ACKNOWLEDGEMENT

Apart from our efforts, completion of this project is attributed to the encouragement and guidelines of many other contributors. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project work.We would like to place on record our thanks to **Dr. Sachin Chawla,** Dean of GNI, Mullana and **Mr. Gaurav Sharma,** Head of ECE Department for their tremendous support and help. We were motivated whenever we had a meeting with them. Without their encouragement and guidance this work would not have materialized.

We wish to express our deepest gratitude to our Project Guide, **Mr. Pradeep Sandhu**, the Assistant Professor, ECE Department for her sincere and invaluable guidance, suggestions and constant encouragement which was very vital for completion of this project. With profound respect, we would also like to place on record our thanks to **Mr. Gaurav Sharma, Project Coordinator, HOD,** ECE Department for his invaluable guidance, motivation and belief. Without his academic help, constant morale support and guidance, it would have been impossible to complete the project. We also feel obiliged to Mr. Naveen Vaid for his moral support, enlightenment of the subject matter and cooperation at every step without which the satisfactory completion of the work would not have been possible.

We express gratitude to other faculty members of ECE Department, GNI for their intellectual support throughout the course of this work. Finally, we are indebted to all whosoever have contributed in this major project work and friendly stay at GNI, Mullana.

Manish Kumar (6315221)

Nabin Karna (6315223)

Dilshad Ali (6315751)

Shiva (6315234)

ABSTRACT

The Internet of Things (IoT) is the internetworking of physical devices, vehicles and other objects which consists of an embedded system with sensors, actuators and network connectivity that enable to collect and exchange data. The IoT allows objects to be sensed and/or controlled remotely across existing network infrastructure, creating opportunities for more integration of the physical world into computer-based systems, and result in improved accuracy, efficiency and economic benefit. The IoT is a rapidly increasing and promising technology which becomes more and more present in our everyday lives. Furthermore, the technology is an instance of the more general class of cyber-physical systems, which also encompasses technologies such as smart grids, smart homes and smart cities. Considering the high-rate development of IoT technologies, and the significant increment in the number of the connected devices, comprehensive overview of the IoT system aims, architecture, challenges, applications, protocols, and market overview were discussed. In order to give an example of IoT solution, a simple IoT demonstrator was implemented using the current affordable hardware, and cloud efficient software. With this demonstrator, the simplicity and design flexibility of IoT solution were highlighted and two of the IBM IoT software, Node-RED and Bluemix, were examined.

COST ESTIMATION

S. No.	COMPONENTS	COST (Rs.)
1.	LDR Module	120
2.	Gas Sensor	300
3.	Soil Moisture Sensor	450
4.	NodeMCU	560
5.	Arduino	500
6.	Transformer	250
7.	Fan	50
8.	Diodes	10
9.	7805	30
10.	7812	30
11.	Relay	100
12.	BJT	10
13.	Wires and connectors	50
14.	PCB	100
15.	Power Cord	30
16	Modelling Materials	800
TOTAL		**3390**

LIST OF FIGURES

LIST OF ABBREVIATIONS

S.No	Abbreviations	Full Form
1.	IDE	Integrated Development Environment
2.	IC	Integrated Circuit
3.	LED	Light Emitting Diode
4.	API	Application Program Interface

CONTENTS

CHAPTER 1
INTRODUCTION TO IOT?

1.1 WHAT IS IOT?

Internet of Things (IoT) is an ecosystem of connected physical objects that are accessible through the internet. The 'thing' in IoT could be a person with a heart monitor or an automobile with built-in-sensors, i.e. objects that have been assigned an IP address and have the ability to collect and transfer data over a network without manual assistance or intervention. The embedded technology in the objects helps them to interact with internal states or the external environment, which in turn affects the decisions taken.

Things are either sensors or actuators. A sensor is something that tells us about our environment. Think of a temperature sensor, or even the GPS receiver on your mobile phone. Actuators are something that you want to control, things like thermostats, lights, pumps, and outlets. The "Internet of Things" brings everything together and allows us to interact with our things. For example, you could have your thermostat control itself based on where you're located.

Fig 1.1 IOT introduction

1.2 WHY IOT?

An article by Ashton published in the RFID Journal in 1999 said, "If we had computers that knew everything there was to know about things - using data they gathered without any help from us - we would be able to track and count everything, and greatly reduce waste, loss and cost. We would know when things needed replacing, repairing or recalling, and whether they were fresh or past their best. We need to empower computers with their own means of gathering

information, so they can see, hear and smell the world for themselves, in all its random glory." This is precisely what IoT platforms does for us. It enables devices/objects to observe, identify and understand a situation or the surroundings without being dependent on human help.

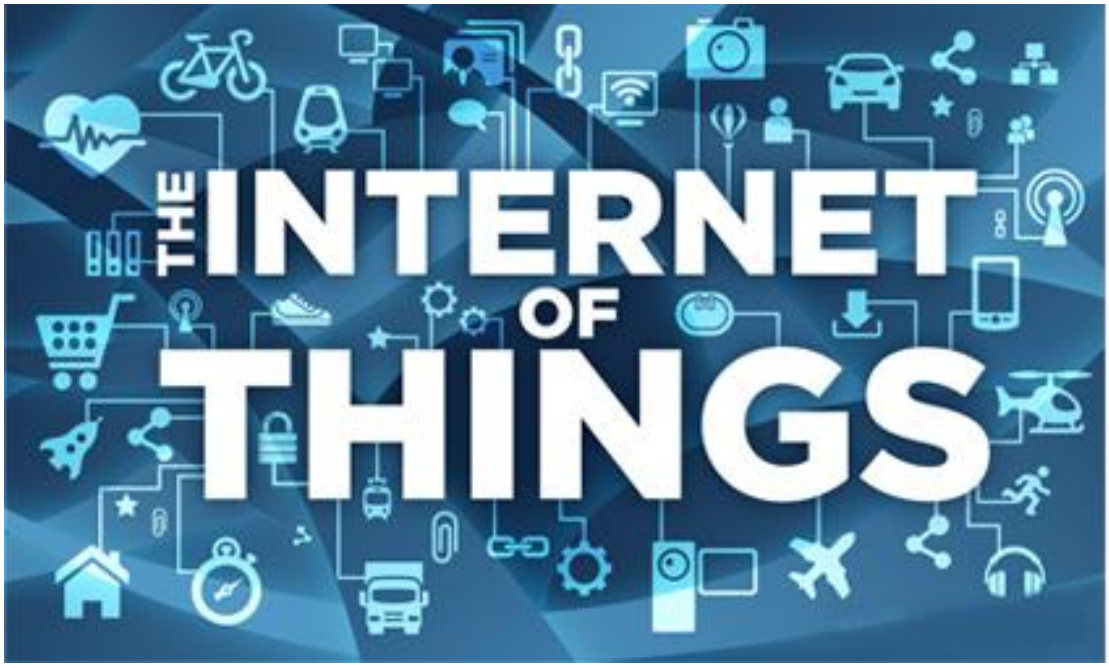


Fig 1.2 Concept of IOT

1.3 SCOPE OF IOT?

Internet of Things can connect devices embedded in various systems to the internet. When devices/objects can represent themselves digitally, they can be controlled from anywhere. The connectivity then helps us capture more data from more places, ensuring more ways of increasing efficiency and improving safety and IoT security.

IoT is a transformational force that can help companies improve performance through IoT analytics and IoT Security to deliver better results. Businesses in the utilities, oil & gas, insurance, manufacturing, transportation, infrastructure and retail sectors can reap the benefits of IoT by making more informed decisions, aided by the torrent of interactional and transactional data at their disposal.

Fig. 1.3 Scope of IOT

1.4 HOW CAN IOT HELP?

IoT platforms can help organizations reduce cost through improved process efficiency, asset utilization and productivity. With improved tracking of devices/objects using sensors and connectivity, they can benefit from real-time insights and analytics, which would help them make smarter decisions. The growth and convergence of data, processes and things on the internet would make such connections more relevant and important, creating more opportunities for people, businesses and industries.

Fig.1.4 Application of IOT

CHAPTER 2
INTRODUCTION TO NODEMCU AND ARDUINO IDE

2.1 WHAT IS NODEMCU?

NodeMCU is a development board featuring the popular ESP8266 WiFi chip. As it turns out, you can program the ESP8266 just like any other microcontroller. Its obvious advantage over the Arduino or PIC is that it can readily connect to the Internet via WiFi. However, the ESP8266 breakout board has limited pins although the chip itself has a lot of output ports. The NodeMCU solves this problem by featuring 10 GPIO pins each capable of using PWM, I2C and 1-wire interface.

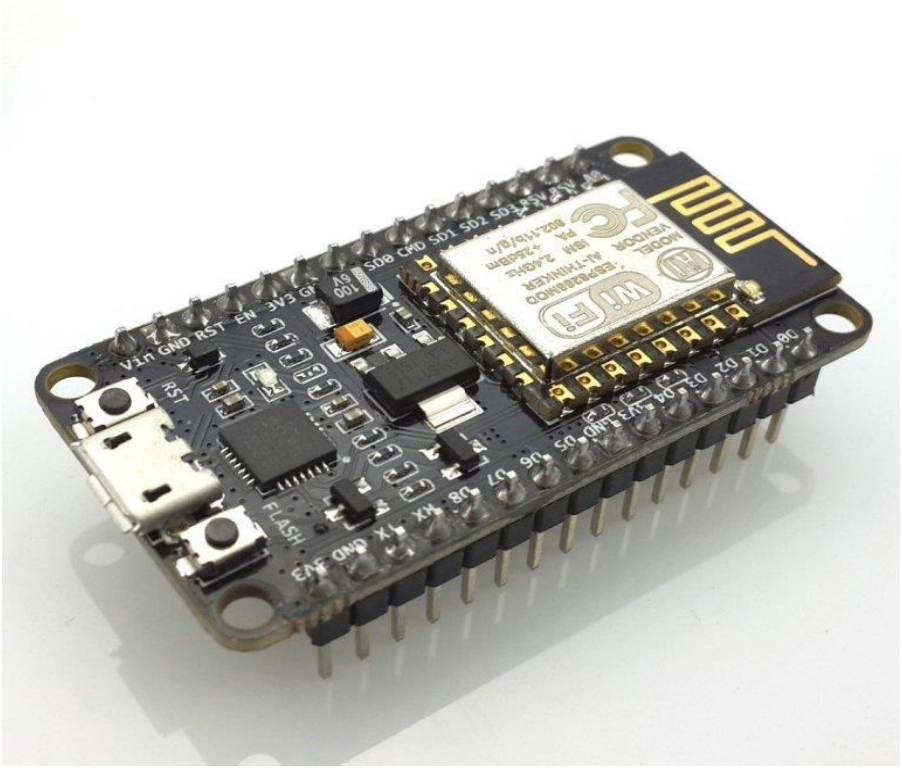


Fig. 2.1 NodeMCU

This ESP8266 development board really looks like an Arduino Nano. Speaking of Arduino, another advantage of this board is that you can connect it directly to your PC or Mac and program it like an Arduino!

Here's how to program the NodeMCU using the Arduino IDE.

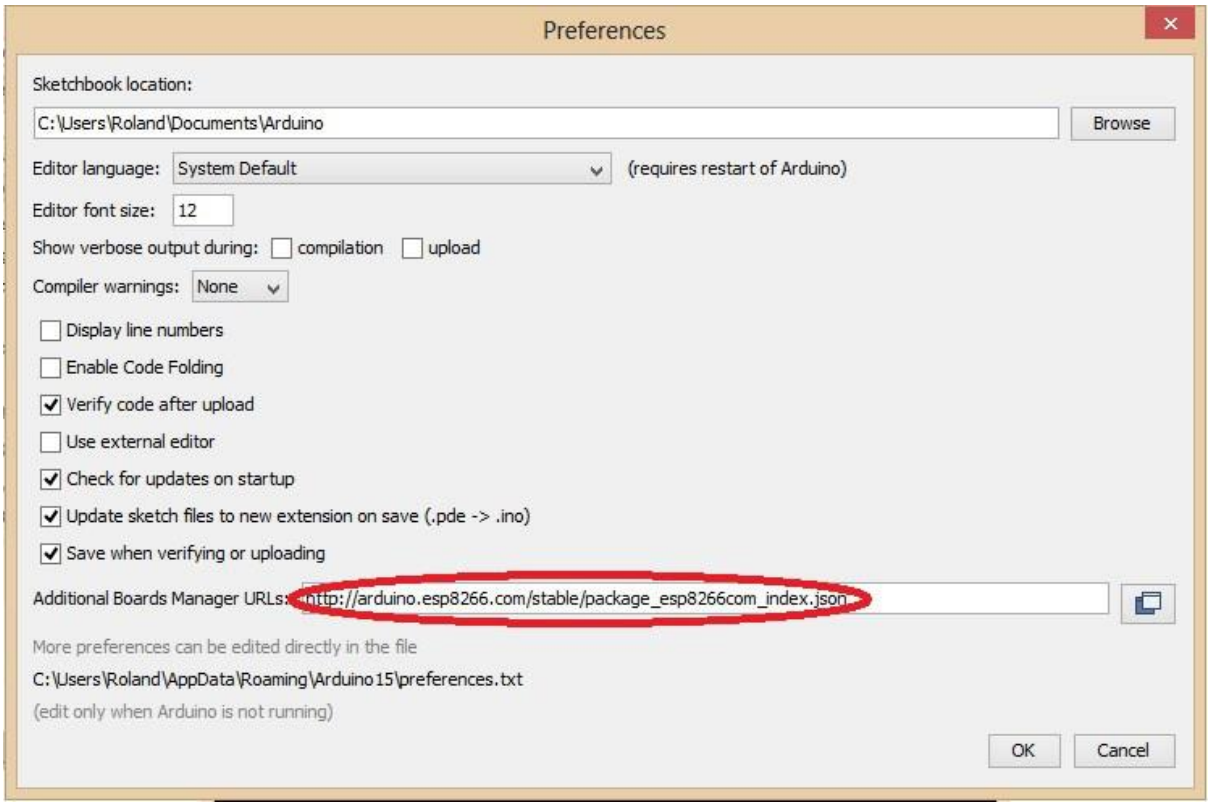
Step 1: Connect your NodeMCU to your computer

You need a USB micro B cable to connect the board. Once you plugged it in, a blue LED will start flashing. If your computer is not able to detect the NodeMCU board, you may need to download the driver on this page.

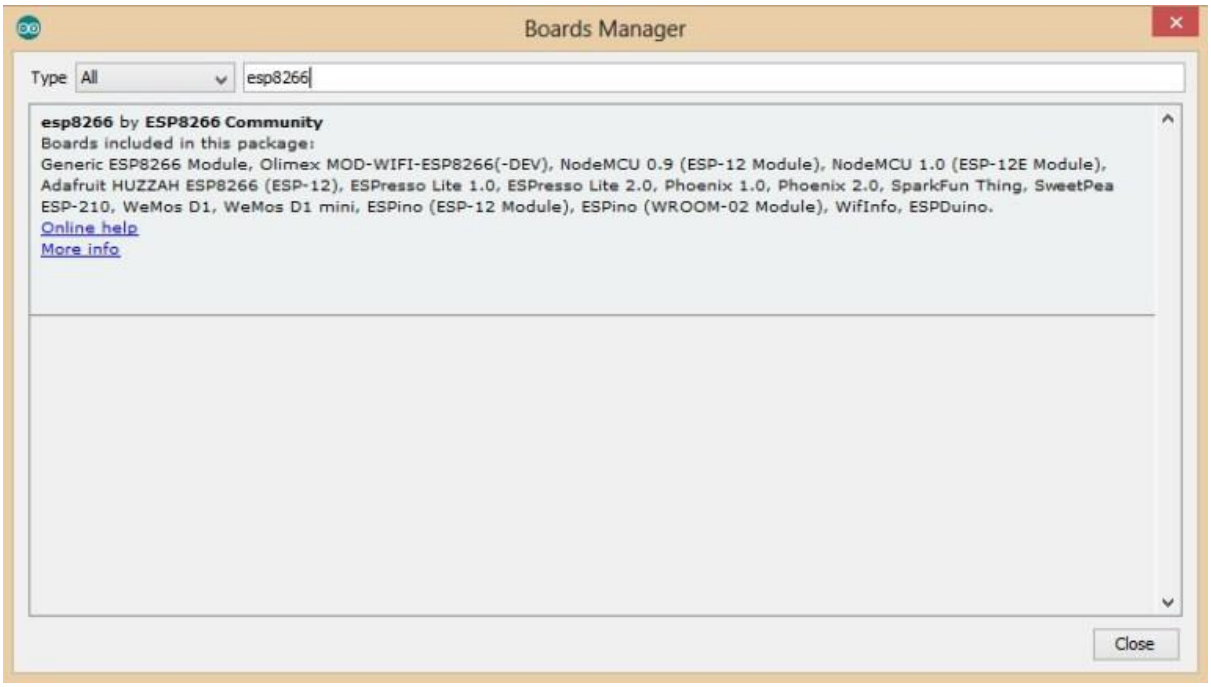
Step 2: Open Arduino IDE

You need to have at least Arduino IDE version 1.6.4 to proceed with this.

Go to File > Preferences. In the "Additional Boards Manager URLs" field, type (or copy-paste) http://arduino.esp8266.com/stable/package_esp8266com_index.json. Click OK!



Then go to Tools > Board > Board Manager. Type "esp8266" in the search field. The entry "esp8266 by ESP8266 Community" should appear. Click that entry and look for the install button on the lower right.



Once the download is complete, you can start coding!

2.2 NODEMCU PINOUT

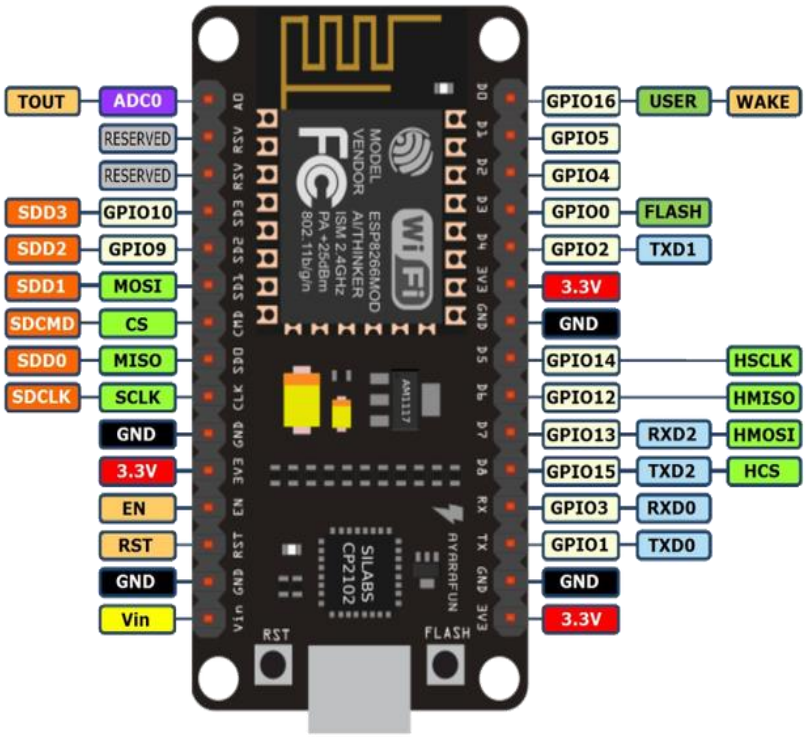


Fig 2.2 NodeMCU Pinout

2.3 WHAT IS AN ARDUINO?

Arduino is an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board.

The Arduino platform has become quite popular with people just starting out with electronics, and for good reason. Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) in order to load new code onto the board -- you can simply use a USB cable. Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program. Finally, Arduino provides a standard form factor that breaks out the functions of the micro-controller into a more accessible package.

What's on the board?

There are many varieties of Arduino boards (explained on the next page) that can be used for different purposes. Some boards look a bit different from the one below, but most Arduinos have the majority of these components in common:

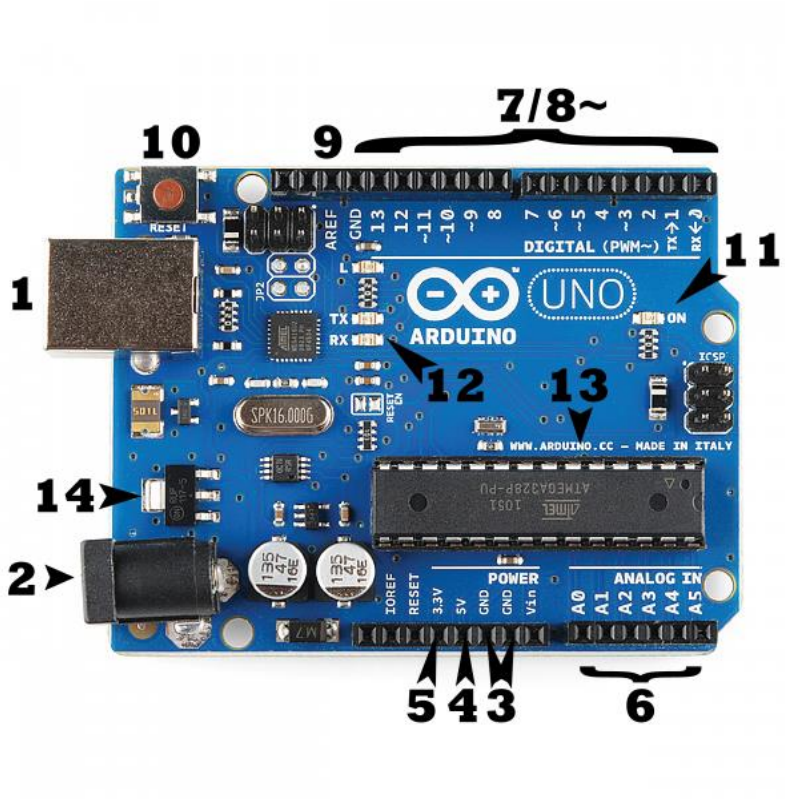


Fig 2.3 Arduino Uno

Power (USB / Barrel Jack)

Every Arduino board needs a way to be connected to a power source. The Arduino UNO can be powered from a USB cable coming from your computer or a wall power supply (like this) that is terminated in a barrel jack. In the picture above the USB connection is labeled (1) and the barrel jack is labeled (2).

The USB connection is also how you will load code onto your Arduino board. More on how to program with Arduino can be found in our Installing and Programming Arduino tutorial.

NOTE: Do NOT use a power supply greater than 20 Volts as you will overpower (and thereby destroy) your Arduino. The recommended voltage for most Arduino models is between 6 and 12 Volts.

Pins (5V, 3.3V, GND, Analog, Digital, PWM, AREF)

The pins on your Arduino are the places where you connect wires to construct a circuit (probably in conjuction with a breadboard and some wire. They usually have black plastic 'headers' that allow you to just plug a wire right into the board. The Arduino has several different kinds of pins, each of which is labeled on the board and used for different functions.

- **GND (3):** Short for 'Ground'. There are several GND pins on the Arduino, any of which can be used to ground your circuit.
- **5V (4) & 3.3V (5):** As you might guess, the 5V pin supplies 5 volts of power, and the 3.3V pin supplies 3.3 volts of power. Most of the simple components used with the Arduino run happily off of 5 or 3.3 volts.
- **Analog (6):** The area of pins under the 'Analog In' label (A0 through A5 on the UNO) are Analog In pins. These pins can read the signal from an analog sensor (like a temperature sensor) and convert it into a digital value that we can read.
- **Digital (7):** Across from the analog pins are the digital pins (0 through 13 on the UNO). These pins can be used for both digital input (like telling if a button is pushed) and digital output (like powering an LED).
- **PWM (8):** You may have noticed the tilde (~) next to some of the digital pins (3, 5, 6, 9, 10, and 11 on the UNO). These pins act as normal digital pins, but can also be used for something called Pulse-Width Modulation (PWM). We have a tutorial on PWM, but for now, think of these pins as being able to simulate analog output (like fading an LED in and out).

- **AREF (9):** Stands for Analog Reference. Most of the time you can leave this pin alone. It is sometimes used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

Reset Button

Just like the original Nintendo, the Arduino has a reset button (10). Pushing it will temporarily connect the reset pin to ground and restart any code that is loaded on the Arduino. This can be very useful if your code doesn't repeat, but you want to test it multiple times. Unlike the original Nintendo however, blowing on the Arduino doesn't usually fix any problems.

Power LED Indicator

Just beneath and to the right of the word "UNO" on your circuit board, there's a tiny LED next to the word 'ON' (11). This LED should light up whenever you plug your Arduino into a power source. If this light doesn't turn on, there's a good chance something is wrong. Time to re-check your circuit!

TX RX LEDs

TX is short for transmit, RX is short for receive. These markings appear quite a bit in electronics to indicate the pins responsible for serial communication. In our case, there are two places on the Arduino UNO where TX and RX appear -- once by digital pins 0 and 1, and a second time next to the TX and RX indicator LEDs (12). These LEDs will give us some nice visual indications whenever our Arduino is receiving or transmitting data (like when we're loading a new program onto the board).

Main IC

The black thing with all the metal legs is an IC, or Integrated Circuit (13). Think of it as the brains of our Arduino. The main IC on the Arduino is slightly different from board type to board type, but is usually from the ATmega line of IC's from the ATMEL company. This can be important, as you may need to know the IC type (along with your board type) before loading up a new program from the Arduino software. This information can usually be found in writing on the top side of the IC. If you want to know more about the difference between various IC's, reading the datasheets is often a good idea.

Voltage Regulator

The voltage regulator (14) is not actually something you can (or should) interact with on the Arduino. But it is potentially useful to know that it is there and what it's for. The voltage regulator does exactly what it says -- it controls the amount of voltage that is let into the Arduino board. Think of it as a kind of gatekeeper; it will turn away an extra voltage that might harm the circuit. Of course, it has its limits, so don't hook up your Arduino to anything greater than 20 volts.

CHAPTER 3

INTRODUCTION TO THINGSPEAK

3.1 WHAT IS THINGSPEAK?

ThingSpeak is an open data platform for the Internet of Things. Your device or application can communicate with ThingSpeak using a RESTful API, and you can either keep your data private, or make it public. In addition, use ThingSpeak to analyze and act on your data. ThingSpeak provides an online text editor to perform data analysis and visualization using MATLAB®. You can also perform actions such as running regularly scheduled MATLAB code or sending a tweet when your data passes a defined threshold. ThingSpeak is used for diverse applications ranging from weather data collection and analysis, to synchronizing the color of lights across the world.

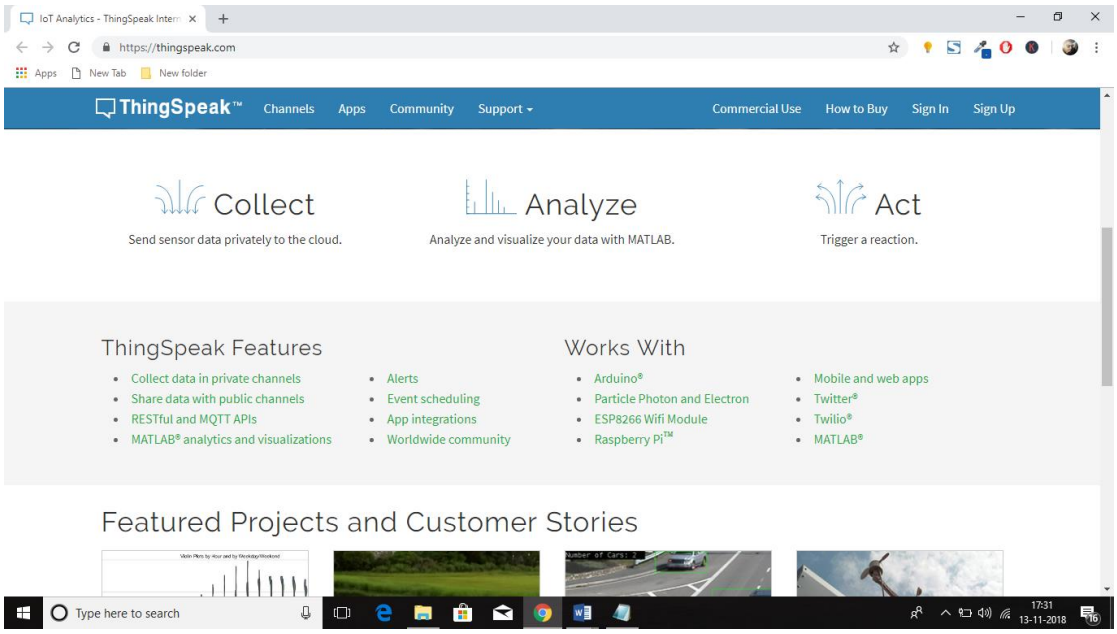


Fig.3.1 ThingSpeak Homepage

At the heart of ThingSpeak is a time-series database. ThingSpeak provides users with free time-series data storage in channels. Each channel can include up to eight data fields. This tutorial provides an introduction to some of the applications of ThingSpeak, a conceptual overview of how ThingSpeak stores time-series data, and how MATLAB analysis is incorporated in ThingSpeak.

ThingSpeak is an application platform for the Internet of Things. ThingSpeak allows you to build an application around data collected by sensors. Features of ThingSpeak include real-time data collection, data processing, visualizations, apps, and plugins.

At the heart of ThingSpeak is a ThingSpeak Channel. A channel is where you send your data to be stored. Each channel includes 8 fields for any type of data, 3 location fields, and 1 status field. Once you have a ThingSpeak Channel you can publish data to the channel, have ThingSpeak process the data, and then have your application retrieve the data.

Here is an example of a light sensor publishing data to ThingSpeak in real-time:

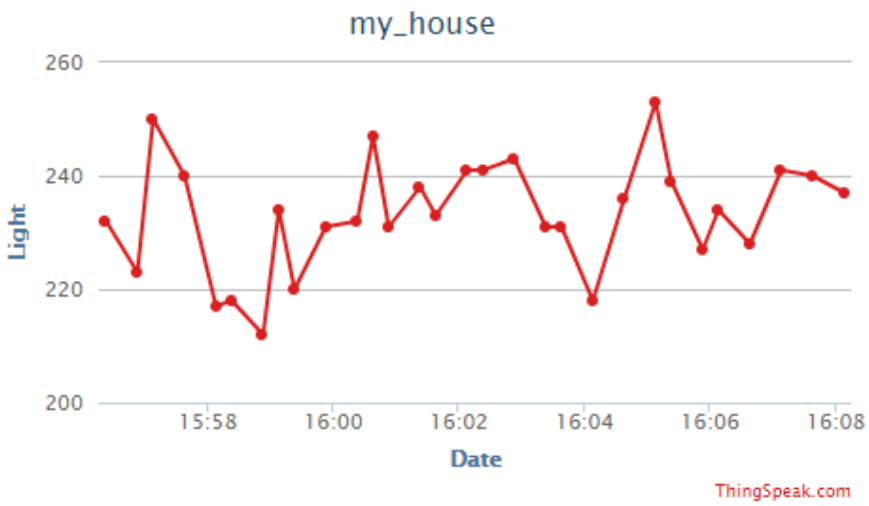
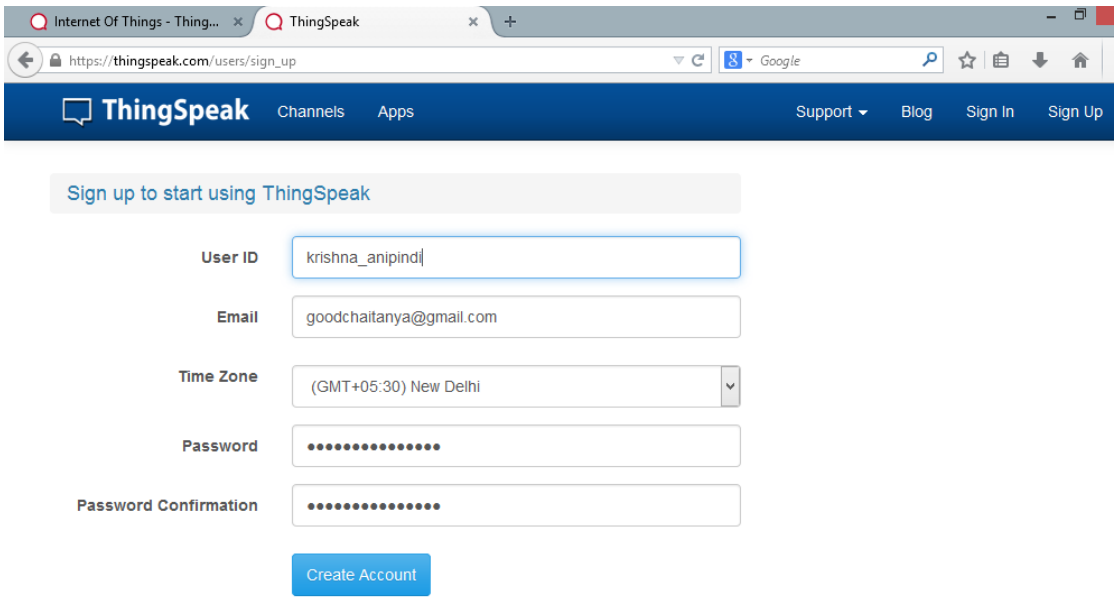


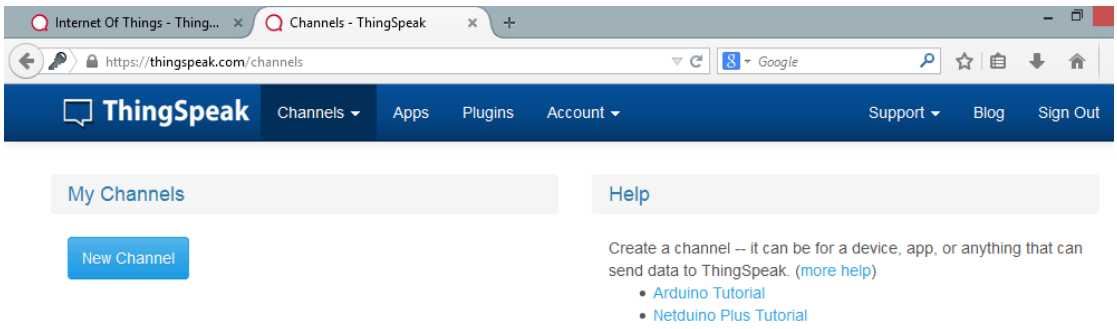
Fig.3.2 Example of Realtime graph on ThingSpeak

3.2 GETTING STARTED

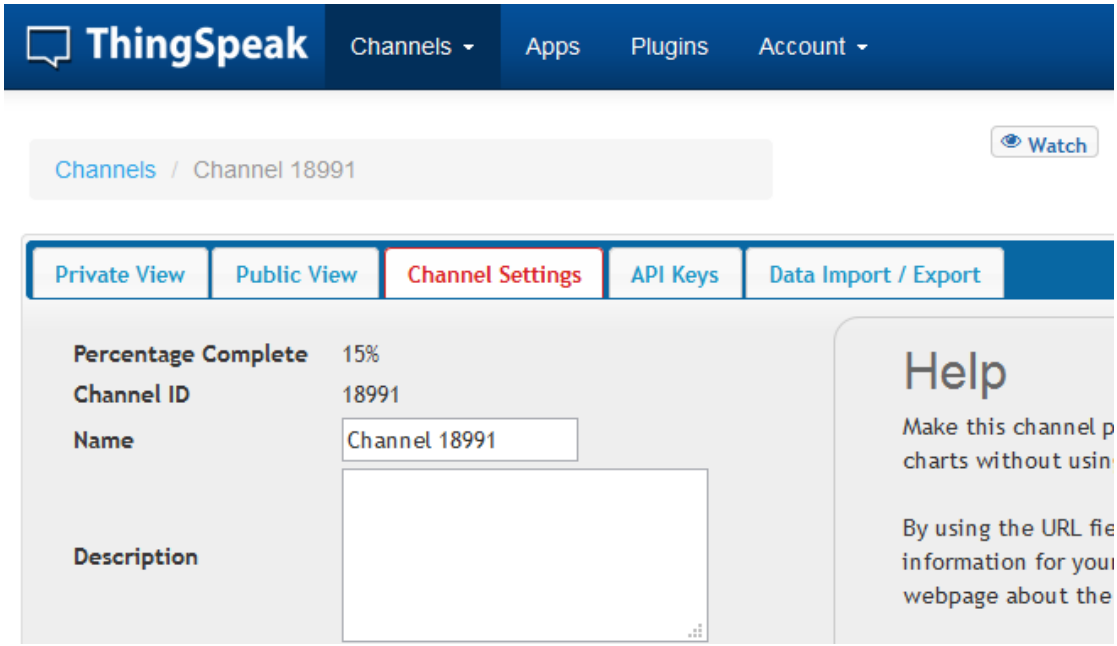
Open https://thingspeak.com/ and click on the 'Get Started Now' button on the center of the page and you will be redirected to the sign-up page(you will reach the same page when you click the 'Sign Up' button on the extreme right). Fill out the required details and click on the 'Create Account' button.



Now you should see a page with a confirmation that the account was successfully created. The confirmation message disappears after a few seconds and the final page should look as in the below screen:

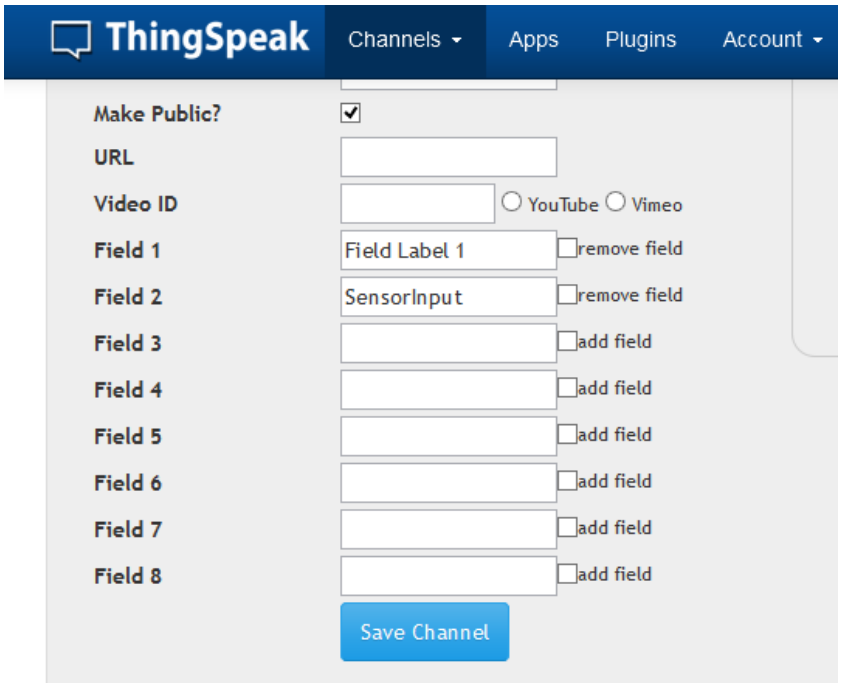


Go ahead and click on ‘New Channel’. You should see a page like the below:

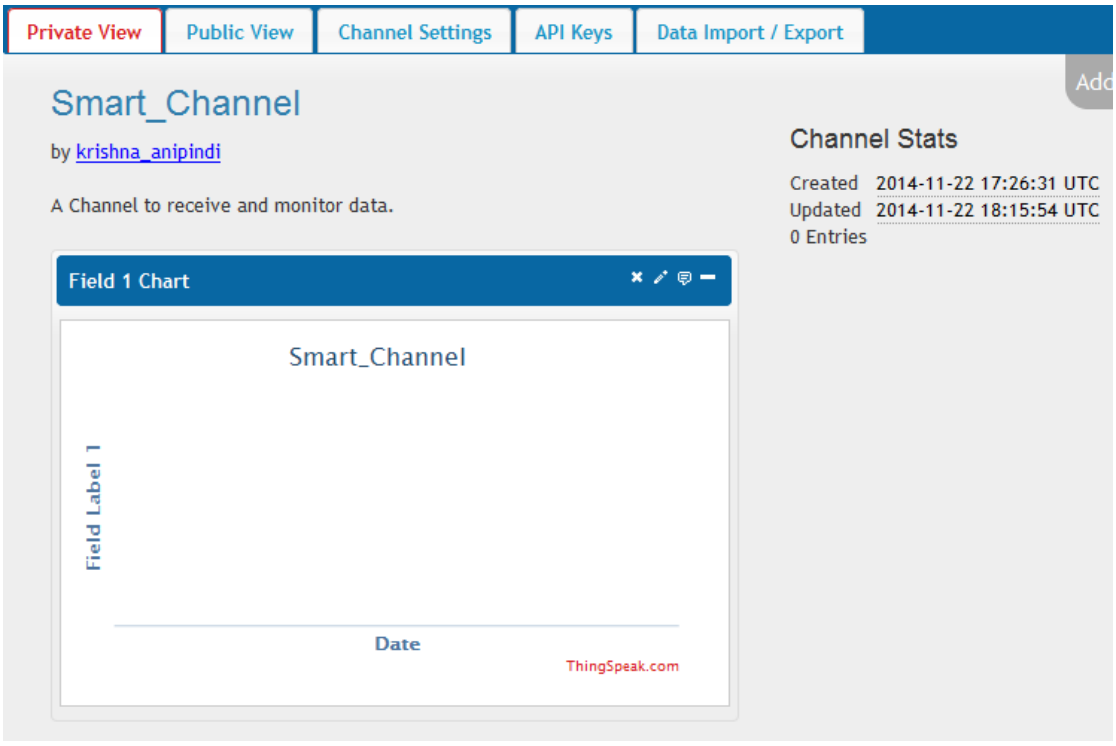


You can change the name to fit your need and you can add a description corresponding to the channel. You can add any other useful description into the metadata field. In the same page, you should see the fields for Latitude, Longitude and Elevation. Also, when you scroll down you should see a check box that says ‘Make Public?’. Let us consider the significance of the various fields and the tabs:

- **Latitude, longitude and elevation -** These fields correspond to the location of a ‘thing’ and are especially significant for moving things.
- **Make Public? -** If the channel is made public, anyone can view the channel's data feed and the corresponding charts. If this check box is not checked, the channel is private, which means for every read or write operation, the user has to pass a corresponding API key.
- **URL -** This can be the URL of your blog or website and if specified, will appear on the public view of the channel.
- **Video ID -** This is the ID corresponding to your YouTube or Vimeo ID. If specified, the video appears on the public view of the channel.
- **Fields 1 to 8 -** These are the fields which correspond to the data sent by a sensor or a ‘thing’. A field has to be added before it can be used to store data. By default, Field 1 is added. In case you try posting to fields that you have not added, your request will still be successful, but you will not be able to see the field in the charts and the corresponding data. You can click on the small box before the ‘add field’ text corresponding to each field to add it. Once you click the ‘add field’ box, a default label name appears in the text box corresponding to each field and the ‘add field’ text changes to ‘remove field’. You can edit the field text that appears by default when a field is added to make more sense. For example, in the below screen, I have modified the text for Field 2 to ‘SensorInput’. To remove a field which is added, just check on the ‘remove field’ box. Once you click this, the text ‘remove field’ changes back to ‘add field’ and the corresponding field text is cleared.



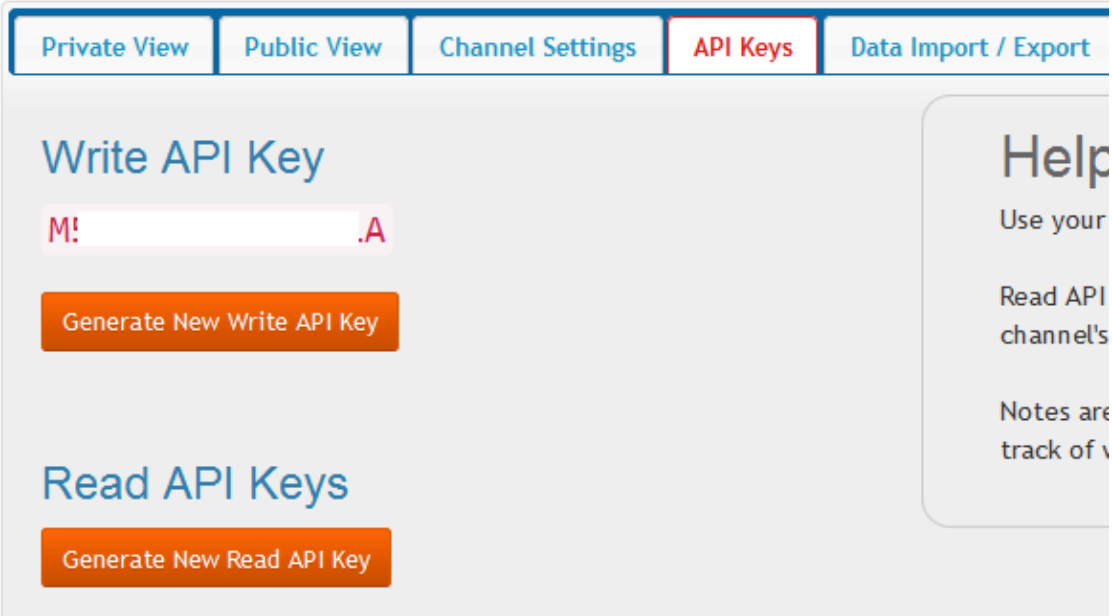
Once you have edited the fields, click on ‘Save Channel’ button. You should now see a page like the below in which the ‘Private View’ tab is defaulted:



The Private View shows a chart corresponding to each of the fields that we have added. Now click on the ‘Public View’ tab. This should look exactly similar to the what we see in the ‘Private View’ tab since our channel is public. In case your channel is not public('make public' check box not checked in the ‘channel settings’ tab), the public view tab shows a message that ‘This channel is not public’.

Now click on the ‘API Keys’ tab. You should see a screen similar to the below. The write API key is used for sending data to the channel and the read API key(s) is used to read the channel data. When we create a channel, by default, a write API key is generated. We generate read API keys by clicking the ‘Generate New Read API Key’ button under this tab.You can also add a note corresponding to each of the read API keys.

Note: Please note that clicking on the ‘Generate New Write API Key’ will over-write the previous key. You will only have one Write API key at any point of time. Also, in case your channel is private, others can only view the channel’s feed and charts by using a Read API key. Please share the Read API keys with people who are approved and authorized to view your channel.



Now click on the ‘Data Import/Export’ tab and you should see a screen similar to the below. This tab is used to import the ‘Comma Separated Values(CSV)’ data from a file into the channel. You can also download the channel’s feed from here in CSV format. This tab also outlines how to send and view data by providing the URIs to the send and view APIs.

CHAPTER 4

VARIOUS SENSORS AND COMPONENTS USED

4.1 DHT11 TEMPERATURE AND HUMIDITY SENSOR

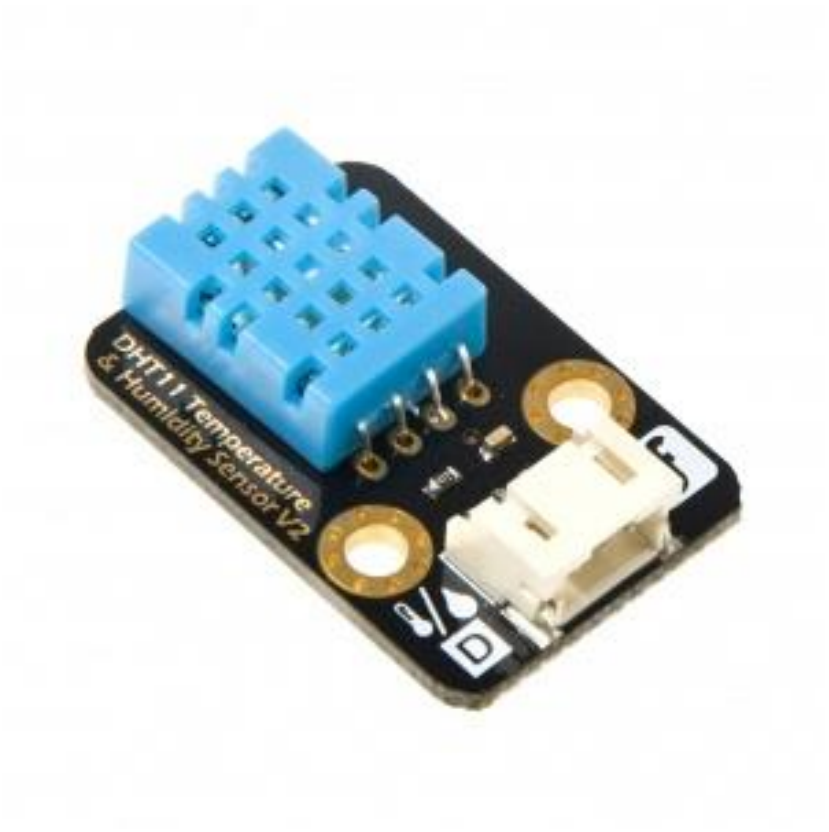


Fig. 4.1 DHT11

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

Each DHT11 sensors features extremely accurate calibration of humidity calibration chamber. The calibration coefficients stored in the OTP program memory, internal sensors detect signals in the process, we should call these calibration coefficients. The single-wire serial interface system is integrated to become quick and easy. Small size, low power, signal

transmission distance up to 20 meters, making it a variety of applications and even the most demanding applications. The product is 4-pin single row pin package. Convenient connection, special packages can be provided according to users need.

SPECIFICATION

Supply Voltage: +5 V

Temperature range :0-50 °C error of ± 2 °C

Humidity :20-90% RH ± 5% RH error

Interface: Digital

4.2 LDR?

LDR an acronym for light dependent resistor is a resistor whose value varies with the amount of light falling on it. It is used as a sensor to detect the light conditions in the surrounding atmosphere. The resistance of the LDR is very high of the order of Mega ohms in absence of light and it drops to a few hundred ohms or even lower in the presence of light.

Fig. 4.2 LDR

The most common type of LDR has a resistance that falls with an increase in the light intensity falling upon the device (as shown in the image above). The resistance of an LDR may typically have the following resistances:

Daylight = 5000Ω

Dark = 20000000Ω

You can therefore see that there is a large variation between these figures. If you plotted this variation on a graph you would get something similar to that shown by the graph shown above.

4.3 SOIL MOISTURE SENSOR

Fig. 4.3 Soil Moisture Sensor

The Soil Moisture Sensor is used to measure the volumetric water content of soil. This makes it ideal for performing experiments in courses such as soil science, agricultural science, environmental science, horticulture, botany, and biology.

DESCRIPTION

The Soil Moisture Sensor uses capacitance to measure the water content of soil (by measuring the dielectric permittivity of the soil, which is a function of the water content). Simply insert this rugged sensor into the soil to be tested, and the volumetric water content of the soil is reported in percent.

SPECIFICATIONS

- Range: 0 to 45% volumetric water content in soil (capable of 0 to 100% VWC with alternate calibration)
- Accuracy: ±4% typical
- Typical Resolution: 0.1%
- Power: 3 mA @ 5VDC
- Operating temperature: –40°C to +60°C
- Dimensions: 8.9 cm × 1.8 cm × 0.7 cm (active sensor length 5 cm)

4.4 GAS SENSOR (MQ6)

The MQ6 (LPG Gas Sensor) is a simple-to-use liquefied petroleum gas (LPG) sensor. It can be used in gas leakage detecting equipment in consumer and industry applications,this sensor is suitable for detecting LPG, iso-butane, propane, LNG. Avoid the noise of alcohol, cooking fumes and cigarette smoke. The sensitivity can be adjusted by the potentiometer.

SPECIFICATION

- Power supply needs: 5V
- Interface type: Analog
- Pin Definition: 1-Output 2-GND 3-VCC
- High sensitivity to LPG, iso-butane, propane
- Small sensitivity to alcohol, smoke
- Fast response
- Stable and long life
- Simple drive circuit
- Size: 40x20mm

PIN DEFINITION

1. Signal Output
2. GND
3. Power

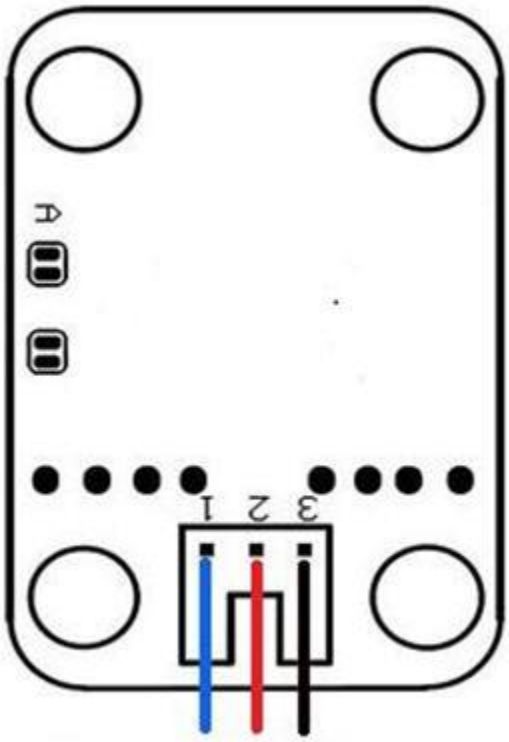


Fig. 4.4 Gas Sensor (MQ6)

4.5 RELAY

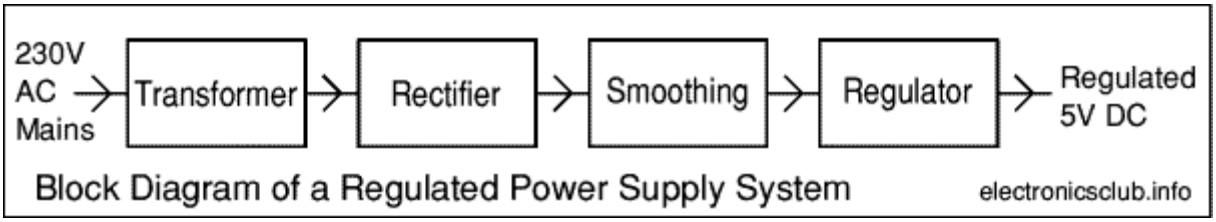
A relay is classified into many types, a standard and generally used relay is made up of electromagnets which in general used as a switch. Dictionary says that relay means the act of passing something from one thing to another, the same meaning can be applied to this device because the signal received from one side of the device controls the switching operation on the other side. So relay is a switch which controls (open and close) circuits electromechanically. The main operation of this device is to make or break contact with the help of a signal without any human involvement in order to switch it ON or OFF. It is mainly used to control a high powered circuit using a low power signal. Generally a DC signal is used to control circuit which is driven by high voltage like controlling AC home appliances with DC signals from microcontrollers.

Fig. 4.5 Relay

4.6 BASIC POWER SUPPLY

There are many types of power supply. Most are designed to convert high voltage AC mains electricity to a suitable low voltage supply for electronics circuits and other devices. A power supply can by broken down into a series of blocks, each of which performs a particular function.

For example a 5V regulated supply:



- Transformer - steps down high voltage AC mains to low voltage AC.
- Rectifier - converts AC to DC, but the DC output is varying.
- Smoothing - smooths the DC from varying greatly to a small ripple.
- Regulator - eliminates ripple by setting DC output to a fixed voltage.

CHAPTER 5

ETCHING A PRINTED CIRCUIT BOARD (PCB)

5.1 INPUT YOUR DESIGN INTO A PCB DESIGN SOFTWARE

Using a decent PCB Design platform can save a lot of time while designing the layout of your PCB. Programs such as Eagle and EasyEDA allow you to design your schematic then build directly from this schematic. This ensures component sizes and connection points are accurate and accounted for on the PCB. Another timesaver of this software is the autorouter tool. This will lay out all traces in paths that only interfere with components and other traces that are connected. After the PCB has been laid out and the circuit connections have traced, verify the circuit one final time prior to printing. Make sure to print using a 1:1 scale so the layout dimensions are accurate.

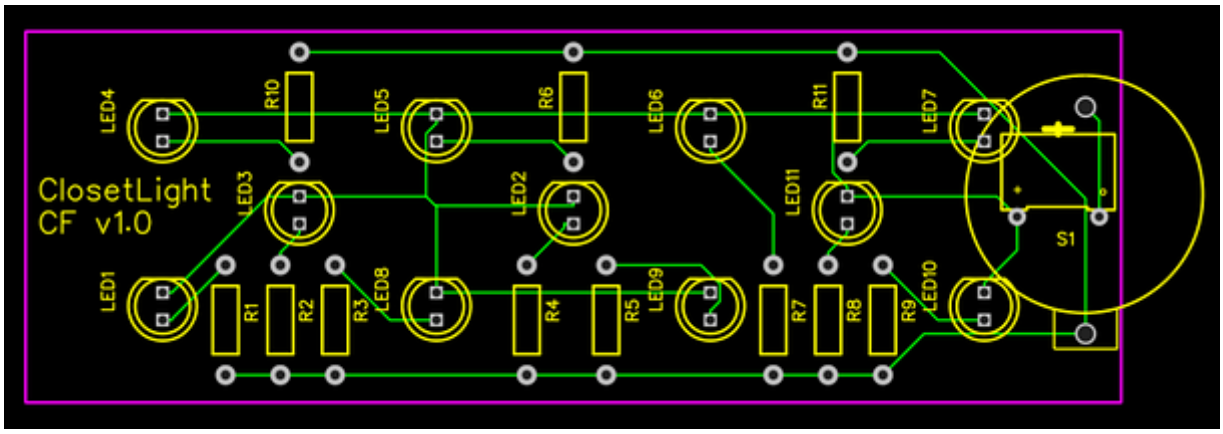


Fig. 5.1 Software Simulated Design

5.2 TRANSFER THE LAYOUT TO THE BOARD

To ensure an accurate transfer, tape this print out over top of a copper clad laminate PCB. Using a pushpin, stab through the paper where each component lead will penetrate the board. This will leave dots on the copper where these components go. The small indentation on the copper will also help the drill bit find the exact location the lead will go. Remove the printed sheet and make sure all components are marked on the copper. Use a 1/32” (maximum) bit to drill out the holes that were marked on the board. Once all the holes are drilled, clean the top of the copper plate with a piece of sandpaper. Using a permanent marker, draw to traces between all the components referencing the printed PCB layout. Allow some time for the ink to dry and touch up the board where the marker is faint.York, Vermont and North Carolina and our projects take us world-wide.

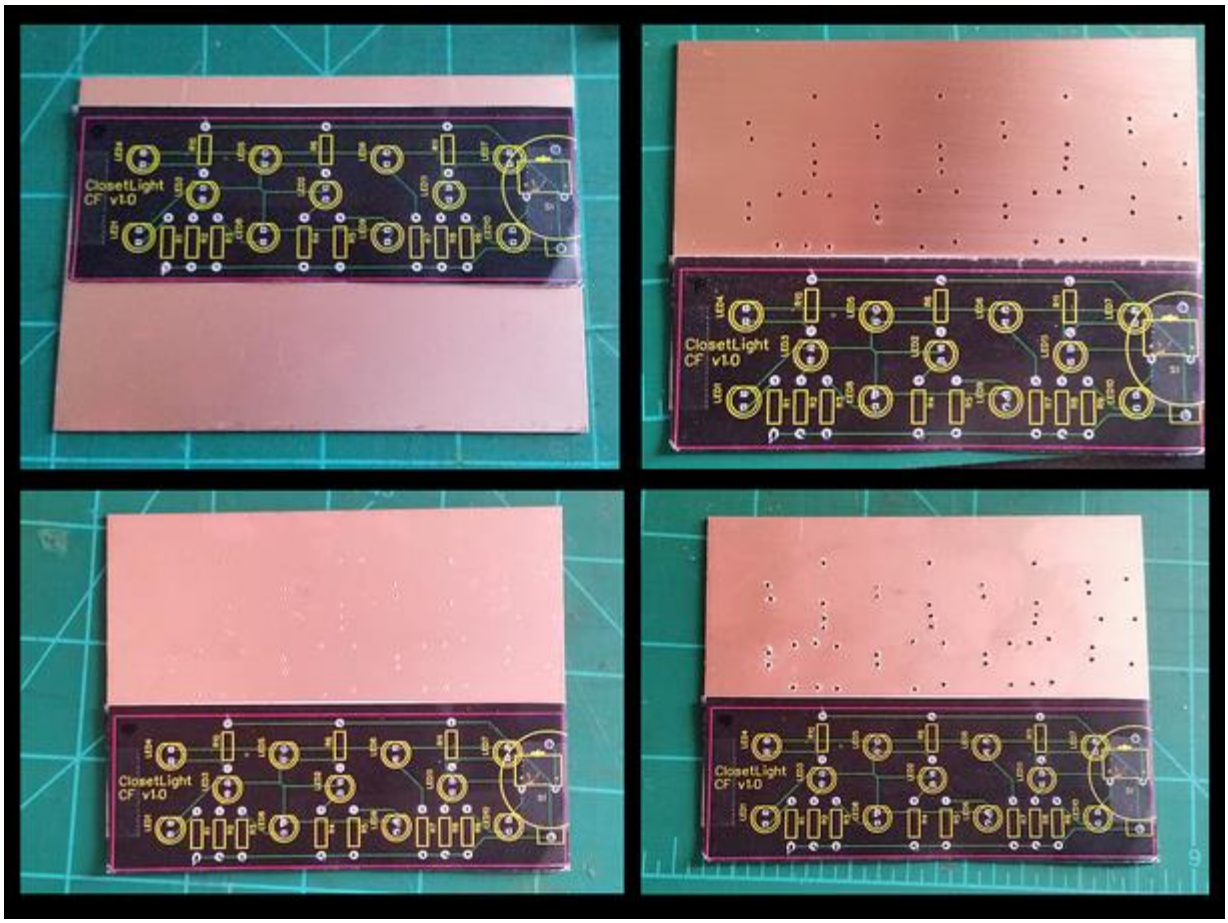


Fig. 5.2 Transfer Of Layout To The Board

5.3 THE ETCHING PROCESS

Once the second coat of marker is completely dry it is time to give the board a bath in ferric chloride. Ferric chloride is a corrosive, acidic chemical compound that will eat away all copper on the board that is not protected by the marker's ink. Pour a modest amount of ferric chloride into a plastic container with a lid; just enough to cover the board completely. Let the board soak for 10 minutes, make sure the lid is properly secured and agitate it every few minutes by rocking the container back and forth. After 10 minutes inspect the board and if no copper is visible, remove the board while wearing a latex glove. Pat the board dry with a disposable rag to remove all ferric chloride from the board. Rinse the board with acetone that will make quick work of the marker ink to reveal your unharmed traces. The etching process is complete!

Fig. 5.3 Etching Process

5.4 POPULATE THE BOARD AND TEST THE CIRCUIT

After the board has been etched, use a multimeter to do a continuity test. Make sure all traces were successful and also begin and terminate in the correct locations before applying power to the circuit. Once all traces have been verified, add the components to their correct position and solder them in. Apply voltage and watch in awe as your perfectly etched circuit functions exactly as planned!

CHAPTER 6
PROJECT DESCRIPTION

6.1 INTRODUCTION:

In today's greenhouses, many parameter measurements are required to monitor and control for the good quality and productivity of plants. But to get the desired results there are some very important factors which come into play like Temperature, Humidity, Light and Water, which are necessary for a better plant growth. Keeping these parameters in mind our team will be building an Automatic Green House Controlling and Monitoring System over IOT using odeMcu & Arduino. This system is very efficient for growing good quality plants. The other important part of this project is that it is fully automatic. Arduino automatically turns on and turns off the appliances.

6.2 BLOCK DIAGRAM:

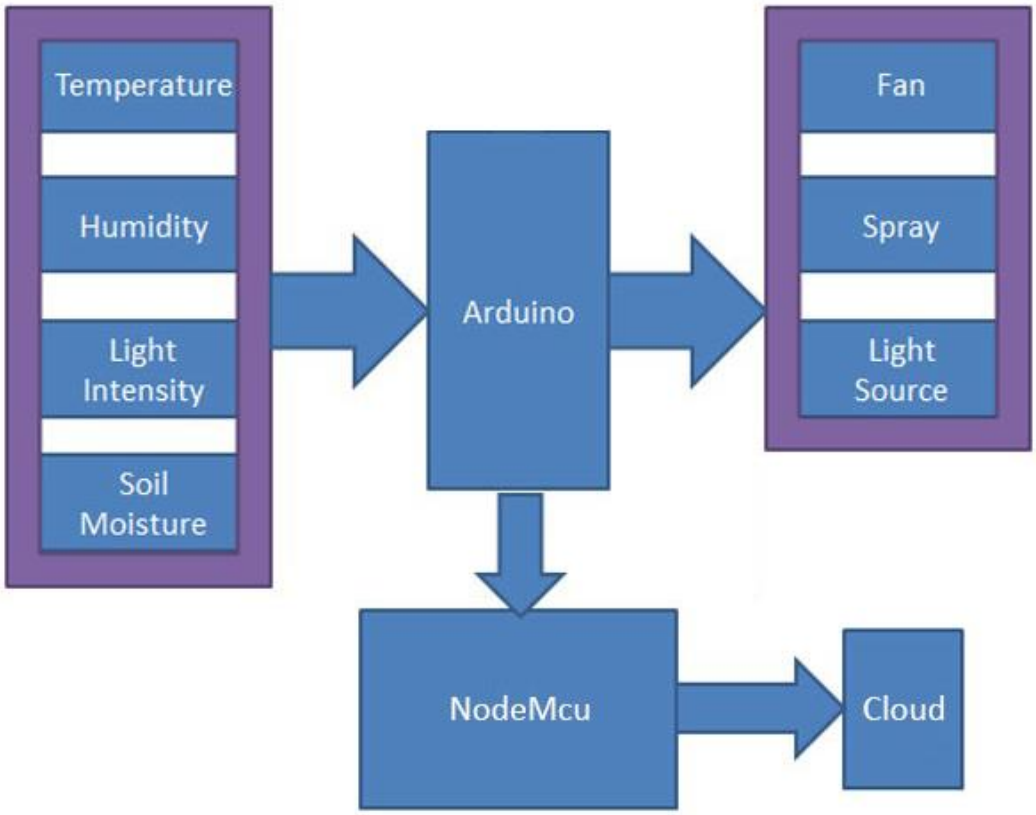


Fig. 6.1 Block Diagram of complete project

6.3 THEORY:

These days NodeMcu is widely used in Green House Monitoring. Here in this project we can keep information about the effects of climate on plants. The system shall also demonstrate climatic changes which affect the plant in its productivity and quality etc. The main purpose of coming up with this project is to build an Automatic Green House Monitoring in which NodeMcu sends the information about Temperature, Humidity, Light intensity, Soil moisture and status of appliances (Fan, Sprays, Artificial Lights and Water pump)to the cloud and which are connected with circuit for controlling Green House effects or Green House parameters (Temperature, Humidity, Light intensity and Water supply for plants).

The four parameters that are essential to discuss are:

TEMPERATURE:

The temperature sensor is used for sensing temperature. When temperature exceeds from a defined level or critical level, the system automatically turns on the fan and a message is also sent to the owner or the operator with information of all parameters (Temperature, Humidity, Light intensity and Electrical appliance on off status). And when the temperature comes in normal range or comes below the defined level the fan turns off automatically.

HUMIDITY:

Humidity is measured by using the humidity sensor. If the humidity of the environment is below the defined levels, sprays are automatically turned on and if the humidity level exceeds from the defined level sprays are automatically turned off. But here in this project instead of a spray we have used CFL light to denote the spray. A status or notification message is also sent to the cloud.

LIGHT INTENSITY:

Light intensity is an important factor for the plant growth. If the light intensity is low then it affects the growth of the plants. To resolve the problem of low light, artificial lights are used. Here in this project 100 watt bulb is used for demonstration. When light intensity is lower than a defined level, the artificial lights turns on, and when the light intensity comes in normal range artificial lights automatically turns off and a notification is sent to cloud. For detecting light intensity LDR is used. Generally light intensity is measured in LUX and therefore for demonstration 100 LUX light is used as defined or threshold level. If light intensity exceeds from 100 LUX, the artificial lights automatically turns on.

SOIL MOISTURE:

Water supply for plants is very important for good growth. So here in this demonstration I have used a water pump and a soil moisture sensor, for detecting soil moisture. Two probes of soil-moisture-sensors are used and placed in soil. When the sensor does not sense moisture in soil then the system turns on the water pump until it reaches the required level. A notification is also sent to the owner with status of water pump like Motor On or Motor Off. Here for sensing soil moisture a transistor is used as a switch.

6.4 CIRCUIT LAYOUT

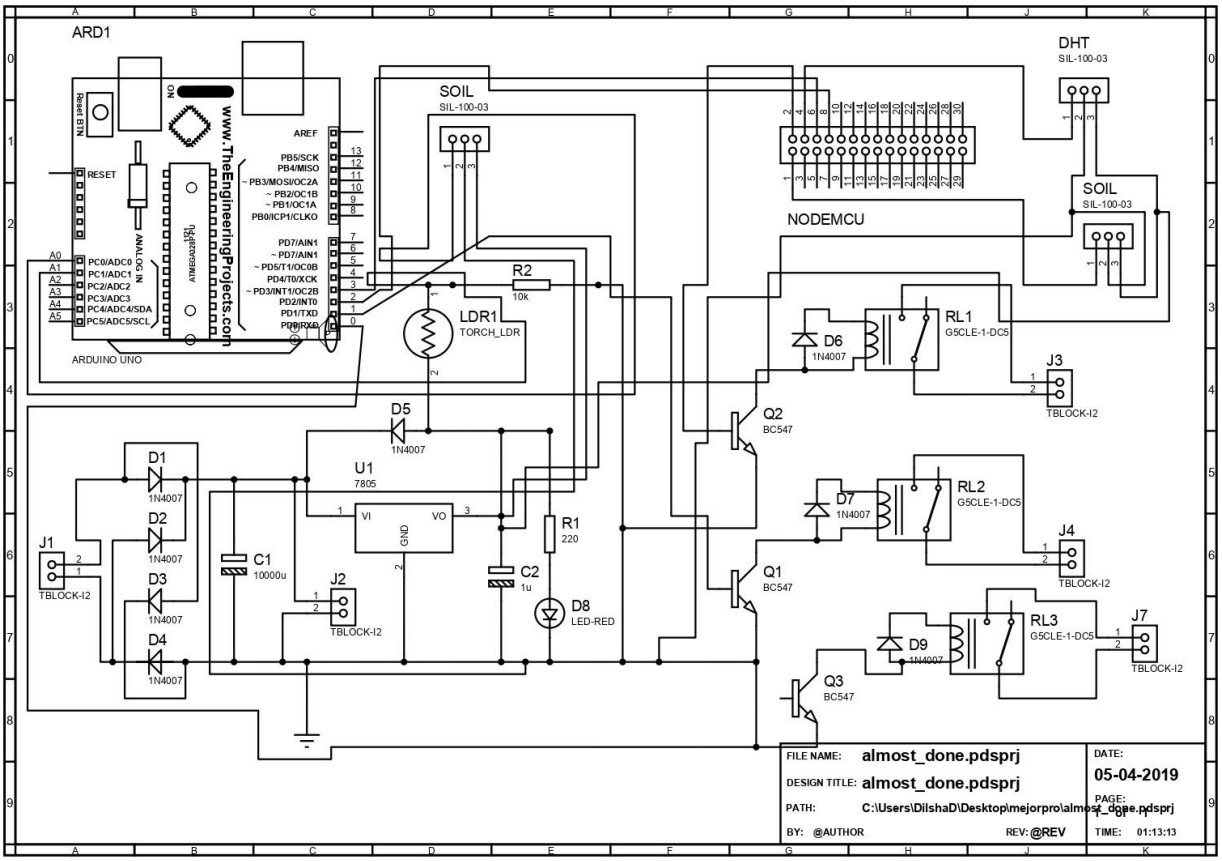


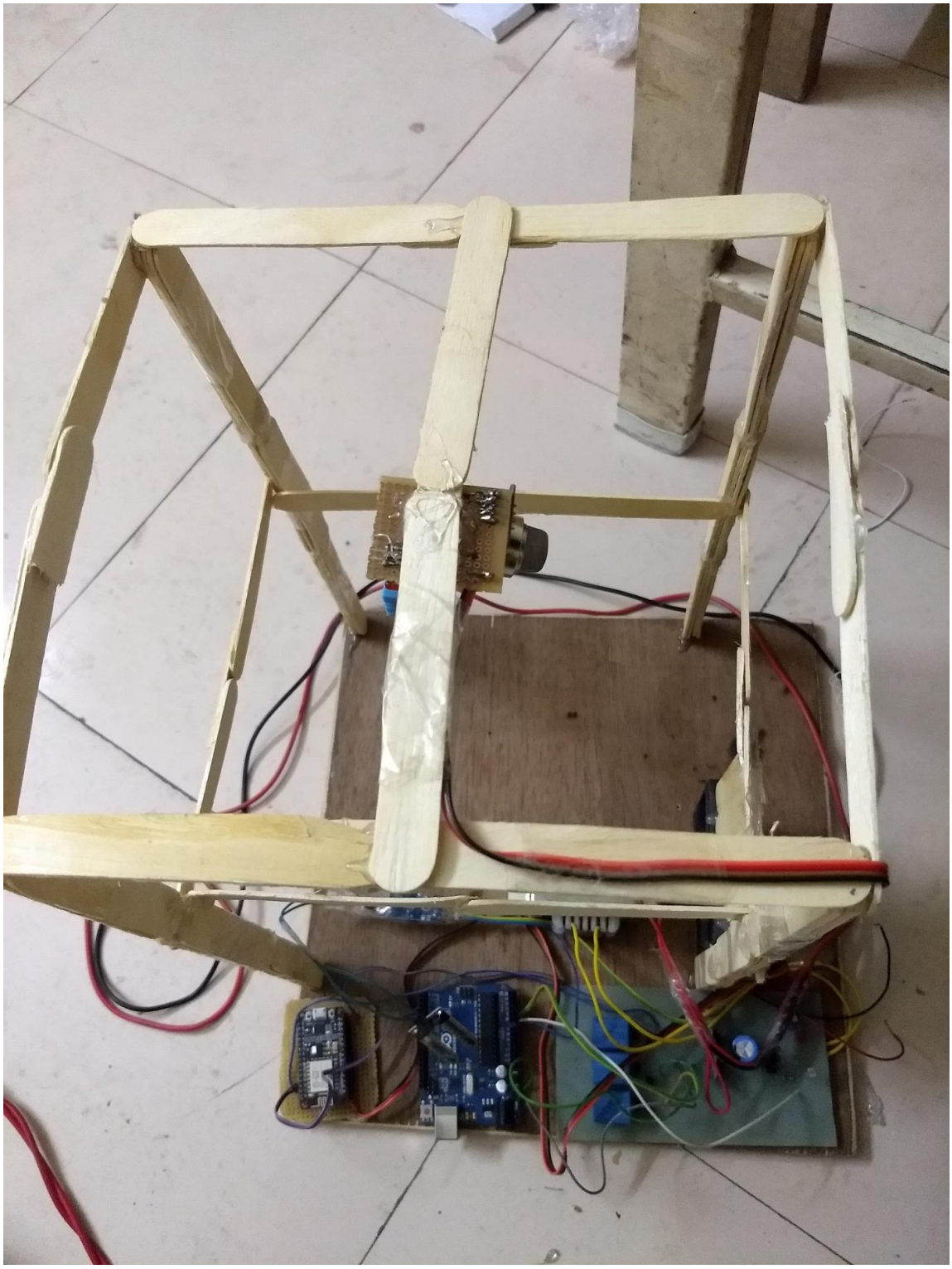
Fig. 6.2 Circuit Diagram

6.5 GETTING API KEY

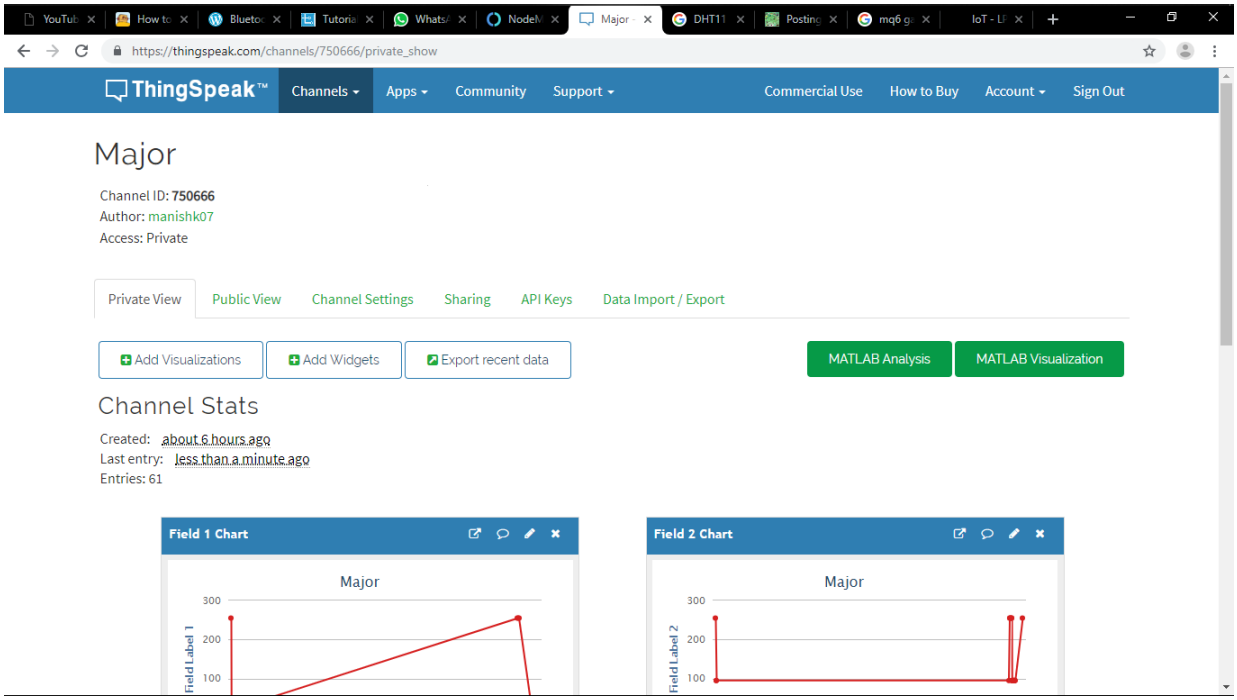
1. Go to https://thingspeak.com/ and create an account if you do not have one. Login to your account.
2. Create a new channel by clicking on the button.Enter basic details of the channel.Than Scroll down and save the channel.

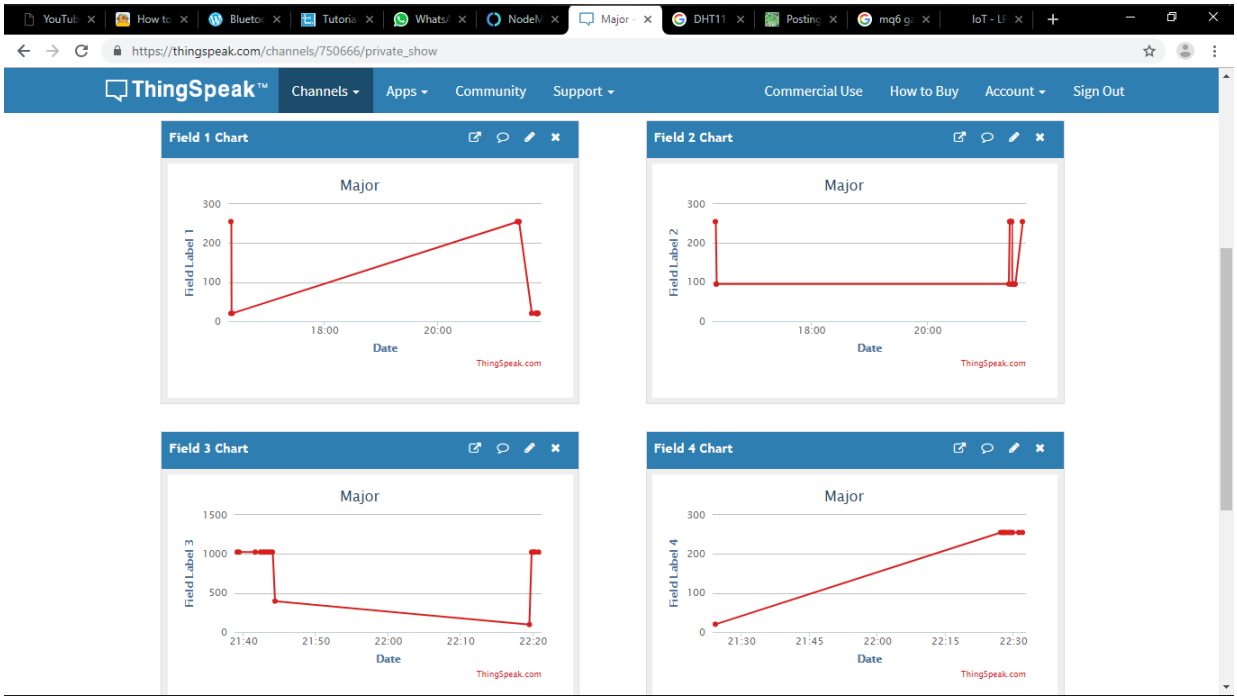
3. Channel Id is the identity of your channel. Note down this. Than go to API keys copy and paste this key to a separate notepad file will need it later.

6.6 PROJECT IMAGE



6.7 OUTPUT


https://thingspeak.com/channels/750666/private_show
ThingSpeak
Channels
Apps
Community
Support
Commercial Use
How to Buy
Account
Sign Out
Major
Channel ID: 750666
Author: manishk07
Access: Private
Private View
Public View
Channel Settings
Sharing
API Keys
Data Import / Export
Add Visualizations
Add Widgets
Export recent data
MATLAB Analysis
MATLAB Visualization
Channel Stats
Created: about 6 hours ago
Last entry: less than a minute ago
Entries: 61
Field 1 Chart
Major
Field Label 1
Field 2 Chart
Major
Field Label 2


https://thingspeak.com/channels/750666/private_show
ThingSpeak
Channels
Apps
Community
Support
Commercial Use
How to Buy
Account
Sign Out
Field 1 Chart
Major
Field Label 1
Date
ThingSpeak.com
Field 2 Chart
Major
Field Label 2
Date
ThingSpeak.com
Field 3 Chart
Major
Field Label 3
Date
ThingSpeak.com
Field 4 Chart
Major
Field Label 4
Date
ThingSpeak.com

CONCLUSION

In conclusion, the Internet of Things is closer to being implemented than the average person would think. Most of the necessary technological advances needed for it have already been made, and some manufacturers and agencies have already begun implementing a small-scale version of it. The main reasons why it has not truly been implemented is the impact it will have on the legal, ethical, security and social fields. Workers could potentially abuse it, hackers could potentially access it, corporations may not want to share their data, and individual people may not like the complete absence of privacy. For these reasons, the Internet of Things may very well be pushed back longer than it truly needs to be.

The future of IoT is virtually unlimited due to advances in technology and consumers' desire to integrate devices such as smart phones with household machines. Wi-Fi has made it possible to connect people and machines on land, in the air and at sea. It is critical that both companies and governments keep in ethics in mind as we approach the fourth Industrial Revolution (Pye, 2014). With so much data traveling from device to device, security in technology will be required to grow just as fast as connectivity in order to keep up with demands.Governments will undoubtable face tough decisions as to how far the private the sector is allowed to go in terms of robotics and information sharing. The possibilities are exciting, productivity will increase and amazing things will come by connecting the world.

FUTURE SCOPE

With the exponential growth in internet usage, the next big thing in the queue of technological advancements is ‘Internet of Things’. The entire world is migrating towards IOT which will have a huge impact on our lives in the coming five years. Leave about connecting computers, laptops and smartphones with the internet, now there will be a multitude of smart devices connected to the internet and each other. Starting from home appliances to big industrial machinery, everything will become smart. Some technology experts call it as the ‘Next Digital Revolution’ while others proclaim it as the ‘Next Generation of Internet’.

IOT is believed to change the entire way people communicate, work and live. Now there will be connectivity for everyone, everything and everywhere. It is going to have an influential impact on how the businesses and government interact with the world.

If we talk about IOT applications in an organized manner, these are broadly divided into Industrial and Consumer segments. The industrial segment covers industrial and retail automation which largely contributes to the development of smart cities. On the other hand, the consumer segment is mainly driven by personal interest and covers smart lifestyle, home, health & fitness automation. Likewise, enterprises and consumers using IOT solutions will be complementing the IOT growth in our country.

REFERENCE

[1] https://www.pantechsolutions.net/iot-projects/iot-based-humidity-and-temperature-monitoring-using-arduino-uno

[2] https://www.engineersgarage.com/contribution/iot-based-humidity-and-temperature-monitoring-rover

[3] https://electronicsforu.com/electronics-projects/humidity-temperature-monitoring-using-arduino-esp8266

[4] https://en.wikipedia.org/wiki/Internet_of_things

APPENDIX

For ARDUINO

```
#include <SoftwareSerial.h>
SoftwareSerial ESP1( 3, 2 );
const int relay1 = 4;
const int relay3 = 5;
int sensor_pin = A1;
const int ldrPin = A0;
int output_value ;

void setup()
{
  Serial.begin(9600);
  ESP1.begin(9600);
  pinMode(ldrPin, INPUT);
  pinMode(relay1,OUTPUT);
  pinMode(relay3,OUTPUT);
  pinMode(LED_BUILTIN, OUTPUT);
  digitalWrite(relay1,HIGH);
  digitalWrite(relay3,HIGH);
  delay(1000);
}

void loop()
```

```
{
  int ldrStatus = analogRead(ldrPin);
  output_value= analogRead(sensor_pin);
  output_value = map(output_value,550,0,0,100);
  ESP1.print("Moisture : ");
  ESP1.print(output_value);
  ESP1.println("%");
  Serial.print("Moisture : ");
  Serial.print(output_value);
  Serial.println("%");
  delay(1000);
  if (ldrStatus <= 200)
  {
    Serial.print("Its DARK, Turn on the LED : ");
    Serial.println(ldrStatus);
    digitalWrite(LED_BUILTIN, HIGH);
    digitalWrite(relay1,HIGH);
    digitalWrite(relay3,HIGH);
    delay(1000);

} else {

  Serial.print("Its BRIGHT, Turn off the LED : ");
  Serial.println(ldrStatus);
  digitalWrite(LED_BUILTIN, LOW);
```

```
  digitalWrite(relay1,LOW);
  digitalWrite(relay3,LOW);
  delay(1000);
}
}
```

For NODE MCU

```
#include <SoftwareSerial.h>
#include <DHT.h>
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ThingSpeak.h>

#define DHTPIN D0
#define DHTTYPE DHT11
#define Relay2 D1

SoftwareSerial UNO (D2, D3);

DHT dht(DHTPIN, DHTTYPE);

const char* ssid = "Bonjour";
const char* password = "MARVEL10";
```

```
WiFiClient client;


unsigned long myChannelNumber = 750666;
const char * myWriteAPIKey = "1UW6Z0JXPOFXQKXK";
uint8_t temperature, humidity;


void setup()
{
  Serial.begin(115200);
  UNO.begin(9600);
  dht.begin();
  delay(10);
  pinMode(Relay2, OUTPUT);
  pinMode(D2, INPUT);
  pinMode(D3, OUTPUT);
  digitalWrite(Relay2, HIGH);
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED)
  {
   delay(500);
   Serial.print(".");
```

```
  }
  Serial.println("");
  Serial.println("WiFi connected");
  // Print the IP address
  Serial.println(WiFi.localIP());
  ThingSpeak.begin(client);
}
void loop()
{
  static boolean data_state = false;
  temperature = dht.readTemperature();
  humidity = dht.readHumidity();
  float gas = analogRead(A0);
  float val = UNO.read();
  Serial.print("Temperature Value is :");
  Serial.print(temperature);
  Serial.println("C");
  Serial.print("Humidity Value is :");
  Serial.print(humidity);
  Serial.println("%");
  Serial.print("Gas Value is :");
  Serial.print(gas);
  digitalWrite(Relay2, LOW);
  if (isnan(gas))
  {
```

```
    Serial.println("Failed to read from MQ-5 sensor!");
    return;
}
   while(UNO.available())
   {
    float val = UNO.read();
    Serial.print(val);
   }
   ThingSpeak.writeField(myChannelNumber, 1, temperature, myWriteAPIKey);
   ThingSpeak.writeField(myChannelNumber, 2, humidity, myWriteAPIKey);
   ThingSpeak.writeField(myChannelNumber, 3, gas, myWriteAPIKey);
   ThingSpeak.writeField(myChannelNumber, 4, val, myWriteAPIKey);
   delay(3000); // ThingSpeak will only accept updates every 15 seconds.
}
```