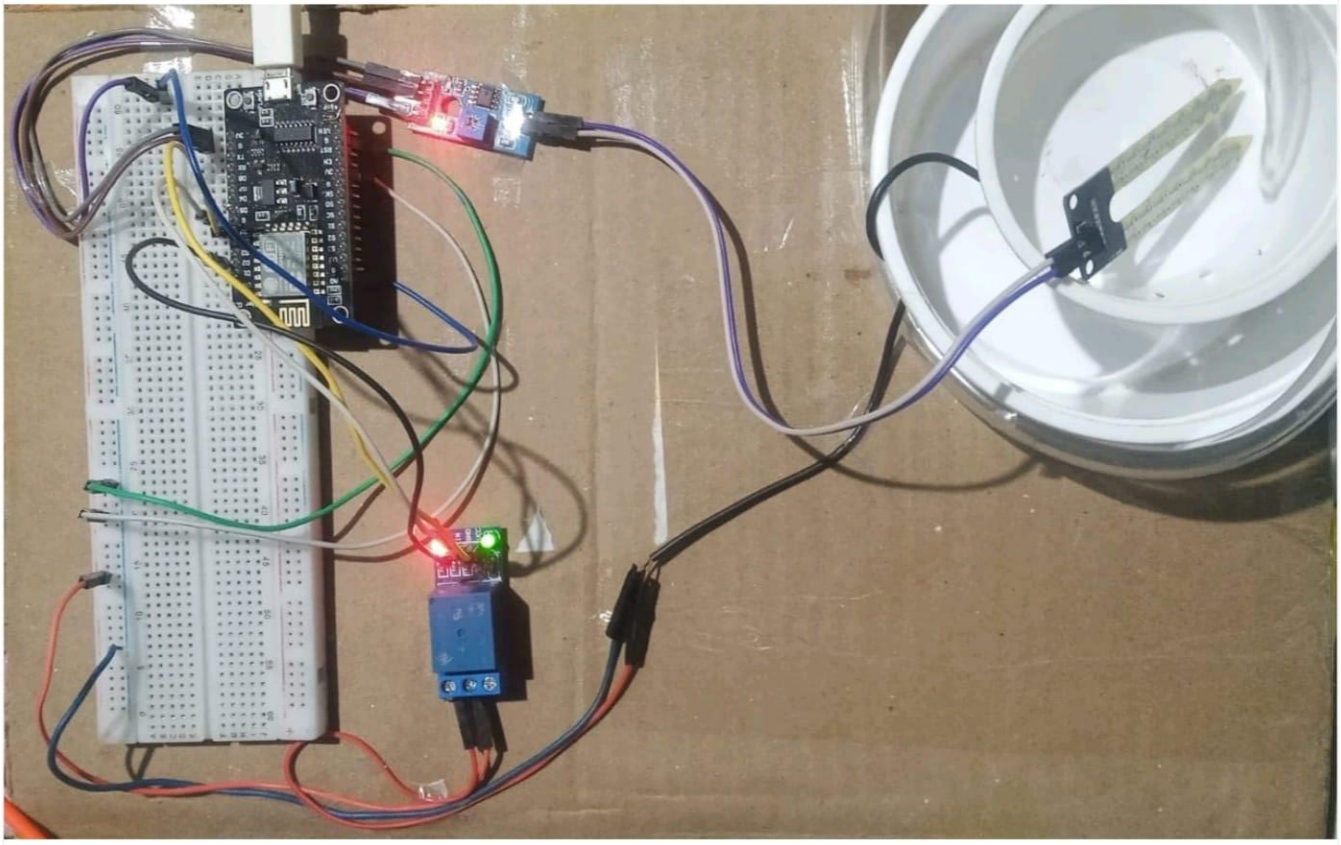
```
#include <dummy.h>

#define BLYNK_TEMPLATE_ID "TMPL3Lfeb0q0H"
#define BLYNK_TEMPLATE_NAME "smart irrigation system"
```

```
#define BLYNK_AUTH_TOKEN
"N3AFJMSzY5fpMeKbwmdIPqBTiaryAhPH"

#define BLYNK_PRINT Serial
#include <dummy.h>

#include <ESP8266WiFi.h>        // WiFi library for ESP8266
#include <BlynkSimpleEsp8266.h> // Blynk library for ESP8266

// Authentication and Wi-Fi credentials
char auth[] = BLYNK_AUTH_TOKEN;
char ssid[] = "Redmi 125G";
char pass[] = "74153475";

// Pins
#define RELAY_PIN 4// Relay connected to D2 (GPIO4)
#define MOISTURE_PIN A0 // Soil moisture sensor connected to A0

// Soil moisture threshold (adjust as necessary)
int moistureThreshold = 500; // This will depend on your sensor

// Blynk virtual button handler for relay
BLYNK_WRITE(V1) {
   int relayState = param.asInt(); // Read relay state (0 or 1)

   if (relayState == 1) {
      digitalWrite(RELAY_PIN, HIGH); // Turn relay ON
      Serial.println("Relay ON");
   } else {
      digitalWrite(RELAY_PIN, LOW); // Turn relay OFF
      Serial.println("Relay OFF");
   }
}

void setup() {
```

```
Serial.begin(115200);

pinMode(RELAY_PIN, OUTPUT); // Set relay pin as OUTPUT
digitalWrite(RELAY_PIN, LOW); // Ensure relay is OFF initially

pinMode(MOISTURE_PIN, INPUT); // Set moisture sensor pin as INPUT

// Connecting to WiFi
Serial.print("Connecting to WiFi...");
WiFi.begin(ssid, pass);
while (WiFi.status() != WL_CONNECTED) {
   delay(300);
   Serial.print(".");
}

Serial.println("\nWiFi Connected!");
Serial.print("IP Address: ");
Serial.println(WiFi.localIP());

// Connecting to Blynk
Blynk.begin(auth, ssid, pass);
Serial.println("Connected to Blynk!");
}

void loop() {
Blynk.run(); // Run Blynk tasks

// Read raw analog value from moisture sensor
int moistureValue = analogRead(MOISTURE_PIN);

// Map raw value to percentage (0–100%)
// Dry soil = high raw value → low percentage
// Wet soil = low raw value → high percentage
int moisturePercent = map(moistureValue, 1023, 0, 0, 100);
```

```
  moisturePercent = constrain(moisturePercent, 0, 100); // Ensure within 0–100 range

  Serial.print("Raw Moisture Value: ");
  Serial.println(moistureValue);
  Serial.print("Soil Moisture (%): ");
  Serial.println(moisturePercent);

  // Send percentage value to Blynk gauge on V8
  Blynk.virtualWrite(V7, moisturePercent);

  // Auto-control relay based on moisture percentage
  if (moisturePercent > 50) { // Below 30% → dry soil
    digitalWrite(RELAY_PIN, HIGH);
    Serial.println("Relay OFF - Moisture Sufficient");
  } else {
    digitalWrite(RELAY_PIN, LOW);
    Serial.println("Relay ON - Moisture Low");

  }
 delay(2000); // Read every 2 seconds
```

RESPECTIVE OUTPUT OF THE PROGRAM:



Viva Questions with Answers

1. What is the purpose of this project?

To automate irrigation by monitoring soil moisture and controlling a water pump through a relay.

2. Why is Blynk used in this project?

Blynk provides a mobile interface to remotely monitor moisture levels and manually control the water pump.

3. What does the moisture sensor measure?

It measures the analog voltage which changes based on the soil moisture content.

4. What is the use of analogRead(A0)?

It reads the raw value from the moisture sensor ranging from 0 (wet) to 1023 (dry).

5. How is the moisture level mapped to percentage?

Using the map() function to convert raw sensor data to a percentage scale (0–100%).

6. Why is digitalWrite(RELAY_PIN, HIGH) used?

To turn the relay ON (depending on your relay logic, it may be active high or low).

7. What happens if moisture is above 50%?

The relay is turned off assuming soil has enough moisture.

8. What does Blynk.virtualWrite(V7, moisturePercent); do?

It sends the calculated moisture percentage to a widget in the Blynk app (e.g., a gauge).

9. What is the delay used for?

To avoid flooding the serial monitor and sensor with too many reads; gives a 2-second gap between each read.