REMOTE MONITORING OF FOOD SPOILAGE USING SMART TECHNOLOGY

A Project report submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

B.Sai Akhilesh(317126512123) **G.Manoj(317126512134)**

P.Uma Prakah(317126512163) **Y.Arjuna Rao(317126512179)**

Under the guidance of

Mr.S.SrinivasM.Tech,Ph.D(IIT Hyd)

Associate Professor

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCE

(Permanently Affiliated to AU, Approved by AICTE and Accredited by NBA & NAAC with 'A' Grade)

Sangivalasa, Bheemili mandal, Visakhapatnam dist.(A.P)

2020-2021

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES
(PermanentlyAf iliatedtoA U,Approved by AICTEandAccreditedbyNBA& NAACwith 'A' Grade)
Sangivalasa, Bheemili Mandal, Visakhapatnam dist.(A.P)

PRAGNANAM BRAHMA
ANITS

CERTIFICATE

This is to certify that the project report entitled **"Remote monitoring of food spoilage using smart technology"** submitted by **B.Sai Akhilesh(317126512123), G.Manoj (316126512134), P.Uma Prakash (317126512163), Y.Arjuna Rao (317126512179)** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Electronics and Communication Engineering** of **Andhra University**, Visakhapatnam is a record of bonafide work carried out under my guidance and supervision.

[illegible]
Project Guide

Dr.S.Srinivas
M.Tech,Ph.D(IIT Hyd)
Associate Professor
Department of E.C.E
ANITS

Associate Professor
Department of E.C.E.
Anil Neerukonda
Institute of Technology & Sciences
Sangivalasa, Visakhapatnam-531 162

[illegible]
Head of the Department

Dr. V. Rajyalakshmi
Professor and HOD
Department of E.C.E
ANITS

Head of the Department
Department of E C E
Anil Neerukonda Institute of Technology & Sciences
Sangivalasa - 531 162

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our project guide **Mrs.S.Srinivas,** Associate Professor, Department of Electronics and Communication Engineering, ANITS, for her guidance with unsurpassed knowledge and immense encouragement. We are grateful to **Dr. V. Rajyalakshmi**, Head of the Department, Electronics and Communication Engineering, for providing us with the required facilities for the completion of the project work.

We are very much thankful to the **Principal and Management, ANITS, Sangivalasa,** for their encouragement and cooperation to carry out this work.

We express our thanks to all **teaching faculty** of Department of ECE, whose suggestions during reviews helped us in accomplishment of our project. We would like to thank **all non-teaching staff** of the Department of ECE, ANITS for providing great assistance in accomplishment of our project.

We would like to thank our parents, friends, and classmates for their encouragement throughout our project period. At last but not the least, we thank everyone for supporting us directly or indirectly in completing this project successfully.

PROJECT STUDENTS

B.Sai Akhilesh(317126512123),
G.Manoj(317126512134),
P.Uma Prakash (317126512163),
Y.Arjuna Rao(317126512179)

CONTENTS

ABSTRACT

- Food safety plays important role in countries economy and human health. The integrated IoT-based online monitoring approach using smart logistics can address the critical needs of reducing food waste, increasing transportation efficiency, and tracking food contamination. Majority of consumers only pay attention to the ingredients used and their nutritional value.
- The parameters like humidity, bacteria, and temperature are major factors on which the rate of decomposition of food depends on. If the temperature of the storage is between 40F to 140F, it is a danger zone because, bacteria grow rapidly, doubling its number in 20 min. Similarly, the humidity in the food storage room should be around 50-55%.
- So in this IoT project, a Food Monitoring device is implemented to monitor the real-time values of the temperature, humidity, and methane gas, which are prime measures in food quality, will be measured and displayed. If the temperature is at the critical value, user will receive a notification through an app.

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

CHAPTER 1

INTORDUCTION

1.1 Introduction:

Food safety and hygiene is a major concern in order to prevent the food wastage. The Quality of the food needs to be monitored and it must be prevented from rotting and decaying by the atmospheric factors like temperature, humidity and dark. Therefore, it is useful to deploy quality monitoring devices at food stores. These quality monitoring devices keep a watch on the environmental factor that cause or pace up decay of the food. Later, the environmental factors can be controlled like by refrigeration, vacuum storage etc.

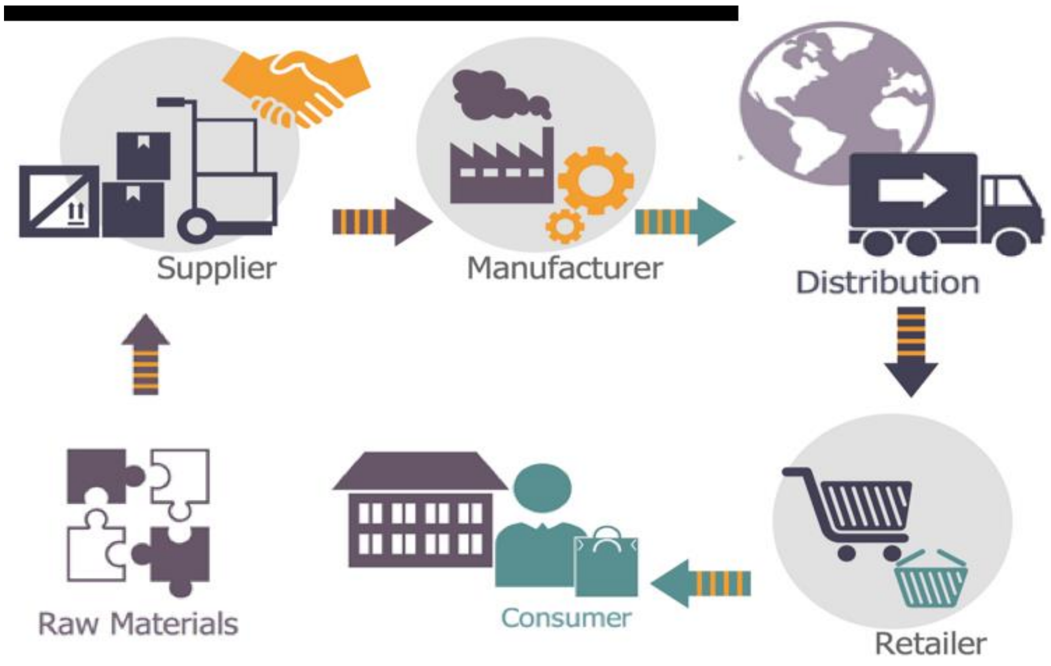


Fig 1.1

In this project, a similar food quality monitoring device will be designed that will keep watch of environmental factors like temperature, humidity, gas content and exposure to light The device is built on Arduino UNO which is a popular prototyping board. The Arduino board is interfaced with various sensors like DHT-22 to monitor temperature and humidity, MQ3 to detect gas content and LDR to measure exposure to light. This is an IoT device and sends the measured sensor data to an IoT platform. The ESP8266 Wi-Fi Modem is interfaced with the Arduino to connect it to the internet via Wi-Fi router. The sensor data is also displayed on a character LCD interfaced with the Arduino UNO. The IoT platform used for logging and monitoring of sensor data is Freeboard.io. With the power of Internet of Things, the environmental factors affecting the food storage can be monitored from anywhere, anytime and from any device.

Many such devices can be installed at a location for better monitoring and quality control. The Arduino Sketch running over the device implements the various functionalities of the project like reading sensor data, converting them into strings, displaying them on character LCD and passing them to the IoT platform. The Sketch is written, compiled and loaded using the Arduino IDE.

The 5 stages of the food supply chain include-

- **Farm** - This is where the ingredients, meat, fruits vegetables, food, beverages originate and are purchased from (either animals or plants).
- **Processing** - At this stage, plants, and animals are converted into edible form.
- **Distributing** - Once the food is edible, it is transported and distributed to the necessary retailer/supplier. Distributors sell items, manage inventories, reduce costs, and do other actions to add value to the food item.
- **Retailer** - This is the process used to deliver the products to the consumers. It encompasses everything from obtaining the distributed items to selling them.
- **Consumer** - The consumer purchases the food item from the retailer.

One-third of the raw food produced over the world i.e.,1.3 billion tonnes is being wasted every year, according to resource Organization of the United Nations. In Singapore, the National Environment Agency (NEA) estimated that 790,000 tonnes of food was thrown in 2014.
Such amount of wastage of resources ridicules the mankind. These wastage may be due to various reasons such as environmental conditions or buying excessive food, all these ambiguities can be reduced if the raw food materials stored will be properly monitored (to keep an eye on...).
Parameters like temperature and humidity play a major role in maintaining the quality of the raw food material. The monitoring unit present in the system is done using various sensors to monitor relative humidity and temperature of the storage area where the raw food is stored, sensors continuously records the sensed values. Now the controlling unit makes sure that the parameters lie in the desired range of that particular raw food material, this is done with actuators, both actuators and sensors should go hand in hand i.e., based on the recorded values by the sensors the actuators must take an action to maintain the temperature and humidity in the desired range if it goes beyond or below the threshold range, all these help in maintain the quality of the raw food.
Quantity is also an important factor to avoid wastage which is done by a sensor for continuously monitoring the weight of the raw food stored. The quantity of the food material is monitored continuously and a certain minimum threshold is set and if the quantity falls below the threshold minimum value set then the user is notified of the shortage of food. Since the presence of food is monitored there will be no excess food bought by the user and this avoids food wastage.
Thus, to keep the user updated about these factors some or the other wireless communication is necessary. There are many technologies nowadays and each differ from other based on range, speed and reliable transmission. Therefore an communication system which is reliable, fast and by which user can have knowledge of data from anywhere would be suitable for this system. Here we use the Internet Of Things technology which keeps the user notified at all intervals of time. The various features of our system: Wood as casing material as it provides proper insulation.
Automated controlling mechanism storage container can have multiple raw food materials but only one material at a time not mixture of raw food. Usage of two peltier modules for temperature as well as humidity control to save power Weekly analysis reports as well as real time notifications through web application .

It is important to maintain the safety and hygiene of the food to keep it fresh and edible which helps in **decreasing the food wastage**. One solution for this is to maintain suitable environmental conditions for the stored food to control the rate of decomposition.
There are different parameters on which food decomposition depends, the parameters like humidity, bacteria, and temperature are major factors on which the rate of decomposition of food depends on. If the temperature of the

storage is between 40F to 140F, it is a danger zone because during that temperature bacteria grow rapidly, doubling its number in 20 min. Similarly, the humidity in the food storage room should be around 50-55% to keep the quality of the food at high, as long as possible.

So in this IoT project, we will build a **Food Monitoring device using NodeMCU and Arduino IDE**, to monitor the temperature and humidity of the stored environment and control it. To control the temperature, we are going to use a DC motor as a cooling mechanism. To find the temperature, and humidity the **DHT22 sensor** module is used, and to determine the status of the food, the **MQ3 gas sensor module** is used. In the future, if needed we can also use an IoT based Weight sensor to also monitor the food quantity in the storage area.

1.2 Internet of Things (IOT)

The term IoT is mainly used for devices that wouldn't usually be generally expected to have an internet connection, and that can communicate with the network independently of human action. For this reason, a PC isn't generally considered an IoT device and neither is a smartphone -- even though the latter is crammed with sensors.
A smartwatch or a fitness band or other wearable device might be counted as an IoT device

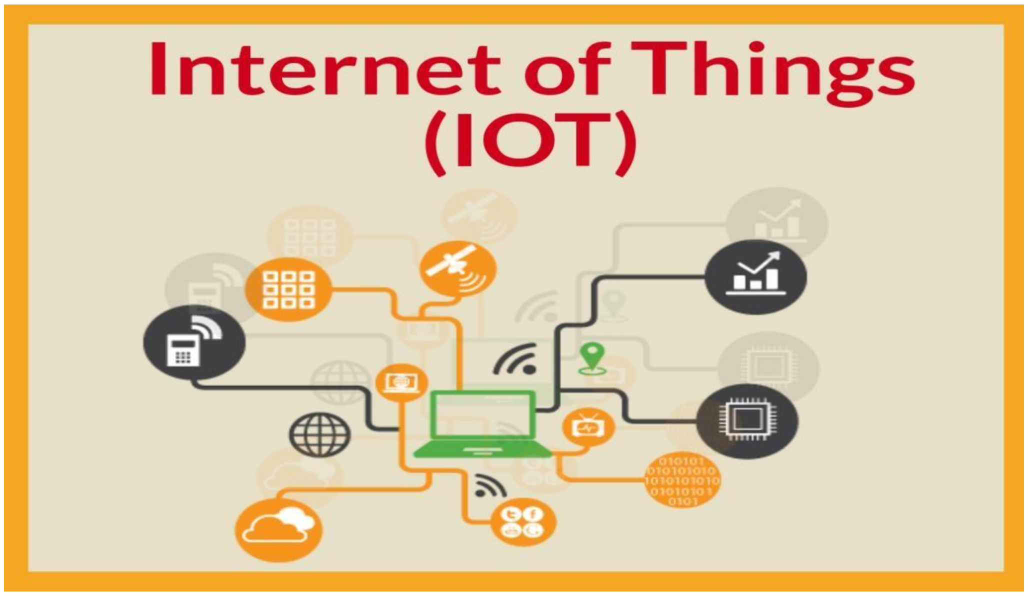


Fig 1.2

Efficient resource utilization: If we know the functionality and the way that how each device work we definitely increase the efficient resource utilization as well as monitor natural resources.
Minimize human effort: As the devices of IoT interact and communicate with each other and do lot of task for us, then they minimize the human effort.
Save time: As it reduces the human effort then it definitely saves out time. Time is the primary factor which can save through IoT platform.
Enhance Data Collection:
Improve security: Now, if we have a system that all these things are interconnected then we can make the system more secure and efficient.

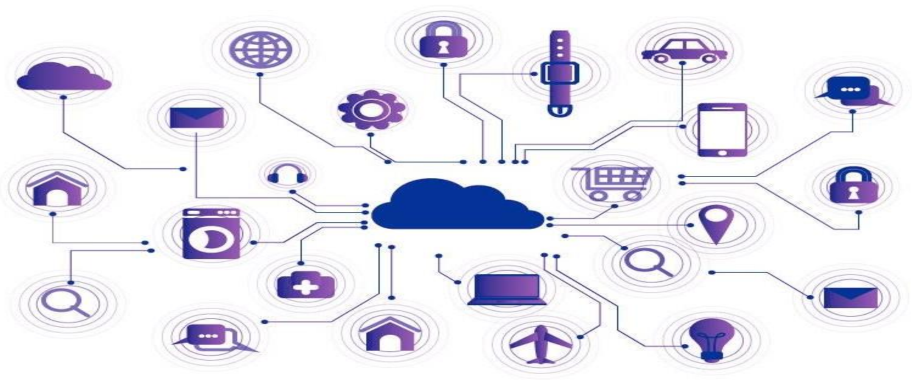


Fig 1.3

IoT devices produce many types of information, including telemetry, metadata, state, and commands and responses. Telemetry data from devices can be used in short operational timeframes or for longer-term analytics and model building. (For more on this diversity, read the overview of Internet of Things.)

Many devices support local monitoring in the form of a buzzer or an alarm panel on-premises. This type of monitoring is valuable, but has limited scope for in-depth or long-term analysis. This article instead discusses remote monitoring, which involves gathering and analyzing monitoring information from a remote location using cloud resources.

Operational and device performance data is often in the form of a time series, where each piece of information includes a time stamp. This data can be further enriched with dimensional labels (sometimes referred to as tags), such as labels that identify hardware revision, operating time zone, installation location, firmware version, and so on.

Time-series telemetry can be collected and used for monitoring. *Monitoring* in this context refers to using a suite of tools and processes that help detect, debug, and resolve problems that occur in systems while those systems are operating. Monitoring can also give you insight into the systems and help improve them.

The state of monitoring IT systems, including servers and services, has continuously improved. Monitoring tools and practices in the cloud-native world of microservices and Kubernetes are excellent at monitoring based on time-series metric data. These tools aren't designed specifically for monitoring IoT devices or physical processes, but the constituent parts—labeled series of metrics, visualization, and alerts—all can apply to IoT monitoring.

1.3 Symptoms of consuming spoiled food:-

- Upset stomach
- Nausea
- Vomitings
- Diarrhea
- Fever
- Dehydration
- Kidney Failure
- Brain
- Nerve damage

1.4 Production of Banana

India is the top country by bananas production in the world. As of 2019, bananas production in India was 30.5 million tonnes that accounts for 26.02% of the world's bananas production. The top 5 countries (others are China, Indonesia, Brazil, and Ecuador) account for 53.94% of it. The world's total bananas production was estimated at 117 million tonnes in 2019.
A banana is an edible fruit botanically a berry produced by several kinds of large herbaceous flowering plants in the genus Musa. The fruit is variable in size, colour, and firmness, but is usually elongated and curved, with soft flesh rich in starch covered with a rind, which may be green, yellow, red, purple, or brown when ripe**.**

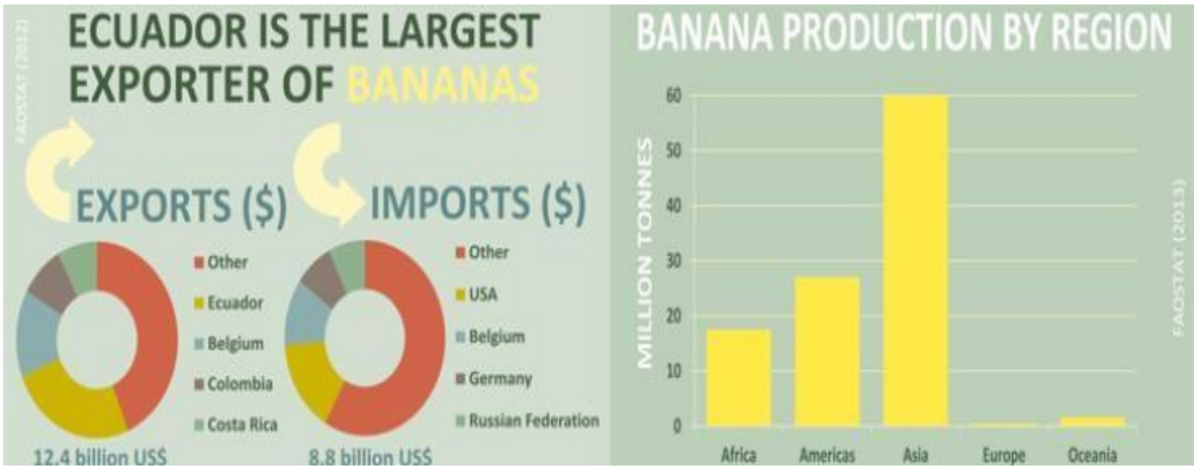


Fig 1.4

Bananas are one of the most popular fruits in the world. They are the fourth most important food crop after wheat, rice, and maize in terms of production and are the world's favourite fruit in terms of consumption quantity. They are not only consumed raw, but are also used in making juices, sauces, baked goods and various dishes .The global consumption for bananas is estimated to grow at a CAGR of 1.21% for the forecast period of 2019-2024. In the medium term, the global banana market is set to continue moderate growth. The expected population growth will drive the consumption of bananas in the world.

Bananas are one of the most popular fruits in the world. In Japan, we can find them at supermarkets, convenience stores and groceries. It is said that 9 to 15 bananas are consumed by one household every month in Japan. However, the production of bananas in Japan is quite low, and we import bananas from overseas to make up for the gap in domestic production, which is far below domestic consumption. This article introduces which country produces bananas and what kind of supply chain process bananas have to go through before coming to your table. This article also compares the difference in banana supply chain management among the world's three largest consumption areas, namely North America, Europe, and East Asia.

A wide variety of banana brands are placed on the banana shelves in supermarkets in Japan, but about 80% of them are imported from the Philippines. By the time bananas reach consumers, they go through the hands of producer, exporter, importer, ripening / distributor, and retailer. The long journey of the banana supply chain begins with the cultivation and planting of bananas in the exporting country.

About 70 to 100 days later, the flowers of the banana trees bloom, the square fruits become rounded, and when the color of the fruits turns light green, it is time to harvest. Unlike other fruits, bananas can be harvested all year round, because banana production areas are distributed in tropical and subtropical areas. After the bananas are harvested, they are shipped to a local packing plant, washed thoroughly with water, and then sorted, weighed, and packed. When bananas are packed in a cardboard box, a number is stamped on the outside of the box. This number is the identity code, which identifies the farm where the banana was harvested, the packing facility, and the date and time of packing, that ensures the traceability of the bananas. Then, after packing is completed, after undergoing export inspection such as phytosanitary inspection, the bananas finally will be on board the vessel and cross over to the other side of the world.
It probably takes weeks for bananas to cross the ocean, but how are they kept fresh? They are transported using temperature-controlled reefer containers. Recently, some shipping companies have also equipped reefer containers with Remote Container Management (RCM) technology. This makes it possible to remotely track data such as power supply, temperature, humidity, ventilation settings, etc. in the reefer containers even while the bananas are

travelling at sea, and safely transport the bananas to the importing countries.
Bananas are not directly distributed to the market after arriving at the importing country via sea transportation and undergoing phytosanitary and import customs clearance on the importing country. Instead, the packed banana pallets are first ripened in the ripening room. This is because yellow-colored bananas are generally banned from import by pest control quarantine rules, so when they arrive in the importing country, the bananas are still green and hard. Green hard bananas are brought into this ripening room, which is divided according to the shipping date, and the green bananas are ripened for about 4 to 7 days in this warm room where the temperature, ethylene gas and hydrocarbons are adjusted. The temperature must be kept constant, and temperatures below 13°C or above 18°C can spoil the appearance of bananas. Bananas that have reached the proper ripened state are then moved to the cooling room. Now the bananas are ready to be edible, the bananas are shipped to wholesalers and retailers, and finally delivered to consumers.

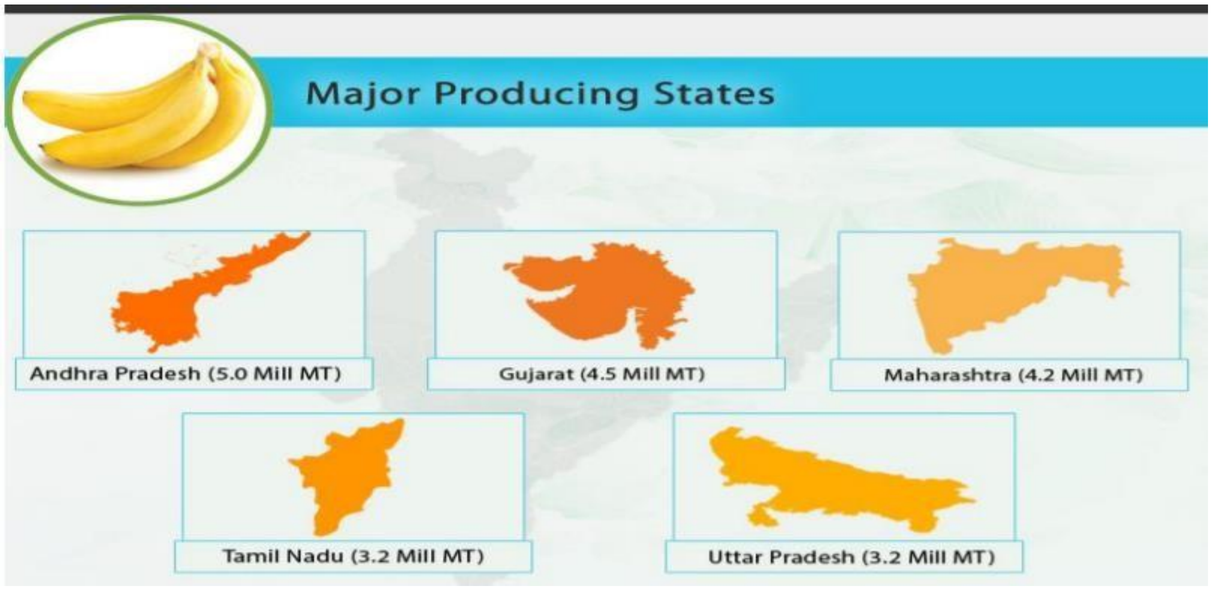

Major Producing States
Andhra Pradesh (5.0 Mill MT)
Gujarat (4.5 Mill MT)
Maharashtra (4.2 Mill MT)
Tamil Nadu (3.2 Mill MT)
Uttar Pradesh (3.2 Mill MT)

Fig 1.5

1.5 Advantages of Bananas

- Bananas Contain Many Important Nutrients.
- Bananas Contain Nutrients That Moderate Blood Sugar Levels.
- Bananas May Improve Digestive Health.
- Bananas May Aid Weight Loss.
- Bananas May Support Heart Health.
- Bananas Contain Powerful Antioxidants.
- Unripe Bananas May Improve Insulin Sensitivity.

1.6 Storage of Bananas

Banana fans like to stock up with the yellow fruit as this vitamin-filled all-rounder makes an ideal snack for in between meals and can be enjoyed in countless different dishes. But how should bananas be stored so that they stay fresh for as long as possible
Keep them cool and protected from the light

Pop them into the fridge

1.7 LITERATURE SURVEY

Food is being wasted everywhere. Majority of consumers only pay attention to the information provided on the packaging, i.e., the amount of ingredients used and their nutritional value, but they forget that they are blindly risking their health by ignoring the environmental conditions to which these packets are subjected. The major concern is that people are consuming the food even after the food is spoiled.
Every year in our country half of the food wastage is because of decomposing and spoilage of food. In this generation people are busy with their works and are not bothering of the kind of food that they are taking and are keeping their life at risk.
As people are not aware that when food will be spoiled.
Food should be stored at correct temperature, humidity and the ethylene gas content should be low, and many people fail at that, resulting the spoilage of food.
One of the world’s most popular fruit having a massive production in Asia itself the “**BANANA**” have a lot of nutrition facts in it which is consumed more than 100 billion annually.Symptoms of eating spoiled Bananas:- Bloating, Gastric, Nausea ,Vomitings etc.,

So in this project, IoT is used to monitor the parameters like temperature, humidity and the ethylene gas content present in the food in order to notify the user before the food gets spoiled. The network of things or objects are embedded with the sensors, software and other technologies for the purpose of connecting and exchanging data(if the food is spoiled or not) with other devices and systems over the internet.

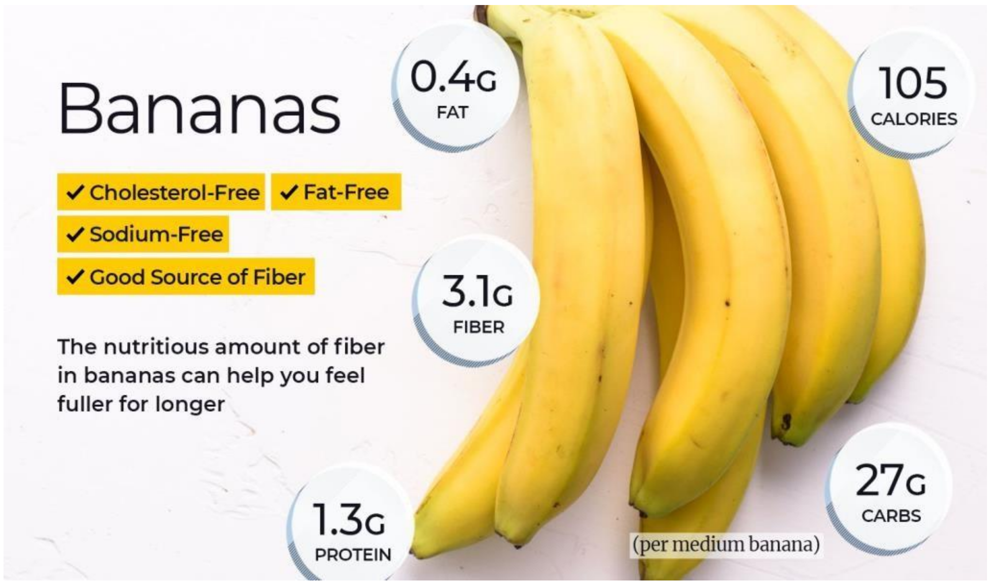


Fig 1.6

Figure 1 Banana's supply chain

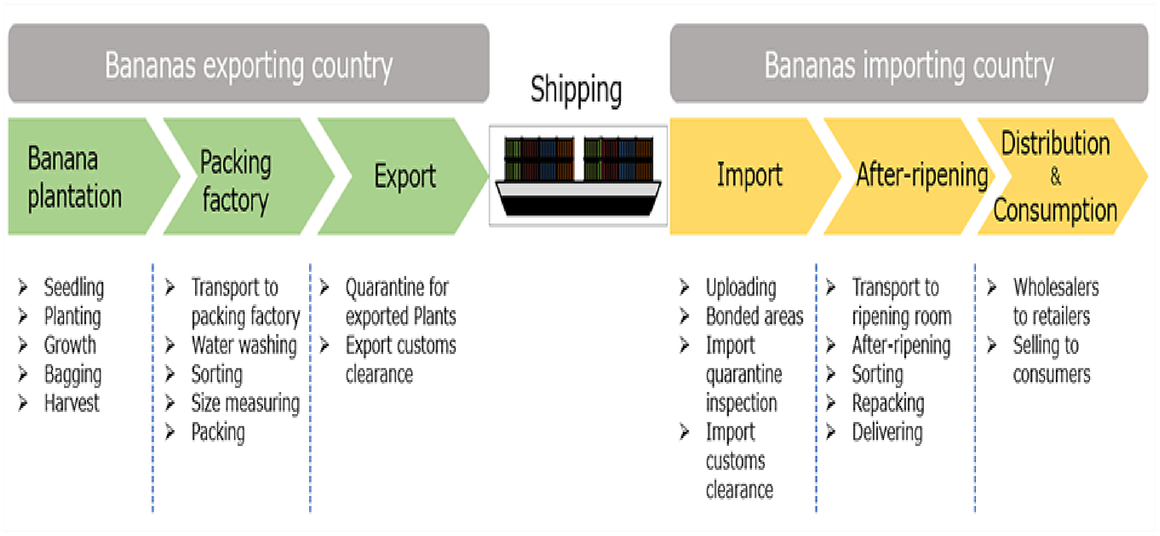


Fig 1.7

Making an international comparison of banana imports and exports, there are the following six key findings.

Point 1. Banana productive capacity ≠ export capacity

Figures 2 and 3 show banana productivity and imports by country, respectively. Comparing these, countries that produce a lot of bananas do not necessarily export a lot of bananas. In particular, although India is ranked first and China is ranked second in banana production, their banana exports are not very high. China ranks 9th in export volume, which is low for a large banana producing country.
Why is the export volume so small even though they produce large amount? Since these two countries have very large populations, India with about 1.26 billion and China about 1.37 billion, the domestic consumption is also large; therefore the export volume is very small. Besides India and China, the same reasoning can be applied for other populous countries, like Brazil and Indonesia.

Point 2. "Transit point" the Netherlands and Belgium

Figure 3 shows the interesting banana exporters, that is the Netherlands and Belgium. Why do the Netherlands and Belgium rank high as banana exporters when they both are not located in tropical and subtropical areas? This is because they would be functioning as transit points for bananas in Europe due to their international ports.

Figure 2 Banana production by country (2018), Unit: tons

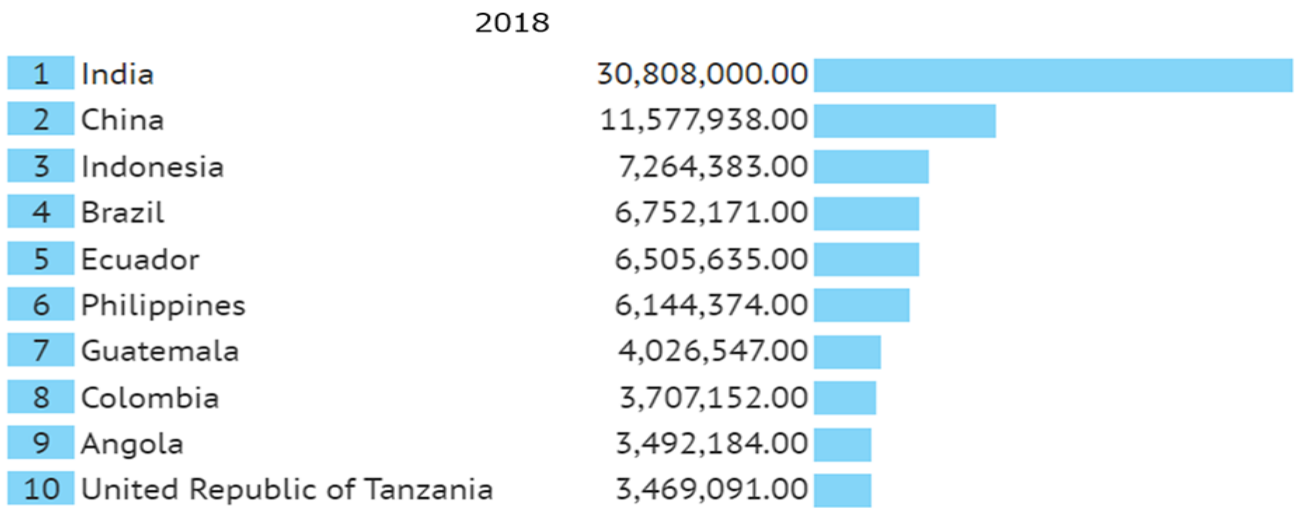


Fig 1.8

CHAPTER-2

HARDWARE REQUIRED

CHAPTER-2

HARDWARE REQUIRED

2.1 Components Required:

- Arduino uno
- NodeMCU ESP8266
- MQ3 Sensor Module
- LDR Sensor
- DHT22 Sensor module
- 16X2 LCD Screen
- Connecting jumper wires

2.2 ARDUINO UNO:

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

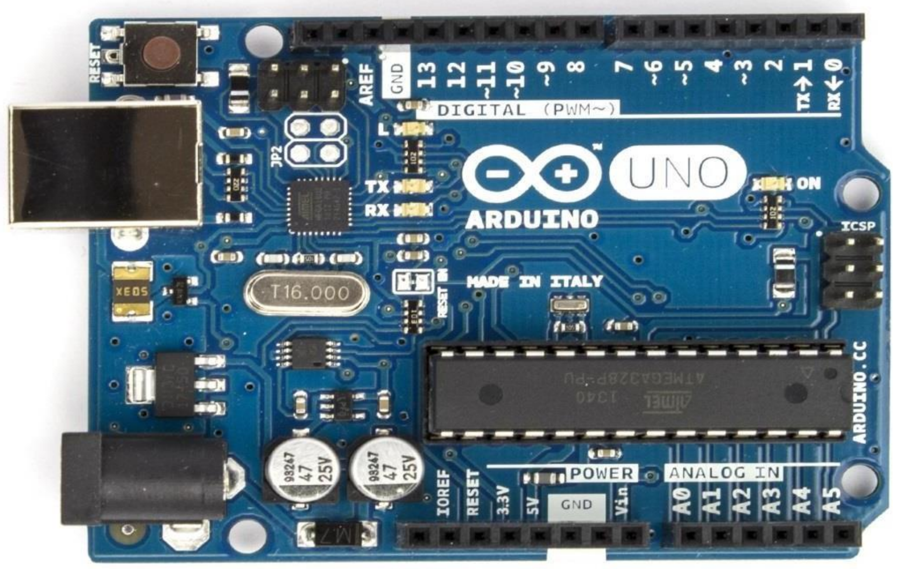


Fig 2.1

2.2.1 POWER

Power The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector. The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts. The power pins are as follows:

- **VIN**:- The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V**:-This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.
- **3V3**:- A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND**:-Ground pins.

2.2.2 MEMORY

The ATmega328 has 32 KB (with 0.5 KB used for the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

2.2.3 INPUT OUTPUT

Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX):-** Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- **External Interrupts: 2 and 3**:- These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 3, 5, 6, 9, 10, and 11**.:- Provide 8-bit PWM output with the analogWrite() function.
- **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication using the SPI library.
- **LED: 13**:- There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. Additionally, some pins have specialized functionality:

- **TWI: A4 or SDA pin and A5 or SCL pin**:-Support TWI communication using the Wire library. There are a couple of other pins on the board:
- **AREF**:- Reference voltage for the analog inputs, Used with analogReference().
- **Reset:-**Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

2.2.4 COMMUNICATION

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1). A SoftwareSerial library allows for serial communication on any of the Uno's digital pins. The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

2.2.5 PROGRAMMING

The Arduino Uno can be programmed with the Arduino software (download). Select "Arduino Uno from the Tools > Board menu (according to the microcontroller on your board). For details, see the reference and tutorials. The ATmega328 on the Arduino Uno comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files). You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details. The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available . The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by:

- On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2.
- On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.

2.2.6 Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload. This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following halfsecond or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data. The Uno contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

2.2.7 Physical Characteristics

The maximum length and width of the Uno PCB are 2.7 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Four screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

2.3 NODE MCU

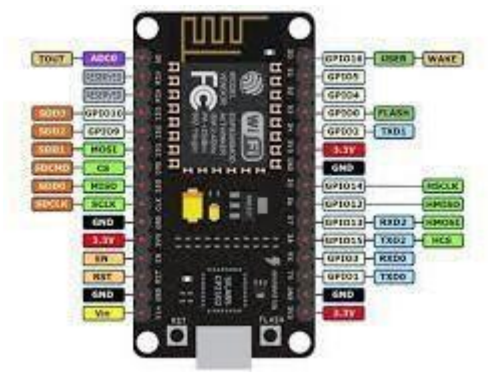


Fig 2.2

The ESP8266 is the name of a micro controller designed by Espressif Systems. The ESP8266 itself is a self-contained WiFi networking solution offering as a bridge from existing micro controller to WiFi and is also capable of running self-contained applications. This module comes with a built in USB connector and a rich assortment of pin-outs. With a micro USB cable, you can connect NodeMCU devkit to your laptop and flash it without any trouble, just like Arduino. It is also immediately breadboard friendly.

2.3.1 Specification

- Voltage:3.3V**.**
- Wi-Fi Direct (P2P), soft-AP.
- Current consumption: 10uA~170mA**.**
- Flash memory attachable: 16MB max (512K normal).
- Integrated TCP/IP protocol stack.
- Processor: Tensilica L106 32-bit.
- Processor speed: 80~160MHz**.**
- RAM: 32K + 80K**.**
- GPIOs: 17 (multiplexed with other functions).
- Analog to Digital: 1 input with 1024 step resolution.
- +19.5dBm output power in 802.11b mode
- 802.11 support: b/g/n**.**
- Maximum concurrent TCP connections: 5.

2.3.2 Features

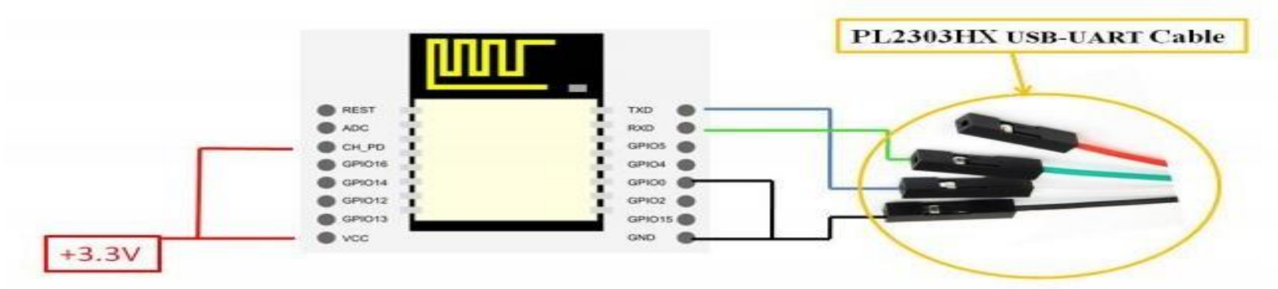
- Open-source, Interactive, Programmable, Low cost, Simple, Smart, WI-FI enabled
- Arduino-like hardware IO
- Lowest cost WI-FI

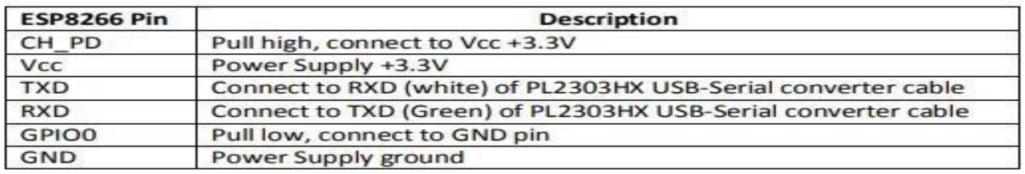
2.3.3 General Purpose Input/Output Interface (GPIO):-

ESP8266EX has 17 GPIO pins which can be assigned to various functions by programming the appropriate registers.

Each GPIO can be configured with internal pull-up or pull-down, or set to high impedance, and when configured as an input, the data are stored in software registers; the input can also be set to edge-trigger or level trigger CPU interrupts. In short, the IO pads are bidirectional, non-inverting and tristate, which includes input and output buffer with tristate control inputs.
These pins can be multiplexed with other functions such as I2C, I2S, UART, PWM, IR Remote Control, etc.

2.3.4 Wiring





ESP8266 Pin	Description
CH_PD	Pull high, connect to Vcc +3.3V
Vcc	Power Supply +3.3V
TXD	Connect to RXD (white) of PL2303HX USB-Serial converter cable
RXD	Connect to TXD (Green) of PL2303HX USB-Serial converter cable
GPIO0	Pull low, connect to GND pin
GND	Power Supply ground

Fig 2.3

2.3.5 Node MCU Connection with Arduino Uno

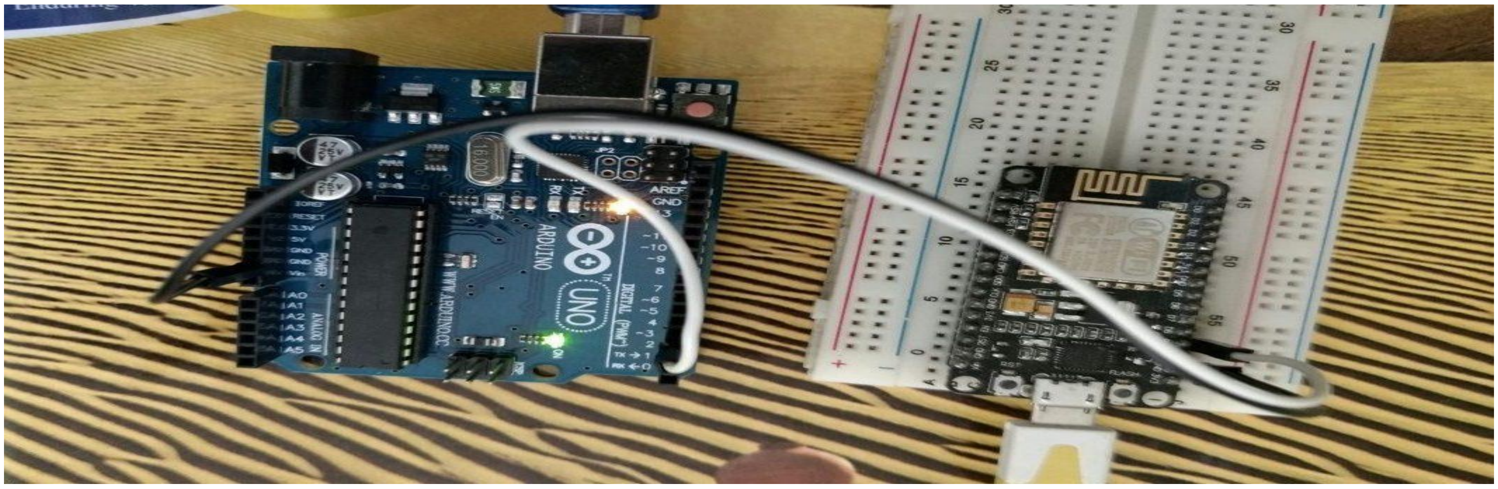


Fig 2.4

2.4 MQ3 GAS SENSOR

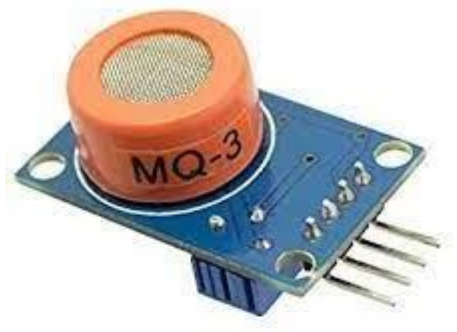


Fig 2.5

Sensitive material of MQ-3 gas sensor is SnO2, which with lower conductivity in clean air. When the target alcohol gas exist, the sensor's conductivity gets higher along with the gas concentration rising. Users can convert the change of conductivity to correspond output signal of gas concentration through a simple circuit.
MQ-3 gas sensor has high sensitivity to alcohol gas and can resistant to the interference of gasoline, smoke and vapour. It is with low cost and suitable for various applications of detecting alcohol at different concentration.

This alcohol gas sensor detects the concentration of alcohol gas in the air and ouputs its reading as an analog voltage. The concentration sensing range of 0.04 mg/L to 4 mg/L is suitable for breathalysers (the legal limit of breath alcohol concentration, or BrAC, in most US states is 0.08 grams per 210 liters, or 0.38 mg/L). The sensor can operate at temperatures from -10 to 50°C and consumes less than 150 mA at 5 V.

Gas Sensor**(**MQ3**)** module is useful for gas **leakage** detection **(in home and industry).** It is suitable for detecting Alcohol, Benzine, CH4, Hexane, LPG, CO. The sensitivity of the sensor can be adjusted by using the potentiometer.

2.4.1 Features

It has good sensitivity to alcohol in wide range, and has advantages such as long lifespan, low cost and simple drive circuit &etc.

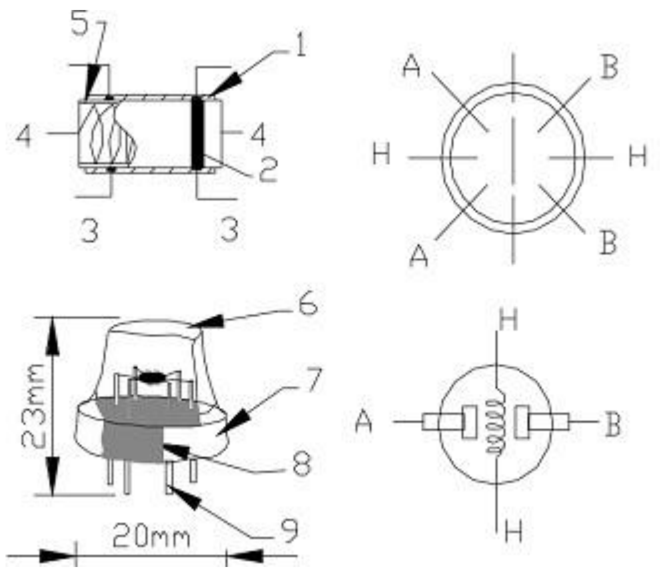
It is widely used in domestic alcohol gas alarm, industrial alcohol gas alarm and portable alcohol detector.

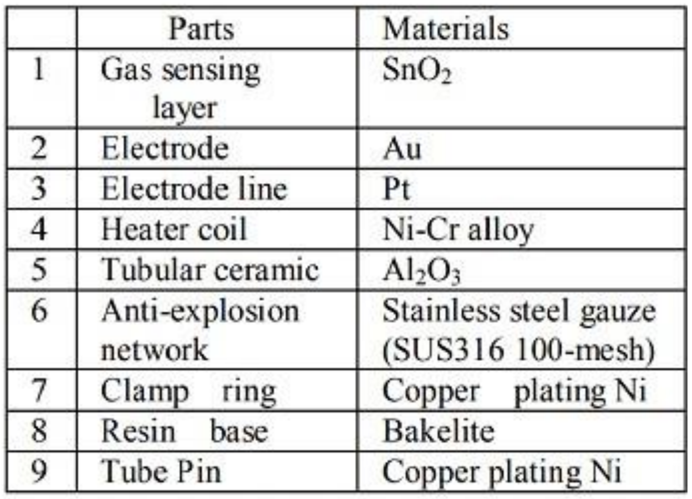
2.4.2 Specifications of MQ-3 Gas Sensor

- Power requirements: 5 VDC @ ~165 mA (heater on) / ~60 mA (heater off)
- Current Consumption: 150mA
- DO output: TTL digital 0 and 1 (0.1 and 5V)
- AO output: 0.1- 0.3 V (relative to pollution), the maximum concentration of a voltage of about 4V
- Detecting Concentration: 0.05-10mg/L Alcohol

- Interface: 1 TTL compatible input (HSW), 1 TTL compatible output (ALR)
- Heater consumption: less than 750mW
- Operating temperature: 14 to 122 °F (-10 to 50°C)

2.4.3 Structure of MQ3 GAS Sensor





	Parts	Materials
1	Gas sensing layer	SnO_2
2	Electrode	Au
3	Electrode line	Pt
4	Heater coil	Ni-Cr alloy
5	Tubular ceramic	Al_2O_3
6	Anti-explosion network	Stainless steel gauze (SUS316 100-mesh)
7	Clamp ring	Copper plating Ni
8	Resin base	Bakelite
9	Tube Pin	Copper plating Ni

Fig 2.6

2.4.4 Applications

- Gas level over-limit alarm
- Breathalyser
- Portable alcohol detector
- Stand-alone/background sensing device
- Environmental monitoring equipment

2.4.5 Mq3 Connection with Arduino Uno

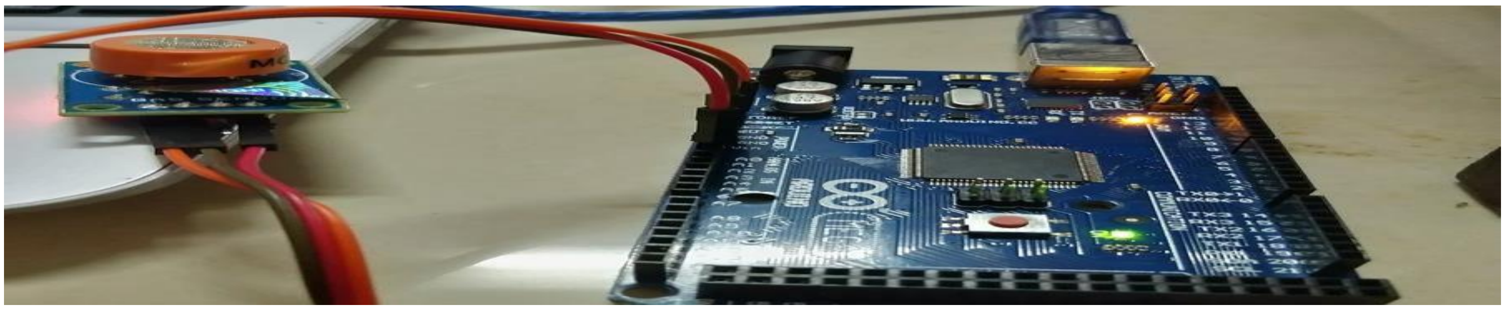


Fig 2.7

2.5 LDR SENSOR

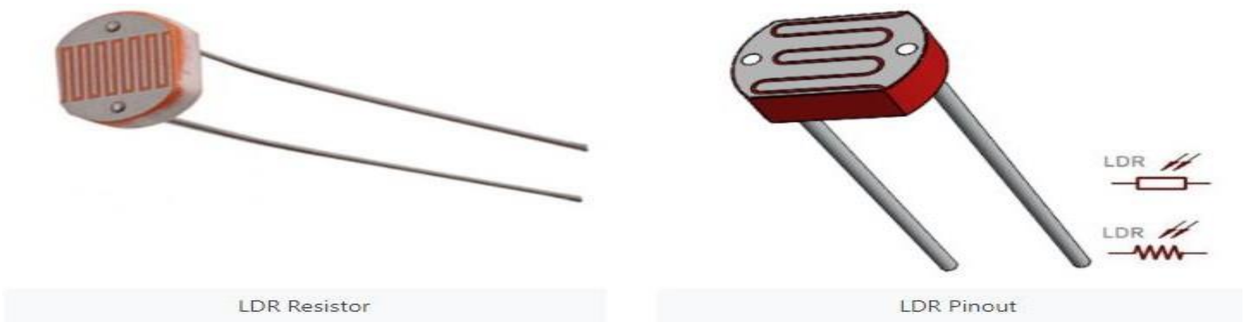


Fig 2.8

The **Light Dependent Resistor** (**LDR**) is just another special type of Resistor and hence has no polarity. Meaning they can be connected in any direction. They are breadboard friendly and can be easily used on a perf board also. The symbol for LDR is just as similar to Resistor but adds to inward arrows as shown above. The arrows indicate the light signals.

A Light Dependent Resistor (LDR) is also called a photoresistor or a cadmium sulfide (CdS) cell. It is also called a photoconductor. It is basically a photocell that works on the principle of photoconductivity. The passive component is basically a resistor whose resistance value decreases when the intensity of light decreases. This **optoelectronic device** is mostly used in light varying sensor circuit, and light and dark activated switching circuits. Some of its applications include camera light meters, street lights, clock radios, light beam alarms, reflective smoke alarms, and outdoor clocks.

2.5.1 Basic Structure

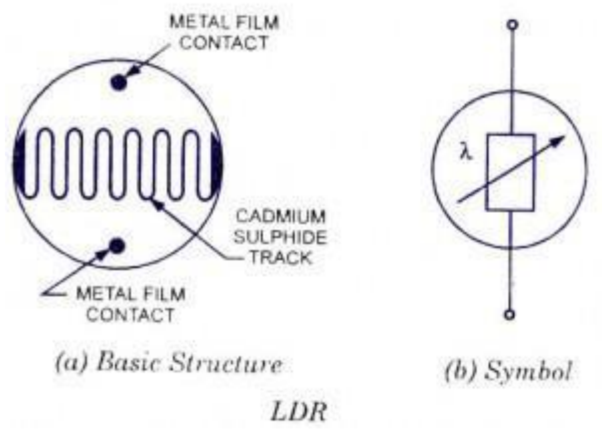


Fig 2.9

The snake like track shown below is the Cadmium Sulphide (CdS) film which also passes through the sides. On the top and bottom are metal films which are connected to the terminal leads. It is designed in such a way as to provide maximum possible contact area with the two metal films. The structure is housed in a clear plastic or resin case, to provide free access to external light. As explained above, the main component for the construction of LDR is cadmium sulphide (CdS), which is used as the photoconductor and contains no or very few electrons when not illuminated. In the absence of light it is designed to have a high resistance in the range of mega ohms. As soon as light falls on the sensor, the electrons are liberated and the conductivity of the material increases. When the light intensity exceeds a certain frequency, the photons absorbed by the semiconductor give band electrons the energy required to jump into the conduction band. This causes the free electrons or holes to conduct electricity and thus dropping the resistance dramatically (< 1 Kilo ohm).

2.5.2 Features

- Can be used to sense Light
- Easy to use on Breadboard or Perf Board
- Easy to use with Microcontrollers or even with normal Digital/Analog IC
- Small, cheap and easily available
- Available in PG5 ,PG5-MP, PG12, PG12-MP, PG20 and PG20-MP series

2.5.3 Applications

- Automatic Street Light
- Detect Day or Night
- Automatic Head Light Dimmer
- Position sensor
- Used along with LED as obstacle detector
- Automatic bedroom Lights
- Automatic Rear view mirror

2.5.4 LDR Connection with Arduino Uno

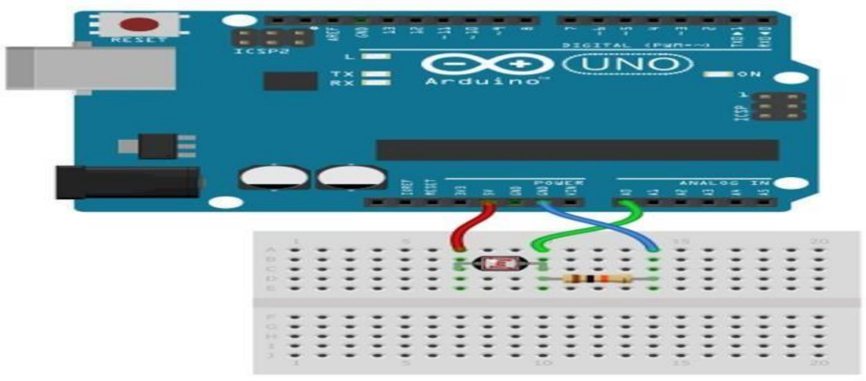


Fig 2.10

2.6 DHT22 SENSOR

DHT22 output calibrated digital signal. It utilizes exclusive digital-signal-collecting-technique and humidity sensing technology, assuring its reliability and stability.Its sensing elements is connected with 8-bit single-chip computer.

Every sensor of this model is temperature compensated and calibrated in accurate calibration chamber and the calibration-coefficient is saved in type of programme in OTP memory, when the sensor is detecting, it will cite coefficient from memory.

Small size & low consumption & long transmission distance(20m) enable DHT22 to be suited in all kinds of harsh application occasions.

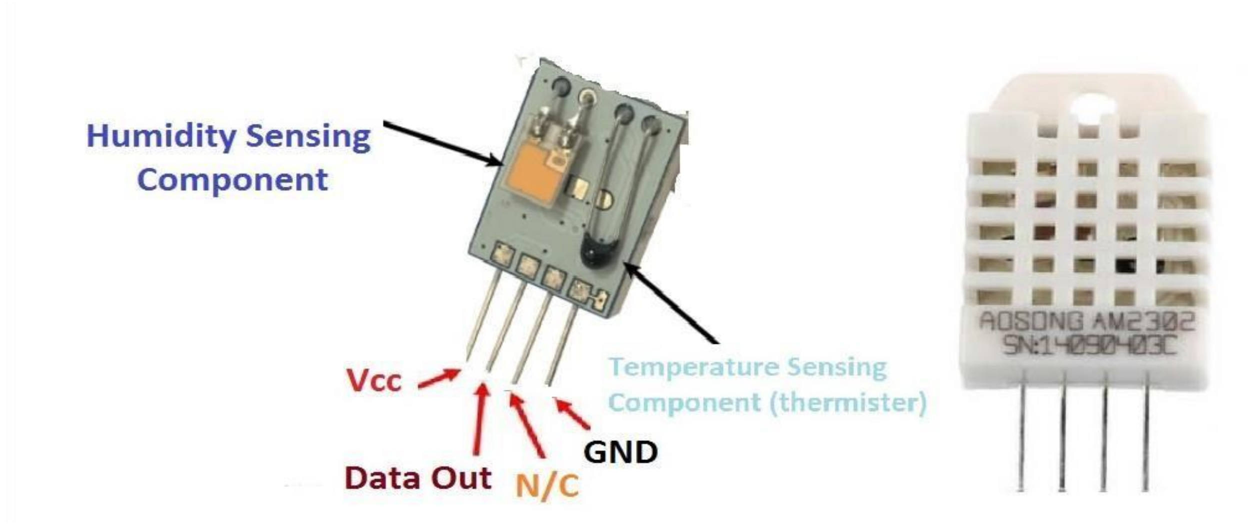

Humidity Sensing Component
Vcc
Data Out
N/C
GND
Temperature Sensing Component (thermister)
AOSONG AM2302
SN:14090403C

Fig 2.11

2.6.1 Pin Description

- Vcc is the power pin. Apply voltage in a range of 3.5 V to 5.0 V at this pin.
- Data Out is the digital output pin. It sends out the value of measured temperature and humidity in the form of serial data.
- N/C is a not connect pin.
- GND: Connect the GND pin to main ground.

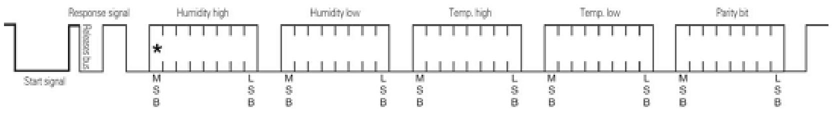
2.6.2 Operating specifications

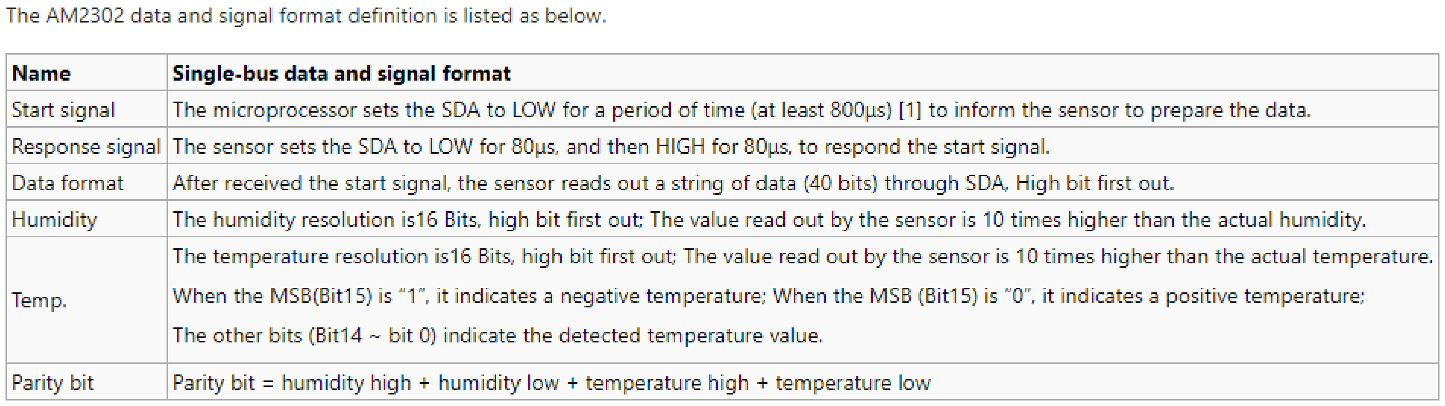
(1) **Power and Pins**:-

Power's voltage should be 3.3-6V DC. When power is supplied to sensor, don't send any instruction to the sensor within one second to pass unstable status. One capacitor valued 100nF can be added between VDD and GND for wave filtering.

(2) **Communication and signal**:-

Single-bus data is used for communication between MCU and DHT22, it costs 5mS for single time communication. Data is comprised of integral and decimal part, the following is the formula for data. DHT22 send out higher data bit firstly! DATA=8 bit integral RH data+8 bit decimal RH data+8 bit integral T data+8 bit decimal T data+8 bit check-sum If the data transmission is right, check-sum should be the last 8 bit of "8 bit integral RH data+8 bit decimal RH data+8 bit integral T data+8 bit decimal T data". When MCU send start signal, DHT22 change from low-power-consumption-mode to running-mode. When MCU finishes sending the start signal, DHT22 will send response signal of 40-bit data that reflect the relative humidity.




The AM2302 data and signal format definition is listed as below.

Name	**Single-bus data and signal format**
Start signal	The microprocessor sets the SDA to LOW for a period of time (at least 800µs) [1] to inform the sensor to prepare the data.
Response signal	The sensor sets the SDA to LOW for 80µs, and then HIGH for 80µs, to respond the start signal.
Data format	After received the start signal, the sensor reads out a string of data (40 bits) through SDA, High bit first out.
Humidity	The humidity resolution is16 Bits, high bit first out; The value read out by the sensor is 10 times higher than the actual humidity.
Temp.	The temperature resolution is16 Bits, high bit first out; The value read out by the sensor is 10 times higher than the actual temperature. When the MSB(Bit15) is "1", it indicates a negative temperature; When the MSB (Bit15) is "0", it indicates a positive temperature; The other bits (Bit14 ~ bit 0) indicate the detected temperature value.
Parity bit	Parity bit = humidity high + humidity low + temperature high + temperature low

2.6.3 2D – model of the sensor

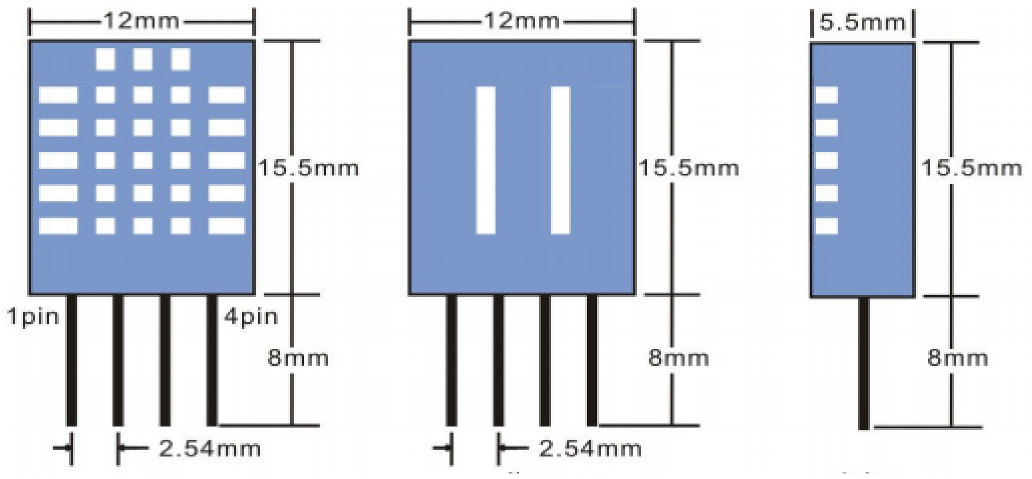


Fig 2.12

2.6.4 Applications

- Weather station for measuring temperature and monitoring environment
- Automatic climate control
- Humidity controller for measuring relative humidity
- Test and detection device

2.6.5 DHT22 Connection with Arduino Uno

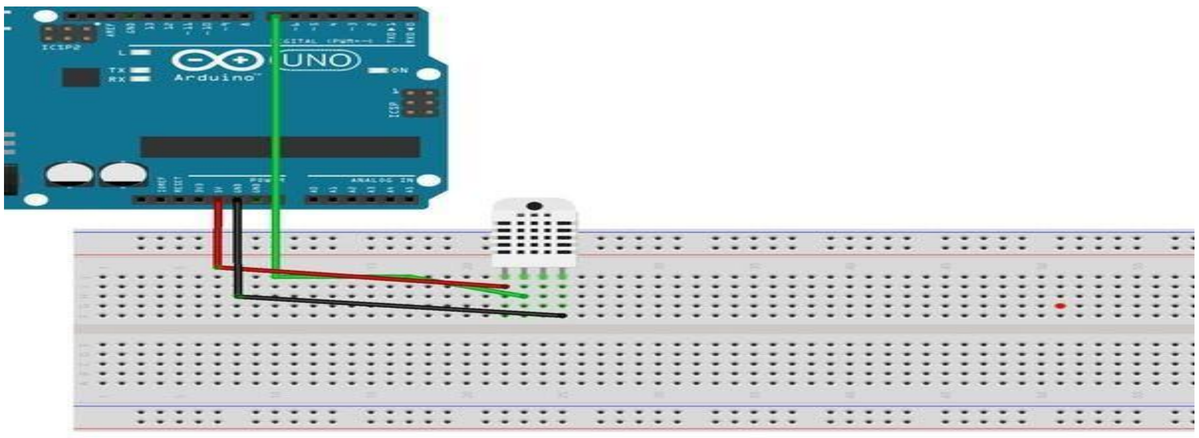


Fig 2.13

2.7 LIQUID CRYSTAL DISPLAY (LCD)

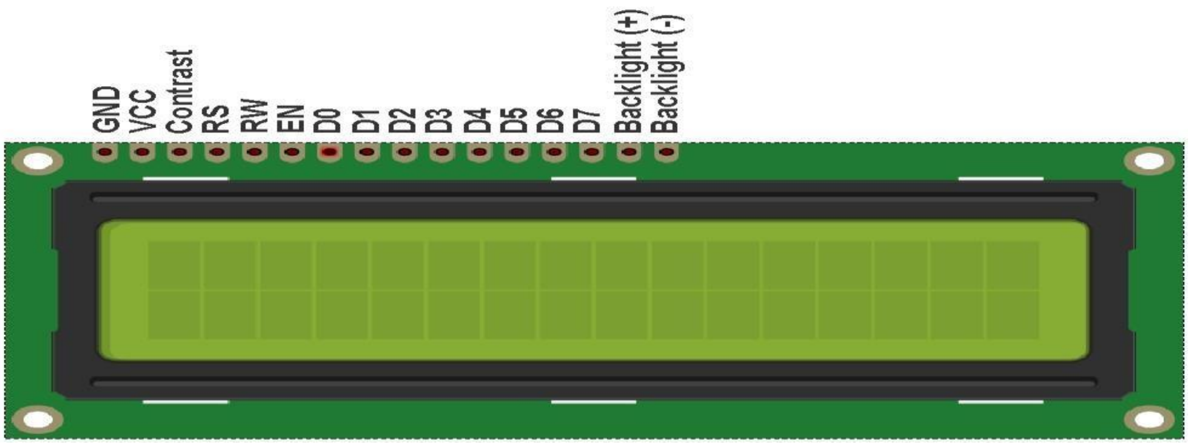


Fig 2.14

A 16x2 LCD display **is very basic module and is very commonly used in various devices and circuits. A** 16x2 LCD **means it can** display **16 characters per line and there are 2 such lines. In this** LCD **each character is** displayed **in 5x7 pixel matrix. This** LCD **has two registers, namely, Command and Data.**

2.7.1 Pin Diagram

The 16×2 LCD pinout is shown below.

- Pin1 (Ground/Source Pin): This is a GND pin of display, used to connect the GND terminal of the microcontroller unit or power source.
- Pin2 (VCC/Source Pin): This is the voltage supply pin of the display, used to connect the supply pin of the power source.
- Pin3 (V0/VEE/Control Pin): This pin regulates the difference of the display, used to connect a changeable POT that can supply 0 to 5V.
- Pin4 (Register Select/Control Pin): This pin toggles among command or data register, used to connect a microcontroller unit pin and obtains either 0 or 1(0 = data mode, and 1 = command mode).
- Pin5 (Read/Write/Control Pin): This pin toggles the display among the read or writes operation, and it is connected to a microcontroller unit pin to get either 0 or 1 (0 = Write Operation, and 1 = Read Operation).
- Pin 6 (Enable/Control Pin): This pin should be held high to execute Read/Write process, and it is connected to the microcontroller unit & constantly held high.
- Pins 7-14 (Data Pins): These pins are used to send data to the display. These pins are connected in two-wire modes like 4-wire mode and 8-wire mode. In 4-wire mode, only four pins are connected to the microcontroller unit like 0 to 3, whereas in 8-wire mode, 8-pins are connected to microcontroller unit like 0 to 7.
- Pin15 (+ve pin of the LED): This pin is connected to +5V
- Pin 16 (-ve pin of the LED): This pin is connected to GND.

2.7.2 Features of LCD16x2

The features of this LCD mainly include the following.

- The operating voltage of this LCD is 4.7V-5.3V
- It includes two rows where each row can produce 16-characters.
- The utilization of current is 1mA with no backlight
- Every character can be built with a 5×8 pixel box
- The alphanumeric LCDs alphabets & numbers
- Is display can work on two modes like 4-bit & 8-bit
- These are obtainable in Blue & Green Backlight
- It displays a few custom generated characters

2.7.3 Registers of LCD

A 16×2 LCD has two registers like data register and command register. The RS (register select) is mainly used to change from one register to another. When the register set is '0', then it is known as command register. Similarly, when the register set is '1', then it is known as data register.

Command Register:-

The main function of the command register is to store the instructions of command which are given to the display. So that predefined tasks can be performed such as clearing the display, initializing, set the cursor place, and display control. Here commands processing can occur within the register.

Data Register:-

The main function of the data register is to store the information which is to be exhibited on the LCD screen. Here, the ASCII value of the character is the information which is to be exhibited on the screen of LCD. Whenever we send the information to LCD, it transmits to the data register, and then the process will be starting there. When register set =1, then the data register will be selected.

2.7.4 LCD Connection with Arduino Uno

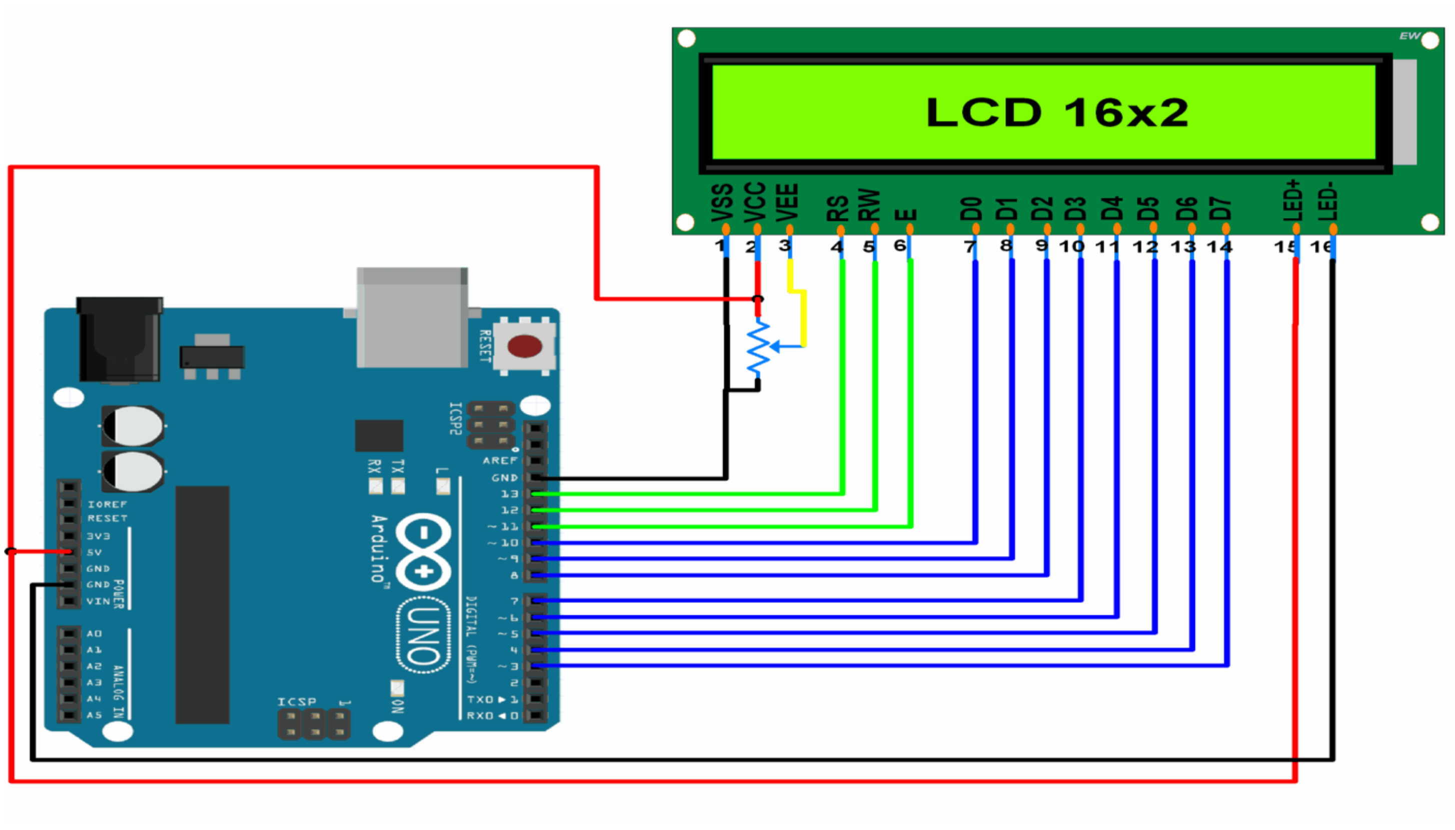


Fig 2.15

CHAPTER-3

SOFTWARE REQUIRED

CHAPTER – 3

SOFTWARE REQUIRED

3.1 Software Required:

- Embedded C
- Arduino IDE
- Installation in Blynk Application

3.2 EMBEDDED -C

- Embedded C is one of the most popular and most commonly used Programming Languages in the development of Embedded Systems. So, in this article, we will see some of the Basics of Embedded C Program and the Programming Structure of Embedded C.

- Embedded C is perhaps the most popular languages among Embedded Programmers for programming Embedded Systems. There are many popular programming languages like Assembly, BASIC, C++, Python etc. that are often used for developing Embedded Systems but Embedded C remains popular due to its efficiency, less development time and portability.

- Before digging in to the basics of Embedded C Program, we will first take a look at what an Embedded System is and the importance of Programming Language in Embedded Systems.An Embedded System can be best described as a system which has both the hardware and software and is designed to do a specific task. A good example for an Embedded System, which many households have, is a Washing Machine.

- We use washing machines almost daily but wouldn't get the idea that it is an embedded system consisting of a Processor (and other hardware as well) and software.

3.2.1 Programming Embedded Systems

As mentioned earlier, Embedded Systems consists of both Hardware and Software. If we consider a simple Embedded System, the main Hardware Module is the Processor. The Processor is the heart of the Embedded System and it can be anything like a Microprocessor, Microcontroller, DSP, CPLD (Complex Programmable Logic Device) or an FPGA (Field Programmable Gated Array).All these devices have one thing in common: they are programmable i.e., we can write a program (which is the software part of the Embedded System) to define how the device actually works.

Embedded Software or Program allow Hardware to monitor external events (Inputs / Sensors) and control external devices (Outputs) accordingly. During this process, the program for an Embedded System may have to directly manipulate the internal architecture of the Embedded Hardware (usually the processor) such as Timers, Serial Communications Interface, Interrupt Handling, and I/O Ports etc.

From the above statement, it is clear that the Software part of an Embedded System is equally important as the Hardware part. There is no point in having advanced Hardware Components with poorly written programs (Software).There are many programming languages that are used for Embedded Systems like Assembly (low-level Programming Language), C, C++, JAVA (high-level programming languages), Visual Basic, JAVA Script (Application level Programming Languages), etc.

In the process of making a better embedded system, the programming of the system plays a vital role and hence, the selection of the Programming Language is very important.

3.3 ARDUINO IDE SOFTWARE

Arduino is a prototype platform (open-source) based on an easy-to-use hardware and software. It consists of a circuit board, which can be programed (referred to as a microcontroller) and a ready-made software called Arduino IDE (Integrated Development Environment), which is used to write and upload the computer code to the physical board.

The key features are: • Arduino boards are able to read analog or digital input signals from different sensors and turn it into an output such as activating a motor, turning LED on/off, connect to the cloud and many other actions. • You can control your board functions by sending a set of instructions to the microcontroller on the board via Arduino IDE (referred to as uploading software). • Unlike most previous programmable circuit boards, Arduino does not need an extra piece of hardware (called a programmer) in order to load a 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 the functions of the microcontroller into a more accessible package.

- Step 1: Install the Arduino Software (IDE)
- Step 2: Get an Uno R3 and USB cable
- Step 3: Connect the board

```
void setup() {
  // put your setup code here, to run once:

}

voi
  //

}
```

1.6.4
ARDUINO
AN OPEN PROJECT WRITTEN, DEBUGGED, AND SUPPORTED BY ARDUINO.CC AND THE ARDUINO COMMUNITY WORLDWIDE
LEARN MORE ABOUT THE CONTRIBUTORS OF ARDUINO.CC on arduino.cc/credits

Fig 3.1

3.4 BLYNK APPLICATION

Blynk was designed for the Internet of Things. It can control hardware remotely, it can display sensor data, it can store data, vizualize it and do many other cool things.
There are three major components in the platform:

- **Blynk App** - allows to you create amazing interfaces for your projects using various widgets we provide.

- **Blynk Server** - responsible for all the communications between the smartphone and hardware. You can use our Blynk Cloud or run your private Blynk server locally. It's open-source, could easily handle thousands of devices and can even be launched on a Raspberry Pi.

- **Blynk Libraries** - for all the popular hardware platforms - enable communication with the server and process all the incoming and outcoming commands.

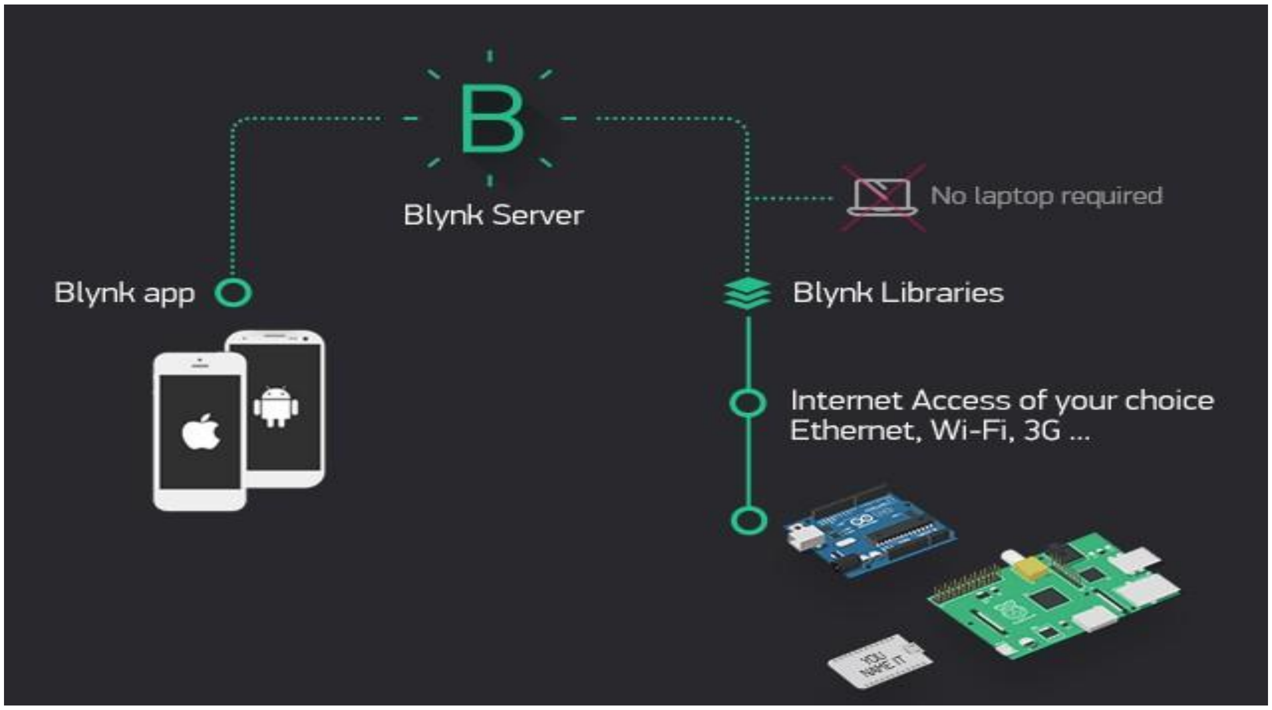


Fig 3.2

3.4.1 FEATURES

- Similar API & UI for all supported hardware & devices

- Connection to the cloud using:
    - WiFi
    - Bluetooth and BLE
    - Ethernet
    - USB (Serial)
    - GSM
    - …
- Set of easy-to-use Widgets
- Direct pin manipulation with no code writing
- Easy to integrate and add new functionality using virtual pins
- History data monitoring via SuperChart widget
- Device-to-Device communication using Bridge Widget
- Sending emails, tweets, push notifications, etc.

1. Hardware.
An Arduino, Raspberry Pi, or a similar development kit.
Blynk works over the Internet. This means that the hardware you choose should be able to connect to the internet. Some of the boards, like Arduino Uno will need an Ethernet or Wi-Fi Shield to communicate, others are already Internet-enabled: like the ESP8266, Raspberri Pi with WiFi dongle, Particle Photon or SparkFun Blynk Board. But even if you don’t have a shield, you can connect it over USB to your laptop or desktop (it’s a bit more complicated for newbies, but we got you covered). What’s cool, is that the list of hardware that works with Blynk is huge and will keep on growing.

2. A Smartphone.
The Blynk App is a well designed interface builder. It works on both iOS and Android.

3.4.2 STEPS

STEP 1**:-** Create account

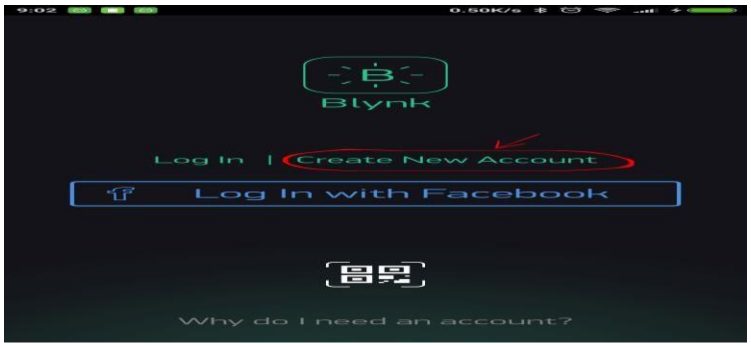


Fig 3.3

STEP 2:- CREATE A NEW PROJECT

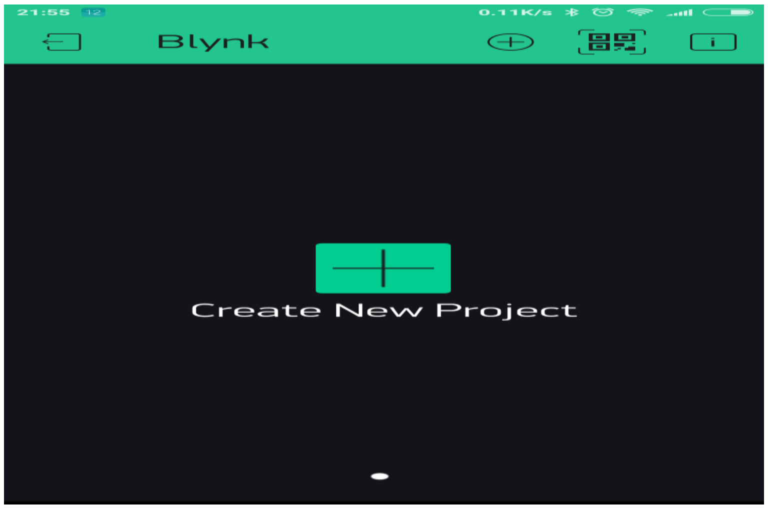


Fig 3.4

STEP 3:- CHOOSE THE HARDWARE

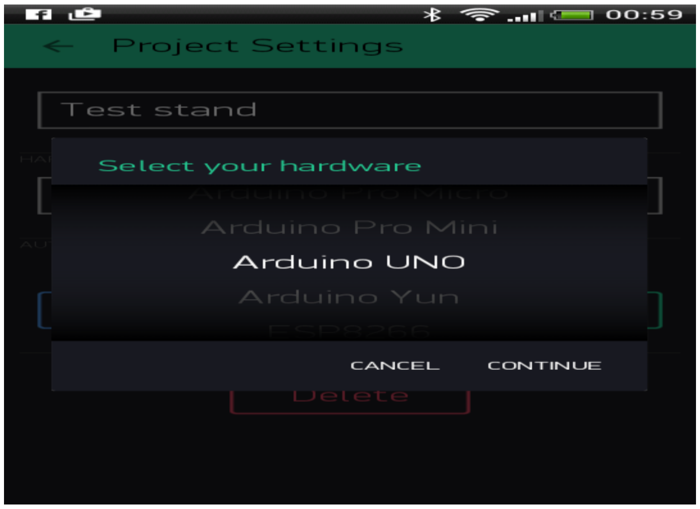


Fig 3.5

STEP 4:- AUTHENTICATION TOCKEN

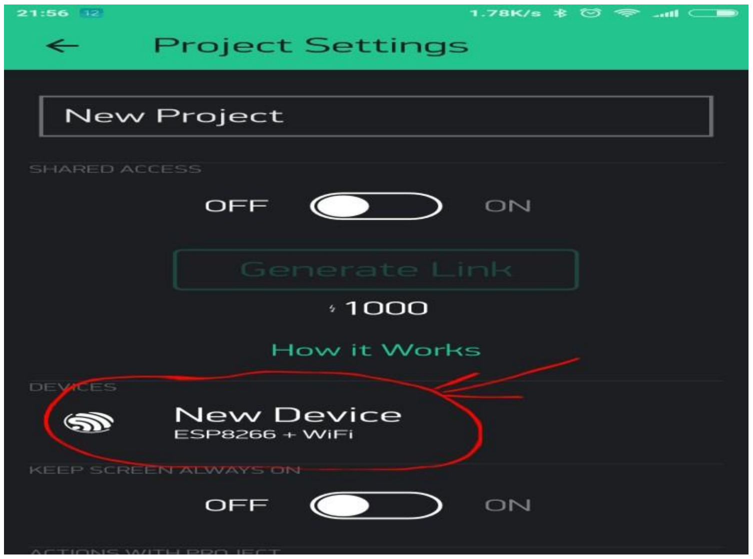


Fig 3.6

STEP 5:- CREATE A NEW PROJECT

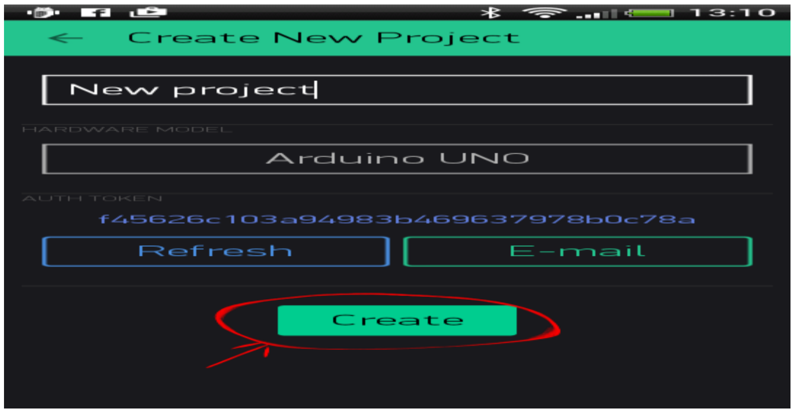


Fig 3.7

STEP 6:- Add a widget

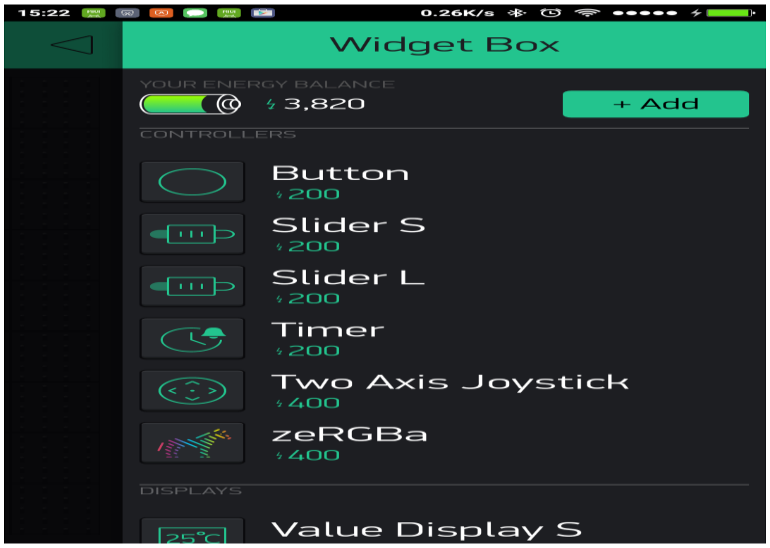
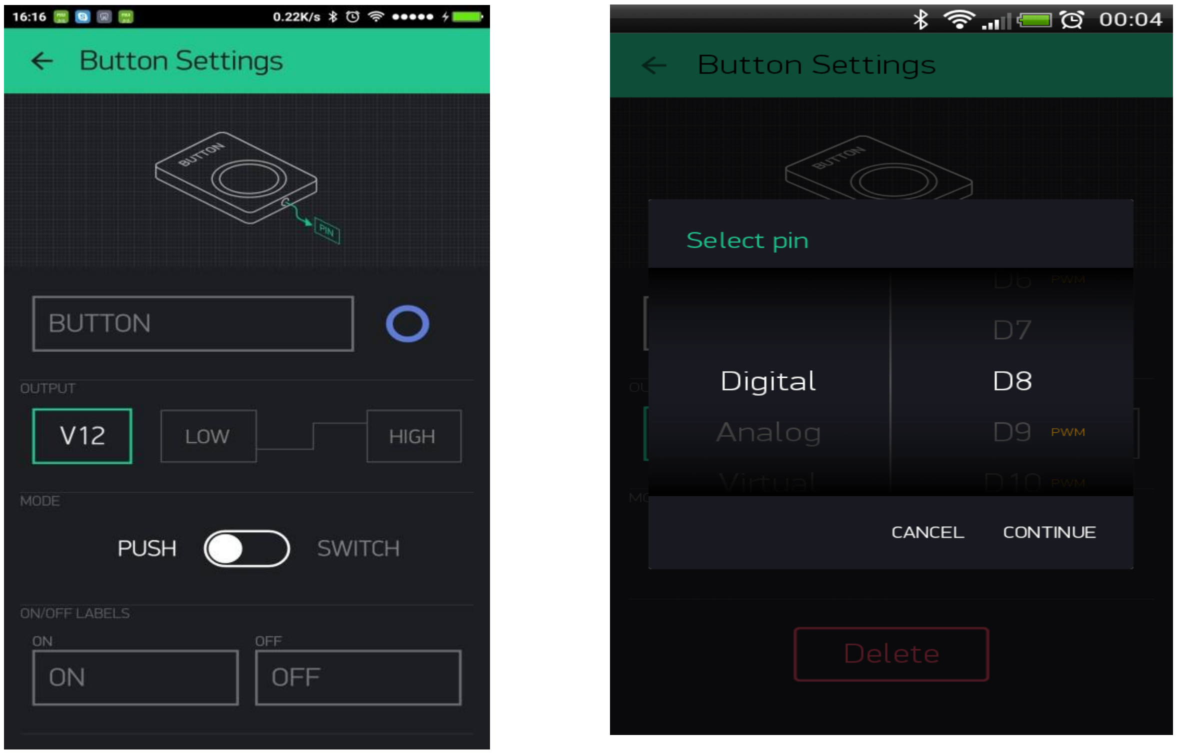


Fig 3.8

STEP 7:- The terminals are set for receiving the data



BUTTON
OUTPUT V12
MODE PUSH SWITCH
ON ON
OFF OFF
Select pin: Digital D8

Fig 3.9

STEP 8:- RUN THE PROJECT

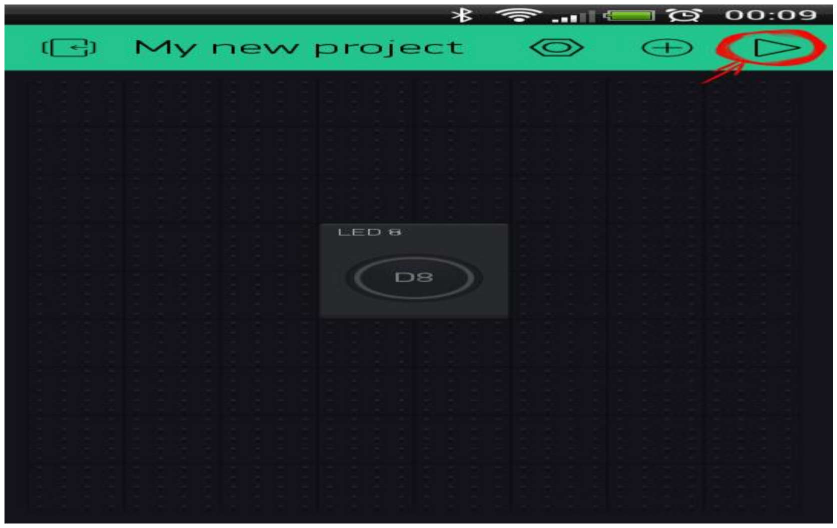


Fig 3.10

3.4.3 CODE for NODEMCU AND BLYNK APPLICATION Connection:-

```
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

// You should get Auth Token in the Blynk App.
// Go to the Project Settings (nut icon).
char auth[] = "HURskrQZ33mTskB1bwDF4EeKqjwOX9z3";

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "iot";
char pass[] = "08910891";

WidgetTerminal terminal(V0);


void setup() {
  // Open serial communications and wait for port to open:
  Serial.begin(9600);
 Blynk.begin(auth, ssid, pass);

 while (!Serial) {
    ; // wait for serial port to connect. Needed for native USB port only
  }
}

void loop() { // run over and over
   Blynk.run();
   if (Serial.available())
   {
     char ch = Serial.read();

    Serial.write(ch);//for pc
  terminal.write(ch);// for terminal
  }

}
```

CHAPTER 4

METHODOLOGY

CHAPTER-4

METHODOLOGY

4.1 Block Diagram

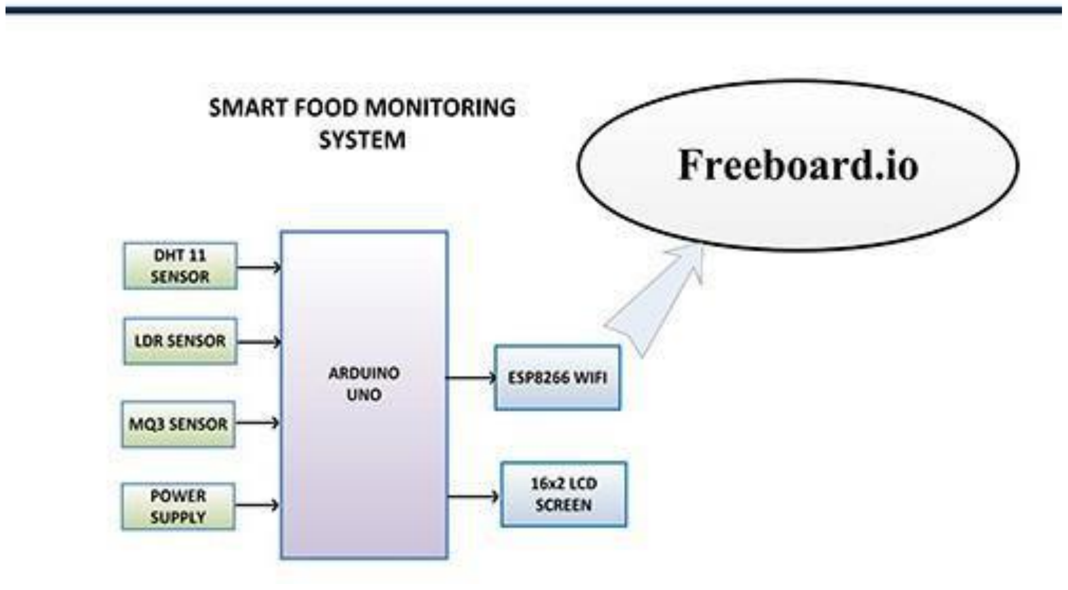


Fig 4.1

4.2 WORKING PRINCIPLE

There is a sensor unit to monitor the critical environmental parameters like temperature, humidity, light, gas content etc.

The DHT-22 sensor which is a digital sensor will sense the humidity and temperature of the food and give it to the Arduino. MQ3 alcohol sensor is an analog sensor module which is used to detect the presence of ethanol. Ethylene is a natural gas given off by fruits that causes in ripening. Apples and Bananas give off more ethylene than other fruits at the stage of ripening.

LDR is used to sense the intensity of light. LCD [Liquid Crystal Display] is 16x2 dipaly which displays the values from the sensors.

The values from these sensors are given to the Arduino UNO [ATmega328 based microcontroller board]. It has 14 GPIO pins, 6 PWM pins, 6 Analog inputs and an on board UART,SPI and Interface. The values from the Arduino are given to NODE MCU. NODE MCU is an open source firmware and development board which has an in built Wifi module and this uploads the data into the cloud and the data from the cloud is taken by the BLYNK application.

4.3 Implementation

IoT Device should be installed at the place where food is present. Once it is properly installed and powered on, it connects with the internet through Wi-Fi modem and starts reading data from the interfaced sensors-DHT22 the successor of the DHT11 module. The **DHT22** is a commonly used Temperature and humidity sensor. The sensor can measure temperature from **-40°C to 80°C** and humidity from **0% to 100%** with an accuracy of ±1°C and ±1%.
DHT22 is a digital sensor and this sensor can be directly connected to the digital pin in the Arduino as the Arduino have 14 digital pins. This sensor is connected to the second digital pin of the Arduino board.
MQ3 Sensor is an alcohol sensor module which is used to detect the presence of ethanol, where the sensitive material used for this sensor is SnO2, whose conductivity is lower in clean air.
MQ3 Sensor is an analog sensor and this sensor is directly connected to the analog pin in the Arduino as the Arduino have 6 analog pins. This sensor is connected to the A0 pin of the Arduino board.
LDR(LIGHT DEPENDENT RESISTOR) is used to sense the intensity of light.
LDR sensor is also an analog sensor and this sensor is directly connected to the analog pin in the Arduino as there are 5 remaining Analaog pins. This sensor is connected to the A1 pin of the Arduino board.

The values from the sensors that are obtained are given to the Arduino as the sensors are connected to the pins of the Arduino board. If the values obtained from the sensors reach the critical values then as per the code written in the Arduino software then we get a display regarding the condition of food.

In this project, Bananas are tested at different conditions. Too high a temperature destroys enzymes, and too low a temperature can break down the cell walls of the fruit so the contents mix and the fruit oxidizes, browns and softens abnormally. The optimum temperature and humidity conditions for ripening are 62 to 68 degrees Fahrenheit and 90 to 95 percent relative humidity. The Critical value of the ethylene gas is also found by testing the banana for about 7-9 days and the value is converted into meter cube(m^3). The main use of LDR in this project is to tell the user to place the bananas in a closed box if the intensity is high (Intensity is high if kept in an open area ie., more light will be present in an open area rather than in a closed box). Here food should be kept in a closed box inside a refrigerator, if we keep the bananas in an open area then the humidity and temperature will effect the bananas.

Millis() Returns the number of milliseconds passed since the Arduino board began running the current program. This number will overflow (go back to zero), after approximately 50 days. This is used ie.,If the food is kept at high temperatures and humidity for 2 or more days then the food gets spoiled and before the food gets spoiled then we give a notification to the user. Generally Bananas kept at nearly room temperatures will last for more time nearly 7 days. Upon crossing this time limit then the food will be spoiled.

The transmitter and receiver pins of the Arduino Uno are connected to the receiver and transmitter pins of the NODE MCU respectively. Initially as the pins are connected then the values in the serial communication port of the Arduino are given to the NODE MCU and as the NODE MCU has an in built Wifi module, it uploads the data collected into the cloud.
The cloud acts as an interface between the BLYNK application and the NODE MCU.
The BLYNK application at the time of installation generates a unique authentication id that is used to get the data inside the cloud. In the BLYNK application, V0 terminal is used in order to receive the data form the cloud. Now the data regarding the spoilage of food can also be seen in mobile.

Even the data is also displayed on a LCD 16X2 Display where is shows the values from the sensors and also the condition of the food is also displayed. This is used as the user can also be at home and he can monitor the bananas.

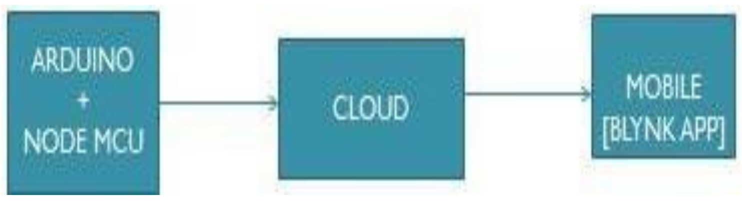


Fig 4.2

4.4 **Practical Circuit**:-

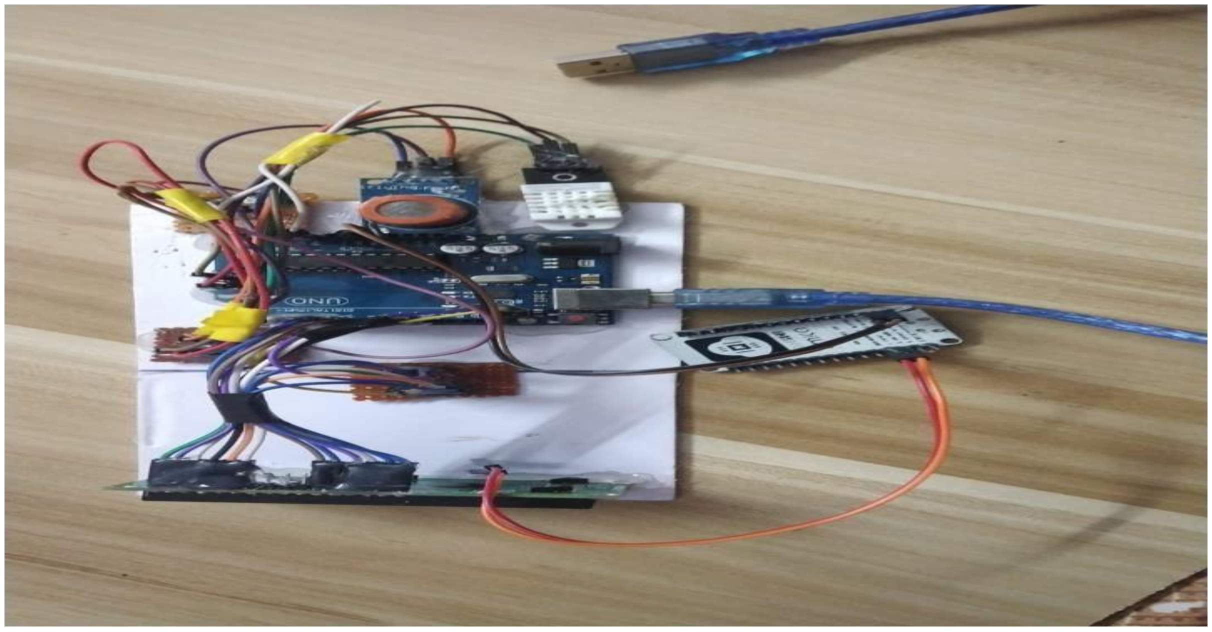


Fig 4.3

4.5 ALGORITHM OF THE CODE

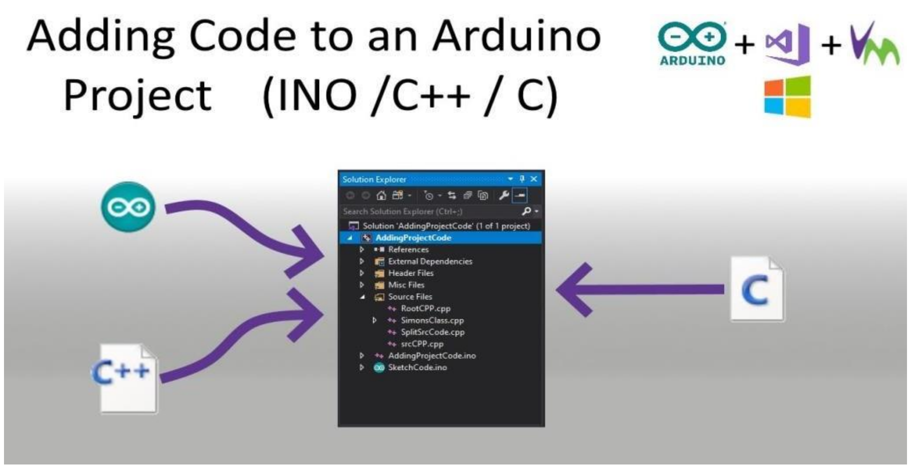


Fig 4.4

- Firstly header files are added for LCD and DHT22 sensor.
- The 2nd pin of Arduino is connected to Data Out pin of DHT 22 sensor.
- MQ3 and LDR sensors Data Out pins are connected to A0 and A1 pins of Arduino respectively.
- The variables representing pin connections to read the sensor data are initialized.
- A function is called in which baud rate for the serial communication for communication with the Wi-Fi modem is set to 9600.
- Header files are installed.
- A unique authentication token is set which is generated at the time of Blynk app installation.
- Ssid and password are also set. A terminal V0 is initailised inside the blynk app inorder to recive the data.
- Another function is called in which the sensor data is fetched and stored in the initialized variables and as the tx and rx pins of the arduino are connected to the nodemcu,data is transferred to nodemcu.
  From nodemcu data is transferred to cloud and then to blynk app.

4.5.1 Initialization of Code for taking inputs from sensors and displaying

```
#include<LiquidCrystal.h>
LiquidCrystal lcd(12,11,7,6,5,4);
#include <SimpleDHT.h>
int pinDHT22 = 2;
SimpleDHT22 dht22(pinDHT22);

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



lcd.begin(16,2);
lcd.print(" FOOD SAFETY ");
lcd.setCursor(0,1);
lcd.print(" MONITORING_SYS");
delay(2000);
lcd.clear();
}

void loop() {
  // start working...

  // read without samples.
  // @remark We use read2 to get a float data, such as 10.1*C
  //    if user doesn't care about the accurate data, use read to get a byte data, such as 10*C.
  float temperature = 0;
  float humidity = 0;
  int err = SimpleDHTErrSuccess;
  if ((err = dht22.read2(&temperature, &humidity, NULL)) != SimpleDHTErrSuccess) {
    Serial.print("Read DHT22 failed, err="); Serial.print(SimpleDHTErrCode(err));
    Serial.print(","); Serial.println(SimpleDHTErrDuration(err)); delay(2000);
    return;
  }
  int gas = analogRead(A1);
  int ldr = analogRead(A0);
  unsigned long myTime;


  /*Serial.print((int)temperature);
  Serial.print(" *C, ");
  Serial.print((int)humidity);
  Serial.println(" RH%");*/

  lcd.setCursor(0,0);
  lcd.print("TEM");     // temperature
  lcd.setCursor(0,1);
  int temp= temperature;
  lcd.print( temp);
  Serial.print("Temp ");
  Serial.print(temp);

  lcd.setCursor(4,0);
  lcd.print("HUM");
  lcd.setCursor(4,1);// humidity
  int hum = humidity;
  lcd.print(  hum);
  Serial.print(" hum:");
  Serial.print(hum);
```

```
lcd.setCursor(8,0);      //ldr
lcd.print("LDR");
lcd.setCursor(8,1);
ldr=ldr/10;
lcd.print( ldr);
Serial.print(" ldr:");
Serial.print(  ldr);



lcd.setCursor(12,0);
lcd.print("GAS");
lcd.setCursor(12,1);
gas = gas/10;              //gas
lcd.print( gas);
Serial.print(" gas:");
Serial.println(gas);
```

CHAPTER 5

RESULTS

CHAPTER 5

RESULTS

5.1 Practical setup

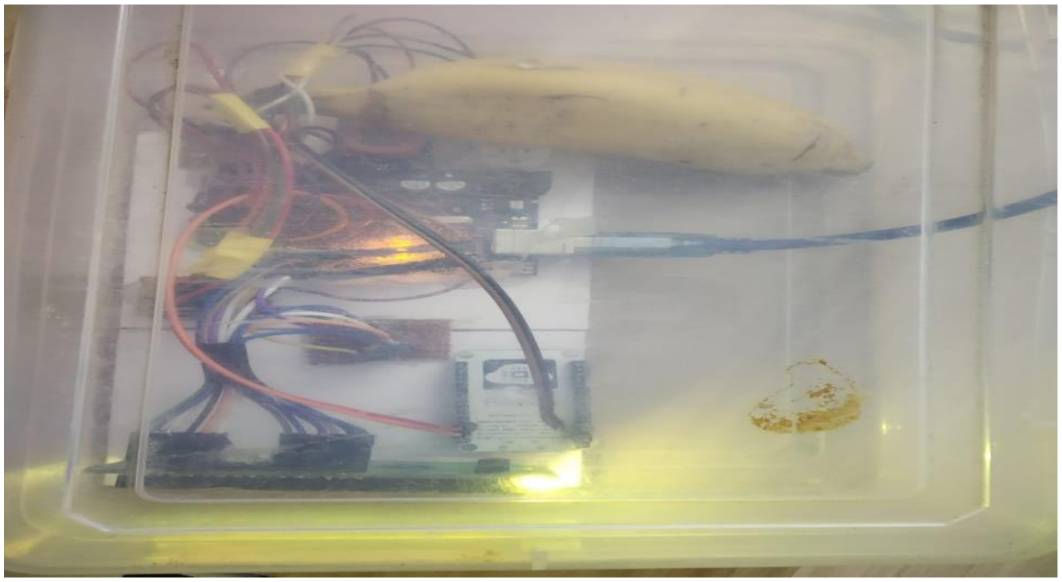


Fig 5.1

5.2 Values displayed in LCD

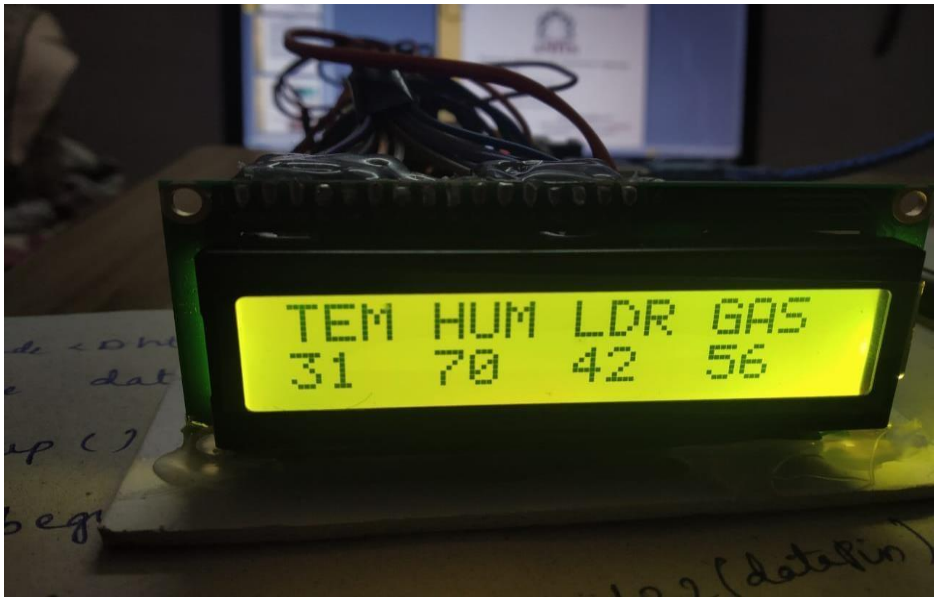
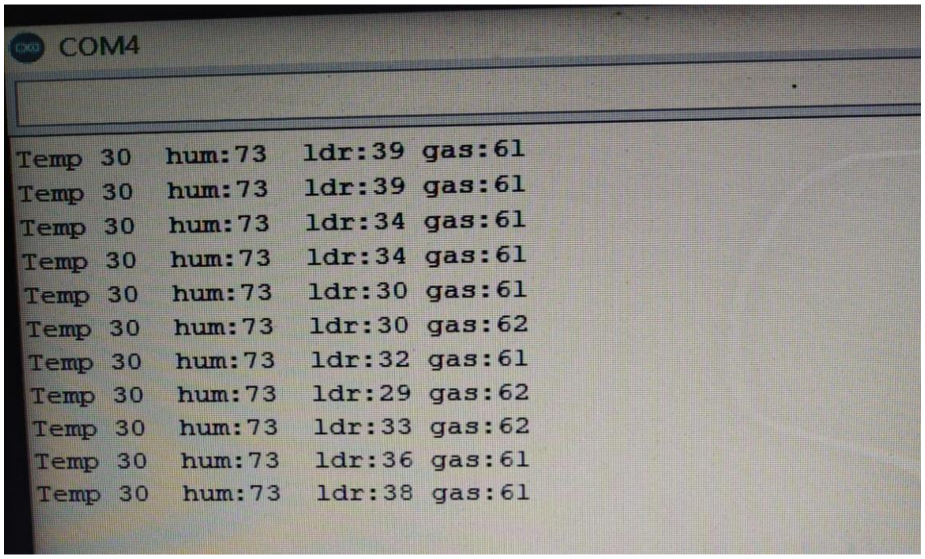


Fig 5.2

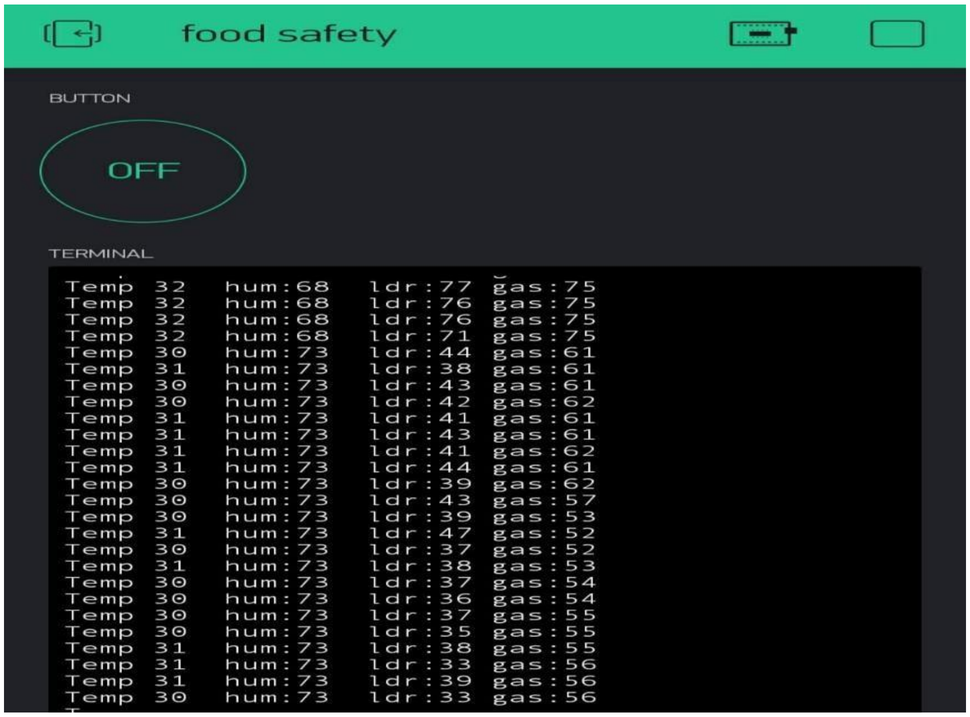
5.3 Values in Serial communication port of Arduino IDE



```
Temp 30  hum:73  ldr:39 gas:61
Temp 30  hum:73  ldr:39 gas:61
Temp 30  hum:73  ldr:34 gas:61
Temp 30  hum:73  ldr:34 gas:61
Temp 30  hum:73  ldr:30 gas:61
Temp 30  hum:73  ldr:30 gas:62
Temp 30  hum:73  ldr:32 gas:61
Temp 30  hum:73  ldr:29 gas:62
Temp 30  hum:73  ldr:33 gas:62
Temp 30  hum:73  ldr:36 gas:61
Temp 30  hum:73  ldr:38 gas:61
```

Fig 5.3

5.4 Values in Blynk app



```
Temp 32  hum:68  ldr:77 gas:75
Temp 32  hum:68  ldr:76 gas:75
Temp 32  hum:68  ldr:76 gas:75
Temp 32  hum:68  ldr:71 gas:75
Temp 30  hum:73  ldr:44 gas:61
Temp 31  hum:73  ldr:38 gas:61
Temp 30  hum:73  ldr:43 gas:61
Temp 30  hum:73  ldr:42 gas:62
Temp 31  hum:73  ldr:41 gas:61
Temp 31  hum:73  ldr:43 gas:61
Temp 31  hum:73  ldr:41 gas:62
Temp 31  hum:73  ldr:44 gas:61
Temp 30  hum:73  ldr:39 gas:62
Temp 30  hum:73  ldr:43 gas:57
Temp 30  hum:73  ldr:39 gas:53
Temp 31  hum:73  ldr:47 gas:52
Temp 30  hum:73  ldr:37 gas:52
Temp 31  hum:73  ldr:38 gas:53
Temp 30  hum:73  ldr:37 gas:54
Temp 30  hum:73  ldr:36 gas:54
Temp 30  hum:73  ldr:37 gas:55
Temp 30  hum:73  ldr:35 gas:55
Temp 31  hum:73  ldr:38 gas:55
Temp 31  hum:73  ldr:33 gas:56
Temp 31  hum:73  ldr:39 gas:56
Temp 30  hum:73  ldr:33 gas:56
```

Fig 5.4

5.5 Table of Observations

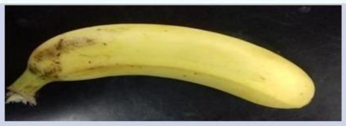
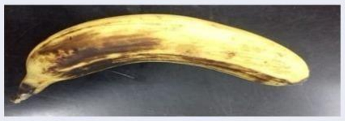
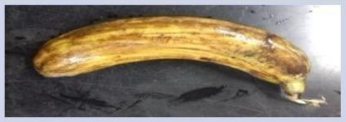
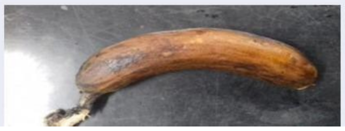
S.no	Condition of Banana (At room temperature)	Gas Value (as shown by the MQ3 sensor) in meter cube
1		Banana is edible [50-58]
2		Banana is edible [60-68]
3		Banana is about to spoil [70-75]
4		Banana is spoiled [More than 75]

Table 5.1

LDR VALUES> 60 to 70 : ❼ Place the food in a closed box

5.6 Practical testing of Banana on different days

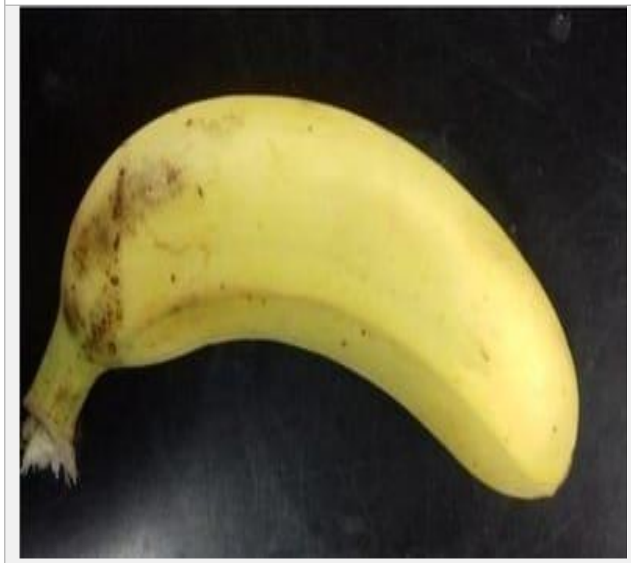
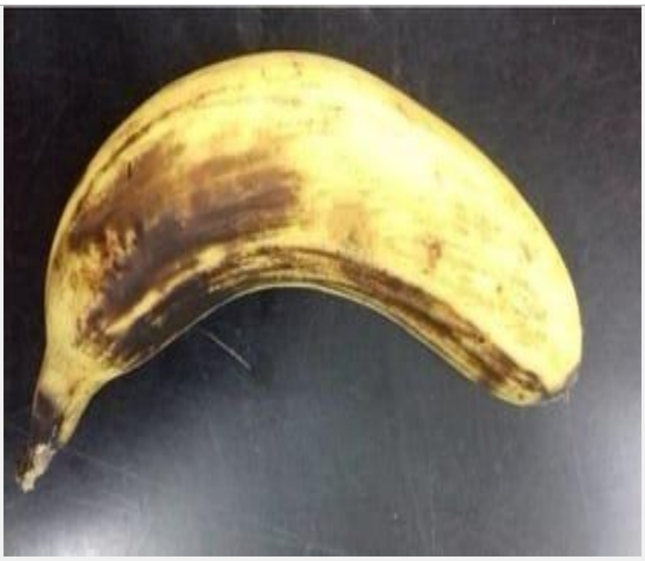
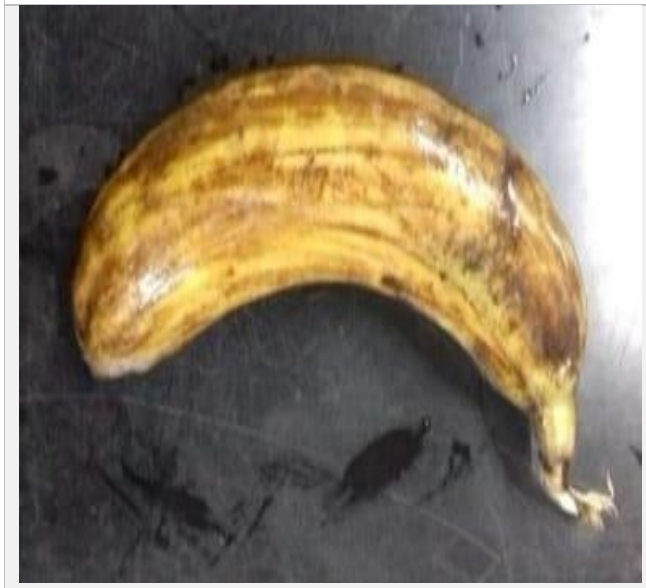
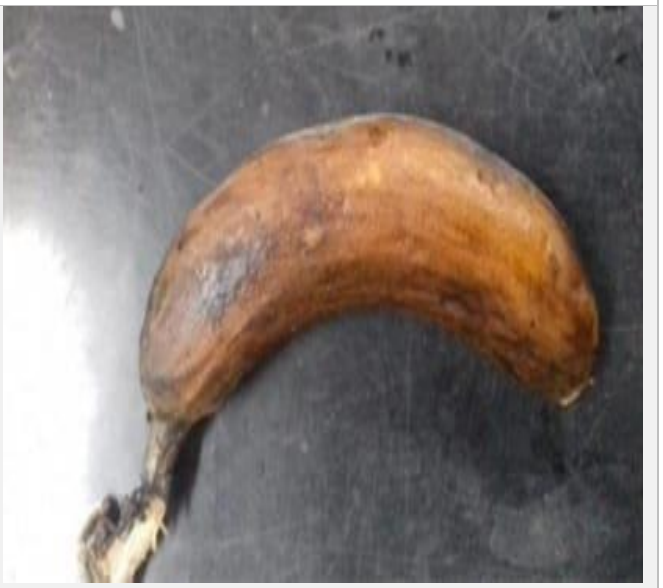
Banana on Day 1: It is Fresh and edible	Banana on Day 3: It is edible
	
Banana on Day 5:It is about to spoil	Banana on Day 8:It is spoiled
	

Table 5.2

Bananas are tested at room temperature and the critical values of the gas content are noted

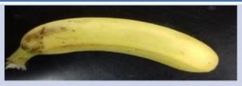
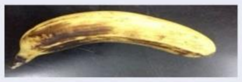
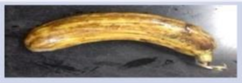
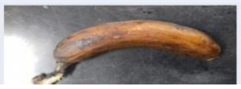
S.no	Condition of Banana (At room temperature)	Gas Value (as shown by the MQ3 sensor) in meter cube
1		Banana is edible [50-58]
2		Banana is edible [60-68]
3		Banana is about to spoil [70-75]
4		Banana is spoiled [More than 75]

Table 5.3 **Table showing the condition of a Banana**

Arduino Uno acts as a central unit and takes care of all the activities of the system. The main role of Arduino is to take action based on the inputs provided by outputs of the sensors (as the outputs from the sensors are connected to the inputs of the Arduino). Embedded C is used for programming in Arduino Uno. Node MCU is connected to the Arduino Uno in order to serve the purpose of the Wifi Module[14] for establishing a connection for IoT purpose.

- If the user is near to the place where food is kept,
  The details of the condition of food is displayed in the LCD.

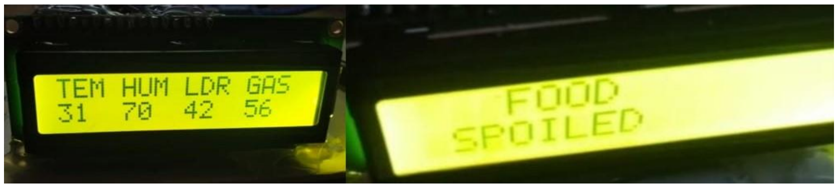
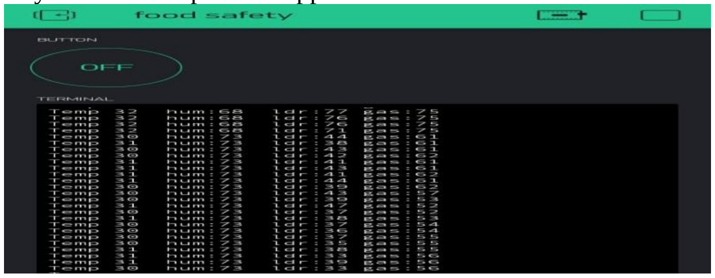


Fig.5.5 LCD Display

At the stage of spoiling, the user gets a notification that ‘Food is about to spoil’ (Here “Bananas” are taken as sample food for practical testing). Whenever the user sees this notification, the user can complete the food as fast as possible. Upon further passing of the days, if the food is spoiled then the user can see in the LCD display that ‘Food spoiled’ so that he can know that he should not consume the spoiled Banana.

- If the user is at a farther place,

For displaying and monitoring, data is uploaded to the cloud server. From the cloud server, the data is given to the Blynk application, where users can get a notification in numerous ways with the help of this application.



```
Temp 32   hum:68   ldr:77   gas:75
Temp 32   hum:68   ldr:76   gas:75
Temp 32   hum:68   ldr:76   gas:75
Temp 32   hum:68   ldr:71   gas:75
Temp 30   hum:73   ldr:44   gas:61
Temp 31   hum:73   ldr:38   gas:61
Temp 30   hum:73   ldr:43   gas:61
Temp 30   hum:73   ldr:42   gas:62
Temp 31   hum:73   ldr:41   gas:61
Temp 31   hum:73   ldr:43   gas:61
Temp 31   hum:73   ldr:41   gas:62
Temp 31   hum:73   ldr:44   gas:61
Temp 30   hum:73   ldr:39   gas:62
Temp 30   hum:73   ldr:43   gas:57
Temp 30   hum:73   ldr:39   gas:53
Temp 31   hum:73   ldr:47   gas:52
Temp 30   hum:73   ldr:37   gas:52
Temp 31   hum:73   ldr:35   gas:53
Temp 30   hum:73   ldr:37   gas:54
Temp 30   hum:73   ldr:36   gas:54
Temp 30   hum:73   ldr:37   gas:55
Temp 30   hum:73   ldr:35   gas:55
Temp 31   hum:73   ldr:38   gas:55
Temp 31   hum:73   ldr:33   gas:56
Temp 31   hum:73   ldr:39   gas:56
Temp 30   hum:73   ldr:33   gas:56
```

Fig 5.6 values in blynk app

CHAPTER 6

CONCLUSION

6.1 CONCLUSION

The integrated IoT-based online monitoring approach using smart logistics can address the critical needs of reducing food waste, increasing transportation efficiency, and tracking food contamination.
The experimentation was a success and from this experimentation we can understand how to measure the temperature and humidity, LDR values, and the ethane gas data given off during the spoilage of food, especially in “BANANAS”. Ethylene is a natural gas given off by fruit that helps in ripening.
Too high a temperature destroys enzymes, and too low a temperature can break down the cell walls of the fruit so the contents mix and the fruit oxidizes, browns and softens abnormally. The optimum temperature and humidity conditions for ripening are 62 to 68 degrees Fahrenheit and 90 to 95 percent relative humidity.
The food is monitored at different conditions and evaluated on a daily basis. Many such devices can be installed at a location for better monitoring and quality control.
So in this IoT project, a Food Monitoring device is implemented to monitor the real-time values of the temperature, humidity, and methane gas, which are prime measures in food quality, will be measured and displayed. If the temperature is at the critical value, user will receive a notification through an app

6.2 Future Enhancements

Food poisoning has been the source of innumerable diseases, to reduce and avoid illness we use biosensors and electrical sensors to determine the freshness of household food items like diary, fruits, to expend the device for more items by adding new sensors and by using existing sensors. User can also get output in the form of notification and also through the lcd display.

CHAPTER 7

REFERENCES

References:

[1] Rajesh Kumar Kaushal, Harini. T, Pavithra Lency.D, Sandhya.T, Soniya.P, “IoT Based Smart Food Monitoring System”, International Journal Of Current Engineering And Scientific Research, Vol 6, Issue 6, 2019, pp. 73-76.
[2] https://en.wikipedia.org/wiki/Food_loss_and_waste
[3] https://learnenglish.britishcouncil.org/skills/reading/advanced-c1/a- threat-to-bananas.
[4]Available: https://www.nationalgeographic.com/environment/article/food-journeys-graphic
[5] Soumya T K et. al , “Implementation of IoT based Smart Warehouse Monitoring System”, International Journal of Engineering Research & Technology,Vol6,Issue5,2018,pp.1-4.
[6] H. L. Yin, Y. M. Wang, "An Effective Approach for the Design of Safety Fresh Food Supply Chain Networks with Quality Competition”, IEEE International Conference on Information and Automation, 2017, pp.921-924.
[7] F. Kamoun, O. Alfandi and S. Miniaoui, "An RFID Solution for the Monitoring of Storage Time and Localization of Perishable Food in a Distribution Center”, Global Summit on Computer & Information Technology,2015,pp.1-6.
[8] Available;http://www.eatbydate.com/fruits/fresh/bananas-shelf-life-expiration-date/
[9] https://docs.blynk.cc/
➢