

INDUSTRIAL TRAINING REPORT

TITLE: AUTOMATIC INDUSTRIAL MONITORING SYSTEM (AIMS)

A project report submitted in partial fulfilment of requirements for the award of the Degree of
B.Sc. Engineering in Computer Science & Engineering

SUBMITTED BY:

Name: Ashikur Rahman
University ID: 17104039

Name: Rayhanuzzaman
University ID: 17104049

Name: Jahid Hasan
University ID: 17104056

SUBMITTED TO:

Department of Computer Science & Engineering

**BANGLADESH ARMY UNIVERSITY OF ENGINEERING & TECHNOLOGY (BAUET)
QADIRABAD CANTONMENT, NATORE-6431**

DECLARATION

I hereby declare that the industrial Training Report entitled AUTOMATIC INDUSTRIAL MONITORING SYSTEM (AIMS) is an authentic record of my own work as requirements of 01-months Industrial Training during the period from February 14, 2020 to March 12, 2020 for the award of degree of B.Sc. Engineering (Computer Science & Engineering), **Bangladesh Army University of Engineering & Technology** , under the guidance of **CEO, MR. K. Nahid Parvez.**

Date: 15th March 2020

Certified that the above statement made by the student is correct the best of our knowledge and belief.

Examined by:

MR. K. Nahid Parvez.

**Head of Department
(Signature and Seal)**

ACKNOWLEDGEMENT

The present project work is the several days of study of the various aspects of the project development. During this the effort in the present study, we have received a great amount of help from our department of CSE, Bangladesh Army University of Engineering and Technology, which we wish to acknowledge and thank from depth of our hearts.

I am thankful to **Mohammad Golam Sarwar Bhuyan** sir, for permitting and encouraging me in doing this project.

I express my sincere thanks to our Academic Dean **Dr. Mirza A.F.M. Rashidul Hassan** mam, for encouragement and suggestions to the successful completion of my project.

My sincere thanks to **Dr. Mirza A.F.M. Rashidul Hassan** sir, Head of the Department in CSE whose motivation and constant encouragement has led to pursue a project in the field of software development.

I am very much obliged and thankful to **CEO, MR. K. Nahid Parvez** sir is my coordinator for providing this opportunity and constant encouragement given by him during the course.

My parents have put myself ahead of themselves. Because of their hard work and dedication, I have had opportunities beyond my wildest dreams. My heartfelt thanks to them for giving me all I ever needed to be successful student and individual.

Finally, I express my thanks to all my other professors, classmates, friends, neighbors and my family members who helped me for the completion of my project and without infinite love and patience this would never been possible.

About Organization/Company

Innovative Soft (An incorporated in Bangladesh bearing Registration No. (0131536) Dated 16th January 2007 having, its registered office at 141/4, Lake Circus Kalabagan, Dhanmondi, Dhaka-1205 (hereinafter referred to as the “Company”). Innovative soft is the registered member of CPTU (Central Procurement Technical Unit) Bangladesh government. Also become a member of Bangladesh Association of Software and Information services (BASIS)

Innovative Soft was a dream now a day it is a largest software and IT company in Bangladesh. Our CEO, MR. K. Nahid Parvez being a CSE graduate, to live this dream and started his own IT company, Innovative Soft was started since 2006.

Innovative Soft is a leading solution provider for internet based applications. Established in 2006, The Company has been promoted by some highly experienced professionals dedicated to provide total IT solutions under one roof. It possesses not only the latest technology implement but also the most knowledgeable and experience hands to offer most user friendly customized solutions.

Office Community

Innovative Soft was a dream now a day it is a largest software and IT company in Bangladesh. Our CEO, MR. K. Nahid Parvez being a CSE graduate, to live this dream and started his own IT company, Innovative Soft was started since 2006. We provide the Open Door culture inside Innovative Soft, we share each other mistake and experience to find the appropriate solution. Innovative soft is an open platform to build yourself and archive success. Our goal is to give the best services you need to create a positive impact in the world through technology

Innovative Soft has been working as a development associated in building up the homeland for different company. Now at the outset of the twenty-first century Innovative Soft is ever more prepared for heeding the challenging demands of the new millennium with highly qualified management team, modern management techniques and software developer. Innovative Soft is focused on total customer satisfaction, expended and repeat business from satisfied customers.

TABLE OF CONTENTS:

| | | |
|---------|---|----------|
| 1. | CHAPTER-I: INTRODUCTION | page 1-4 |
| 1.1 | OVERVIEW | page 7 |
| 1.2 | IOT TECHNOLOGY & INDUSTRIAL MONITORING | page 7 |
| 1.2.1 | IOT: CONCEPT & DEFINITION | page 7 |
| 1.2.2 | IOT ENABLING TECHNOLOGIES | page 8 |
| 1.2.3 | BENEFITS OF IOT IN INDUSTRIAL MONITORING | page 9 |
| 2. | CHAPTER-II: OVERVIEW OF THE PROJECT | page 10 |
| 2.1 | DEFINITION IOT BASED AUTOMATIC INDUSTRIAL MONITORING SYSTEM | page 10 |
| 2.2 | COMPONENTS & MODULES | page 10 |
| 2.2.1 | ARDUINO UNO | page 10 |
| 2.2.2 | WIFI MODULE -ESP 8266 | page 11 |
| 2.2.3 | ARDUINO ETHERNET SHIELD | page 11 |
| 2.2.4 | NODE MCU 12E | page 12 |
| 2.2.5 | 4 CHANNEL 5V RELAY CONTROL | page 12 |
| | 2.2.6 SENSORS | page 13 |
| 2.2.6.1 | TEMPERATURE AND MOISTURE SENSOR –DHT22 | page 13 |
| 2.2.6.2 | TEMPERATURE SENSOR LM35 | page 13 |
| 2.2.6.2 | WATER LEVEL SENSOR | page 14 |
| 2.2.7 | MULTIMETER | page 14 |
| 2.2.8 | USB CABLES AND JUMPER WIRES | page 15 |
| 3. | CHAPTER-III: DESCRIPTION OF PROJECT WITH NUMBER OF PHASES | page 16 |
| 3.1 | ER DIAGRAM | page 16 |
| 3.2 | CLASS DIAGRAM | page 17 |
| 3.3 | USE CASE DIAGRAM | page 18 |
| 3.4 | PHASE I, PHASE II, PHASE III | page 19 |
| 4. | CHAPTER-IV: CONCLUSION & FUTURE SCOPE | page 33 |
| 4.1 | CONCLUSION | page 33 |
| 4.2 | FUTURE SCOPE | page 33 |
| | REFERENCE & SOURCES | page 34 |

ABSTRACT

Internet of Things (IoT) technology has brought revolution to each and every field of common man's life by making everything smart and intelligent. IoT refers to a network of things which make a self-configuring network. The AUTOMATIC INDUSTRIAL MONITORING SYSTEM IoT based devices are day by day turning the face of industrial production by not only enhancing it but also making it cost-effective and reducing risk, fatal accidents and malfunctions. The aim of this project is to ensure the full automation of the production facility and gain its control and overall monitoring from any part of the world. With the help of this smartphone application we can easily get the updates and condition of the facility along with the status of different machineries which have been connected with this software and the system. The automation readily reduces the flaws and errors in monitoring the devices.

CHAPTER I: INTRODUCTION

1.1 OVERVIEW

The objectives of this report is to propose IoT based Automatic Industrial Monitoring System which will enable user to have real time reading of temperature from different machinery and equipment, reading of humidity from the production facility, water level readings from tanks and boilers and dual mode monitoring system which ensures both manual and automatic control over the electrical devices and equipment. The structure of the report is as follows: chapter I will cover over of overview of IoT Technology and Industrial monitoring and definition, IOT enabling technologies, IOT application in Industry, benefits of IOT in Industry. Chapter II will cover definition of IOT based Industrial Monitoring System, the components and modules used in it and working principal of it. Chapter III will cover algorithm and use case diagram of the overall process carried out in the system and its final graphical output chapter IV consist of conclusion, future scope and references.

1.2 IOT TECHNOLOGY AND INDUSTRIAL MONITORING

1.2.1 IOT: CONCEPT AND DEFINITION

Internet of things IOT consists of two words Internet and Things. The term things in IOT refers to various IOT devices having unique identities and have capabilities to perform remote sensing, actuating and live monitoring of certain sort of data. IOT devices are also enable to have live exchange of data with other connected devices and application either directly or indirectly, or collected data from other devices and process the data and send the data to various servers. The other term internet is define as Global communication Network connecting Trillions of computers across the planets enabling sharing of information .Thus the IOT can be define as : “A dynamic Global Network Infrastructure with self -configuring capabilities based on standard and inter operable communication to protocol where physical and virtual things have identities, physical attributes ,and virtual personalities and use intelligent interfaces and are seamlessly integrated into the information network ,often communicate data associated with user and their environment.”

An ideal IoT device consists of various interfaces for making connectivity to other devices which can either be wired or wireless.

Any IoT based device consists of following components:

- I/O interface for Sensors.
- Interface for connecting to Internet.
- Interface for Memory and Storage.
- Interface for Audio/Video.

IoT devices can be of various forms like wearable sensors, smart watches, IoT smart home monitoring, IoT intelligent transport systems, IoT smart health devices etc.

1.2.2 IOT ENABLING TECHNOLOGIES

Internet of Things has a strong backbone of various enabling technologies- Wireless Sensor Networks, Cloud Computing, Big Data, Embedded Systems, Security Protocols and Architectures, Protocols enabling communication, web services, Internet and Search Engines.

Wireless Sensor Network (WSN): It consists of various sensors/nodes which are integrated together to monitor various sorts of data.

Cloud Computing: Cloud Computing also known as on-demand computing is a type of Internet based computing which provides shared processing resources and data to computers and other devices on demand. It can be in various forms like IaaS, PaaS, SaaS, DaaS etc.

Big Data Analytics: Big data analytics is the process of examining large data sets containing various forms of data types—i.e. Big Data – to uncover hidden patterns, unknown correlations, market trends, customer preferences and other useful business information.

Communication Protocols: They form the backbone of IoT systems to enable connectivity and coupling to applications and these protocols facilitate exchange of data over the network as these protocols enable data exchange formats, data encoding and addressing.

Embedded Systems: It is a sort of computer system which consists of both hardware and software to perform specific tasks. It includes microprocessor/microcontroller, RAM/ROM, networking components, I/O units and storage devices.

1. 2. 3 BENEFITS OF IOT IN INDUSTRIAL MONITORING

The following are the benefits of IoT in **Industrial Monitoring**:

1. IoT enables easy collection and management of tons of data collected from sensors and with integration of firebase data can be accessed live from anywhere and everywhere enabling live monitoring and end to end connectivity among all the parties concerned.
2. IoT is regarded as key component for AUTOMATIC INDUSTRIAL MONITORING as with accurate sensors and smart equipment's, as a result it is possible to reduce the number of machine operators in certain industries. Automation mode will follow particular conditions to activate or deactivate the devices or machines with help of timers, sensors and logics.
3. With IoT industrial monitoring system cost can be reduced to a remarkable level which will in turn increase profitability and sustainability.
4. With IoT, efficiency level can also be ensured for certain machines and devices which are very sensitive and reduce their decay rate.
5. With IoT, various factors would also lead to the protection of industrial environment and reduce accidents.

CHAPTER II: OVERVIEW OF THE PROJECT

2.1 DEFINITION IOT BASED AUTOMATIC INDUSTRIAL MONITORING SYSTEM

IoT based AUTOMATIC INDUSTRIAL MONITORING SYSTEM is regarded as IoT gadget focusing on Live Monitoring of Industry in terms of Temperature, Humidity and other types depending on the sensors integrated with it. The system provides the concept of “Plug & Sense” in which user can directly implement manual or automatic control as per condition by putting the System in the industry and getting the readings from the connected devices and machineries. The data generated via sensors can be easily viewed by the user anywhere remotely via Smart Phone connected with Internet and it is also possible to control those devices from any part of the world

2.2 COMPONENTS AND MODULES

In this section, various components and Modules being used for IoT based AUTOMATIC INDUSTRIAL MONITORING SYSTEM development is discussed:

2.2.1 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 crystal oscillator, a USB connection ,a power jack, an ICSP header , and a reset button.



Fig-2.1: Arduino UNO

2.2.2 WIFI MODULE-ESP 8266

ESP8266 Wi-Fi Module is SOC with TCP/IP protocol stack integrated which facilitates any microcontroller to access Wi-Fi network. ESP8266 module is cost effective module and supports APSD for VOIP Applications and Bluetooth co-existence interfaces. Technical Specifications: 802.11b/g/n; Wi-Fi Direct, 1MB Flash Memory, SDIO 1.1/2.0, SPI, UART, Standby Power Consumption of <1.0mW.

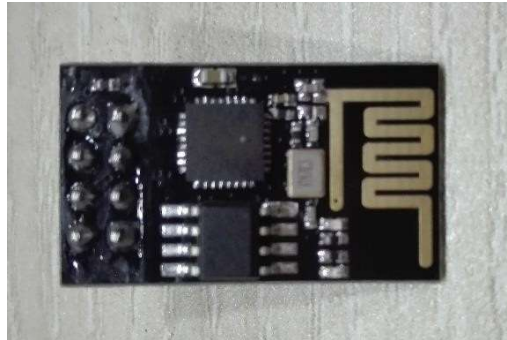


Fig-2.2: ESP8266.

2.2.3 ARDUINO ETHERNET SHIELD

The **Arduino Ethernet Shield** allows an **Arduino** Board to connect to the internet. It is based on the (Wiznet W5500 **Ethernet** chip). The Wiznet W5500 provides a network (IP) stack capable of both TCP and UDP. ... The **Ethernet Shield 2** connects to an **Arduino** Board using long wire-wrap headers extending through the **Shield**.



Fig-2.3: Arduino Ethernet Shield.

2.2.4 Node MCU E12

NodeMCU is an open source firmware for which open source prototyping board designs are available. The firmware uses the **Lua** scripting language. It uses many open source projects, such as **lua-cjson** and **SPIFFS**. It is a development board which runs on the **ESP8266** with the Espressif Non-OS SDK, and hardware based on the ESP-12 module. The device **features** 4MB of flash memory, 80MHz of system clock, around 50k of usable RAM and an on chip Wifi Transceiver.



Fig-2.4: Node MCU E12.

2.2.5 4 CHANNEL 5V RELAY CONTROL

The **4 Channel Relay Module** is a convenient board which can be used to **control** high voltage, high current load such as motor, solenoid valves, lamps and AC load. It is designed to interface with microcontroller such as Arduino, PIC and etc. The **relays** terminal (COM, NO and NC) is being brought out with screw terminal.



Fig-2.5: 4 Channel Relay Module.

2.2.6 SENSORS

2.2.6.1 TEMPERATURE AND MOISTURE SENSOR –DHT22

The *DHT22* is a basic, low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital signal on the data pin (no analog input pins needed). It's fairly simple to use, but requires careful timing to grab data.

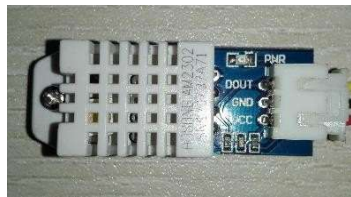


Fig-2.6: Temperature & Humidity Sensor DHT22

2.2.6.2 TEMPERATURE SENSOR LM35

LM35 is a temperature sensor that outputs an analog signal which is proportional to the instantaneous temperature. The output voltage can easily be interpreted to obtain a temperature reading in Celsius. ... Many low-end products take advantage of low cost, greater accuracy and used **LM35** in their products.



Fig-2.7: Temperature Sensor LM35.

2.2.6.3 ARDUINO WATER LEVEL SENSOR

This **Arduino water level indicator** uses an ultrasonic **sensor** or Ping **sensor** to determine the **level** of **water** in the tank. The Ping **sensor** measures distance using sonar. The following project can be interfaced to an **Arduino** board if you have one or directly to an ATmega 328 microcontroller on a breadboard.



Fig-2.8: Water level sensor.

2.2.7 MULTIMETER

A multimeter or a multimeter, also known as a VOM (volt-ohm-milliammeter), is an electronic measuring instrument that combines several measurement functions in one unit. A typical multimeter can measure voltage, current, and resistance. Analog multimeters use a microammeter with a moving pointer to display readings.



Fig-2.9: Multi-meter.

2.2.8 USB CABLES AND JUMPER WIRES

USB Cable Overview The term USB stands for "Universal Serial Bus". USB cable assemblies are some of the most popular cable types available, used mostly to connect computers to peripheral devices such as cameras, camcorders, printers, scanners, and more. A jump wire (also known as jumper wire, or jumper) is an electrical wire, or group of them in a cable, with a connector or pin at each end (or sometimes without them – simply "tinned"), which is normally used to interconnect the components of a breadboard or other prototype or test circuit, internally or with other equipment or components, without soldering.



Fig-2.10: USB cable.

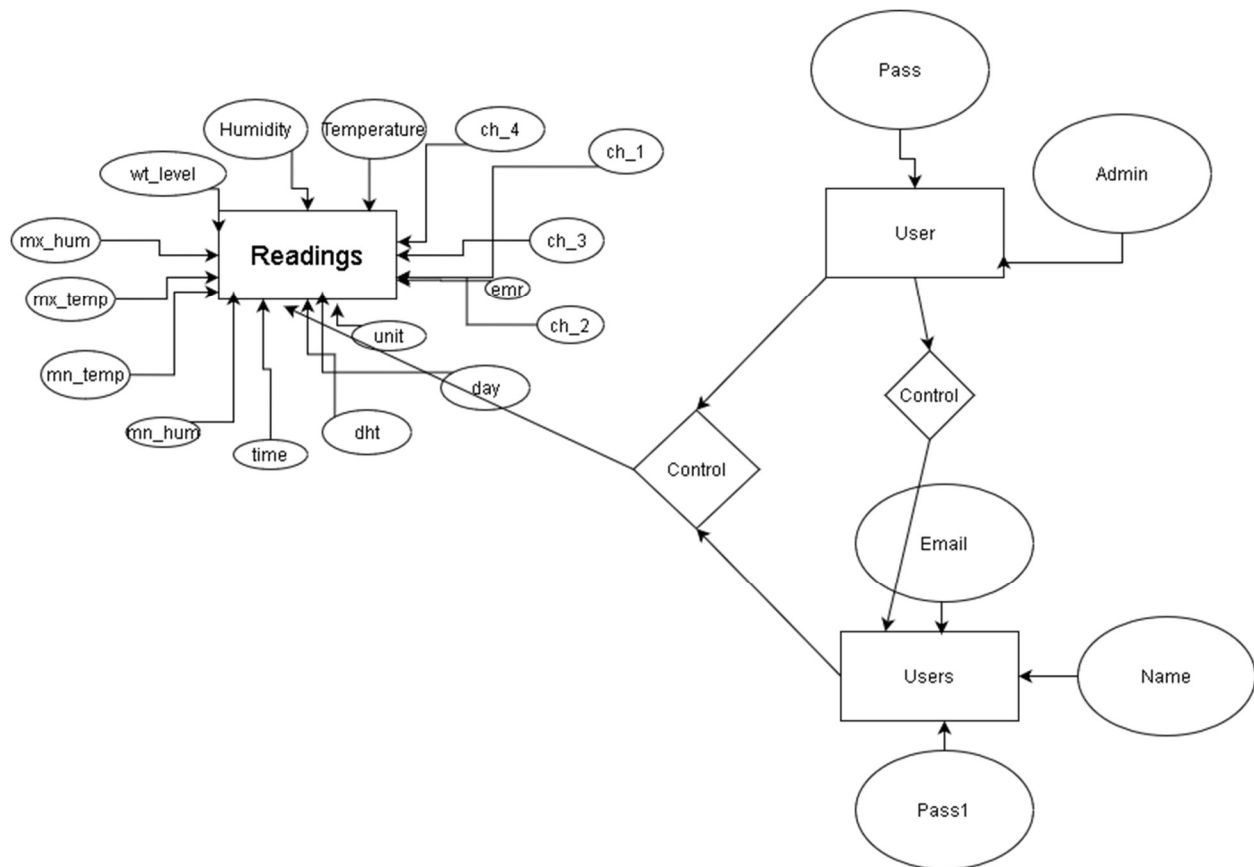


Fig-2.11: Jumper Wires

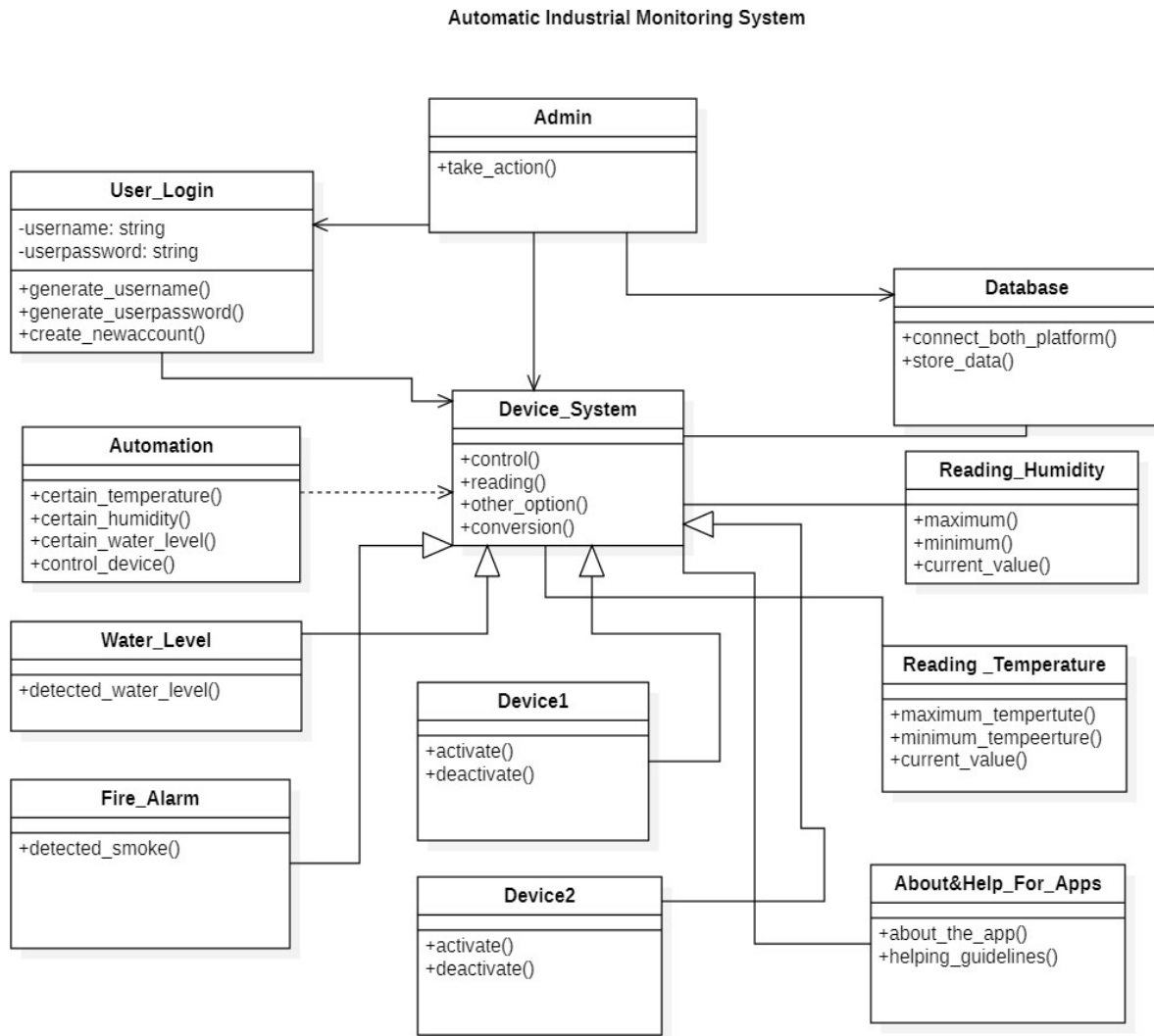
CHAPTER-III:

E-R DIAGRAM, CLASS DIAGRAM, USE CASE DIAGRAM AND PHASES

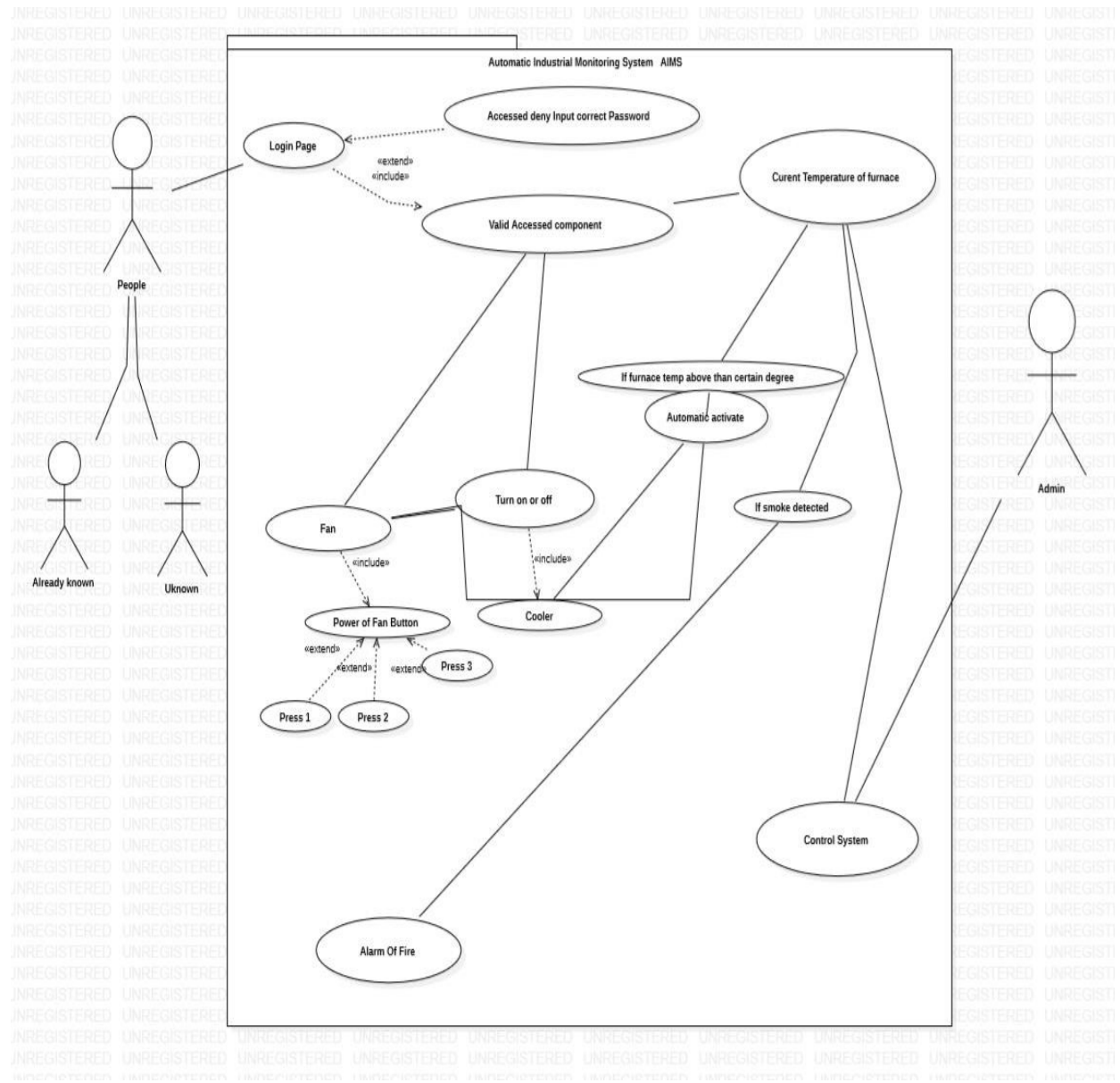
3.1 CLASS DIAGRAM:



3.2 CLASS DIAGRAM:



3.3 USE CASE DIAGRAM:



3.4 PHASES:

PHASE I

MONITORING TEMPERATURE AND HUMIDITY WITH CONTROLLING RELAY BOARD WITH ARDUINO UNO OVER ETHERNET

It is required to monitor the temperature and humidity of the facility with the help of sensors via Arduino UNO over Ethernet connection. It is also required to take control over the connected devices with the system respectively.

APPARATUS:

- i. DHT22.
- ii. Arduino UNO.
- iii. Ethernet Shield.
- iv. 5V 4 Channel Relay Module.
- v. Breadboard.
- vi. Multi-meter.
- vii. USB cable, LAN cable, Jumper wires.

NECESSARY DIAGRAMS:

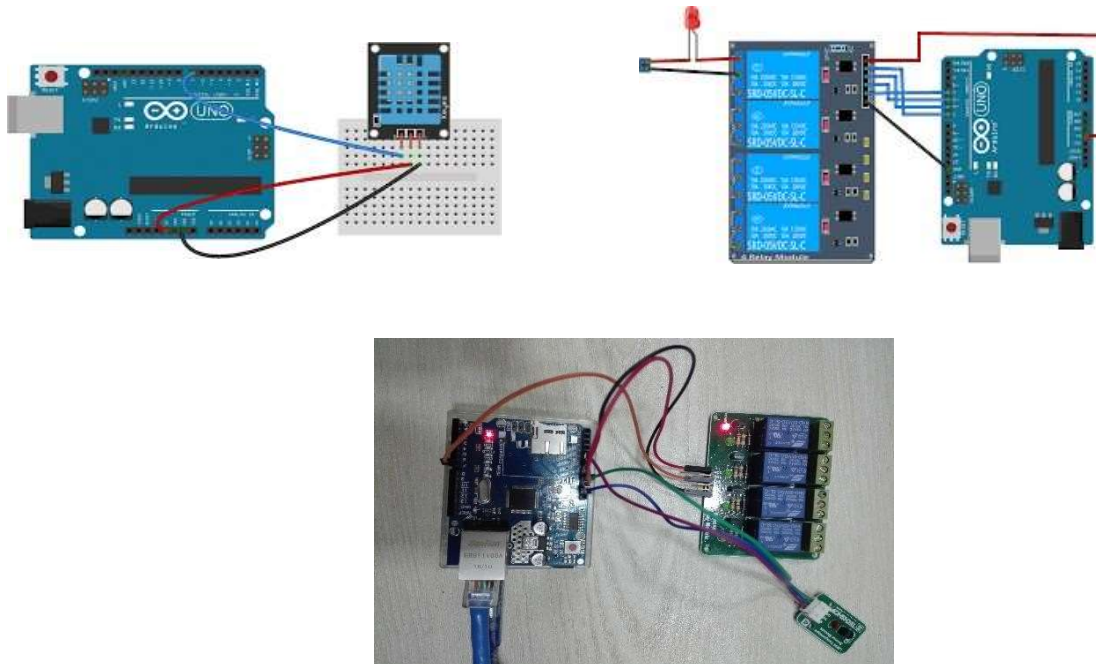
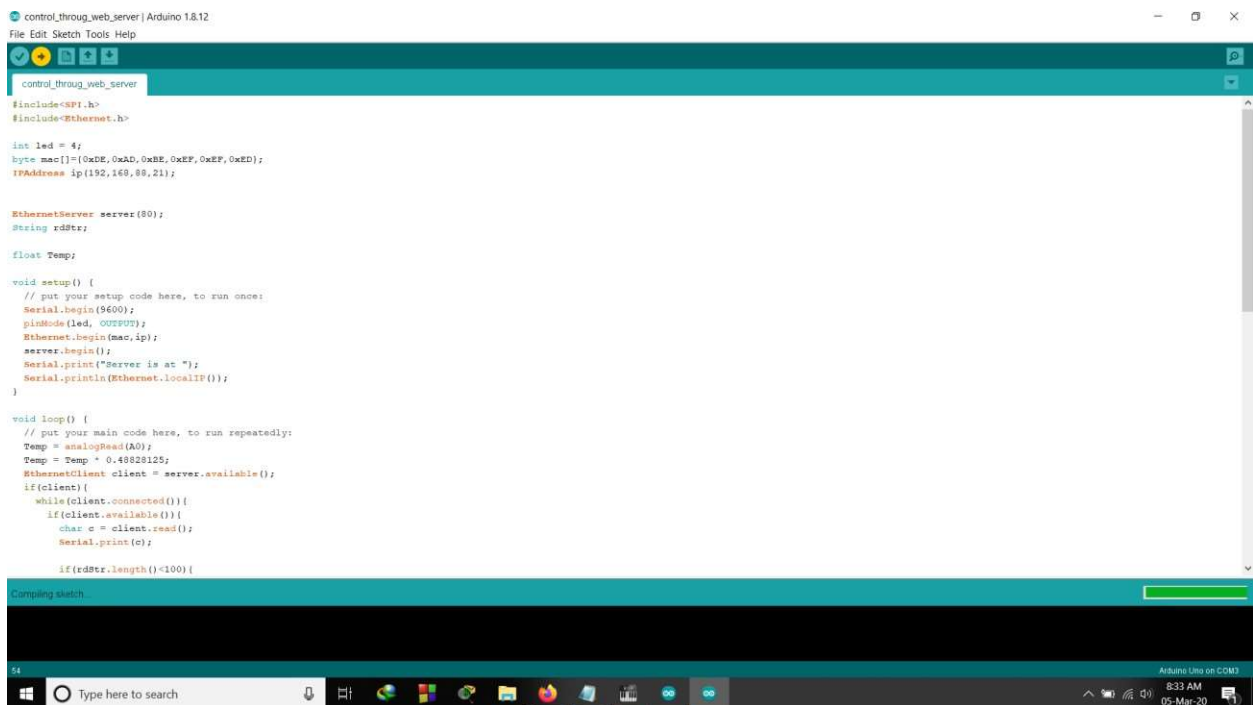
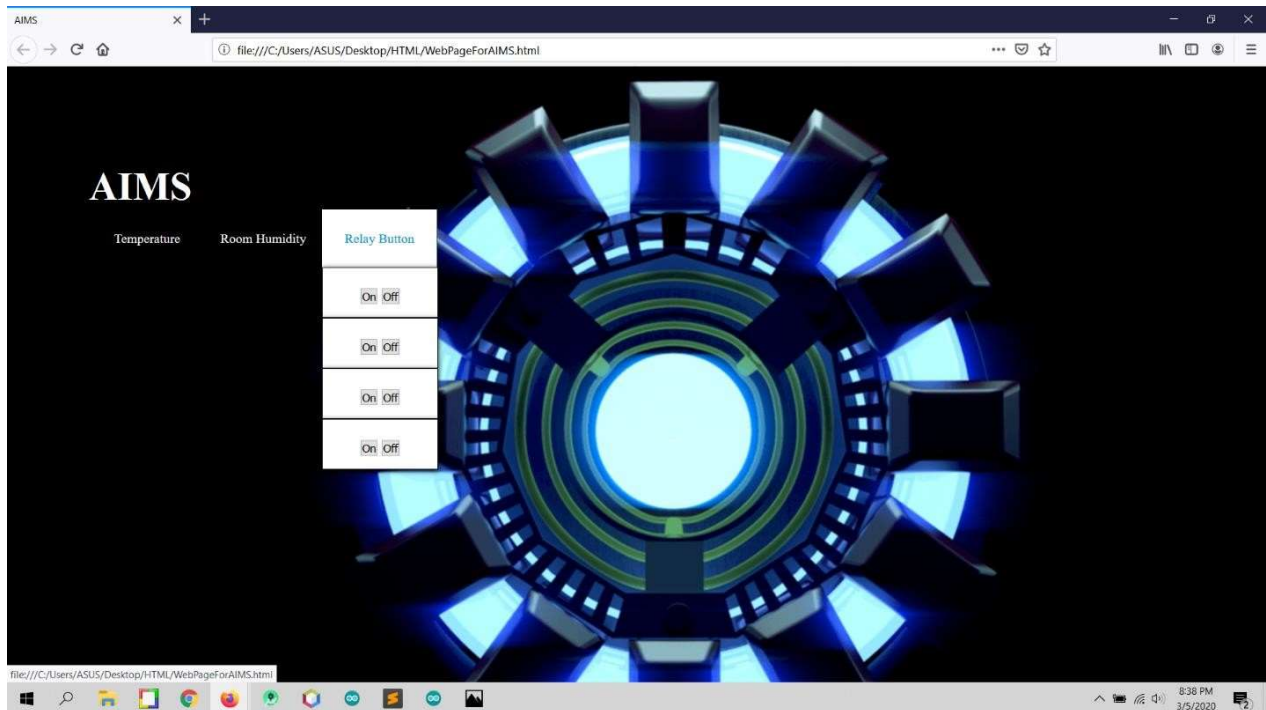


Fig-3.1: Connection of Arduino UNO with Relay and Sensor.

SNAPSHOTS:



PHASE II

REMOTELY MONITOR THE TEMPERATURE AND HUMIDITY WITH CONTROLLING RELAY BOARD BY ARDUINO UNO

Now we have to consider our equipment to connect with an internet connection via router with help of ESP8266 WIFI module for remotely monitoring the sensors reading and control the machineries or devices inside the production facility.

APPARATUS:

- i. DHT22.
- ii. Arduino UNO.
- iii. ESP8266 WIFI Module.
- iv. 5V 4 Channel Relay Module.
- v. Breadboard.
- vi. Multi-meter.
- vii. USB cable, Jumper wires.

NECESSARY DIAGRAMS:

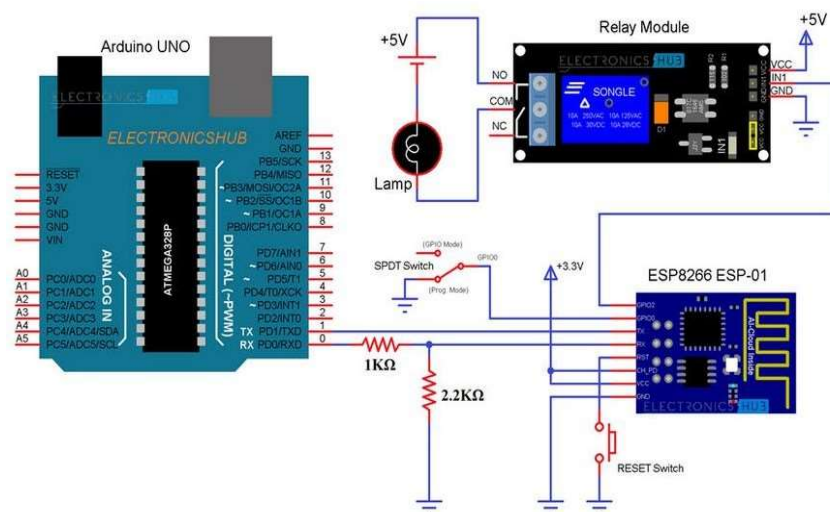
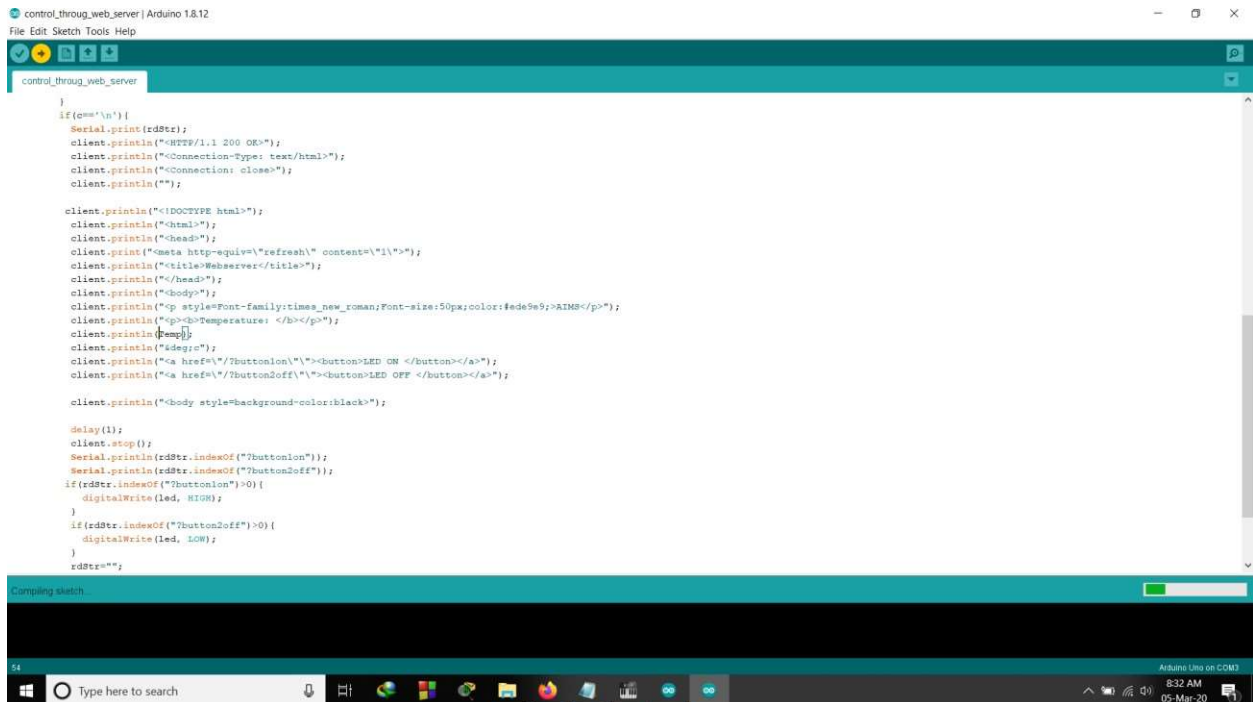
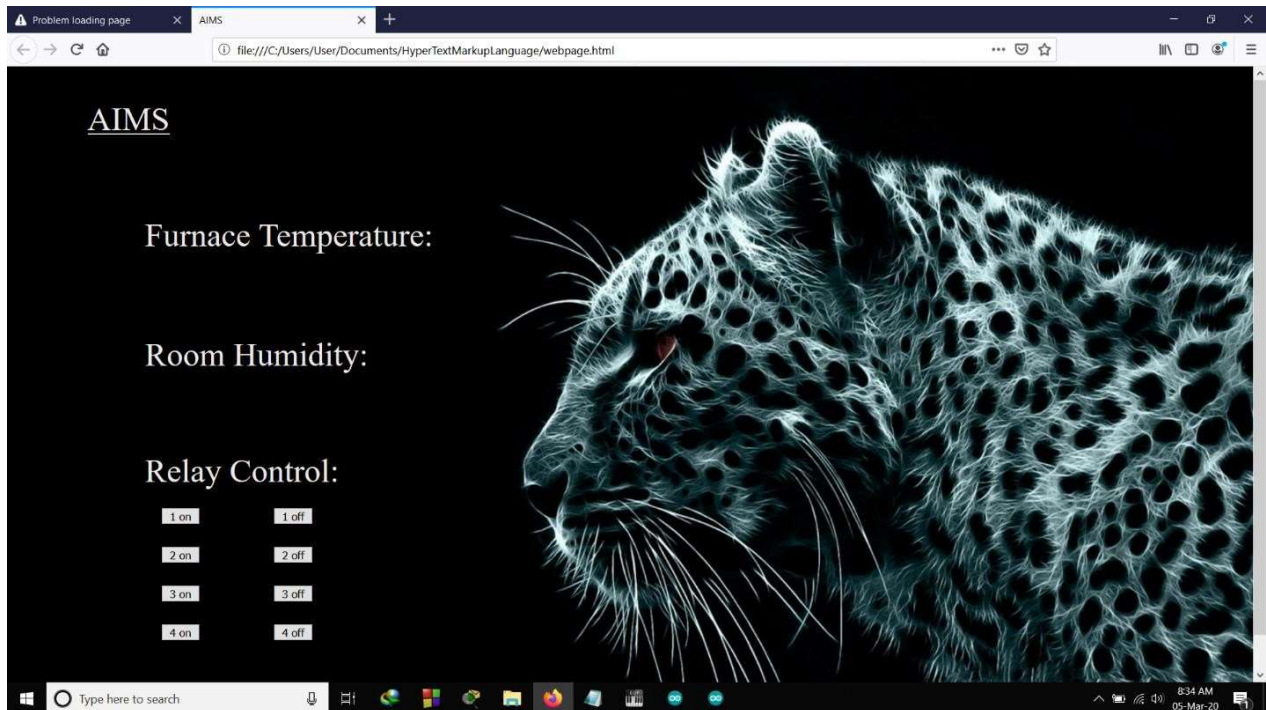


Fig-3.2: Arduino UNO connected with ESP8266.

SNAPSHOTS:



PHASE III

MONITOR THE TEMPERATURE AND HUMIDITY WITH CONTROLLING RELAY BOARD BY ANDROID APPLICATION

An Android Application is developed for viewing the information and controlling the devices over internet connection. The user can easily monitor and maintain the machineries using the software. Moreover, there is an Automatic system which will initiate at user defined temperature or condition to activate or deactivate any important device instantly. Both Manual and Automatic mode are operational and reliable.

APPARATUS:

- i. DHT22.
- ii. Arduino UNO.
- iii. Android device.
- iv. ESP8266 WIFI Module.
- v. 5V 4 Channel Relay Module.
- vi. Breadboard.
- vii. Multi-Meter.
- viii. USB Cable, Jumper Wires.
- ix. Liquid Level Sensor.

NECESSARY DIAGRAMS:

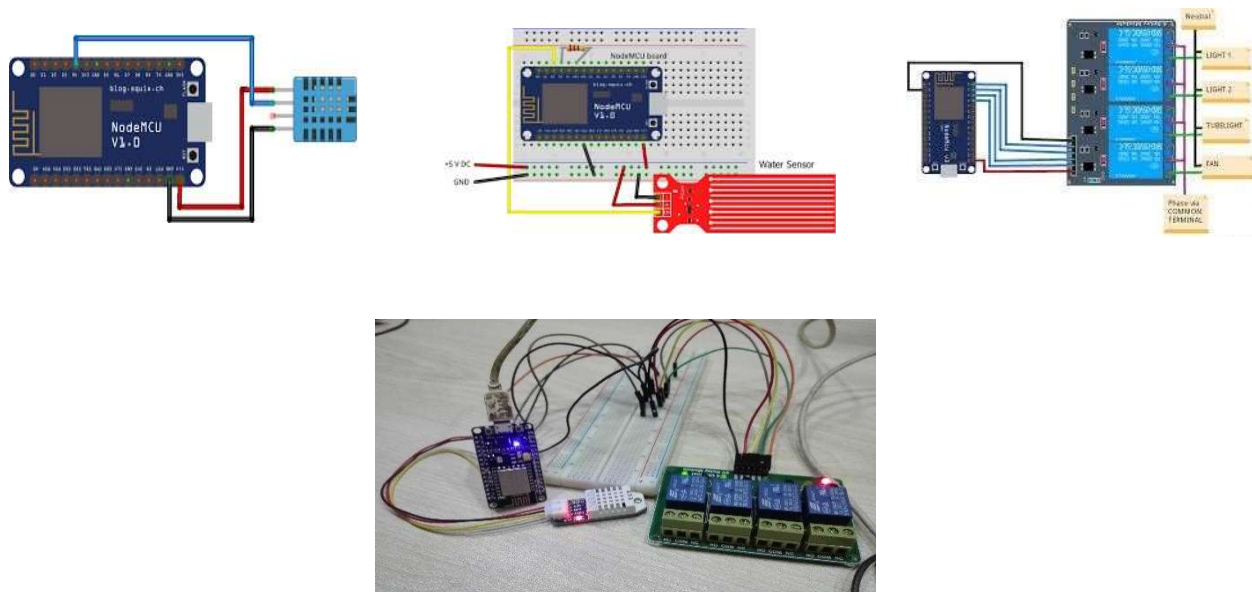


Fig-3.3: NodeMCU connected with DHT22 and 5V 4 Channel Relay board.

PROCEDURE:

1. First of all, we established the circuit by connecting all the apparatus properly. We ensured the input voltage and amperage of each and every equipment with the help of multi-meter.
2. The NodeMCU requires 3.3V 1amp current to activate. The sensors require 5V 1amp current to activate and the Relay board requires 5V to activate itself.
3. We could manage the voltage and current supply as per required for each apparatus and prepared them for functioning properly.
4. The NodeMCU contains 4MB of space where we can upload our codes and commands for operating the whole equipment. We have established some logics and written codes using the Arduino IDE and then uploaded to the NodeMCU via USB cable from Computer.
5. Now it is time for establishing a connection between NodeMCU with the Firebase. We have set the mac address, SSID, Static IP, Firebase secret within our code and then we set up the Firebase server following the directions of the site. NodeMCU could easily pass the data to the Firebase and also accepted the data passed through Firebase.
6. Now it is time for establishing a connection between our Android application and Firebase. We have entered the serial number of our software in Firebase. Our smart-phone could easily access the data of Firebase and also send data to it.

Software Requirements:

| Name of the component | Specification |
|-----------------------|-----------------------------------|
| Operating System | Windows 10 |
| IDE | Android Studio, Arduino IDE |
| Database | Firebase Server |
| Language | Java, XML, Arduino Uno, HTML, CSS |
| Browser | Chrome, Firefox |

DEVELOPMENT OF ANDROID APPLICATION:

Login Page:

The login page gives the option for Administration and Registration for the purpose of system security. There are textboxes for taking the ID and Password and finally log inside the Application.

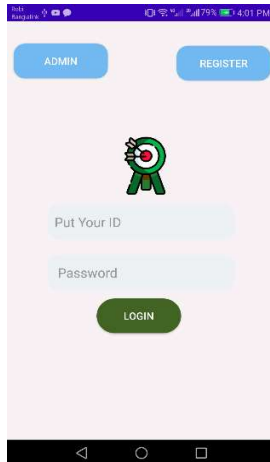


Fig-3.4 : Login Page.

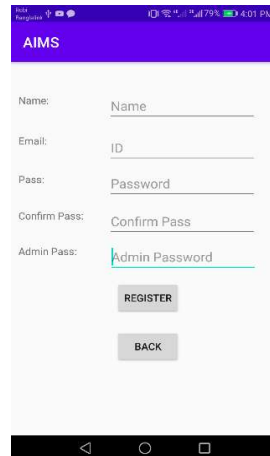


Fig-3.5: Admin Page.

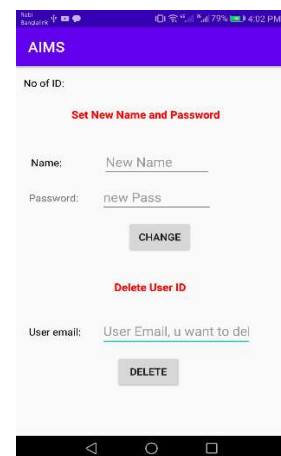


Fig3.6: Registration Page.

Main page:

The main page is displaying the readings of temperature, humidity and also the water level. It also shows the readings and status of different devices connected to the system. There are several toggle buttons to control the devices manually.



Fig-3.7: Main Page.

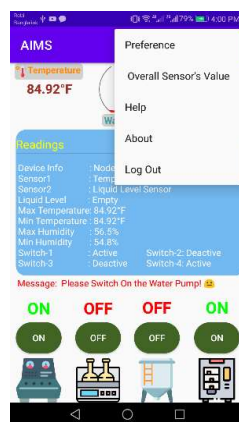


Fig-3.8: Menu Bar.

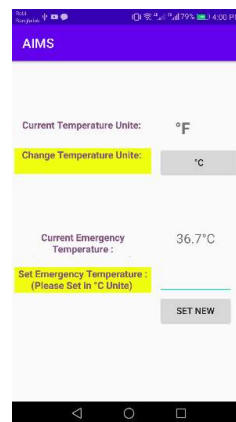


Fig-3.9: Preferences.



Fig-3.10: Overall values.

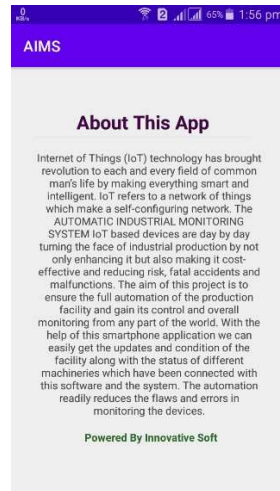


Fig-3.11: About the Application.

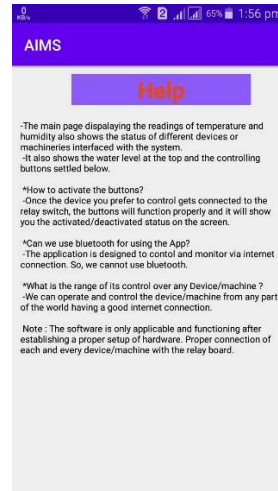


Fig3.12: Helps and Instruction.

There are some other features which includes preferences, overall sensor value and help options.

Android Source Codes Snapshot:

```
myRef = FirebaseDatabase.getInstance().getReference().child("Readings");
try {
    myRef.addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {
            // This method is called once with the initial value and again
            // whenever data at this location is updated.
            tm = dataSnapshot.child("temperature").getValue(String.class).toString();
            hm = dataSnapshot.child("humidity").getValue(String.class).toString();
            ch1 = dataSnapshot.child("ch_1").getValue(String.class).toString();
            ch2 = dataSnapshot.child("ch_2").getValue(String.class).toString();
            ch3 = dataSnapshot.child("ch_3").getValue(String.class).toString();
            ch4 = dataSnapshot.child("ch_4").getValue(String.class).toString();
            day = dataSnapshot.child("day").getValue(String.class).toString();
            hr = dataSnapshot.child("time").getValue(String.class).toString();
            wt = dataSnapshot.child("w_level").getValue(String.class).toString();
            enr_gncy = dataSnapshot.child("enr").getValue(String.class).toString();
            unt = dataSnapshot.child("unit").getValue(String.class).toString();

            time = " Device Date : "+day+" Device Time: "+hr;
            wt_tank = Integer.parseInt(wt);
            //Log.d(TAG, "Value is: " + value);
            SetTemperature_Humidity();
            getNewValueData();
            setWaterTankValue(wt_tank);
            setDetailsBox(wt_tank);
            setMessage();
            setToggleButton();
        }
        @Override
        public void onCancelled(DatabaseError error) {
            // Failed to read value
            flag1++;
        }
    });
} catch (Exception e) {
    Toast.makeText(context, this, "Database Connection Error", Toast.LENGTH_SHORT).show();
}
```

Fig-3.13: Variables and function calling

```
public void SetTemperature_Humidity() {
    try {
        double cel = Double.parseDouble(tm);
        farhenheit = (cel * (9.00 / 5.00)) + 32;
        DecimalFormat dec = new DecimalFormat("###.00");
        kelvin = cel + 273.15;
        temp = Double.parseDouble(tm);
        if (unt.equals("K"))
            temperature.setText(dec.format(kelvin) + unt);
        else if (unt.equals("F"))
            temperature.setText(dec.format(farhenheit) + unt);
        else
            temperature.setText(temp + unt);
        humidity.setText(hm + "%");
        Time.setText(time);
    } catch (Exception e) {
        Toast.makeText(getApplicationContext(), "Failed to Set Value, Please Switch on sensor", Toast.LENGTH_SHORT).show();
    }
}
```

Fig-3.14: Functions for temperature and humidity.

```

public void SetDetails(int x){
    float = Double.parseDouble(h);
    if(h<minHum)
        minHum = h;
    if(h>maxHum)
        maxHum = h;
    if (x<700){...}
    else if (x<700 && x<800){...}
    else if (x<800 && x<900){...}
    else if (x<900 && x<1000){...}
    else if (x<1000){...}
    String ch01, ch02, ch03, ch04;
    if(ch01=="1")
        ch01="Action ";
    else
        ch01="Deactive";
    if(ch02=="1")
        ch02="Action ";
    else
        ch02="Deactive";
    if(ch03=="1")
        ch03="Action ";
    else
        ch03="Deactive";
    if(ch04=="1")
        ch04="Action ";
    else
        ch04="Deactive";

    String Details = "Lafale Device Info:      : NodeMCU ESP8266 Sensor1      : Temperature & Humidity Sensor2\n";
    Details += "Liquid Level Sensor1 Liquid Level      : \"Level\" in Max Temperature: \"-\" Del. format(printH);\" in Max Temperature: \"-\" \n";
    Details += "Del. format(printH);\" in Max Humidity      : \"maxHum\" in Max Humidity      : \"minHum\" in\n";
    Details += "Switch-1      : \"ch01\" in Switch-2      : \"ch02\" in Switch-3      : \"ch03\" in Switch-4      : \"ch04\" in\n";
    board.setText(Details);
}

```

Fig-3.15: Functions for detail box.

```

public void getMaxValueData(){
    if (hr=="23:23"){...}
    if(temp>mxtemp)
        mxtemp = temp;
    if(temp<mntemp)
        mntemp = temp;
    if (unt.equals("°K")) {
        printMnT = mntemp + 273.15;
        printMxT = mxtemp + 273.15;
        u="°K";
    }
    else if(unt.equals("°F")) {
        double x1,x2;
        printMnT = (mntemp * (9.00/5.00))+32;
        printMxT = (mxtemp * (9.00/5.00))+32;
        u="°F";
    }
    else {
        printMxT = mxtemp;
        printMnT = mntemp;
        u = "°C";
    }
}

```

Fig-3.16: Functions for max/min temperature and humidity.

```

public void setToggleButton(){
    Double emergency = Double.valueOf(emr_sncy);
    if (temp<emergency){
        t1.setOnCheckedChangeListener((compoundButton, b) -> {
            if (b) {...} else {...}
        });
        t2.setOnCheckedChangeListener((compoundButton, b) -> {
            if (b) {...} else {...}
        });
        t3.setOnCheckedChangeListener((compoundButton, b) -> {
            if (b) {...} else {...}
        });
        t4.setOnCheckedChangeListener((compoundButton, b) -> {
            if (b) {...} else {...}
        });
    }
    else
    {
        if (temp>emergency){
            myRef.child("ch_1").setValue("0");
            myRef.child("ch_2").setValue("0");
            myRef.child("ch_3").setValue("0");
            myRef.child("ch_4").setValue("0");
            off_.setSpan(fcsRed, start: 0, end: 3, Spanned.SPAN_EXCLUSIVE_EXCLUSIVE);
            s4.setText(off_);
            s3.setText(off_);
            s2.setText(off_);
            s1.setText(off_);
        }
    }
}

```

Fig-3.17: Functions for Toggle button.

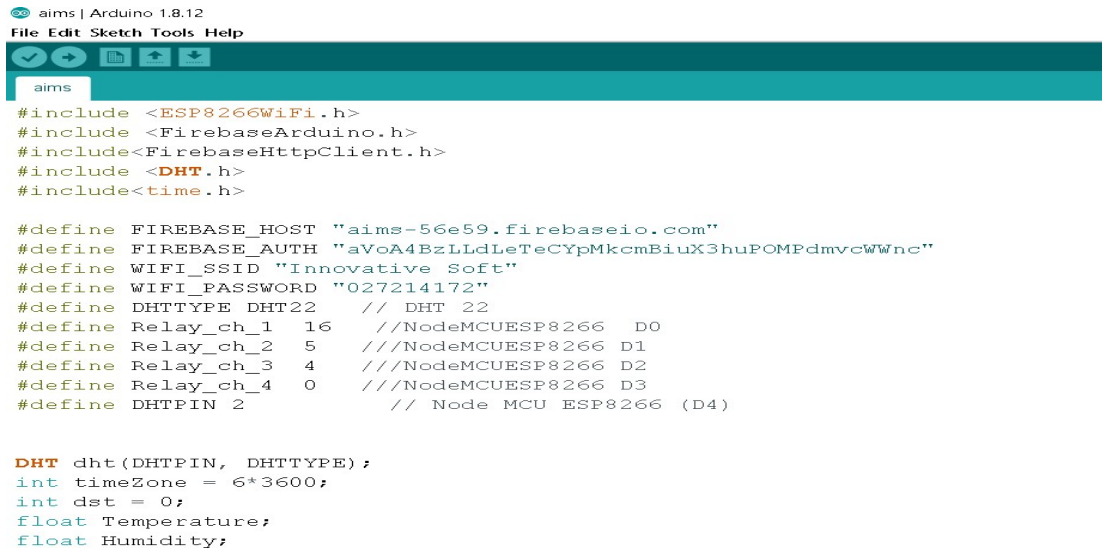
```

public void setWaterTankValue(int x){
    double per = (x-700);
    per=(per/324)*100;
    if (x<=700){
        mWaveloadingView.setProgressValue(0);
    }
    else {
        mWaveloadingView.setProgressValue(((int) per));
    }
    if (per<=0){...}
    else if (per<=10 && per>=0){...}
    else if (per>10 && per<=20){...}
    else if (per>20 && per<=30){...}
    else if (per>30 && per<=40){...}
    else if (per>40 && per<=50){...}
    else if (per>50 && per<=60){...}
    else if (per>60 && per<=70){...}
    else if (per>70 && per<=80){...}
    else if (per>80 && per<=90){...}
    else if (per>90 && per<=100){...}
}

```

Fig-3.18: Function for water level.

Arduino Source Codes Snapshot:



```
aims | Arduino 1.8.12
File Edit Sketch Tools Help

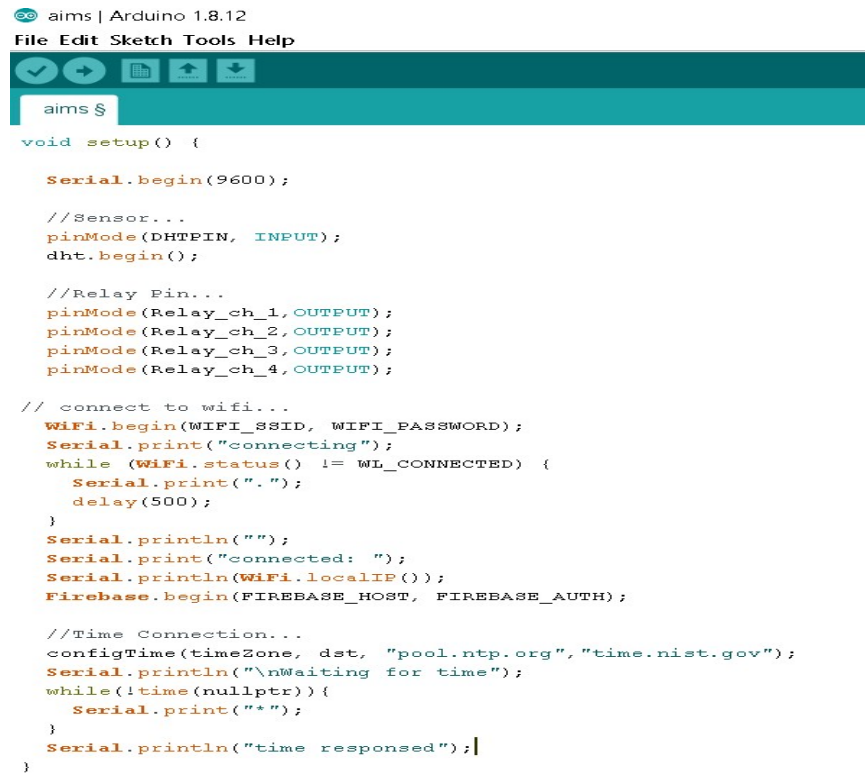
aims

#include <ESP8266WiFi.h>
#include <FirebaseArduino.h>
#include <FirebaseHttpClient.h>
#include <DHT.h>
#include <time.h>

#define FIREBASE_HOST "aims-56e59.firebaseio.com"
#define FIREBASE_AUTH "aVoA4BzLLdLeTeCYpMkcmBiuX3huPOMPdmvcWWnc"
#define WIFI_SSID "Innovative Soft"
#define WIFI_PASSWORD "027214172"
#define DHTTYPE DHT22 // DHT 22
#define Relay_ch_1 16 //NodeMCUESP8266 D0
#define Relay_ch_2 5 //NodeMCUESP8266 D1
#define Relay_ch_3 4 //NodeMCUESP8266 D2
#define Relay_ch_4 0 //NodeMCUESP8266 D3
#define DHTPIN 2 // Node MCU ESP8266 (D4)

DHT dht(DHTPIN, DHTTYPE);
int timeZone = 6*3600;
int dst = 0;
float Temperature;
float Humidity;
```

Fig-3.19: Declaring variables and Functions.



```
aims | Arduino 1.8.12
File Edit Sketch Tools Help

aims $

void setup() {

    Serial.begin(9600);

    //Sensor...
    pinMode(DHTPIN, INPUT);
    dht.begin();

    //Relay Pin...
    pinMode(Relay_ch_1, OUTPUT);
    pinMode(Relay_ch_2, OUTPUT);
    pinMode(Relay_ch_3, OUTPUT);
    pinMode(Relay_ch_4, OUTPUT);

    // connect to wifi...
    WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
    Serial.print("connecting");
    while (WiFi.status() != WL_CONNECTED) {
        Serial.print(".");
        delay(500);
    }
    Serial.println("");
    Serial.print("connected: ");
    Serial.println(WiFi.localIP());
    Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);

    //Time Connection...
    configTime(timeZone, dst, "pool.ntp.org", "time.nist.gov");
    Serial.println("\nWaiting for time");
    while(!time(nullptr)){
        Serial.print("*");
    }
    Serial.println("time responded");
}
```

Fig-3.20: Initiating pin-mode and Firebase.

```

void loop() {
|
//Sensor Data Reading...
Temperature = dht.readTemperature(); // Gets the values of the temperature
Humidity = dht.readHumidity();

// Gets the values of the humidity
if((String)Humidity=="nan" || (String)Temperature=="nan"){
    Firebase.setString("Readings/Temperature", "0.0");
    Firebase.setString("Readings/Humidity", "00.00");
    Firebase.setString("Readings/dht", "0");
}
else
{
    Firebase.setString("Readings/dht", "1");
    Serial.print("Temperature: ");
    Serial.println(Temperature);
    Firebase.setString("Readings/Temperature", (String)Temperature);
    Serial.print("Humidity: ");
    Serial.println(Humidity);
    Firebase.setString("Readings/Humidity", (String)Humidity);
}
}

```

Fig-3.21: Codes for Temperature and Humidity.

```

//Relay Control...
String ch1_sts = Firebase.getString("Readings/ch_1");
String ch2_sts = Firebase.getString("Readings/ch_2");
String ch3_sts = Firebase.getString("Readings/ch_3");
String ch4_sts = Firebase.getString("Readings/ch_4");
//channel 1...
if(ch1_sts=="1")
    digitalWrite(Relay_ch_1,HIGH);
else
    digitalWrite(Relay_ch_1,LOW);
//Channel 2...
if(ch2_sts=="1")
    digitalWrite(Relay_ch_2,HIGH);
else
    digitalWrite(Relay_ch_2,LOW);
//Channel 3...
if(ch3_sts=="1")
    digitalWrite(Relay_ch_3,HIGH);
else
    digitalWrite(Relay_ch_3,LOW);
//Channel 4...
if(ch4_sts=="1")
    digitalWrite(Relay_ch_4,HIGH);
else
    digitalWrite(Relay_ch_4,LOW);

```

Fig-3.22: Codes for Controlling Relay from Firebase.

```

    ///Emergency.....
    if(Temperature>=40.00){
        digitalWrite(Relay_ch_1,LOW);
        digitalWrite(Relay_ch_2,LOW);
        digitalWrite(Relay_ch_3,LOW);
        digitalWrite(Relay_ch_4,LOW);
        Firestore.setString("Readings/ch_1", "0");
        Firestore.setString("Readings/ch_2", "0");
        Firestore.setString("Readings/ch_3", "0");
        Firestore.setString("Readings/ch_4", "0");
    }
    ...

```

Fig-3.23: Conditions for Handling Emergency Situation.

```

//Time Setup...
time_t now = time(nullptr);
struct tm* p_tm = localtime(&now);
int day = (int)p_tm->tm_mday;
int month = (int)p_tm->tm_mon+1;
int year = (int)p_tm->tm_year + 1900;
String date = (String)day + "/" + (String)month + "/" + (String)year;
Firestore.setString("Readings/day", date);
Serial.println(date);
int hr = (int)p_tm->tm_hour;
int mnt = (int)p_tm->tm_min;
String Time_ = (String)hr + ":" + (String)mnt;
Firestore.setString("Readings/time", Time_);
Serial.println(Time_);

//Liquied Level Sensor...
int liquied = analogRead(A0);
Serial.println(liquied);
Firestore.setString("Readings/w_level", (String)liquied);
|
// handle error...
if (Firestore.failed()) {
Serial.print("setting /number failed:");
Serial.println(Firestore.error());
return;
}

```

Fig3.24: Setting time, Sensor, and handling error.

Connecting to Database:

We have connected to the database for keeping the login ID and password of multiple and verified users. The database also keeps the record of the real-time readings of the sensors and the status of the connected devices. The NodeMCU sends the data to the database and our Application can access the database to collect those data and show them on the display. Even the change or input given in the Application can be send to the database and our equipment will work according to that instruction.

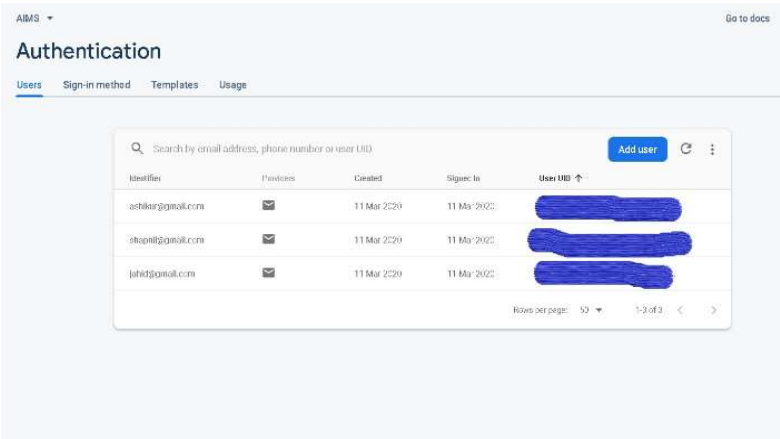


Fig-3.25: Authentication in Database.

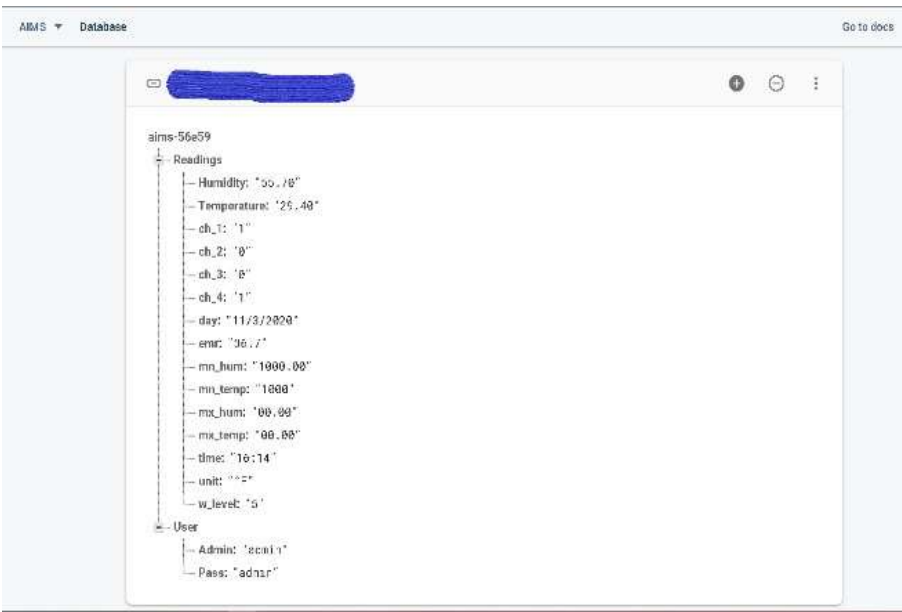
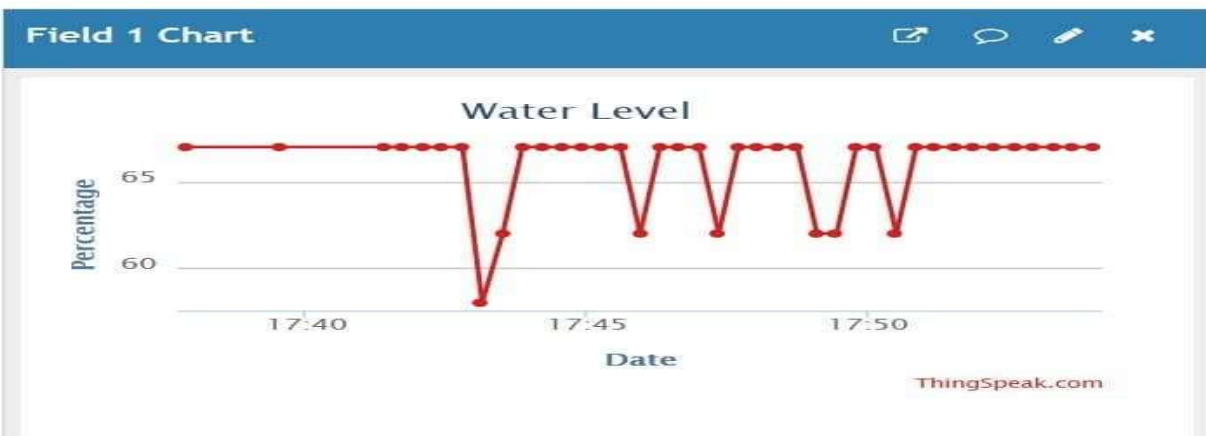
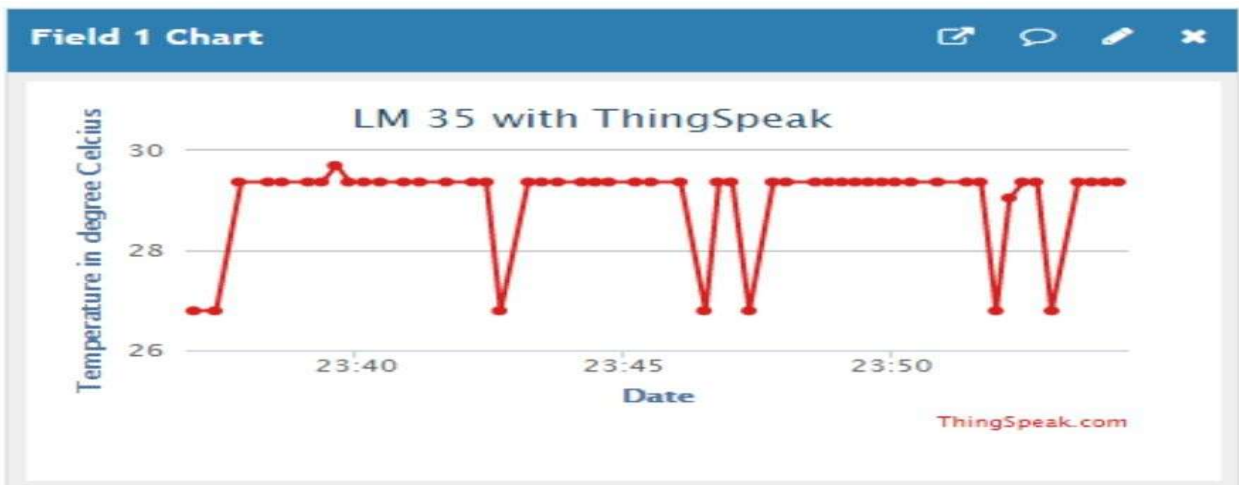
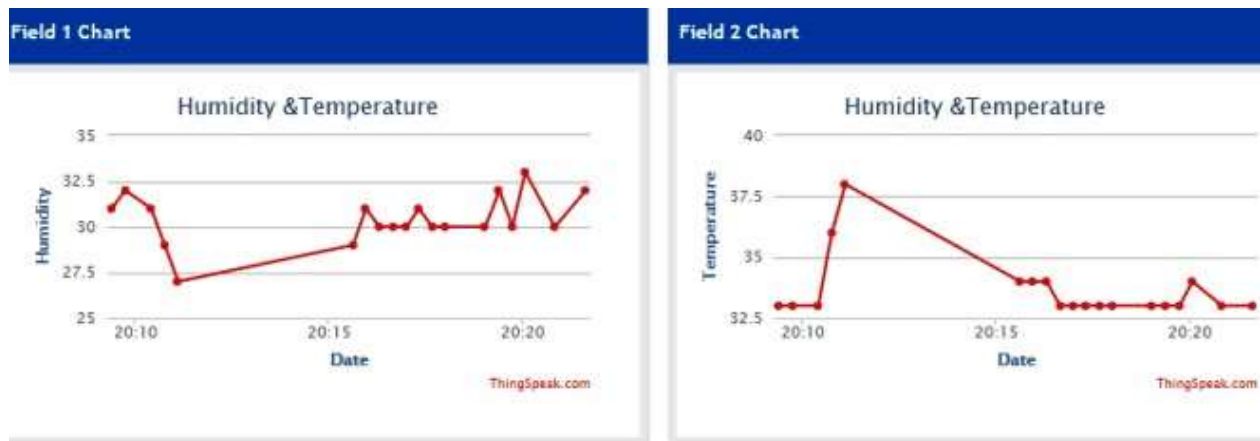


Fig-3.26: Real Time Database.

Output Graphs:



CHAPTER-IV: CONCLUSION & FUTURE SCOPE

4.1 CONCLUSION

IoT based AUTOMATIC INDUSTRIAL MONITORING SYSTEM for Live Monitoring of Temperature, humidity and water level has been proposed using Arduino and Firebase. The System has high efficiency and accuracy in fetching the live data from the sensors. The IoT based AUTOMATIC INDUSTRIAL MONITORING being proposed via this report will assist the user in monitoring and maintaining the production facility or the industry and take efficient care of the expensive machineries as the System will always provide helping hand to the user for getting accurate live feed and status of the whole system and the devices/ machineries connected or controlled with it.

4.2 FUTURE SCOPE

Future work would be focused more on increasing sensors on this system to fetch more data especially with regard to AUTOMATION and SECURITY. It will be our target to make this project more reliable and compatible with any kind of industry monitoring system. We will work hard for enhancing the efficiency of our project and keep it updated with the upcoming features and necessities.

REFERENCE & SOURCES

1. https://create.arduino.cc/projecthub/dhairya-parikh/an-ethernet-powered-server-for-home-automation-0f87ee?ref=search&ref_id=nodemcu%20set%20up&offset=3
2. <https://www.geeksforgeeks.org/adding-firebase-to-android-app/>
3. <https://roboindia.com/tutorials/nodemcu-connecting-internet-arduino/>
4. <https://www.electronicshub.org/dht11-humidity-sensor-arduino/>