

# **IOT based weather monitoring system with android app support**

A research project presented to the Jahangirnagar University in partial fulfillment of the requirement for the degree of Master of Science in Computer Science & Engineering.

**By**

**Name: Mustakim Billah Bedar**

**Registration no: 2153**

**Roll no: CSE202001009**

**Session: Spring 2020 [22nd Batch]**

**Hall Name: Bangabandhu Shaikh Mujibur Rahman hall**

**Supervised By**

**MD. ABUL KALAM AZAD**

Professor

Dept. of CSE

Jahangirnagar University



**Jahangirnagar University, Savar, Bangladesh**

**August 2021**

# Approval

This project report entitled “**IOT based weather monitoring system with android app support**” prepared and submitted by **Mustakim Billah Bedar**, roll no: CSE202001009 to the Department of Computer Science and Engineering, Jahangirnagar University, Savar, Bangladesh, for the degree of Master of Science in Computer Science and Engineering, has been examined and recommended for approval and acceptance as to its style and contents.

**Signature of Supervisor**

.....

**Md. Abul Kalam Azad**

Professor

Department of Computer Science and Engineering

Jahangirnagar University

**Signature of 2<sup>nd</sup> Examiner**

.....

**Dr. Musfique Anwar**

Associate Professor

Department of Computer Science and  
Engineering

Jahangirnagar University

**Signature of Coordinator**

.....

**Md. Golam Moazzam**

Professor

Department of Computer Science and  
Engineering

Jahangirnagar University

# Declaration

I, hereby declare that the project work entitled “**IOT based weather monitoring system with android app support**” is done by me under the supervision of **Md. Abul Kalam Azad**, Professor, Department of CSE, Jahangirnagar University, Savar. I also state that the outcomes depicted in this research project is not been handed to any other university or institute for the honor of any degree.

**Countersign:**



.....  
**Mustakim Billah Bedar**

(Candidate)

.....  
**Md. Abul Kalam Azad**

(Supervisor)

# Acknowledgement

At first, I would like to show gratitude to Almighty Allah, the most merciful for keeping His blessings on us in every sphere of life specially in this COVID 19 situation.

Then I would like to prompt my cavernous appreciation and deep regards to my supervisor **Md. Abul Kalam Azad**, Professor, Department of CSE, Jahangirnagar University for his supervision, intensive care and constant inspiration throughout this project. The blessing, help and leadership given by him 24/7 will carry us a long way in the journey of CSE life.

I also express a deep sense of gratitude to **DR. Liton Jude Rozario**, Professor, **Chairman**, Department of CSE, Jahangirnagar University, for inspiring us and giving me the opportunity to participate in this project.

I also value the insights and guidance of **Md. Golam Mowazzam**, Professor, **Coordinator**, Department of CSE, Jahangirnagar University.

At last, but not least gratitude goes to all of my teachers and stuffs of our department.

[If there is Anyone left in this brief acknowledgement does not mean lack of appreciation.]

# **Abstract**

This project describes the design and development of an IOT based weather monitoring system with android app support. The aim of this weather monitoring system is to detect entity from environment and display various weather parameters such as temperature humidity and Heat index as well. The system engages a Node MCU that comes with built in Wi-Fi module and DHT-11 sensor. This system makes use of the DHT-11 sensor for detecting weather parameters and then the collected information is sent to the web server by Node MCU which can be accessed using the IP address via Wi-Fi network. The data displayed as an output can be observed and forecasted through an android app named Mercury droid. The app is capable of showing temperature in Celsius, Fahrenheit and Kelvin. It also displays the humidity level and Heat index. Depending upon the threshold value the app is capable of giving alert for unwanted temperature arising. The design includes facts of Arduino programming, electrical circuit combination with coding and some architectures as well as basic mechanical engineering principal for the development.

# Acronyms

**IoT: Internet of Things**

**Wi-Fi: Wireless Fidelity**

**APP: Application**

**GSM: Global System for Mobile Communications**

**DHT-11: Dihydrotestosterone Humidity and Temperature**

**UV Sensor: Ultraviolet sensor**

**LCD: Liquid Crystal Display**

**OLED: Organic Light-Emitting Diode**

**IDE: Integrated Development Environment**

**AVR: Automatic Voltage Regulator**

**JDK: Java Development Kit**

**SDK: Software Development Kit**

**APK: Android Application Package**

**USB: Universal Serial Bus**

**RST: Reset**

**GND: Ground**

**SPI: Serial Peripheral Interface**

**UART: Universal Asynchronous Receiver-Transmitter**

**NTC: Negative Temperature Coefficient**

**VCC: Voltage Common Collector**

**JSON: JavaScript Object Notation**

**IP: Internet Protocol**

**REST : Representational State Transfer**

**API : Application Programming Interface**

**URL : Uniform Resource Locator**

**UI: User Interface**

# Table of Content

## Content

## Page No:

### Chapter 1 Introduction

1.1 Introduction	2
1.2 What is weather?	3
1.2.1 Weather and Climate	3
1.2.2 Air pressure and Weather	4
1.2.3 Temperature and Humidity	4
1.2.3.1 Temperature	4
1.2.3.2 Humidity	5
1.2.3.3 Relationship between Temperature and Humidity	5
1.2.4 Heat Index	5
1.3 Internet of Things (IOT)	7
1.3.1 Importance of IOT	7
1.3.2 Applications of IOT	8
1.3.3 Challenges of IOT	8
1.4 Existing System Review	9
1.4.1 Limitations of Existing System	9
1.5 Proposed System	10
1.5.1 Advantages of proposed system	10

## **Chapter 2 Literature Review**

<b>2.1 Literature Review</b>	<b>12</b>
<b>2.2 Weather monitoring system using real time database</b>	<b>12</b>
<b>2.3 Arduino Uno based weather monitoring system</b>	<b>13</b>
<b>2.4 Zigbee based weather monitoring system</b>	<b>14</b>
<b>2.5 Cloud based weather monitoring system</b>	<b>14</b>

## **Chapter 3 Requirement Analysis**

<b>3.1 Components</b>	<b>16</b>
<b>3.2 Software Component</b>	<b>16</b>
<b>3.2.1 Arduino IDE</b>	<b>16</b>
<b>3.2.2 Features of Arduino IDE</b>	<b>17</b>
<b>3.2.3 Android Studio</b>	<b>18</b>
<b>3.2.4 Features of Android Studio</b>	<b>18</b>
<b>3.3 Hardware Components</b>	<b>19</b>
<b>3.3.1 IoT Wi-Fi Module (Node MCU)</b>	<b>19</b>
<b>3.3.1.1 Features of Node MCU:</b>	<b>20</b>
<b>3.3.1.2 Pin Function of Node MCU</b>	<b>20</b>
<b>3.3.1.3 Table to describe Pin Function of Node MCU</b>	<b>21</b>



## **Content**

## **Page No:**

3.3.2 Bread Board	22
3.3.3 DHT-11 sensor	23
3.3.3.1 Pin functions of DHT-11	23
3.3.3.2 DHT-11 specifications	24
3.3.4 Budget Estimation:	24

## **Chapter 4 Design**

4.1 Connection establishment of DHT-11 and Node MCU	26
4.2 Block Diagram for IP generation	27
4.3 Procedure of getting IP information	28
4.4 Design for working principal of weather monitoring system	29

## **Chapter 5 Implementation**

5.1 Basic working principal of this weather monitoring system	32
5.2 Hardware setting	32
5.2.1 Pin configuration	32
5.2.2 Hardware assembly for the system	33

## **Content**

## **Page No:**

<b>5.3 Software setting</b>	<b>33</b>
<b>5.3.1 Library Installation</b>	<b>34</b>
<b>5.3.2 Communication port selection</b>	<b>36</b>
<b>5.3.3 Node MCU Board selection</b>	<b>37</b>
<b>5.3.4 Setting for IP generation</b>	<b>38</b>
<b>5.4 User interface of the Android App</b>	<b>39</b>
<b>5.4.1 Weather entity observation</b>	<b>40</b>
<b>5.4.2 Threshold value setting and alert system</b>	<b>41</b>
<b>5.5 RESULTS</b>	<b>42</b>
<b>5.6 Limitations</b>	<b>42</b>

## **Chapter 6 Conclusion and Future work**

<b>6.1 CONCLUSION</b>	<b>44</b>
<b>6.2 Future Work</b>	<b>44</b>
<b>References</b>	<b>45</b>
<b>Appendix:</b>	<b>47</b>

# List of figures

<b><u>Content</u></b>	<b><u>Page No:</u></b>
Fig 1.1: Heat index chart	6
Fig 2.1: Arduino Uno based weather monitoring system	13
Fig 3.1: Node MCU (ESP 8266)	19
Fig 3.2: Pin functions of Node MCU	20
Figure 3.3: Bread Board (400 holes)	22
Fig 3.4: DHT-11 sensor with pin functions	23
Fig 4.1: Connection establishment of DHT-11 and Node MCU	26
Fig 4.2: Block Diagram for IP generation	27
Fig 4.3: Procedure of getting IP information	28
Fig 4.4: Flowchart for working principal of weather monitoring system	29
Fig 4.5: UI design for Android APP	30
Fig 5.1: Hardware assembly for the system	33
Fig 5.2: Installing procedure of Wi-Fi library	35
Fig 5.3: Installing procedure of Sensor library	36
Fig 5.4: Communication port selection	37
Fig 5.5: Node MCU board selection	37
Fig 5.6: Compiling for the Program	38
Fig 5.7: IP generation for the system	39
Fig 5.8: User interface of Android App	40
Fig 5.9: Weather information of my home	41
Fig 5.10: warning for out of threshold	42

# **Chapter 1**

## **Introduction**

## 1.1 Introduction

The Weather Monitoring System is a center for providing information on weather forecasts and weather forecasts in the vicinity of IoT instruments. The weather forecast these days cannot predict that it could be accurate because of the general climate change. That is why the Weather Monitoring System is widely used to monitor climate change and climate control to control areas such as housing, industry, agriculture etc.

In modern times, climate monitoring is important and has been used in many different areas from tracking the agricultural climate of agriculture to monitoring industrial conditions. Climate monitoring helps us to track various weather parameters including temperature and humidity. The System can be linked to wireless or wireless technology. In the event of a wireless connection, the communication will be effective and easy to use. Without climate control it would not require human presence in the area. Temperature and humidity indicate indoor as well as outdoor space. Buzzers are set and available in the monitoring system which shows that they are not in the distance.

The system that we built is one kind of IoT (Internet of things) Embedded system based on Android Mobile Application which is capable to measure & monitor small area weather activity. it is very inexpensive home weather monitoring system that works perfectly. We don't need more than 10\$ to build this system. The microcontroller unit used in this project is very inexpensive and efficient containing in built Wi-Fi module. Since this project solves a specific problem within a fixed system context that's why it can be considered as an embedded system. To program the microcontroller embedded C is used which is basically a dialect of C programming language.

The basic idea is to get the weather entity from the environment through a sensor specifically DHT-11 sensor which is capable of converting analog entity of the weather parameters into digital form. This digital form can be represented as many ways as we like. But the most suitable and easiest way should be applied to get those digital data. That's why we choose to represent these data to an android device since almost all of us nowadays likes to get every information in our mobile app.

## **1.2 What is weather?**

A climate is a natural state that reflects how hot or cold it is, wet or dry, quiet or agitated, clear or cloudy. Weather is a collection of events that occur daily in space. Most of the climate use has been observed in the troposphere, part of the world's closest atmosphere. Heavy rain and fog, blue skies, cold snow, and low temperatures are all considered climate.

The climate is determined by air pressure, temperature, and humidity that vary from place to place. The climate must vary from one part of the world to another over a period of minutes, hours, days, and weeks. The climate varies because of the differences that occur due to the Sun entering any particular area, which varies in width.

### **1.2.1 Weather and Climate**

Climate is the average climate in the area for a few decades. Different regions have different geographical conditions. For example, the climate of European countries is very different from the climate of Asian countries. In news reports or other international weather news channels it provides a global climate based on the average of all regional climates. As the climate changes, climatic patterns also change. It is very difficult to say whether the weather of a particular day is influenced by climate change but it is easy these days to predict how patterns may change. For example, Weather specialists predict extreme weather events such as global warming.

Climate change is a major problem worldwide. With the increasing amount of global warming, there is likely to be more droughts and storm surges. Researchers anticipate the long-standing effects of climate change that will lead to declining sea ice and water resources in arid regions. It can increase the melting of ice, heat waves, and heavy rainfall. Asia The availability of freshwater is expected to decline in Central, South, East and Southeast Asia by the 2050s. Asian areas will be at risk due to increased flooding. The death toll from flood-related diseases and drought is expected to rise in some areas including Bangladesh.

## 1.2.2 Air pressure and Weather

Air pressure is the heaviness of air molecules that press down on the planet. Air pressure is caused by the abundance of air molecules that make up the atmosphere. The pressure of air molecules changes as we ascend from the ocean to the atmosphere. the highest stress is at sea level because the density of air molecules is plenty more there. Climate change in any region is controlled by changes in air pressure. it's miles often seen when the air stress is excessive and the sky is obvious and blue. excessive stress creates thick air and evaporates because it tactics the floor, preventing clouds from forming their herbal kingdom. When the air pressure is too low, the air flows completely and goes up to where it meets, rises, cools and builds clouds. [01] On cloudy days it can cause rain or other types of rain which is why we may need to take precautionary measures such as carrying an umbrella.

## 1.2.3 Temperature and Humidity

### 1.2.3.1 Temperature

Temperature is a numerical value that helps us understand the coldness or warmth of anything generally measured in Celsius and Fahrenheit. it is a famous fact that room temperature is 25 ° Celsius and the common frame temperature is usually said as 36.5-37 ° C. someone's frame temperature varies as it depends on gender, age, time of day, stage of workout, health reputation (inclusive of illness and menstruation), what body measure is measured, attention (waking, sound asleep, sitting) and many different parameters.

Followings are the representations of temperature values and formulas to convert:

Celsius to Fahrenheit formula	$^{\circ}\text{F} = 9/5 (^{\circ}\text{C}) + 32$
Fahrenheit to Celsius formula	$^{\circ}\text{C} = 5/9 (^{\circ}\text{F}) - 32$
Celsius to Kelvin formula	$\text{K} = ^{\circ}\text{C} + 273$
Kelvin to Celsius formula	$^{\circ}\text{C} = \text{K} - 273$
Fahrenheit to Kelvin formula	$\text{K} = 5/9 (^{\circ}\text{F} - 32) + 273$

### **1.2.3.2 Humidity**

Temperature is the intensity of the heat produced but humidity is the water level that is present in the air. Simply we can say that Humidity determines the moisture present in the air. Relative humidity represents a percent of water vapor present within the air that modifications while the air temperature gets modified. whilst air temperature receives increased, its relative humidity decreases considering that air can hold extra water molecules at that point and when temperatures drop, relative humidity increases. Relative humidity is the amount of moisture in the air gift at that time the air can "maintain" at that temperature. whilst the air can't "preserve" all the moisture, then it transforms as dew.

### **1.2.3.3 Relationship between Temperature and Humidity**

These two perspectives are different but show a significant impact on climate and sometimes depend on each other. The relationship between humidity and temperature formula simply states that they are equally distinct. If the temperature rises the relative humidity will decrease and the air will dry out. On the other hand, when the temperature drops, the air will be wet, which means that the relative humidity will increase. So, finally, we can conclude that temperature and humidity are some of the most important and fundamental concepts of climate in which one determines the temperature of an object in any place and another describes the humidity in the air.

### **1.2.4 Heat Index**

In weather reports we get to know the actual temperature and what its feels like temperature. Generally, the feels like temperature are higher the actual temperature. This feels like temperatures are known as heat index. Heat index is based on the actual temperature and humidity or relative humidity to be exact. Warmer areas can hold more moistures in the air. That's why in the summer time there is much more moistures than the winter. more moisture inside the air slows down the proportion of the evaporation. Evaporation removes moistures or sweat from our pores and skin which additionally removes heat to chill our frame. With excessive temperature and high moisture, we can sweat however not capable of put off the heat from our bodies. Therefore, our internal body temperature increases which can affect our bodies and cause cramps, exhaustion, heat stroke etc. The heat index is developed from a mathematical formula by Lans P. Rothfusz stated as: [02]



$$HI = -42.379 + 2.04901523T + 10.14333127R - 0.22475541TR - 6.83783 \times 10^{-3}T^2 - 5.481717 \times 10^{-2}R^2 + 1.22874 \times 10^{-3}T^2R + 8.5282 \times 10^{-4}TR^2 - 1.99 \times 10^{-6}T^2R^2$$

where T = ambient dry bulb temperature (°F)  
R = relative humidity (integer percentage).

The following figure is useful for heat index calculation. [03]

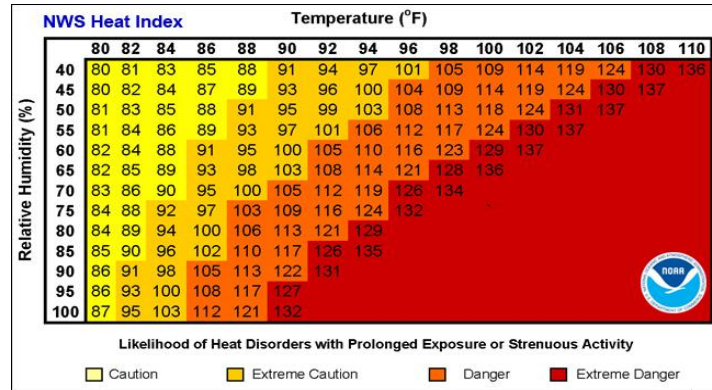


Fig 1.1: Heat index chart

For instance, if the actual temperature is 90-degree Fahrenheit and the relative humidity is 60% it generates a heat index of 100 degrees. The national weather service gives alert when the heat index exceeds 105 degrees for two consecutive days. The heat index should be used as a guide when planning an outdoor summer activity.

Moisture has a profound effect on the temperature we feel. If the air temperature is 25 ° C and the relative humidity is 0%, then the air temperature is about 22 ° C. If the relative humidity is 100% at the same air temperature, then it appears to be 28° ° C. In other words, if the air is 25 ° C and contains a lot of water vapor, then the human body cools to the same degree as it would if it was 28 ° C and dry. [04]

Low humidity causes dry nasal passages to become infected with cold germs. Low humidity can be a major cause of occasional nosebleeds. The humidity associated with the house should be kept above 30% to reduce the chances of drying the patient's lungs dry. The human body feels comfortable within a wide range of humidity with temperatures ranging from 30% to 70% but well between 50% and 60%. Too low humidity can make you feel uncomfortable, have trouble breathing, and increase allergies to most of us. In winter, it is recommended to keep the relative humidity at 30% or more. [05]

## **1.3 Internet of Things (IOT)**

internet of things (IoT) is a community of actual-lifestyles devices geared up with sensors, software program, and different modules for the reason of speaking with different gadgets and applications thru the internet. net of factors is the modern-day generation that creates a international network of machines and devices that can make connections and exchange statistics with the assist of the internet. there is a fundamental difference between the net of things and the net. The internet of factors can provide you with facts approximately connected gadgets, analyze it, and make choices. consequently, we can say that the net of factors is smarter than the net. safety cameras, sensors, motors, buildings, software program and greater are interchangeable and are examples of the net of factors.

### **1.3.1 Importance of IOT**

nowadays the net of factors has end up one of the maximum good sized and crucial technologies of the 21st century. we can hyperlink everyday items like kitchen appliances, automobiles, thermostats, toddler video display units and so on. online with embedded devices. Wireless communication can occur between people, processes, and objects. Cloud computing, big data, mobile technology and material can share and collect data without human effort as little as possible. In this global valley, digital systems can record, monitor, and adjust every communication between connected devices. [06]

Industrial IoT simply exchanges data by connecting devices, systems, smart people and real-world applications to empower areas such as automated factories, smart cities and health services. Traditional embedded systems with embedded system engineering can be considered as the heart of the object, transforming from standalone systems into a network of devices and connected systems. Tracking is one of the IoT applications that plays a similar role.

### **1.3.2 Applications of IOT**

- Smart Homes
- Smart City
- Self-driven Cars
- IoT Retail Shops
- Agriculture
- Industrial
- Telehealth
- Smart Supply chain Management
- Transportation / Mobility
- Wearables
- Water supply

### **1.3.3 Challenges of IOT**

#### **Smart communication**

Sensors and devices linked and linked collectively with the help of internet of things want to be changed or updated.

#### **Privacy and security**

Thousands of IoT devices can connect to potentially dangerous networks. Connecting to this type of massive quantity of devices calls for excessive safety requirements to prevent scams and permit a excessive degree of information safety.

#### **Big data treatment**

The maximum vital task of using internet of factors is the big quantity of records control that transfers among linked gadgets known as massive statistics.

#### **Reduce overall data delays**

Shared data for IoT devices is increasing exponentially. This causes a communication delay or data delivery delay between all connected devices.

## **1.4 Existing System Review**

current meteorological systems often use meteorological stations inclusive of thermometers, barometers, air vehicles, rain gauge and so forth. to measure weather and climate exchange. most of those devices use simple analog era this is later recorded and manually saved in the facts machine. This data is compiled digitally and sent to news stations and radio stations where the weather report is furnished.

Previous systems existed for collecting weather data or transmitting this data using ZigBee, GSM, or other remote means. Although all of these systems measure parameter values but they lack one common feature and that is accuracy. As human beings we need an accurate weather report of our environment. We need to know the right weather information so that we can thrive and adapt. These methods only apply to those areas where there is little climate change, which means that they are stable throughout the year.

In many previous studies, one method of communicating with a single master microcontroller has been developed. The microcontroller can detect using a unicast connection. Which means the master gives orders to one of the slave addresses through a star-studded slave network. Then a slave with the same address requested would respond or take action as ordered by the master.

### **1.4.1 Limitations of Existing System**

1. existing climate tracking structures used old technology and heavy hardware modules which calls for regular protection and want to be manually monitored and changed frequently.
2. strength necessities are one of the major constraints in such structures. existing systems used contraptions are typically sited a long way from important electricity supply. That's why the price receives increasing the use of such devices.
3. facts amassed from the units desires to be transferred manually from the logger to a computer or pc through a cable or any wi-fi community.

4. existing systems include huge and heavy devices occupies plenty of area that's why it's miles tough to put in them in far off place and locations that have limited area.
5. The units used inside the current systems are pricey and upkeep price is better.
6. Most of the existing systems sometimes faces problems like delaying in warning people about bad weather.

## **1.5 Proposed System**

In our project, we are going to develop IOT based weather monitoring system with android app support using Node MCU and DHT-11 sensor. The aim of this weather monitoring system is to detect entity from environment and display various weather parameters such as temperature and humidity. Temperature is displayed in both Celsius and Fahrenheit. Heat index is also present in the app. This system makes use of sensors for detecting weather parameters and then the collected information is sent to the web server which can be accessed using the IP address via mobile data hotspot. The data displayed as an output can be observed and forecasted through an android app named Mercury droid.

During the designing period of this project, I faced several problems related with the accuracy, environment, fact and figures. I will talk in details about them later where I have shown that our project is accurate enough to be put to practical use.

### **1.5.1 Advantages of proposed system**

- Safety measure in industry
- Realtime temperature and humidity feedback for greenhouse effects
- Agricultural activities
- Room thermal activities
- Less expensive
- Easy to understand.

# **Chapter 2**

## **Literature Review**

## **2.1 Literature Review**

Monitoring the weather is not a modern time research. Our ancestors and their immediate descendants observed the weather conditions and learned to monitor the weather entities manually. They realized its impact on our daily activities especially our outdoor activities like travelling, agriculture and events. Then they learnt the use of different IoT instruments like the rain gauge, barometer, anemometer etc. After few decades instruments have been used over the centuries to predict the weather. But these instruments were never enough and the need for devices that can interpret the atmospheric condition without the intervention of human effort was required. Nowadays sensors are used for automated weather monitoring stations because of its ability to sense the conditions and real feel of the atmosphere and record the data obtained more accurately without the problem of human errors.

## **2.2 Weather monitoring system using real time database**

This monitoring system monitors temperature, humidity, rainfall, and live weather reports in a particular area and makes the data observable anywhere in the planet. The purpose of this program is to provide an effective environmental monitoring system by measuring and monitoring weather information based on Internet of Things (IoT) technology and displaying weather data using a mobile app for quick and easy access to users. In this research project, the Arduino MKR Wi-Fi 1010 Board microcontroller is used to collect weathering materials from temperature and humidity (DHT11), rain module, and UV Sensor (ML8511). Data collected from sensors is stored in the Firebase real-time data system. A cloud-based database and mobile app is also created using MIT App Inventor to show real-time local weather conditions on an android phone. The main advantage of such a system is lower profitability compared to other similar weather station projects due to the use of the Arduino MKR Wi-Fi 1010 microcontroller designed to connect to Wi-Fi module for IoT applications with a cost-effective solution and low power consumption. [07]

## 2.3 Arduino Uno based weather monitoring system

The following shows hardware configuration for their project. Although other sensors are not present here but the general configuration is same for other sensors.

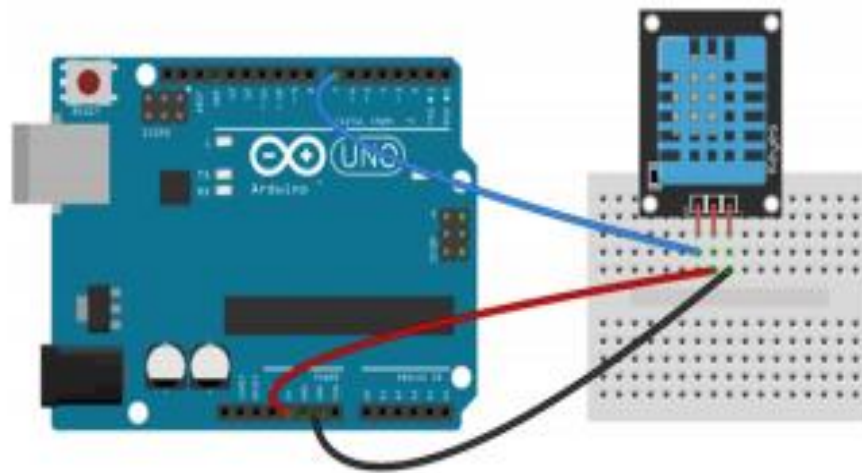


Fig 2.1: Arduino Uno based weather monitoring system

on this Io-based weather monitoring machine research undertaking, Arduino Uno combines four climatic capabilities the usage of a 4-temperature sensor, a humidity sensor, a humidity sensor and a rain-sensor. even though the sensor of warmth and humidity is lively in both climate systems. those 4 sensors are connected without delay to the Arduino Uno microcontroller. Arduino Uno microcontroller inbuilt analog to virtual converter. Arduino steps, calculate, and display this weather data on an integrated lcd display. It then sends these parameters on line the usage of IoT techniques and packages. The direction to sending information to the internet using wi-fi is updated after the extension. After that, at that point the purchaser wishes to go to a specific area to look this climate records. obligations collect and keep records at the internet server. From now on the client receives a live exposure of the weather. Web access or Internet connection and Wi-Fi are required for this weather-based project looking at IoT. [08]



## **2.4 Zigbee based weather monitoring system**

The venture became to promote the Sensor Networking and weather Station tracking system without human intervention using wireless ZigBee generation. This commitment is based on the integrity of the pollutants control Framework. WMS video display units temperature, adhesion, wind speed and identify, quantity of rain and greater. The outline reflects this ongoing analyzing in the exhibition. in addition it appears at statistics that can be verified in an hourly and daily schedule. This information may be generated from the liquid crystal display. diverse strategies are used to screen the climate inclusive of satellites, radars, microcontrollers and many different fundamental gadgets. The weather can likewise be detected by means of the use of distant sensors. Zigbee is the maximum current weather monitoring strategy. Current viewing features of the Weather Monitoring System are not handmade. We need human help to do that. There are obstacles to people's thinking beyond the boundaries of the environment. There are so many possibilities for human error. Since a person's evaluation may not be accurate over and over again. Then again, one cannot cover a large area. We need a framework that will help measure boundaries. In this application, the Wireless sensor organization can deal with this issue, where the limitations and controls will be more specific in the larger region. [09]

## **2.5 Cloud based weather monitoring system**

Here They have proposed Cloud based Weather Monitoring System. The point of their climate checking framework was to distinguish record and show different climate boundaries like temperature and mugginess. This framework utilizes sensors for recognizing and checking climate boundaries and afterward this gathered data is shipped off the cloud which can be gotten to utilizing the web. The information showed as a yield can be noticed and gauged. The framework draws in an Arduino UNO board, sensors, WIFI Module which sends information to distributed computing administrations. A page is likewise made which shows the information and showcases it to clients.[10]

---

# **Chapter 3**

## **Requirement Analysis**

## **3.1 Components**

To develop this Weather monitoring system, we used two types of components.

1. Software Component
2. Hardware Component

The software Component is used to simulate the circuit diagram. So that we can check the Weather monitoring system if it works properly and The Hardware component is used to develop the system.

## **3.2 Software Component**

- Arduino IDE
- Wi-Fi Manager & DHT-11 library
- Android Studio
- Mercury Droid Android Mobile Application.

### **3.2.1 Arduino IDE**

Arduino IDE (integrated development environment) is a cross-platform software for home windows, MacOS and Linux and is written with capabilities from C and C ++. it's miles used to jot down and upload programs to well suited Arduino boards consisting of Arduino Uno, Arduino Nano, Node MCU with the help of 0.33-party cores.

Arduino IDE supports C and C ++ programming languages the use of special guidelines that could range slightly within the system in their code. Arduino IDE has built a software program library from the Wiring venture, which affords many trendy set up and extraction processes. Arduino IDE makes use of this system to transform transportable code into a text document into a hexadecimal code uploaded to the Arduino board via the download program at the firmware of the board. [11]

### **3.2.2 Features of Arduino IDE**

#### **Inexpensive:**

Arduino forums are less expensive in comparison to other microcontroller platforms. The least luxurious form of Arduino module can be assembled manually, and even pre-assembled Arduino modules price less than \$ 50.

#### **Cross Platform:**

Arduino IDE software program runs on nearly all applications such as windows, Macintosh OSX. most microcontroller packages are restrained to home windows.

#### **Simple and clear system environment:**

The surroundings of the Arduino machine is simple to use for novices, however it's far bendy sufficient for superior customers to apply.

#### **Open source and extended software:**

Arduino software program is an open source tool, available for extended programming. Language can be extended with C ++ libraries, and those who are inclined to apprehend technical info can create a transition from Arduino to AVR C based totally programming language. in addition, we can add AVR-C code directly to our Arduino programs if we want to do so.

#### **Project Documents:**

Arduino IDE works for editors and gives the choice to document their initiatives. this feature facilitates them hold tune of their development and any modifications they make every time. then again, texts allow a few human beings to without difficulty use drawings on their forums.

#### **Great Library:**

Arduino IDE has extra than seven-hundred libraries included into its library management. those libraries were written and shared through individuals of the Arduino community in order that other users ought to use them for their own initiatives without input from the internet.

### 3.2.3 Android Studio

Android Studio is the (IDE) framework of Google based on the JetBrains IntelliJ idea packages and is sincerely planned for Android improvement. available for home windows, MacOS and Linux applications. Android Studio exchanges Eclipse Android development (E-ADT) equipment as an important IDE for Android app improvement. First, we introduce an critical characteristic called the Java improvement kit (JDK). After that, we quick down load and release Android Studio because the Android software improvement kit (SDK), which is a hard and fast of modifying gear needed to combine Android programs.

To assist enhance the device within the Android app, Android Studio makes use of a Gradle-primarily based form framework, emulator, code codecs, and a Github combination. each mission in Android Studio has at the least one technique with source code and asset facts. these modules include Android app modules, Library modules, and Google App Engine modules.

Android Studio makes use of the rush object to press code and asset changes into the working system. The code modifying manager assists the fashion designer with encoding and provides coding, extraction, and testing. Apps that run on Android Studio and are integrated into the APK house constructing inside the Google Play save. [12]

### 3.2.4 Features of Android Studio

The following features are provided in the current stable version:

- capable of android-particular refactoring and brief solving.
- Has constructed in template-based wizards to create innovative Android designs and components.
- provides wealthy format editor which allows customers to pull-and-drop user components and offers choice to preview layouts on multiple screen configurations.
- offers built-in aid for Google Cloud Platform and Google App Engine.
- presents Android virtual device to run and debug packages.

### 3.3 Hardware Components

- IoT Wi-Fi Module. (Node MCU)
- Temperature and Humidity measuring Sensor. (DHT-11)
- Power bank. [optional]
- Jumper ware. [Male to Male]
- USB Cable. [Micro USB]
- Android Mobile. [with USB micro]
- Breadboard [400 holes, (35\*47mm)]

#### 3.3.1 IoT Wi-Fi Module (Node MCU)

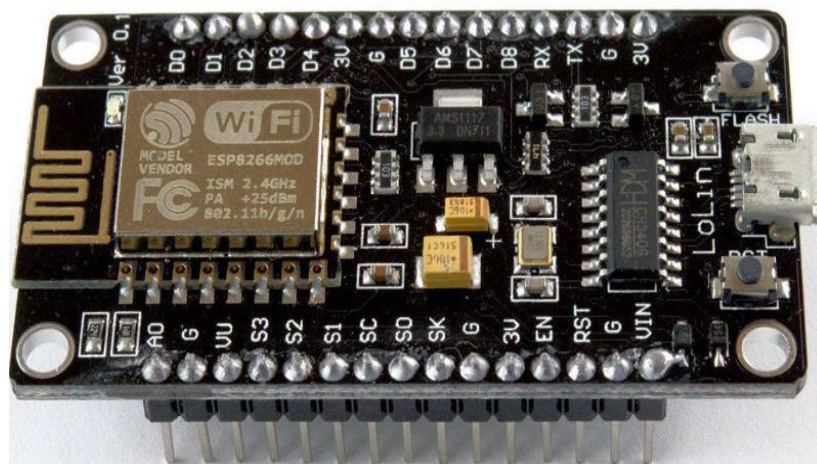


Fig 3.1: Node MCU (ESP 8266)

The Node MCU is a reasonably-priced, effective, open and open source platform. devoted firmware set up on it really works on ESP8266 wi-fi SoC and hardware based on ESP-12 module. The Node MCU is an open source firmware in which it is possible to configure open source forums. There are two variations of NodeMCU based on their variations as model 0.nine & 1.0 type zero.nine contains ESP-12 and version 1.0 includes ESP-12E wherein E stands for "advanced". This MCU Node is likewise called a community Jammer that may be used for correct hacking of wi-fi community facts.

### 3.3.1.1 Features of Node MCU:

- Node MCU affords advanced API for hardware IO that could amazingly lessen the replica work for configuring and manipulating hardware.
- It offers an event-driven API for network applications that facilitates builders writing code in Nodejs fashion.
- Its now not more than 2-dollar wi-fi MCU ESP8266 integrated and easy to prototyping with development package.
- Open-supply, Interactive and Programmable.
- Low fee, easy and clever.
- wireless enabled and Arduino-like hardware IO.
- Nodejs style community API.
- able to accelerate IoT software developing process.

### 3.3.1.2 Pin Function of Node MCU

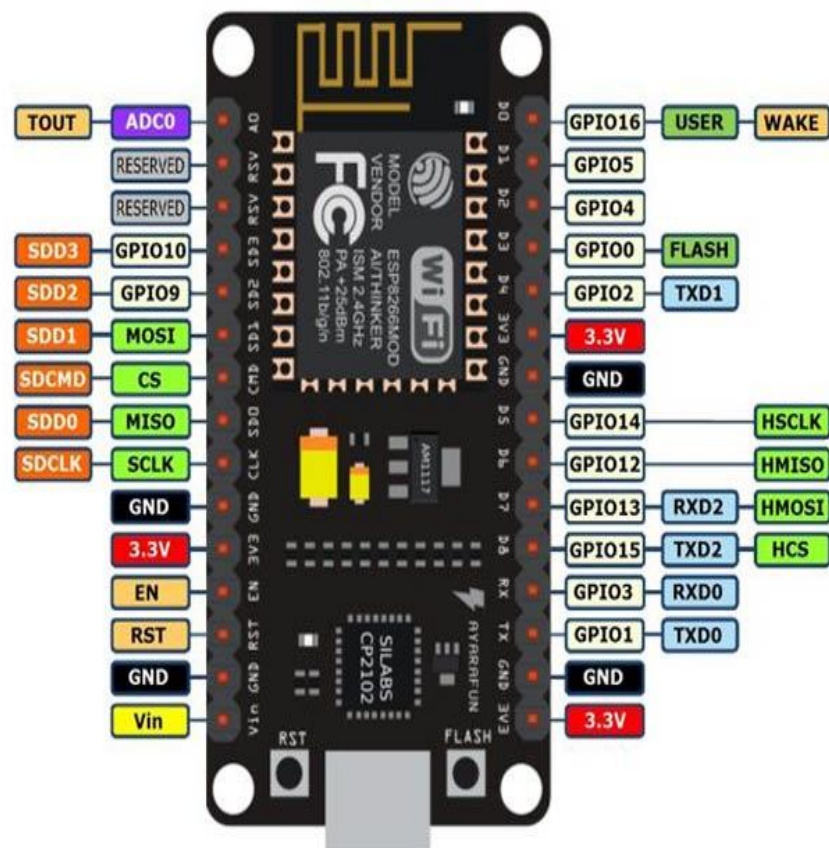


Fig 3.2: Pin functions of Node MCU

### 3.3.1.3 Table to describe Pin Function of Node MCU

Pin Category	Name	Description
Power	Micro-USB 3.3V GND Vin	Node MCU may be powered by means of the micro-USB port. Regulated 3.3V can be furnished to this pin to strength the board. Vin is used for external strength deliver.
Control Pins	EN, RST	For resetting the microcontroller.
Analog Pin	A0	This pin is used to calculate analog voltage in particular range.
GPIO Pins	GPIO1 to GPIO16	Node MCU has sixteen (16) preferred reason IO (Input-Output) pins on its board.
SPI Pins	SD1, CMD, SD0, CLK	Node MCU has 4 pins to be had for SPI verbal exchange.
UART Pins	TXD0, RXD0, TXD2, RXD2	NodeMCU has 2 UART interfaces, UART0 (RXD0 & TXD0) and UART1 (RXD1 & TXD1)
I2C Pins		Node MCU gives I2C capability however because of the inner functionality of those pins it could be hard to discover.



### 3.3.2 Bread Board

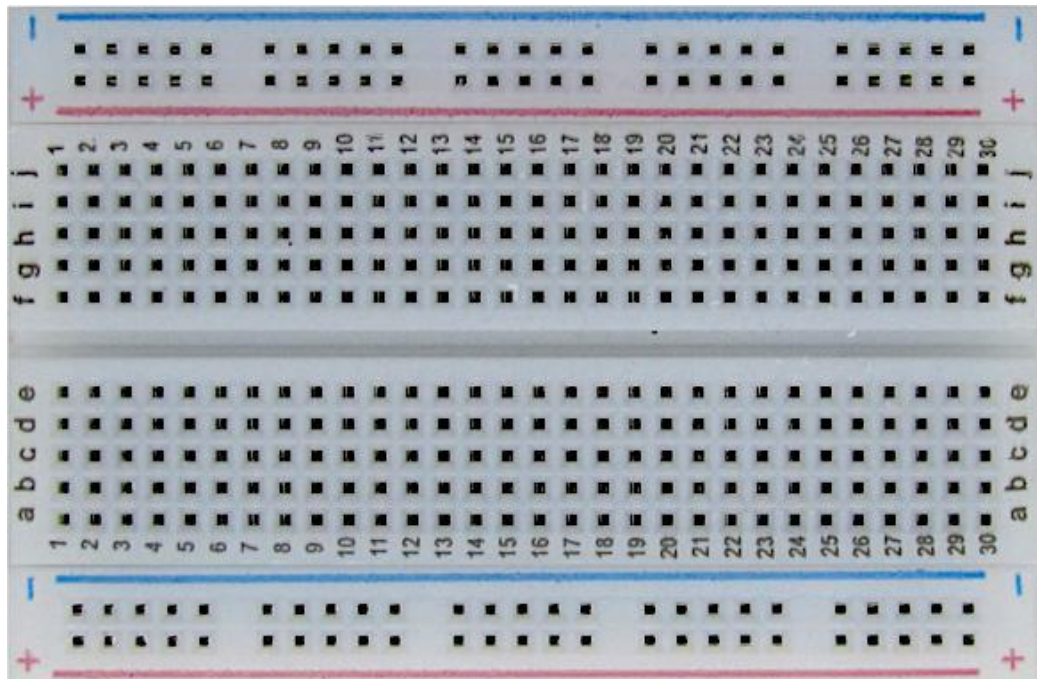


Figure 3.3: Bread Board (400 holes)

Bread board is the basis for building prototyping our IoT modules. is the basis for the development of prototyping of electrical devices. Modern bread boards have great strength, very high resistance, and reliable connectivity. On Signaling breadboards it is limited to about 10 Mega Hertz and sometimes works very poorly under that frequency. A common use of bread board is that it can hold a microcontroller (MCU). This board is often used to develop a specific type of healthy project because performing prototyping is costly and harmless. The modern breadboard socket contains plastic with high phosphors with glass, copper or silver nickel alloy in the lower spring. The gap between the clips is usually 0.1 inches. Due to the high potency of insects and not very resistant to contact reproduction. Non-standard bread boards are limited to low frequency especially 10 MHz and also vary depending on the type of circuit. [14]

### 3.3.3 DHT-11 sensor

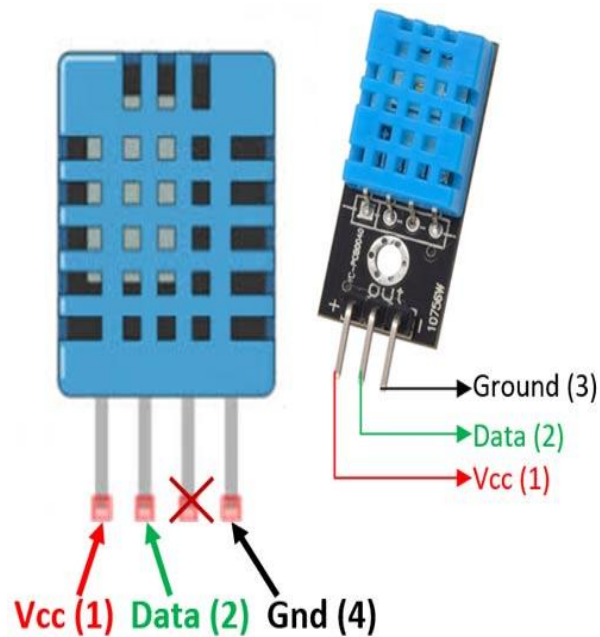


Fig 3.4: DHT-11 sensor with pin functions

The DHT-11 sensor is the most widely used temperature and humidity sensor has NTC which is dedicated to measuring temperature and humidity. It is a composite sensor that provides digital signal output of heat and humidity. This sensor has the opposite of the wet part and the NTC temperature measuring device. It can be connected to an 8-bit high-performance microcontroller.

#### 3.3.3.1 Pin functions of DHT-11

Vcc	Power supply 3.5V to 5.5V
Data	Outputs both Temperature and Humidity through serial Data
NC	No Connection and hence not used
Ground	Connected to the ground of the circuit

### **3.3.3.2 DHT-11 specifications**

- Operating Voltage: 3.5V to 5.5V
- Operating current: 0.3mA (measuring) 60uA (standby)
- Output: Serial data
- Temperature Range: 0°C to 50°C
- Humidity Range: 20% to 90%
- Resolution: Temperature and Humidity both are 16-bit
- Accuracy:  $\pm 1^{\circ}\text{C}$  and  $\pm 1\%$

### **3.3.4 Budget Estimation:**

1. IoT Wi-Fi module: 200 tk
2. Temperature and Humidity measuring sensor: 150 tk
3. Jumper wire: 50 tk
4. USB cable: 100 tk
5. Breadboard: 50 tk

Total =  $200+150+50+100+50=550$  tk (approximate)

# **Chapter 4**

## **Design**

## 4.1 Connection establishment of DHT-11 and Node MCU

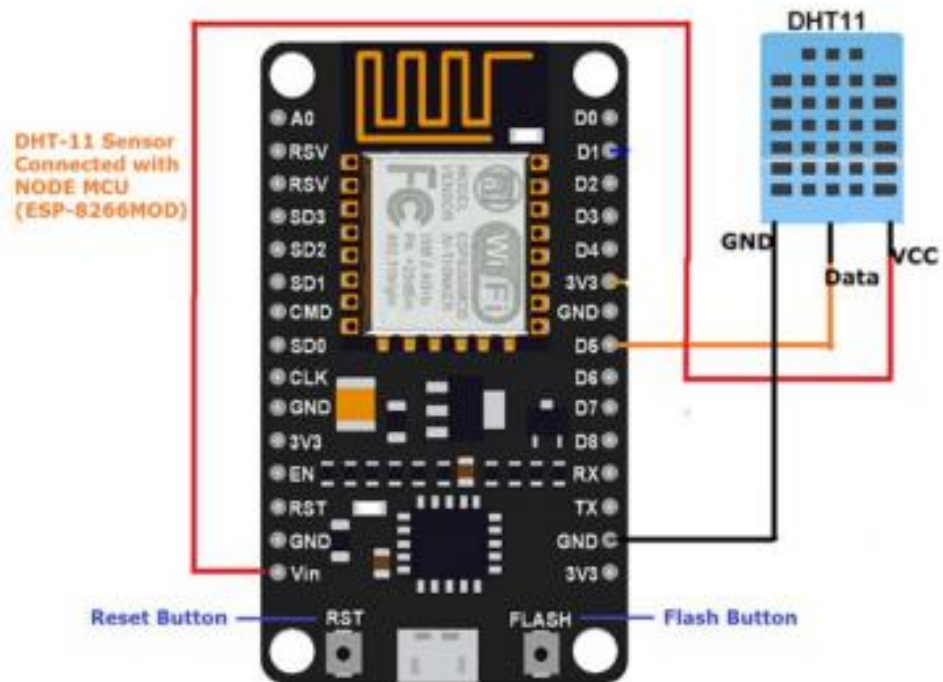


Fig 4.1: Connection establishment of DHT-11 and Node MCU

At first, we need to set the Node MCU and DHT-11 sensor to the breadboard. Then we have to connect them through jumper wire. In above design lines represents jumper wires. Since DHT-11 sensor has 3 working pin we have to connect them with specific pin numbers of Node MCU. We have to connect them as shown in the above figure and described as follows:

DHT-11 data pin to Node MCU D6

DHT-11 GND pin to Node MCU GND

DHT-11 VCC pin to Node MCE Vin

## 4.2 Block Diagram for IP generation

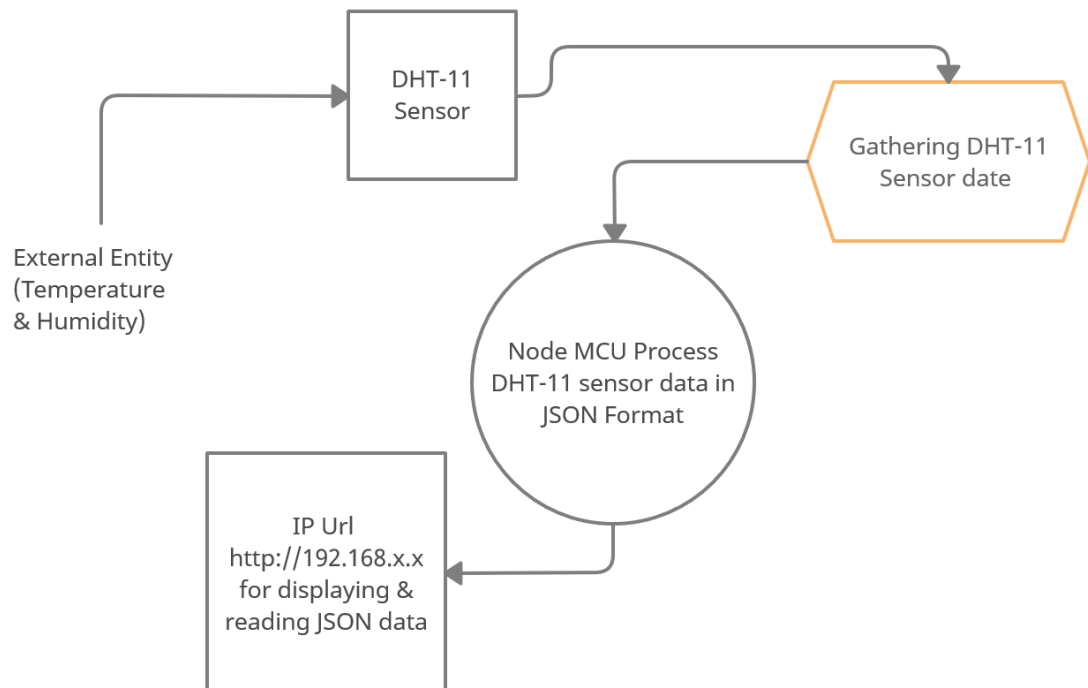


Fig 4.2: Block Diagram for IP generation

The DHT-11 sensor will sense the environmental data and provide it to Node MCU to process the data in JSON format. Node MCU has its own web server that can hold DHT-11 Sensor information which can be retrieved by hitting the IP address generated by the Node MCU. Now this information can be observed through various ways. One of the easiest ways is to get the IP address from Arduino IDE and place it to a web browser with appropriate parameters of the root or we can place the IP address to a weather monitoring website designed specifically for this purpose only such as Things Speak or we can display the information through an Android device. We choose to display this information through a android app named Mercury Droid.

### 4.3 Procedure of getting IP information

When we will get IP information our half of work is done since we just need to place the IP address to the mobile app and get the result. But the internal procedure tells us more about the working mechanism. Once Node MCU generates the IP address it should be stored somewhere that can be accessed via certain communication media. The generated IP address can be gathered from the Arduino IDE serial monitor. Then we will be able to place the IP address to the android mobile app and get the weather information.

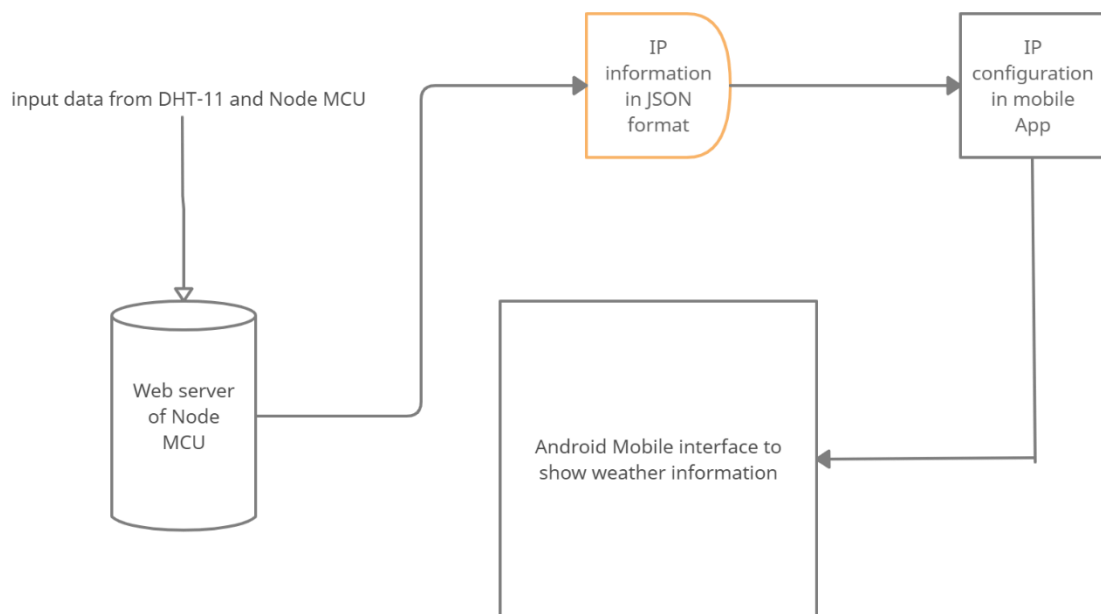


Fig 4.3: Procedure of getting IP information

JSON data provides weather entities as objects. By reading those objects we get values of weather entities. Mobile applications are not connected directly with the database. For mobile applications to connect with database a technology is used known as Rest API. In mobile apps to connect with database we need a middleware. Middleware could be a piece of code that will help to connect through a link. For the link URL is required. When we inspect the URL all the data from the database will represent as a format known as JSON format. This JSON format is one kind of Rest API. This means database data is accessed by third party. This format can be read by third party.

#### 4.4 Design for working principal of weather monitoring system

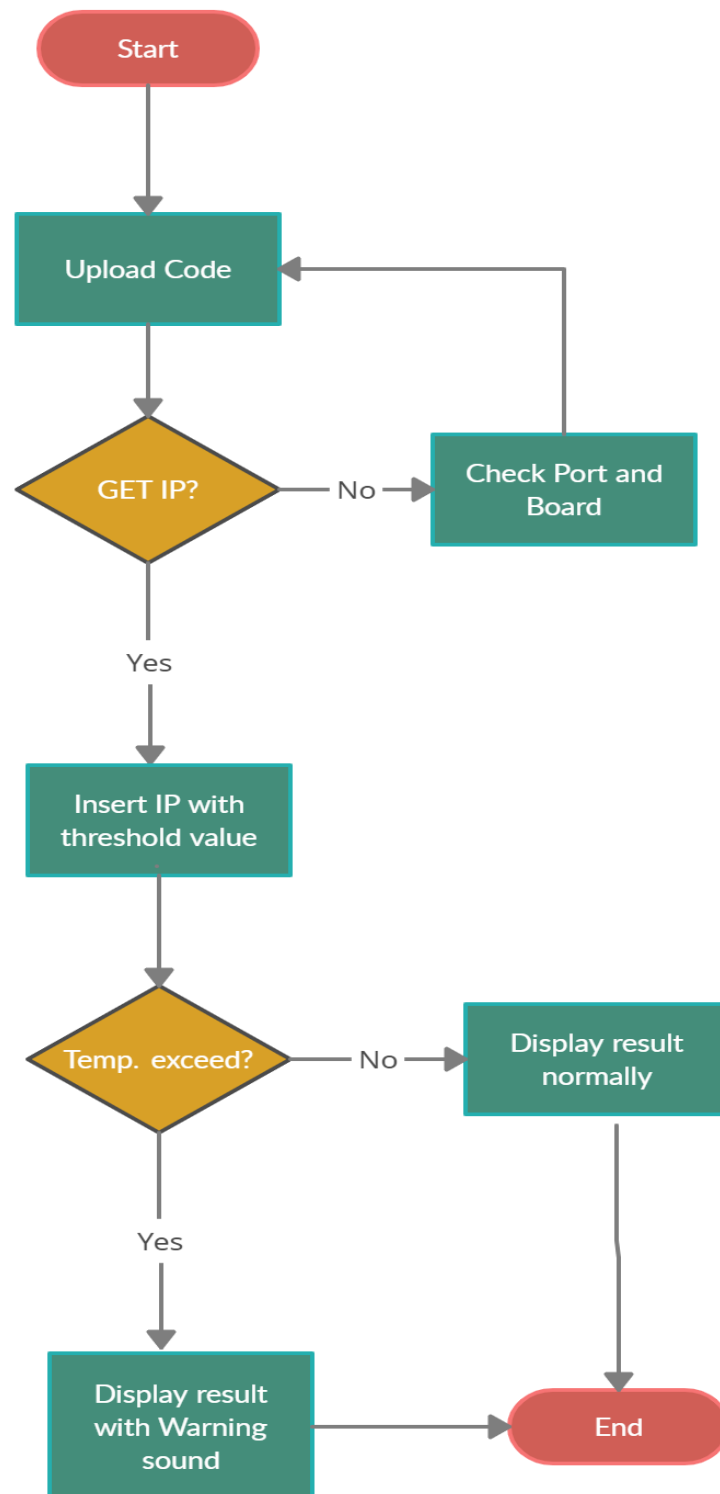


Fig 4.4: Flowchart for working principal of weather monitoring system



First of all, we will configure DHT-11 and Node MCU and connect it with a USB cable to our laptop. Then we will open our Arduino IDE and Compile and Upload our code given in the appendix to the microcontroller. Then we will get the IP generated by the Node MCU which should be available in serial monitor of Arduino IDE. If the IP is not generated then it can be the reason of not connecting the port or the specific board number of the Node MCU. We will see those specific port or Board number in the implementation phase. But if everything goes right then we should get the IP and place it to the Android App. The UI design for the app should look like following:

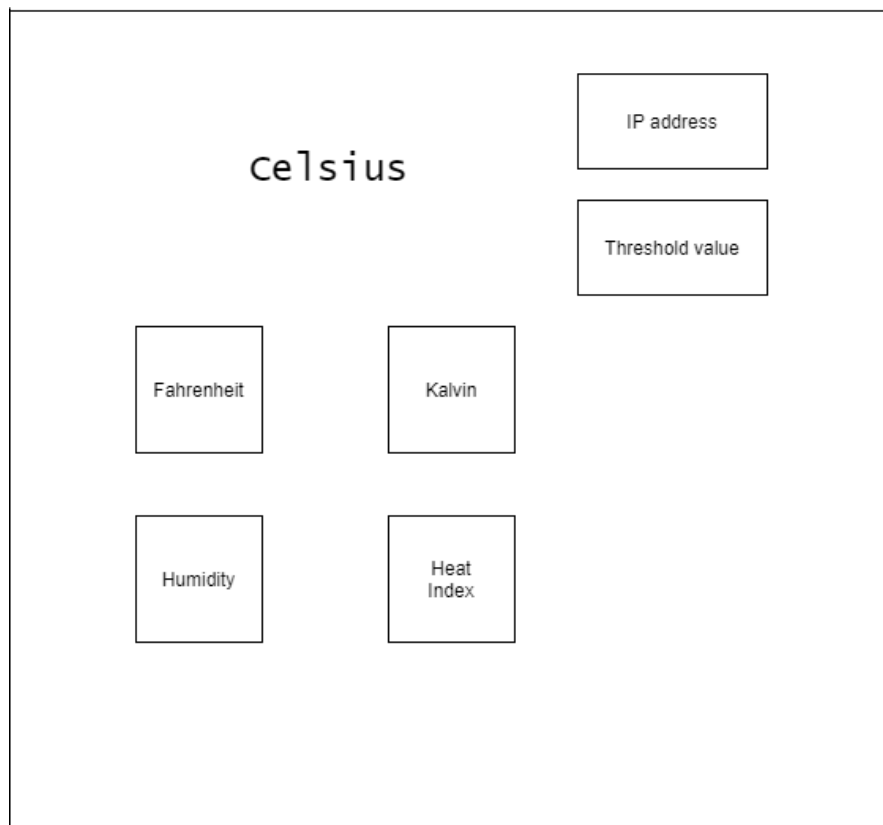


Fig: 4.5 UI design for Android APP

We should provide the exceeding value which is the threshold for the temperature. Then when the system runs, we should observe the results in comfort or discomfort conditions. For any inconvenience that is for the exceeding value the system should provide alert. Thus, the way we can monitor weather condition of a specific location.

# **Chapter 5**

## **Implementation**

## **5.1 Basic working principal of this weather monitoring system**

In this project I have used Node MCU (ESP-8266) Wi-Fi IoT module to process and store weather information. The Node MCU is working as a brain of Our Android app. The DHT11 temperature and humidity sensor measures the real time temperature and humidity and send them in Node MCU. When Node MCU get all the DHT11 Sensor data then it converts this data into a “JSON” String or Data and send them it’s Webserver. Now the Mercury Droid mobile application reads this JSON data from the Node MCU Webserver and show this data to its UI (User interface). This application has also a special feature to measure the excessive temperature value and compare it with user given threshold value. Like that if our current weather temperature is 30 Degree Celsius but the threshold value is less than that then the application gives us alert. If the Threshold value is bigger than Home Current temperature it’s not give you any alert. But we prefer to set the threshold more than 40 degree Celsius since home temperature is 25 degree and Normal temperature of Bangladesh is around 30 to 25 Degree Celsius.

## **5.2 Hardware setting**

Hardware configuration is important because any mistake can result sort circuit and can damage IoT module.

### **5.2.1 Pin configuration**

DHT-11 Data Pin Connected to the Node MCU D5 pin

DHT-11 VCC Pin Connected to the Node MCU Vin pin

DHT-11 GND Pin Connected to the Node MCU GND pin

## 5.2.2 Hardware assembly for the system

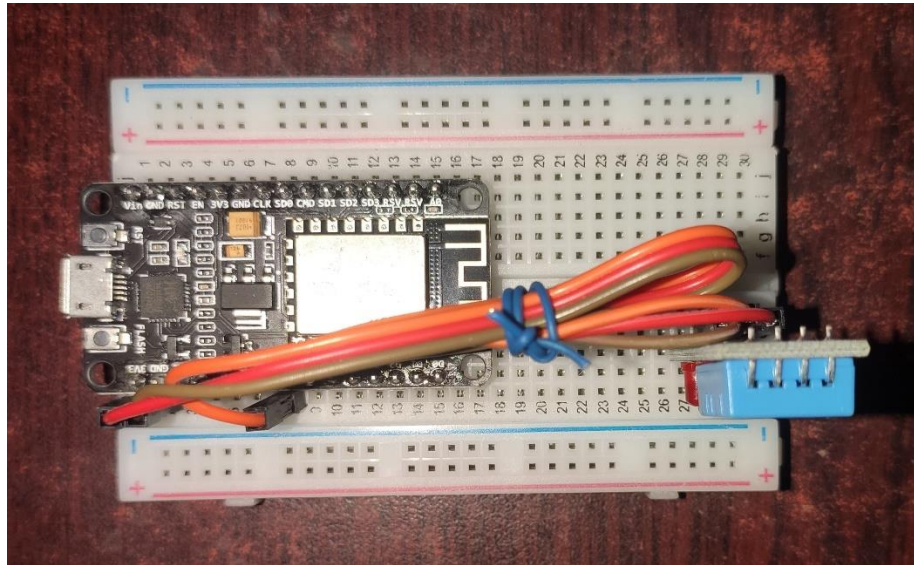


Fig 5.1: Hardware assembly for the system

After successfully connected to DHT-11 with Node MCU we are ready to configure our Node MCU Web Server and Mercury Droid Application.

Bear in mind that,

- ✓ Node MCU RST (Reset) Button helps us to connect mobile hotspot with Wi-Fi module
- ✓ Node MCU FLASH Button Erase all our code and configuration from it.

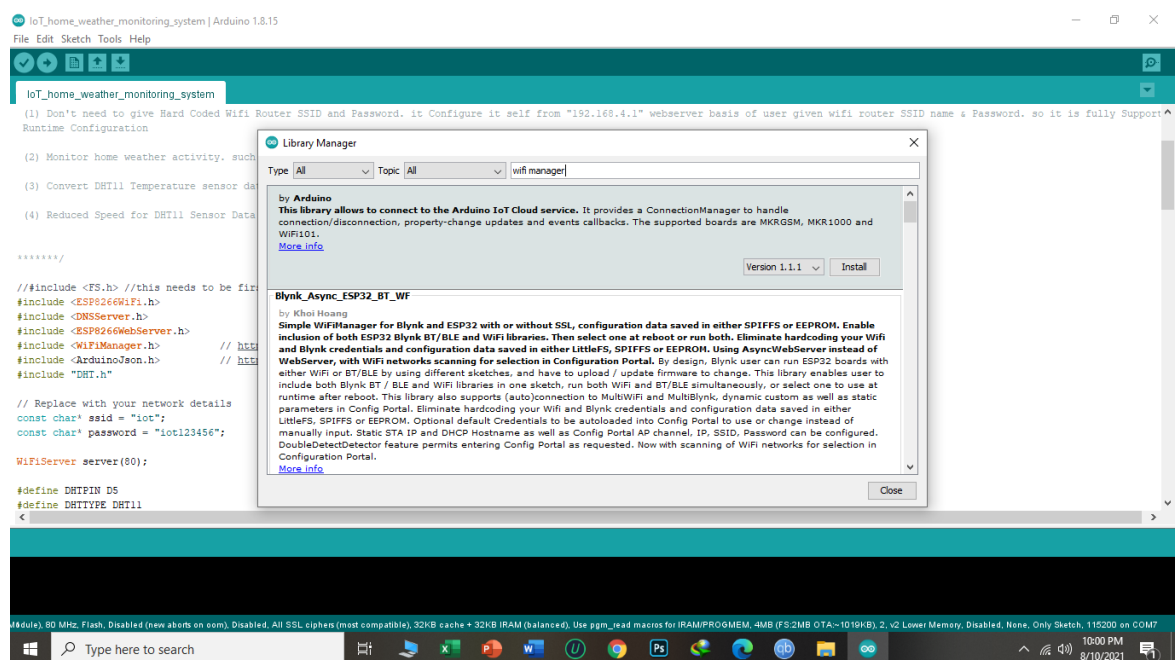
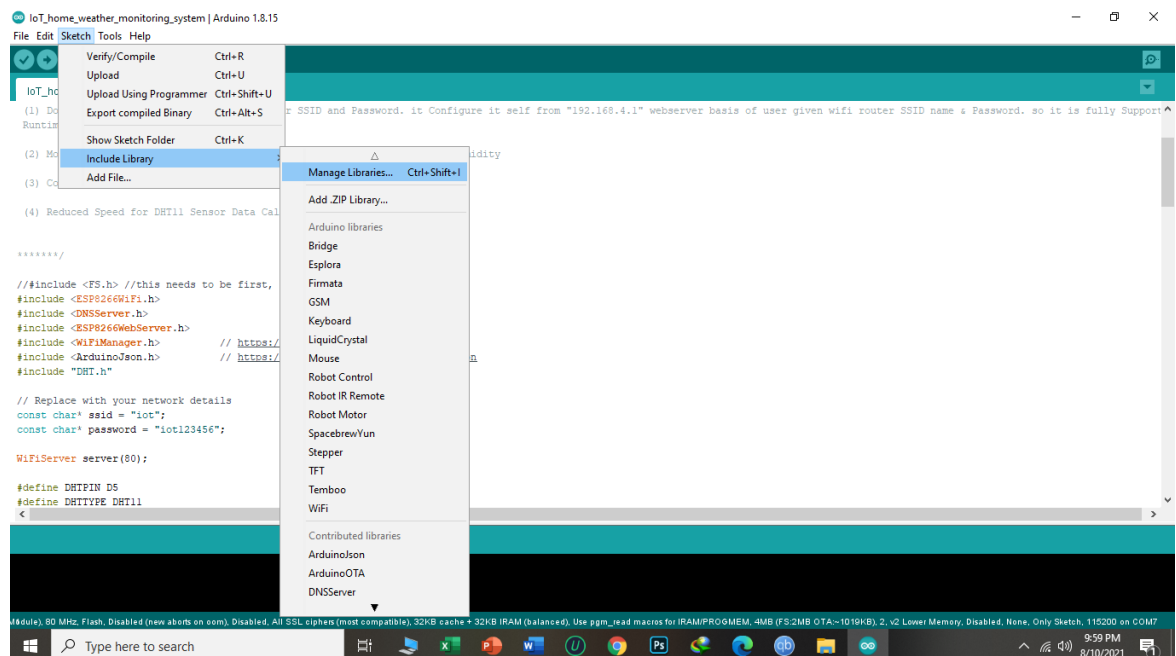
## 5.3 Software setting

Software setting is also crucial because lack of installation packages or selection can result failure or crash the android app.

## 5.3.1 Library Installation

Installing Arduino IDE we have to install some important libraries for our microcontroller and Sensor.

To install libraries for Wi-Fi module we have to open our Arduino IDE and press Sketch >> Include Library >> Manage Libraries Then in “Filter Your Search” Bar Write “Wi-fi Manager”. It will show the Wi-Fi manager library, press drop down menu and select version of Wi-Fi manager and press install.



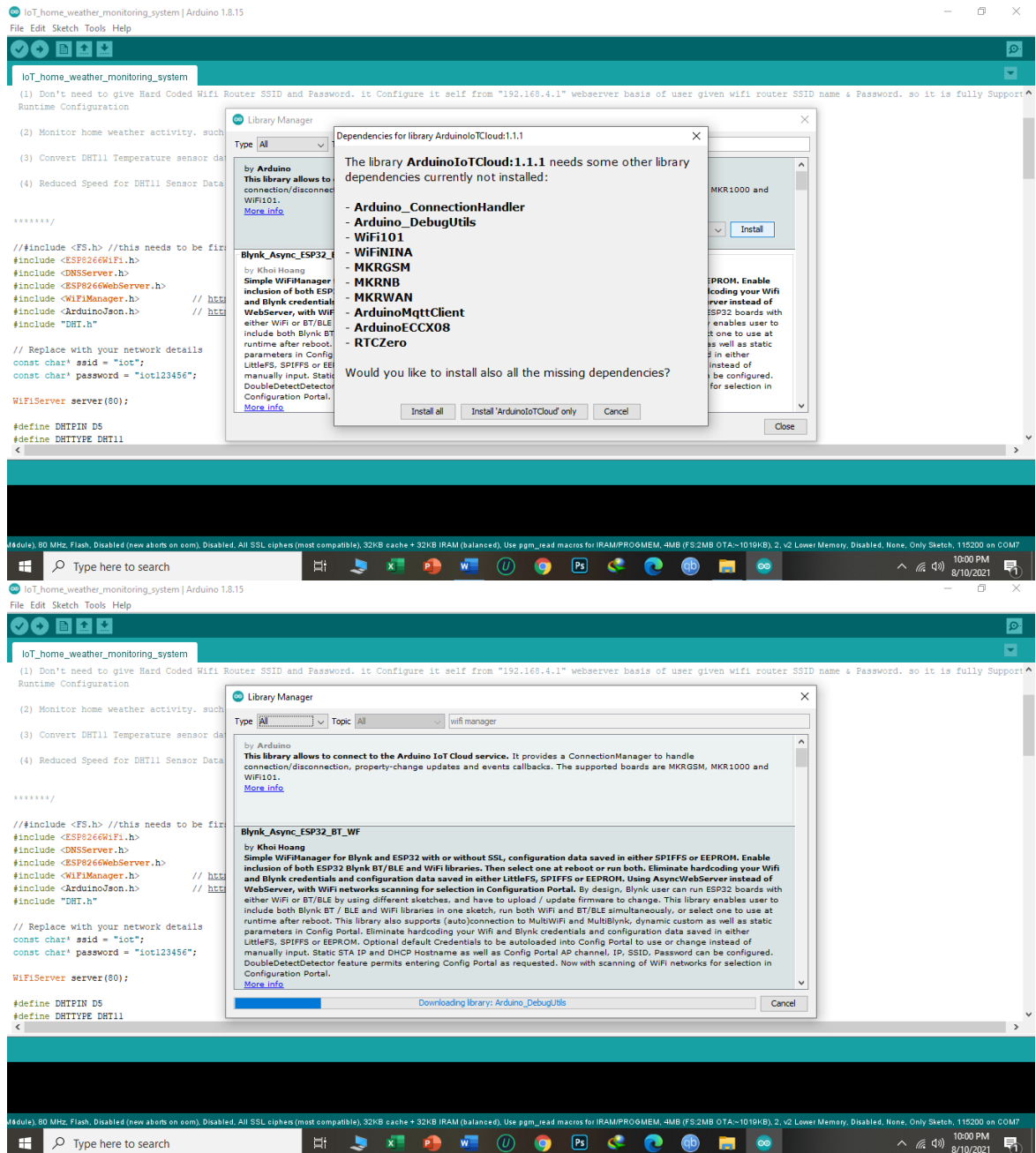


Fig 5.2: Installing procedure of Wi-Fi library

To install libraries for DHT sensor we need to install Libraries the same way we installed Wi-Fi manager library but this time we choose “DHT sensor library by Adafruit Version” and select desired version and install. But it is good to select latest version of both DHT-11 sensor and Wi-Fi manager Library.

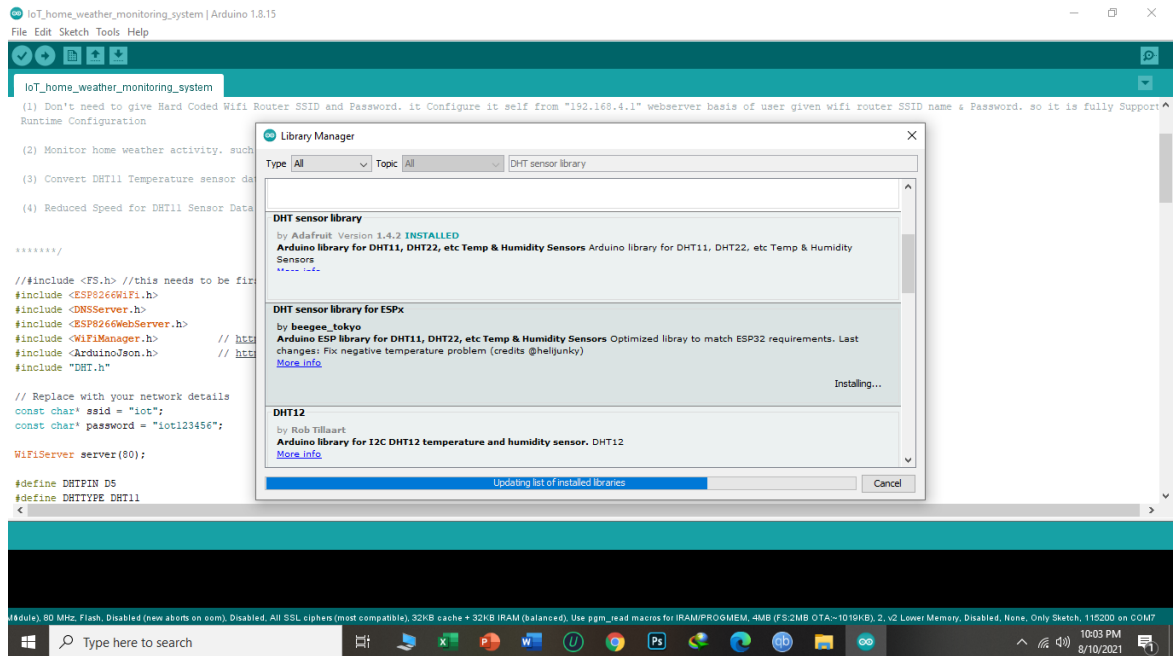


Fig 5.3: Installing procedure of Sensor library

### 5.3.2 Communication port selection

At first, we have to connect the system to a PC with a USB cable. Our PC should give a notification or simply make a sound that will tell us that our PC acknowledged our system. The light attached to the DHT-11 sensor should On. But still we are not sure whether our system is ready to work with Arduino IDE or not. To do so we have to go to Arduino IDE >> Tools>> Port

Hence, we will see a port number will be given if the system is connected. The port number is generally COM20, COM7 etc. If the port is checked then its already selected otherwise, we have to check this COM port for successfully connect with the system.

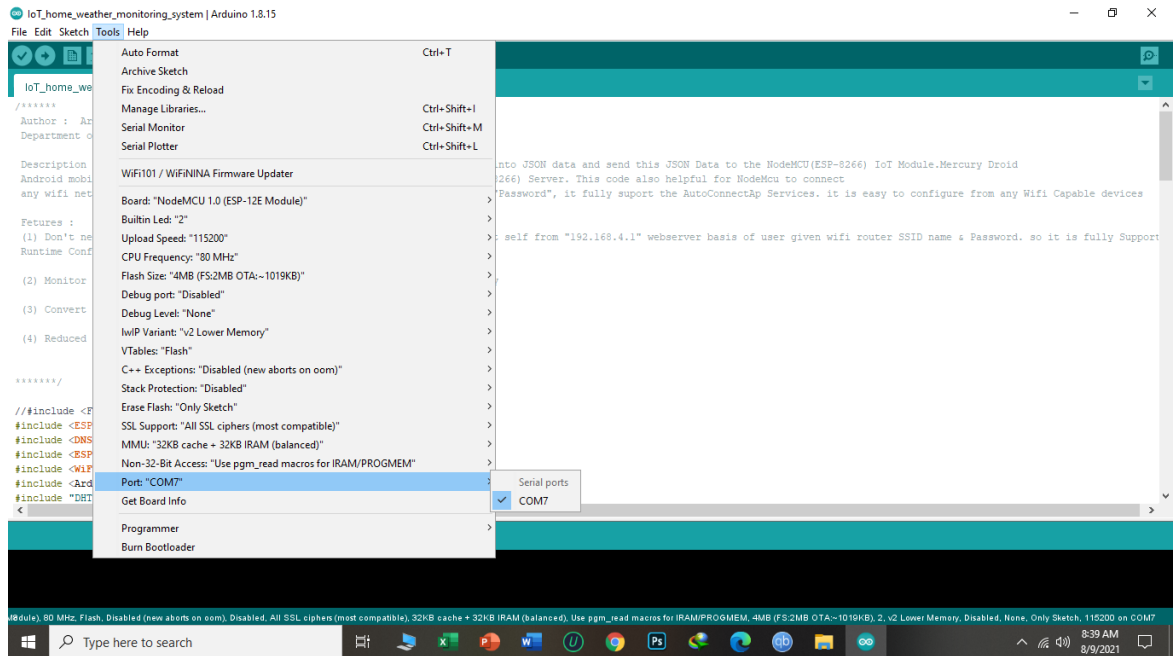


Fig 5.4: Communication port selection

### 5.3.3 Node MCU Board selection

Node MCU board should be auto checked but we have to be sure and go to tools>>board>>ESP 8266 board >> and select Node MCU 1.0

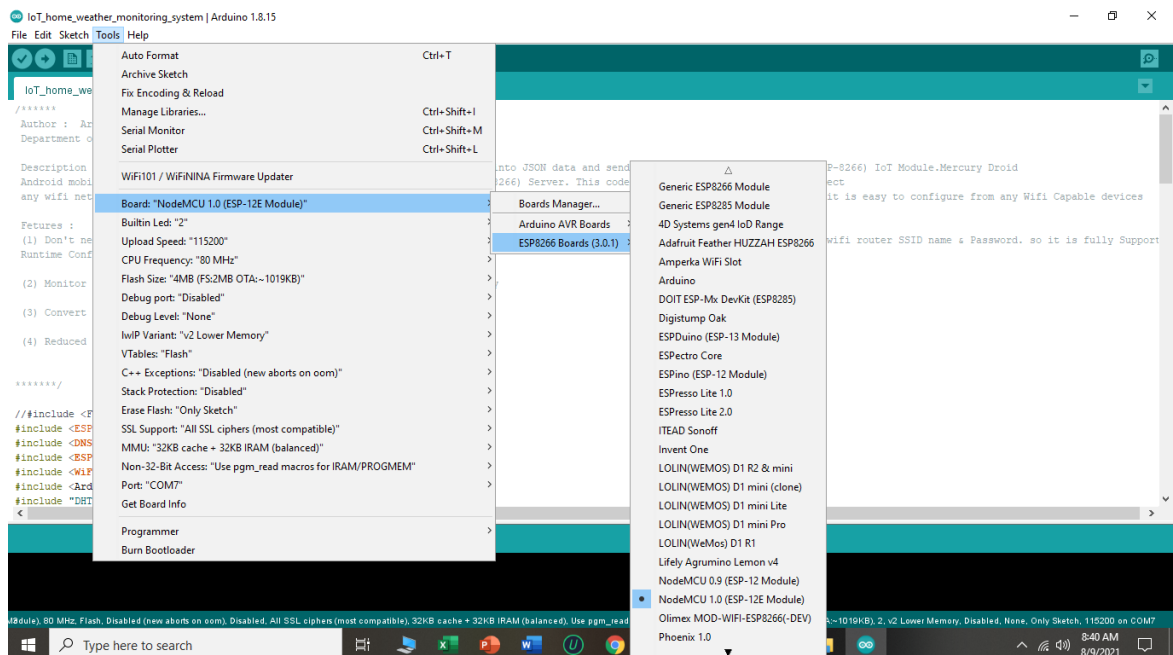
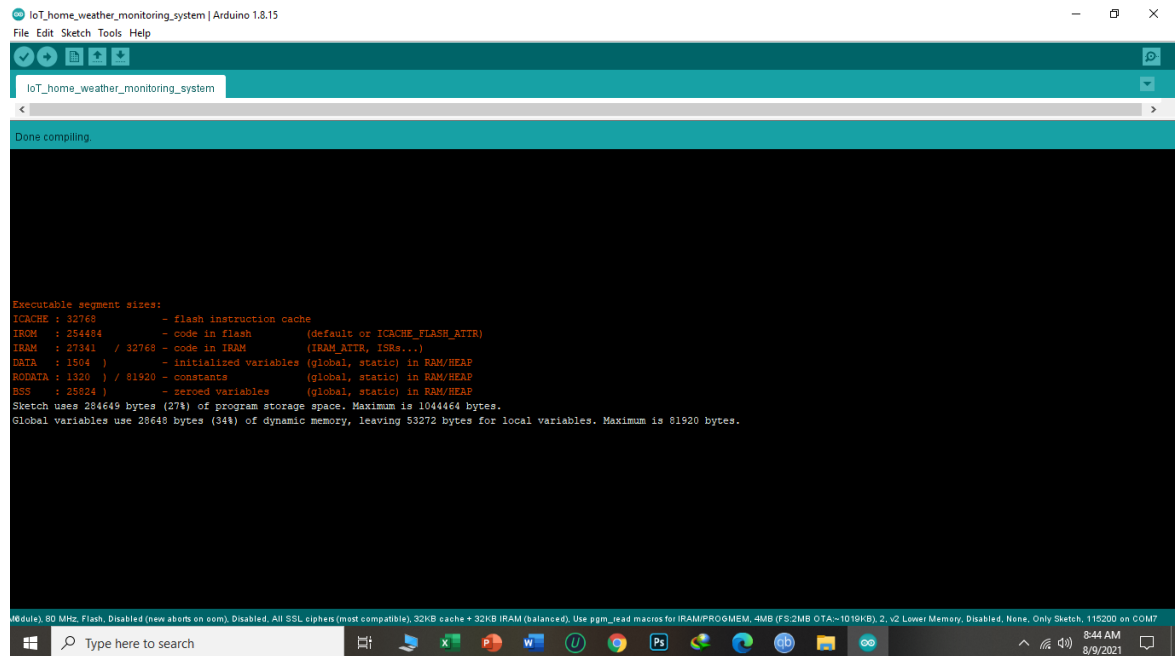


Fig 5.5: Node MCU board selection



### 5.3.4 Setting for IP generation

At First, we place the code written in embedded c and check or verify by compiling. If compilation is successful then we will upload or code to the Node MCU. For IP generation we have to go to serial monitor and select 115200 baud. Then we need to create a Wi-Fi network from the device the app is installed. After that we simply press the RST button and see the magic that is serial monitor will provide our desired IP address.



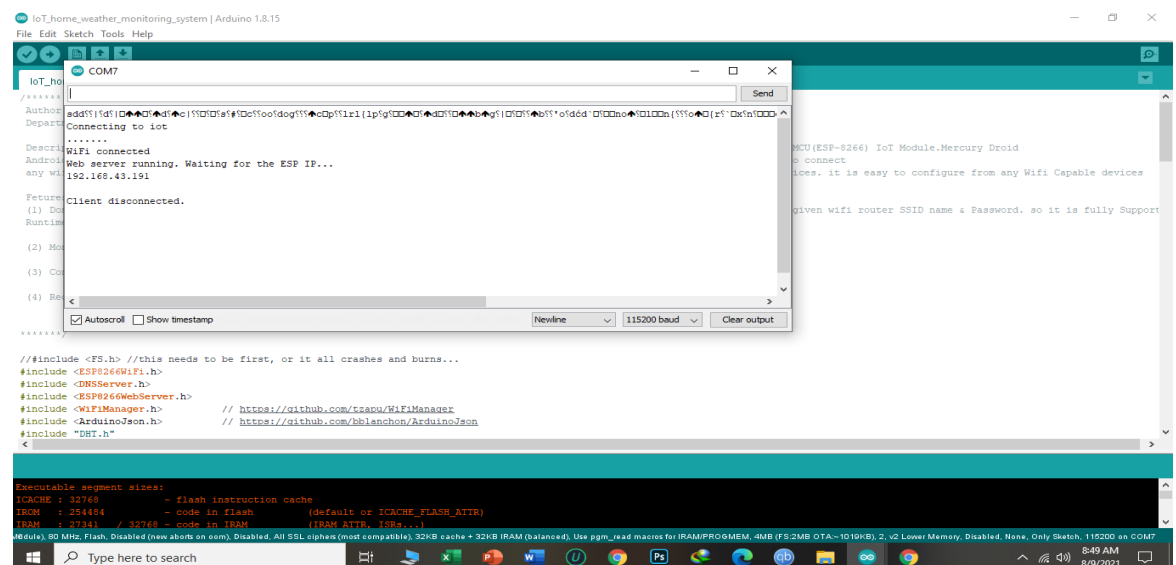
```
IoT_home_weather_monitoring_system | Arduino 1.8.15
File Edit Sketch Tools Help

Done compiling.

Executable segment sizes:
ICACHE : 32768      - flash instruction cache
IRAM   : 254484     - code in flash (default or ICACHE_FLASH_ATTR)
IRAM   : 27341 / 32768 - code in IRAM (IRAM_ATTR, ISRs...)
DATA   : 1504 )     - initialized variables (global, static) in RAM/HEAP
RODATA : 1320 ) / 81920 - constants (global, static) in RAM/HEAP
BSS    : 25824 )     - zeroed variables (global, static) in RAM/HEAP
Sketch uses 284649 bytes (27%) of program storage space. Maximum is 1044464 bytes.
Global variables use 28648 bytes (34%) of dynamic memory, leaving 53272 bytes for local variables. Maximum is 81920 bytes.

80 MHz, Flash, Disabled (new aborts on com). Disabled. All SSL cipher (most compatible). 32KB cache + 32KB IRAM (balanced). Use pgm_read macros for IRAM/PROGMEM. 4MB (FS:2MB OTA~1019KB), 2, v2 Lower Memory, Disabled, None, Only Sketch, 115200 on COM7
```

Fig 5.6: Compiling for the Program



```
IoT_home_weather_monitoring_system | Arduino 1.8.15
File Edit Sketch Tools Help

COM7

*****
Author:
Depart:
Connecting to iot
.....
Description:
WiFi connected
Web server running. Waiting for the ESP IP...
192.168.43.191
Future:
Client disconnected.
(1) Do
Runtime
(2) Ma
(3) Cu
(4) Re
*****
Autoscroll Show timestamp Newline 115200 baud Clear output

// #include <FS.h> //this needs to be first, or it all crashes and burns...
#include <ESP8266WiFi.h>
#include <DNSServer.h>
#include <ESP8266WebServer.h>
#include <WiFiManager.h> // https://github.com/tspas/WiFiManager
#include <ArduinoJson.h> // https://github.com/bblanchon/ArduinoJson
#include "DHT.h"
<

Executable segment sizes:
ICACHE : 32768      - flash instruction cache
IRAM   : 254484     - code in flash (default or ICACHE_FLASH_ATTR)
IRAM   : 27341 / 32768 - code in IRAM (IRAM_ATTR, ISRs...)
DATA   : 1504 )     - initialized variables (global, static) in RAM/HEAP
RODATA : 1320 ) / 81920 - constants (global, static) in RAM/HEAP
BSS    : 25824 )     - zeroed variables (global, static) in RAM/HEAP
Sketch uses 284649 bytes (27%) of program storage space. Maximum is 1044464 bytes.
Global variables use 28648 bytes (34%) of dynamic memory, leaving 53272 bytes for local variables. Maximum is 81920 bytes.

80 MHz, Flash, Disabled (new aborts on com). Disabled. All SSL cipher (most compatible). 32KB cache + 32KB IRAM (balanced). Use pgm_read macros for IRAM/PROGMEM. 4MB (FS:2MB OTA~1019KB), 2, v2 Lower Memory, Disabled, None, Only Sketch, 115200 on COM7
```

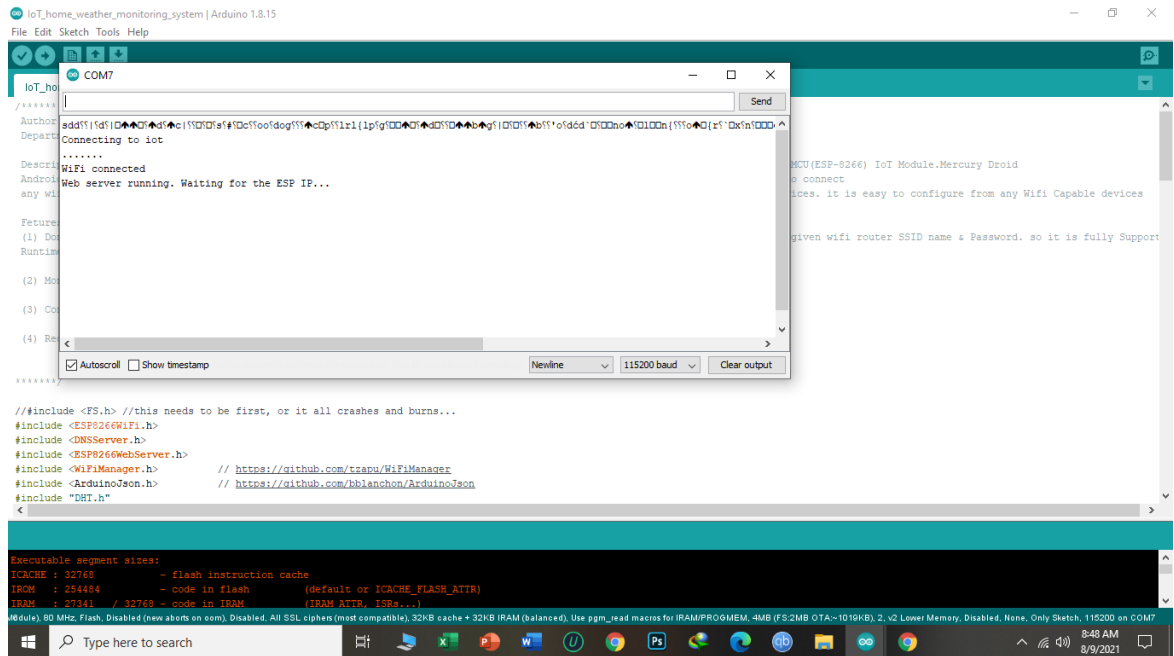


Fig 5.7: IP generation for the system

Now we can place this IP to any weather monitoring website to observe the weather condition. But since we want to observe those entities in our mobile app we will place there.

## 5.4 User interface of the Android App

Android app contains temperatures in Fahrenheit and Kelvin. It also shows IP address and Threshold value field for setting up. The humidity and hit index field is also there.

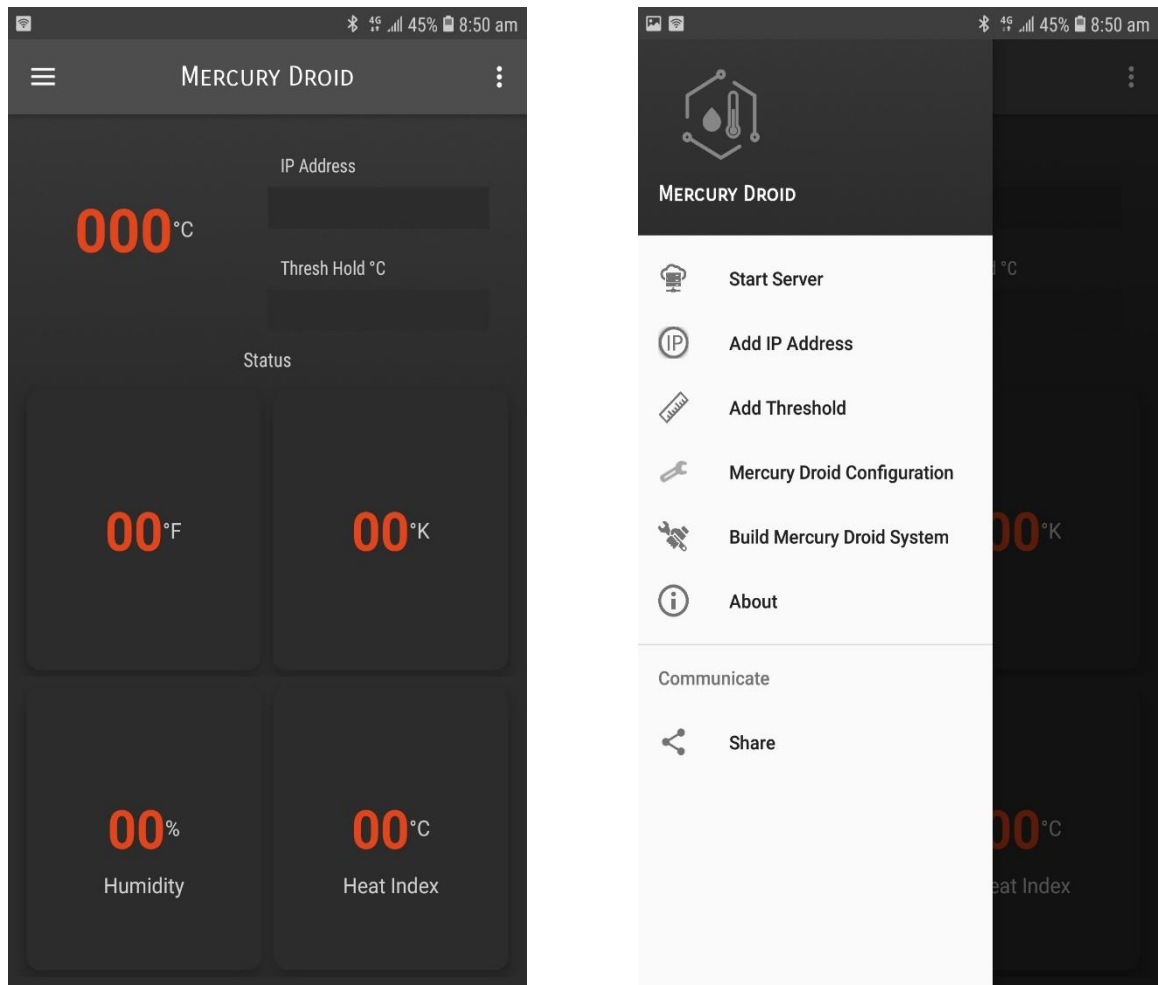


Fig 5.8: User interface of Android App

### 5.4.1 Weather entity observation

From the serial monitor of the Arduino IDE we got the IP address 192.168.43.191 we placed the IP to the android App by touching add IP address. When we touch start server all the weather information shows as following

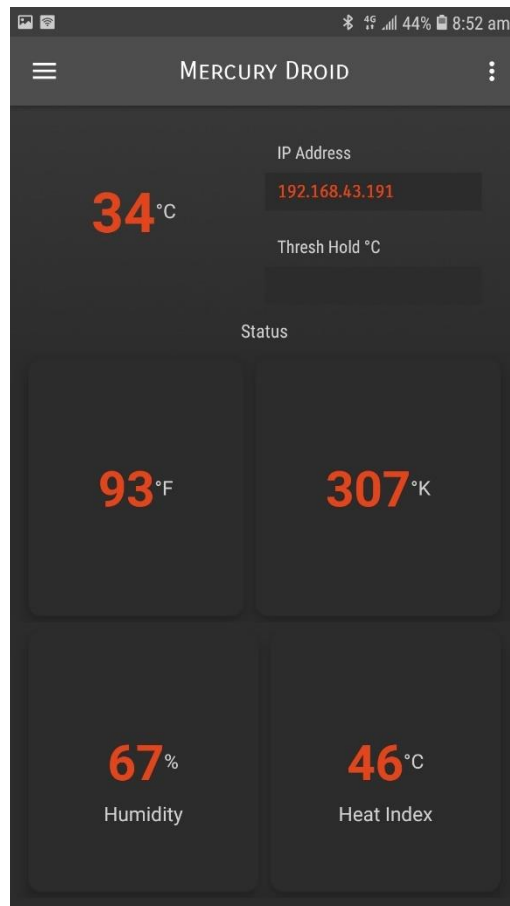


Fig 5.9: Weather information of my home

### 5.4.2 Threshold value setting and alert system

Now to observe alert system we can use two methods. One is to make our home so hot that will increase the temperature more than the room temperature. We can do it by firing something in front of DHT-11 sensor or simply we can reduce the threshold value for testing purpose. Since firing in home condition can make unwanted phenomena that why we choose to reduce the threshold value below the current temperature. For testing purpose, we set the threshold value at 30 and its creating continuous beep sound with alert message shown in given screenshot.

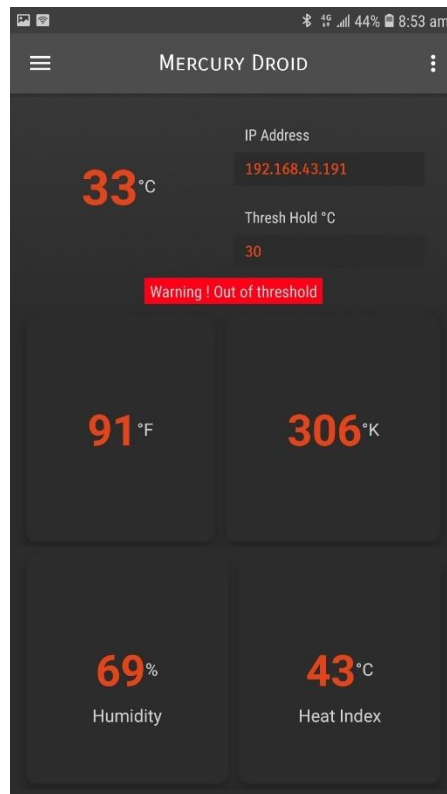


Fig 5.10: warning for out of threshold

## 5.5 RESULTS

I compared this system to standard weather station parameters and found that all the weather information is accurate. This IoT based weather monitoring system runs successfully with small bugs which can be easily resolved.

## 5.6 Limitations

1. Due to the generation of Dynamic IP for every device we have to configure and set up for each device for weather monitoring.
2. User Interface of the Android App is too simple.
3. Its only useful for small area.
4. We have to change user id and password of our mobile device according to the code or vice versa.

## **Chapter 6**

### **Conclusion and future work**

## 6.1 CONCLUSION

The system that has been built can be distinguished from the other weather monitoring system. Heat index is a cool feature in this system that is not present in most of the existing system. In fact, most of us don't even know the factors of heat index. Although this is just a prototype the basic idea behind its working principal can be used in Big IoT projects. This system is very low budget and can be used by anyone and anywhere. Thus, the purpose of making **IoT based weather monitoring system with android app** compatibility was fulfilled.

## 6.2 Future Work

In the future, we plan to add more features such as:

1. Raindrop Module
2. Air Pressure
3. Internal OLED setup
4. Eye catching User Interface
5. Web-Cam integration

# References

- [01] “Weather and Climate”. Weather. Retrieved 12<sup>th</sup> June 2021. Available: <https://scied.ucar.edu/learning-zone/how-weather-works/weather>
- [02] Lans P. Rothfusz. The Heat Index "Equation" (or, More Than You Ever Wanted to Know About Heat Index) NWS Southern Region Headquarters, Fort Worth, TX
- [03] “What is heat index” National Weather Service. Retrieved 28<sup>th</sup> June 2021. Available: <https://www.weather.gov/ama/heatindex>
- [04] “Relative Humidity”. Humidity and Health. Retrieved 13<sup>th</sup> June 2021 Available: <https://en.wikipedia.org/wiki/Humidity#Health>
- [05] "Heat Index". Weather Parameters. Retrieved 14<sup>th</sup> June 2021. Available: <https://www.weather.gov/ama/heatindex>
- [06] “Internet of Things”. Importance of IOT. Retrieved 16<sup>th</sup> June 2021. Available: <https://www.oracle.com/internet-of-things/what-is-iot/>
- [07] Zaw Lin Oo, Theint Win Lai, Su Mon Ko. “IoT based Weather Monitoring System Using Firebase Real Time Database with Mobile Application”. “ResearchGate”
- [\(PDF\) IoT based Weather Monitoring System Using Firebase Real Time Database with Mobile Application \(researchgate.net\)](#)
- [08] Anita M. Bhagat, Ashwini G. Thakare<sup>1</sup>, Kajal A. Molke<sup>1</sup>, Neha S. Muneshwar<sup>1</sup>, Prof. V. Choudhary. “IOT Based Weather Monitoring and Reporting System Project”. International Journal of Trend in Scientific Research and Development (IJTSRD). Volume: 3 | Issue: 3 | Mar-Apr 2019  
Available Online: [www.ijtsrd.com](http://www.ijtsrd.com)



[09] Nisha Gahlot, Varsha Gundkal, Sonali Kothimbire, Archana. “Zigbee based weather monitoring system”. The International Journal Of Engineering And Science (IJES) || Volume || 4 || Issue || 4 || Pages || PP.61-66|| 2015 || ISSN (e): 2319 – 1813 ISSN (p): 2319 – 1805

[10] Mohit Tiwari, Deepak Narang, Priya Goel. WEATHER MONITORING SYSTEM USING IOT AND CLOUD COMPUTING.  
[\(PDF\) WEATHER MONITORING SYSTEM USING IOT AND CLOUD COMPUTING \(researchgate.net\)](#)

[11] “Arduino IDE” Wikipedia. Retrieved 1<sup>st</sup> July 2021. Available:  
[https://en.wikipedia.org/wiki/Arduino\\_IDE](https://en.wikipedia.org/wiki/Arduino_IDE)

[12] “Android Studio” TechTarget. Retrieved 1<sup>st</sup> July 2021. Available:  
<https://searchmobilecomputing.techtarget.com/definition/Android-Studio>

[13] “ESP8266 Node MCU” Component101. Retrieved 1<sup>st</sup> July 2021.  
Available:  
<https://components101.com/development-boards/nodemcu-esp8266-pinout-features-and-datasheet>

[14] “Bread Board” Wikipedia. Retrieved 14<sup>th</sup> April 2021. Available:  
<https://en.wikipedia.org/wiki/Breadboard>

## Appendix-A:

```
#include <ESP8266WiFi.h>
#include <DNSServer.h>
#include <ESP8266WebServer.h>
#include <WiFiManager.h>
#include <ArduinoJson.h>
#include "DHT.h"

// Replace with your network details
const char* ssid = "iot";
const char* password = "iot123456";

WiFiServer server(80);

#define DHTPIN D5
#define DHTTYPE DHT11
//
//IPAddress ip(192, 168, 0, 107); //set static ip
//IPAddress gateway(192, 168, 0, 1); //set gateway
//IPAddress subnet(255, 255, 255, 0); //set subnet

//For Measuring Heat Index
#define c1 (-42.379)
#define c2 (2.04901523)
#define c3 (10.14333127)
#define c4 (-0.22475541)
#define c5 (-0.00683783)
#define c6 (-0.05481717)
#define c7 (0.00122874)
#define c8 (0.00085282)
#define c9 (-0.00000199)
```

```

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(115200);
  delay(10);

  dht.begin();

  // Connecting to WiFi network
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");

  // Starting the web server
  server.begin();
  Serial.println("Web server running. Waiting for the ESP IP...");
  delay(10000);

  // Printing the ESP IP address
  Serial.println(WiFi.localIP());
}

void loop(){

```

```

WiFiClient client = server.available();

// here
// h = Humidity
// t = celsius
// k = kelvin
// f = fahrenheit

// Read the first line of the request
float h = dht.readHumidity();
// Read temperature as Celsius (the default)
float t = dht.readTemperature();
// Conversion of Celsius to Kelvin
float k = t + 273 ;
// Conversion of Celsius to fahrenheit
float f = (t*9)/5 + 32;
//Heat Index
float heatIndex, heatIndexCelsius;
heatIndex
c1+c2*(f)+c3*(h)+c4*(f)*(h)+c5*(pow(f,2))+c6*(pow(h,2))+c7*(pow(f,
2))* (h)+c8*(f)*(pow(h, 2))+c9*(pow(f, 2))*(pow(h, 2));
heatIndexCelsius = (((heatIndex)-32)*5)/9);

String req = client.readStringUntil('\r');
Serial.println(req);
client.flush();

if (req.indexOf("/data") != -1){
  client.flush();
  client.println("HTTP/1.1 200 OK");          // This tells the browser that the request
to provide data was accepted
  client.println("Access-Control-Allow-Origin: *"); //Tells the browser it has
accepted its request for data from a different domain (origin).
  client.println("Content-Type: application/json;charset=utf-8"); //Lets the browser
know that the data will be in a JSON format

```

```

    client.println("Server: Arduino");          // The data is coming from an Arduino
Web Server (this line can be omitted)
    client.println("Connection: close");        // Will close the connection at the end of
data transmission.

    client.println();                          // need to include this blank line - it tells the
browser that it has reached the end of the Server reponse header.

        // This is the starting bracket of the JSON data
    client.print("{ \"temperature\": \"");
    client.print(t,0);
    client.print("\", \"kelvin\": \"");
    client.print(k,0);
    client.print("\", \"fahrenheit\": \"");
    client.print(f,0);
    client.print("\", \"heatindex\": \"");
    client.print(heatIndexCelsius,0);
    client.print("\", \"Humidity\": \"");
    client.print(h,0);
    client.print("\"}");

}
else {
    client.flush();

    client.println("HTTP/1.1 200 OK");          // This tells the browser that the request
to provide data was accepted

    client.println("Access-Control-Allow-Origin: *"); //Tells the browser it has
accepted its request for data from a different domain (origin).

    client.println("Content-Type: application/json;charset=utf-8"); //Lets the browser
know that the data will be in a JSON format

    client.println("Server: Arduino");          // The data is coming from an Arduino
Web Server (this line can be omitted)

    client.println("Connection: close");        // Will close the connection at the end of
data transmission.

    client.println();                          // You need to include this blank line - it tells the
browser that it has reached the end of the Server reponse header.

```

```
        // This is the starting bracket of the JSON data
client.print("{\"Response\": ");
client.print("Invalid");
client.print("}");
        // This is the final bracket of the JSON data
    }

    delay(1);
    Serial.println("Client disconnected.");
    Serial.println("");
}
```

