

ABSTRACT

The Internet of things (IOT) is the mostly used technology for this type of systems. It makes future realistic things for daily life and it is also used for making advanced level things for better world. The Weather Monitoring and Reporting System project is helpful to get live reporting condition of weather. And it show the conditions like humidity, temperature, and rain sensor to monitor weather and provide live reporting statistics, that machine always sends the sensor data to the microcontroller to transmit the data on online web servers through WIFI connections. This system always used to monitorize the changing the climate in different conditions. It should always display the weather parameters.

CONTENTS

CHAPTER 1

1.INTRODUCTION TO THE PROJECT

- 1.1.Introduction
- 1.2.Role of IOT in Weather Reporting System
- 1.3.Necessity of IOT Based Weather Reporting System
- 1.4.Existing System
- 1.5.Limitations of Existing System

CHAPTER 2

2.APPROACH ANALYSIS

- 2.1.Problems of Traditional Method
- 2.2.Approach
- 2.3.Scope of Testing and Evaluation

CHAPTER 3

3.PROPOSED SYSTEM

- 3.1.The Proposed System
- 3.2.Features of Proposed System
- 3.3.Advantages of Proposed System
- 3.4.Proposed Hardware Architecture

CHAPTER 4

4.SYSTEM DESIGN

- 4.1.Design Architecture
- 4.2.Overview of Functional Requirements
- 4.3.Circuit Diagram
- 4.4.Block Diagram

CHAPTER 5

5.HARDWARE DESCRIPTION

- 5.1.List of Hardware Components
- 5.2.Details of Hardware
 - 5.2.1.ESP 32
 - 5.2.2.Featured Solutions
 - 5.2.3.ESP32 Peripheral Features
 - 5.2.4.Applications
- 5.3.DHT 11 Sensor
 - 5.3.1.Pin Configuration of DHT 11
 - 5.3.2.Specifications of DHT 11
- 5.4.Rain Sensor
 - 5.4.1.How Rain Sensor Works?
 - 5.4.2.Hardware Overview
 - 5.4.3.Pin Configuration
- 5.5.LDR Sensor
 - 5.5.1.Applications
- 5.6.16X2 LCD
- 5.7.Buzzer
- 5.8.Jumper Wires
- 5.9.Regulated Power Supply(RPS)

CHAPTER 6

6.SOFTWARE DESCRIPTION

6.1.Software Introduction

CHAPTER 7

7.AIGORITHM AND SOURCE CODE

7.1.Flow Chart of the System

7.2.Source Code

7.3.Implementation of Hardware

CHAPTER 8

8.RESULTS

CHAPTER 9

9.APPLICATIONS AND ADVANTAGES

9.1.Applications

9.2.Advantages

CHAPTER 10

10.CONCLUSION

LIST OF FIGURES

List of Figures	Description of figures	Page no.
Fig.4.3	Circuit Representation of weather reporting system using IOT	09
Fig.4.4	Block Diagram using ESP-32	10
Fig.5.2.1	ESP-32	11
Fig.5.3	DHT-11	15
Fig.5.4.1	Rain Sensor	16
Fig.5.4.2	Rain Sensor Sensing Pad	17
Fig.5.4.3	Electronic Module	17
Fig.5.4.4	Connections between Sensing Pad and Electronic Module	18
Fig.5.5.1	Semiconductor of LDR	19
Fig.5.5.2	LDR Sensor	19
Fig.5.6	Liquid Crystal Display	20
Fig.5.7	Buzzer	21
Fig.5.8	Jumper wires	22
Fig.5.9	Regulated Power Supply	22
Fig.6.1	Opening Arduino-Nightly-Windows.zip	23
Fig.6.2	Launch Arduino IDE	24
Fig.6.3	Create a New Project	24
Fig.6.4	Open an Existing Project Example	25
Fig.6.5	Select your Serial Port	25
Fig.6.6	Function of Each Symbol appearing in the Arduino IDE Toolbar	26
Fig.6.7	Bare Minimum code	27
Fig.7.1	Work Flow	28
Fig.7.3	Hardware	35
Fig.8.1	Result Sample	38

CHAPTER-1

INTRODUCTION TO THE PROJECT

1.1 INTRODUCTION:

Here we introduce a smart weather reporting system over the Internet. Our introduced system allows for weather parameter reporting over the Internet. It allows the people to directly check the weather states online without the need of a weather forecasting agency. System uses temperature, humidity as well as rain with humidity sensor to monitor weather and provide live reporting of the weather statistics. The system constantly monitors temperature using temperature sensor, humidity using humidity sensor and also for rain. Weather monitoring system deals with detecting and gathering various weather parameters at different locations which can be analysed or used for weather forecasting. The aim of this system is achieved by technologies such as Internet of Things (IOT) and Cloud. The idea of internet of things is to connect a device to the internet and to other required connected devices. Using Internet, the information from the IOT device can easily be transferred to the cloud and then from the cloud to the end user. Weather Monitoring is an essential practical implementation of the concept of Internet of Things, it involves sensing and recording various weather parameters and using them for alerts, sending notifications, adjusting appliances accordingly and also for long term analysis. Also, we will try to identify and display trends in parameters using graphical representation. The devices used for this purpose are used to collect, organize and display information. It is expected that the internet of things is going to transform the world by monitoring and controlling the phenomenon of environment by using sensors/devices which are able to capture, process and transmit weather parameters. Cloud is availability of computer system resources like data storage, computing power without direct active management of user. The data captured is transmitted to the cloud so that the data could be further displayed. Besides this, the system consists of components which is a microcontroller, a USB connection and everything used to support microcontroller; DHT11 is Temperature and humidity sensor which is used for detecting these mentioned parameters; WIFI module is used to convert the data collected from the sensors and then send it to the web server. So, in this way weather conditions of any location can be monitored from any remote location in the world. The system constantly transmits this data to the micro controller which now processes this data and keeps on transmitting it to the online web server over a Wi-Fi connection. This data is live updated to be viewed on the online server system. Also, system allows user to set alerts for particular instances.

We connect three sensors 1) LDR Sensor 2) DHT11 and 3) Rain Sensor to NodeMCU. By using these three sensors, we can collect the required weather data for monitoring purpose. This pooled data is stream over the Internet to display it or read it from anywhere. After the successfully programmed hardware, the NodeMCU get one IP address. We can browse this IP address from WEB browser or any webpage.so we display the required live data which fetched

by sensors in beautiful Graphical User Interface format. The weather parameters that we monitor are Temperature, Humidity and Rain. Also, you can check whether data through anywhere using Internet as we hosted this server publicly. We developed an android application for easy access to our weather monitoring system.

This data is live updated to be viewed on the online server system. Also, system allows user to set alerts for instances. In today's world many pollution monitoring systems are designed by different environmental parameters. Existing system model is presented IOT based Weather monitoring and reporting system where you can collect, process, analyse, and present your measured data on web server. Wireless sensor network management model consists of end device, router, gateway node and management monitoring center.

1.2 ROLE OF IOT IN WEATHER REPORTING SYSTEM:

Weather forecasting is an essential and practical skill that significantly impacts various aspects of human life. With the advent of the Internet of Things (IoT) concept, the field of meteorology has witnessed remarkable advancements, transforming the way meteorological characteristics are sensed, recorded, and utilized for real-time alerts, appliance adjustments, long-term analysis, and notifications. In the IoT-based weather reporting system, a range of specialized instruments and sensors are employed to capture and process critical weather data. These sensors are designed to monitor various aspects of weather and climate, such as temperature, humidity, wind speed, wetness, light intensity, UV radiation, and even airborne carbon monoxide levels. The collected data is then transmitted to the cloud, where it is further processed, analyzed, and presented to users in the form of graphical statistics. Furthermore, the vast amount of data collected over time by the IoT-based weather reporting system offers valuable insights for long-term weather analysis. This accumulated data can be utilized to study weather trends, climate patterns, and potential impacts of climate change on specific regions.

By facilitating in-depth analysis, this system contributes to the development of more accurate and reliable weather forecasting models, benefiting various sectors, including agriculture, transportation, tourism, and disaster management. In conclusion, the integration of IoT in weather reporting systems has revolutionized the way meteorological data is sensed, processed, and disseminated. Through the utilization of specialized sensors, cloud-based data storage, and user-friendly applications, this technology enables global access to real time weather information, empowering individuals and communities to make informed decisions and respond effectively to weather changes.

1.3 WHY IS AN IOT BASED WEATHER REPORTING SYSTEM IS NECESSARY?

An IoT-based weather monitoring system is indispensable due to its ability to provide simple access from anywhere in the world to real-time local weather monitoring. Moreover, the system allows for both short- and long-term archiving of weather and environmental data, enabling researchers to study changes in weather patterns and gain insights into how locally produced climate change has impacted weather over time. This vast repository of historical weather data becomes a valuable resource for climate research and helps us comprehend the evolving

climate and its potential implications. Additionally, the ease of deploying an IoT-based weather monitoring system for monitoring local atmospheric variables and microclimates is a significant advantage. This simplicity empowers various sectors, such as agriculture and disaster management, to harness accurate weather forecasting and prediction, facilitating proactive planning and resilience-building measures. Ultimately, an IoT-based weather monitoring system is necessary to foster global awareness, enhance preparedness, and ensure a sustainable future in the face of weather-related challenges.

1.4 EXISTING SYSTEM:

The existing weather monitoring systems typically involve an array of sophisticated sensors, satellite imagery, and data processing units, enabling comprehensive data collection across large geographic areas. These systems can monitor multiple weather parameters, including temperature, humidity, rainfall, wind speed, and air quality, among others. The collected data is often transmitted to central weather monitoring centers, where it is processed, analyzed, and disseminated to the public and relevant authorities for decision-making purposes. While these high-end devices serve critical roles in weather forecasting and disaster management on a large scale, their implementation for monitoring weather in a small region raises several challenges. Firstly, the initial setup and installation costs of such systems can be prohibitively high for smaller communities or organizations with limited resources. Secondly, the maintenance and operational expenses required to sustain the continuous functionality of these complex systems may not be economically viable for a small region. As a result, the need for an alternative and practical weather monitoring system for small regions becomes apparent. The proposed research aims to address this gap by developing an IoT-based weather monitoring system that is cost-effective, easy to deploy, and tailored to the specific needs of smaller regions.

1.5 LIMITATIONS OF EXISTING SYSTEM:

- It is impossible to implement such a system in a tiny area.
- It's a web-based program.
- Such systems have a very high maintenance cost for a small region.

CHAPTER -2

APPROACH AND ANALYSIS

2.1 PROBLEMS OF TRADITIONAL METHOD

- Traditional methods often rely on manual observations and measurements, which may result in limited accuracy compared to automated sensor systems. Human errors and subjective interpretations can affect the precision of the data.
- Manual data collection and reporting processes are time-consuming. The delay in obtaining and disseminating weather information can impact its relevance for decision-making, especially in rapidly changing weather conditions.
- Manual reporting systems may not provide real-time updates. This delay can be critical, especially during severe weather events where timely information is essential for public safety.
- Expanding the coverage of traditional reporting systems can be challenging and costly. Scaling up to cover larger geographical areas may require substantial investments in infrastructure and personnel.
- Traditional reporting may focus on a limited set of weather parameters. Modern weather reporting systems can integrate a broader range of sensors, providing more comprehensive and detailed information.
- Human errors, misinterpretations, and inconsistencies in manual reporting may lead to inaccuracies. Automation reduces these risks and enhances data reliability.
- Managing large volumes of historical data from manual reporting can be cumbersome. Digital systems and databases associated with IoT solutions offer more efficient storage and retrieval mechanisms.
- Communicating weather information to the public may be less efficient using traditional methods. Modern systems allow for automated alerts, notifications, and widespread dissemination through various channels.

2.2 APPROACH

IoT technologies enhances the accuracy, speed, and efficiency of weather reporting, addressing many of the limitations associated with manual processes. The transition to more automated and data-driven approaches has become crucial for improving the effectiveness of weather monitoring and forecasting systems.

- Determine communication protocols for data transmission between IoT devices and the central server or cloud platform.
- Set up a central server or leverage a cloud platform for data aggregation, storage, and processing.

- Choose a scalable solution to accommodate growing data volumes.
- Develop firmware for IoT devices to interface with sensors and transmit data.
- Implement algorithms for processing raw sensor data on the central server or cloud.
- Design a user-friendly web interface or mobile application for end-users.
- Include features like real-time updates, historical data visualization, and customizable alerts.
- Integrate an alerting system to notify users of critical weather conditions.
- Define thresholds for alerts and choose reliable notification mechanisms.
- Monitor power consumption and optimize to extend device lifespan.
- Conduct thorough testing at each stage of development, including unit testing, integration testing, and system testing.
- Simulate various weather scenarios to validate the system's accuracy and responsiveness.

2.3 SCOPE OF TESTING AND EVALUATION

- All of the parts of cart will be checked and evaluated that either they are working mechanically correct.
- The parts of this project will be ESP 32,DHT11,LDR,Rain sensor,LCD,Buzzer and power supply from RPS.
- The tolerance value for all the connections in assembly will be highest accuracy, will be tried to achieve so that the project can function according to the objective.
- Schedule risk
- Cost Risk
- Analysis method maturity
- Equipment and material availability

CHAPTER-3

PROPOSED SYSTEM

3.1 THE PROPOSED SYSTEM

Our project proposes an innovative weather monitoring and analysis system that utilizes multiple sensors to measure various weather and environmental variables, such as temperature and humidity. The core of the system is the ESP32 microcontroller, which processes the sensor readings and efficiently stores them in a Google Firebase database for further analysis. To provide users with real-time updates, the collected readings are also displayed on an onboard LCD screen for convenient viewing.

By evaluating these sensor readings, we can determine the weather features specific to any given location and monitor the weather trend over time. These measured factors, which vary from place to region, play a pivotal role in accurate weather forecasting. To aid in weather analysis, the system tracks and records these essential specifications over time, facilitating the creation of weather charts for specific regions. These charts offer insights into past weather forecasts' performance, enhancing our understanding of regional weather patterns. The ESP32's serial output enables real-time access to the readings read from the sensors, facilitating continuous monitoring and evaluation. Overall, our proposed system presents a robust and user-friendly approach to weather monitoring, analysis, and forecasting.

3.2 FEATURES OF PROPOSED SYSTEM

In IOT enabled weather monitoring system project, ESP32 measures 3 weather parameters using respective 3 sensors. These sensors are a DHT11, rain sensor, and LDR. These 3 sensors are directly connected to ESP32 since it reduces the complexity of wiring. It typically involves connecting sensor pins (analog or digital) directly to GPIO (General Purpose Input/Output) pins on the ESP32. For prototyping and small-scale projects, direct connections simplify the development process. It allows for quick testing and validation of sensor functionality. Direct connections enable the microcontroller to have direct control over the sensors. This facilitates customization and optimization of sensor readings based on specific project requirements.

The weather monitoring system gives high accuracy and reliability for weather monitoring and climate changing. It uses the renewable energy source like solar panel for charging the connected battery. Through the web, it access real time weather information and data. This system can be communicated over general packet radio service (GPRS) network. Low maintenance is required for end users. It is capable for storing data and providing it to the users as required.

3.3 ADVANTAGES OF PROPOSED SYSTEM

- Decreased field damaging conditions
- Improved safety and security
- High-quality receiving data
- Less power consumption
- Accuracy is High
- Smart way to monitor Environment
- The low cost and efforts are less in this system

3.4 PROPOSED HARDWARE ARCHITECTURE

The implemented system consists of a microcontroller (ESP32) as a main processing unit for the entire system and all the sensor and devices can be connected with the microcontroller. The sensors can be operated by the microcontroller to retrieve the data from them and it processes the analysis with the sensor data and updates it to the internet through Wi-Fi module connected with via webpage then we can measure temperature, humidity, day/night mode and rain fall.

CHAPTER-4

SYSTEM DESIGN

4.1 DESIGN ARCHITECTURE

1. **Data Acquisition:**The system should be able to collect data on key meteorological parameters, including temperature, humidity, barometric pressure, wind speed, wind direction, and rainfall.
2. **Communication:**The system must establish reliable communication between IoT devices and the central server or cloud platform.
3. **Data Processing and Analytics:**Implement algorithms for data analysis, trend identification, and potentially use machine learning for predictive analytics to enhance weather forecasting.
4. **User Interface:**Develop a web or mobile application that allows users to view real-time weather updates, historical data, and forecasts. Include features like interactive maps and customizable alerts.
5. **Alerting System:**Implement an alerting mechanism that notifies users of severe weather conditions, ensuring public safety and facilitating proactive decision-making.
6. **Testing and Quality Assurance:**Establish a testing plan covering unit testing, integration testing, and system testing to identify and address any issues.

4.2 OVERVIEW OF FUNCTIONAL REQUIREMENTS

1. **Temperature Sensing and Averaging:** The system must include a temperature sensor (DHT11) to measure the environmental temperature accurately. The system will perform two consecutive temperature readings and calculate their average to ensure data accuracy.
2. **Autonomous Temperature Regulation:** The system should have the capability of autonomously regulating the temperature in the target region. If the measured temperature is lower than the desired level, the system will inject hot air into the area to moderate the temperature. Conversely, if the temperature exceeds the set threshold, the system will blow cold air to lower it.
3. **Safety Measures:** The system should include safety features to prevent any potential hazards related to temperature regulation. For instance, it should have temperature limits for hot air injection and cold air blowing to avoid extreme conditions.

By fulfilling these functional requirements, the proposed system will be able to effectively measure temperature ,maintain temperature levels within the desired range, demonstrate the autonomous capabilities of its individual components, ensure user safety, and provide an

intuitive user interface for seamless interaction. The successful implementation of these functionalities will contribute to the advancement of temperature regulation systems, potentially finding applications in various real-world scenarios.

4.3 CIRCUIT DIAGRAM

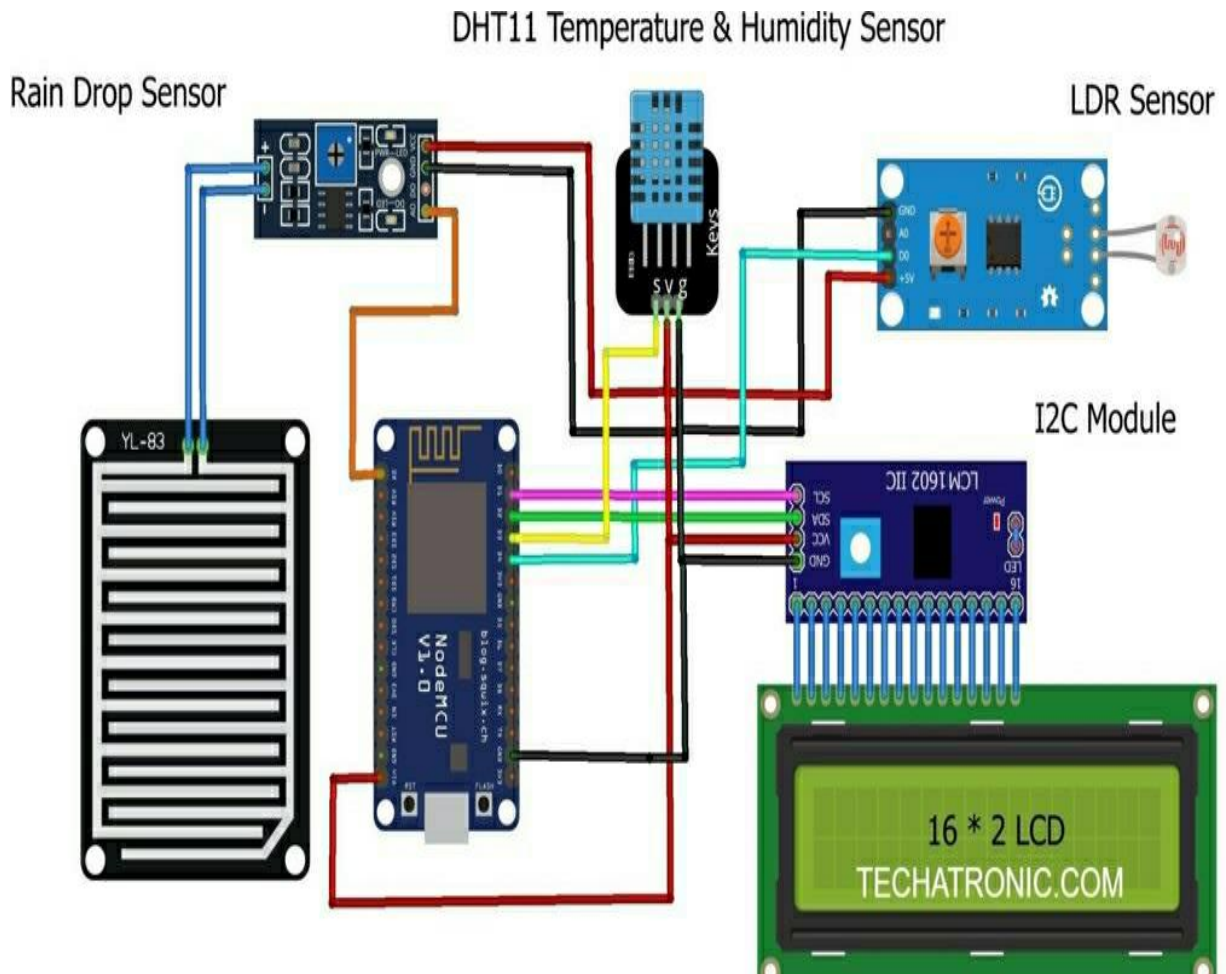


Fig 4.3 : Circuit representation of Weather Reporting System using IOT

4.4 BLOCK DIAGRAM

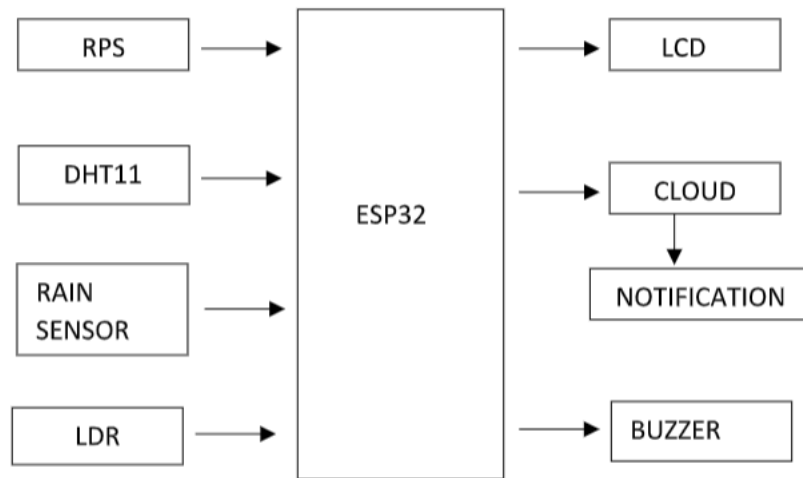


Fig 4.4 : Block Diagram using ESP32

CHAPTER-5

HARDWARE DESCRIPTION

5.1 LIST OF HARDWARE COMPONENTS

1. ESP 32
2. DHT11 Sensor
3. Rain Sensor
4. LDR Sensor
5. 16 X 2 LCD Display
6. Buzzer
7. RPS
8. Jumper Wires

5.2 DETAILS OF HARDWARE

5.2.1 ESP 32

ESP32 is a single 2.4 GHz Wi-Fi-and-Bluetooth combo chip designed with the TSMC ultra-low-power 40 nm technology. It is designed to achieve the best power and RF performance, showing robustness, versatility and reliability in a wide variety of applications and power scenarios. The ESP32 is a series of low-cost, low-power system-on-chip (SoC) microcontrollers with integrated Wi-Fi and Bluetooth capabilities. ESP32 is a single 2.4 GHz Wi-Fi-and-Bluetooth combo chip designed with the TSMC ultra-low-power 40 nm technology. Developed by Espressif Systems, it features a dual-core processor, Wi-Fi and Bluetooth connectivity, and a wide range of peripheral interfaces. The ESP32 is an improved version of the earlier ESP8266 microcontroller and offers more features and capabilities.



Fig 5.2.1 : ESP32

It is a powerful and versatile microcontroller that belongs to the ESP (Espressif) family of chips. It is known for its integrated Wi-Fi and Bluetooth capabilities, making it a popular choice for a wide range of IoT (Internet of Things) applications. Here's a detailed description of the ESP32:

1. Microcontroller:

- The ESP32 is powered by a Tensilica Xtensa LX6 microcontroller, which is a dual-core processor running at 160 or 240 MHz. It has a 32-bit RISC architecture.

2. Wireless Connectivity:

- **Wi-Fi:** The ESP32 has built-in 802.11 b/g/n Wi-Fi support, allowing it to connect to local wireless networks. It supports various Wi-Fi security protocols, including WPA, WPA2, and WEP.
- **Bluetooth:** It features integrated Bluetooth 4.2 and BLE (Bluetooth Low Energy), making it suitable for applications that require short-range wireless communication.

3. Peripheral Interfaces:

- **GPIO (General Purpose Input/Output):** The ESP32 has a large number of GPIO pins, which can be configured for various purposes such as digital input or output, PWM (Pulse Width Modulation), and more.
- **SPI (Serial Peripheral Interface):** Used for interfacing with devices like sensors, displays, and other microcontrollers.
- **I2C (Inter-Integrated Circuit):** Another communication protocol for connecting to various peripherals and sensors.
- **UART (Universal Asynchronous Receiver/Transmitter):** For serial communication with other devices.

4. Analog-to-Digital Converter (ADC):

- The ESP32 has a 12-bit ADC that allows it to read analog sensor values.

5. Memory:

- It typically comes with a range of Flash memory options, allowing developers to choose based on their storage needs.

6.Power Management:

- The ESP32 is designed to operate with low power consumption, making it suitable for battery-powered applications.

7.Development Environment:

- The ESP32 can be programmed using the Arduino IDE, MicroPython, and the Espressif IoT Development Framework (ESP-IDF), providing flexibility for developers.

8.Community Support:

- The ESP32 has a large and active community of developers. This community support includes forums, documentation, and a variety of libraries and examples to help developers get started with their projects.

9. Applications:

- Due to its versatile features, the ESP32 is used in a wide range of applications, including IoT devices, home automation systems, wearable electronics, robotics, and more.

10.ESP32-CAM:

- There is a variant called ESP32-CAM, which comes with a camera module, making it particularly suitable for projects involving image and video processing.

The ESP32's combination of wireless connectivity, a powerful microcontroller, and a range of peripheral interfaces makes it a popular choice for a diverse set of IoT applications. Its open-source nature and extensive community support contribute to its widespread adoption in the maker and development communities.

5.2.2 FEATURED SOLUTIONS

1. Ultra Low Power Solution: ESP32 is designed for mobile, wearable electronics, and Internet-of-Things (IoT) applications. It features all the state-of-the-art characteristics of low-power chips, including fine-grained clock gating, multiple power modes, and dynamic power scaling. For instance, in a low-power IoT sensor hub application scenario, ESP32 is woken up periodically and only when a specified condition is detected. Low-duty cycle is used to minimize the amount of energy that the chip expends. The output of the power amplifier is also adjustable, thus contributing to an optimal trade-off between communication range, data rate and power consumption

2. Complete Integration Solution: ESP32 is a highly-integrated solution for Wi-Fi-and-Bluetooth IoT applications, with around 20 external components. ESP32 integrates an antenna switch, RF balun, power amplifier, low-noise receive amplifier, filters, and power management modules. As such, the entire solution occupies minimal Printed Circuit Board (PCB) area. ESP32 uses CMOS for single-chip fully-integrated radio and baseband, while also integrating advanced calibration circuitries that allow the solution to remove external circuit imperfections or adjust to changes in external conditions. As such, the mass production of ESP32 solutions does not require expensive and specialized Wi-Fi testing equipment uses CMOS for single-chip fully-integrated radio and baseband, while also integrating advanced calibration circuitries that allow the solution to remove external circuit imperfections or adjust to changes in external conditions.

As such, the mass production of ESP32 solutions does not require expensive and specialized Wi-Fi testing equipment.

5.2.3 ESP32 PERIPHERALS FEATURES

- 18 Analog-to-Digital Converter (ADC) channels
- 10 Capacitive sensing GPIOs
- UART interfaces
- SPI interfaces
- 2 I2C interfaces
- 16 PWM output channels
- 2 Digital-to-Analog Converters (DAC)
- 2 I2S interfaces

5.2.4 APPLICATIONS

- Home Automation
- Smart Agriculture
- Live streaming Devices
- Health Care Applications
- Wi-Fi enabled Toys

- Wearable Electronics
- Mesh Network

5.3 DHT 11 SENSOR

The DHT-11 Digital Temperature and Humidity Sensor is a basic, ultra-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). The DHT11 sensor consists of a capacitive humidity sensor and a thermistor to measure temperature and humidity. It is a relatively simple sensor to use, and it communicates with microcontrollers or other devices using a digital signal. The DHT11 operates on a single-wire communication protocol, allowing it to transmit temperature and humidity data to microcontrollers or single-board computers with minimal wiring. The sensor contains a calibrated digital signal output with high precision and stability.

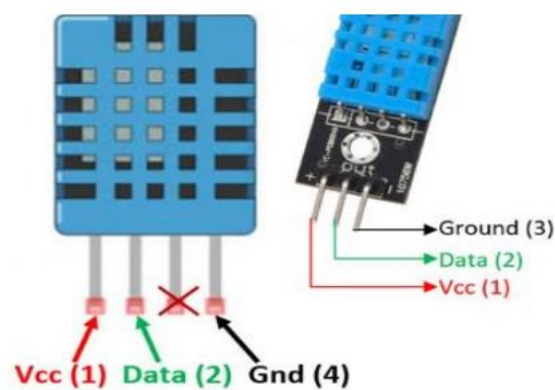


Fig 5.3 : DHT 11(Temperature and Humidity Sensor)

5.3.1 PIN CONFIGURATION OF DHT11

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

5.3.2 SPECIFICATIONS OF DHT11

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

5.4 RAIN SENSOR

With the weather being as unpredictable as ever, it's easy to leave your skylights open, only for it to suddenly start raining, leaving the interior below at risk. With this rain sensor, however, you can stop this from happening. You can use this sensor to monitor rain or slushy snow/hail and send closure requests to electronic shutters, windows, awnings or skylights whenever the rain is detected.

5.4.1 HOW RAIN SENSOR WORKS?

The working of the rain sensor is pretty straightforward. The sensing pad with series of exposed copper traces, together acts as a variable resistor (just like a potentiometer) whose resistance varies according to the amount of water on its surface.

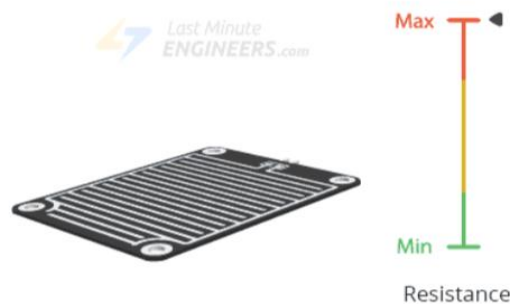


Fig 5.4.1 : Rain Sensor

This resistance is inversely proportional to the amount of water:

- The more water on the surface means better conductivity and will result in a lower resistance.
- The less water on the surface means poor conductivity and will result in a higher resistance.
- The sensor produces an output voltage according to the resistance, which by measuring we can determine whether it's raining or not.

5.4.2 HARDWARE OVERVIEW

A typical rain sensor has two components.

The Sensing Pad: The sensor contains a sensing pad with series of exposed copper traces that is placed out in the open, possibly over the roof or where it can be affected by rainfall. Usually, these traces are not connected but are bridged by water.

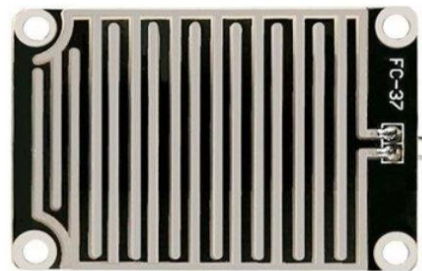


Fig 5.4.2 Sensing Pad

The Module The sensor also contains an electronic module that connects the sensing pad to the Arduino. The module produces an output voltage according to the resistance of the sensing pad and is made available at an Analog Output (AO) pin. The same signal is fed to a LM393 High Precision Comparator to digitize it and is made available at an Digital Output (DO) pin.



Fig 5.4.3 Electronic Module

The module has a built-in potentiometer for sensitivity adjustment of the digital output (DO). You can set a threshold by using a potentiometer; So that when the amount of water exceeds the threshold value, the module will output LOW otherwise HIGH. Rotate the knob clockwise to increase sensitivity and counterclockwise to decrease it.

Apart from this, the module has two LEDs. The Power LED will light up when the module is powered. The Status LED will light up when the digital output goes LOW. Rain Sensor Pin-out the rain sensor is super easy to use and only has 4 pins to connect.

5.4.3 PIN CONFIGURATION

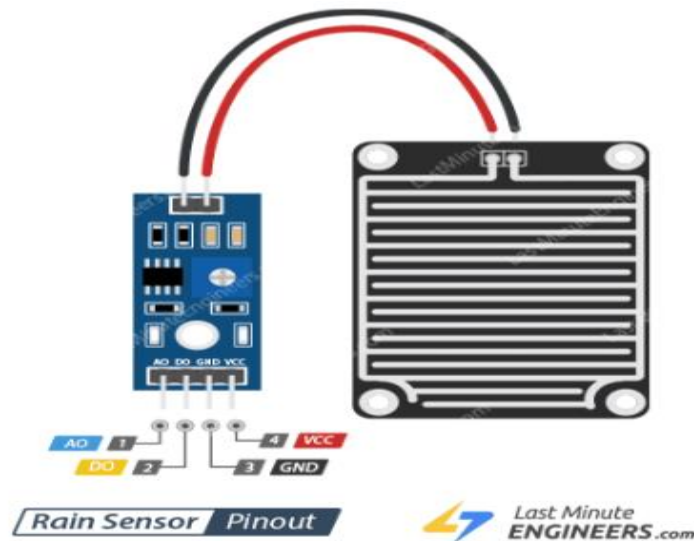


Fig 5.4.4 Connections between sensing pad and electronic module

- **AO (Analog Output)** pin gives us an analog signal between the supply value (5V) to 0V.
- **DO (Digital Output)** pin gives Digital output of internal comparator circuit. You can connect it to any digital pin on an Arduino or directly to a 5V relay or similar device.
- **GND** is a ground connection.
- **VCC** pin supplies power for the sensor. It is recommended to power the sensor with between 3.3V – 5V. Please note that the analog output will vary depending on what voltage is provided for the sensor.

5.5 LDR SENSOR

A photo resistor or Light Dependent Resistor or CdS (Cadmium Sulphide) Cell is a resistor whose resistance decreases with increasing incident light intensity. It can also be referred to as a photoconductor.

A photo resistor is made of a high resistance semiconductor. If light falling on the device is of high enough frequency, photons absorbed by the semiconductor give bound electrons enough energy to jump into the conduction band. The resulting free electron (and its hole partner) conduct electricity, thereby lowering resistance.

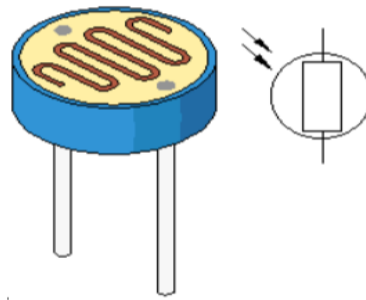


Fig 5.5.1 : Semiconductor of LDR

A photoelectric device can be either intrinsic or extrinsic. An intrinsic semiconductor has its own charge carriers and is not an efficient semiconductor, e.g., silicon. In intrinsic devices the only available electrons are in the valence band, and hence the photon must have enough energy to excite the electron across the entire band gap.

Extrinsic devices have impurities, also called dopants, added whose ground state energy is closer to the conduction band; since the electrons don't have as far to jump, lower energy photons (i.e., longer wavelengths and lower frequencies) are sufficient to trigger the device. If a sample of silicon has some of its atoms replaced by phosphorus atoms (impurities), there will be extra electrons available for conduction. This is an example of an extrinsic semiconductor.

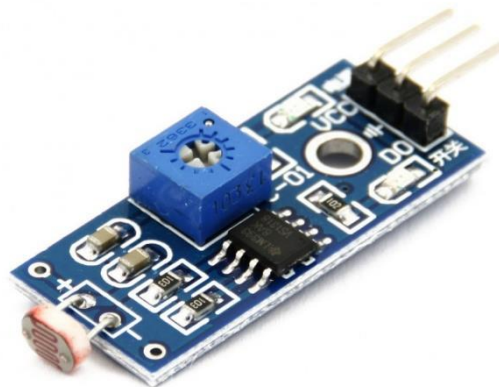


Fig 5.5.2 : LDR Sensor

5.5.1 APPLICATIONS

Photo resistors come in many different types. Inexpensive cadmium sulphide cells can be found in many consumer items such as camera light meters, clock radios, security alarms, street lights and outdoor clocks.

They are also used in some dynamic compressors together with a small incandescent lamp or light emitting diode to control gain reduction.

Lead sulphide and indium antimonite LDRs are used for the mid infrared spectral region. Ge:Cu photoconductors are among the best far-infrared detectors available, and are used for infrared astronomy and infrared spectroscopy.

5.6 16X2 LCD(LIQUID CRYSTAL DISPLAY)

To display interactive messages, we are using LCD Module. We examine an intelligent LCD display of two lines, 16 characters per line that is interfaced to the controllers. The protocol(handshaking) for the display is as shown. Whereas D0 to D7th bit is the Data lines, RS, RW and EN pins are the control pins and remaining pins are +5V, -5V and GND to provide supply. Where RS is the Register Select, RW is the Read Write and EN is the Enable pin.

The display contains two internal byte-wide registers, one for commands (RS=0) and the second for characters to be displayed (RS=1). It also contains a user-programmed RAM area (the character RAM) that can be programmed to generate any desired character that can be formed using a dot matrix. To distinguish between these two data areas, the hex command byte 80 will be used to signify that the display RAM address 00h will be chosen. Port1 is used to furnish the command or data type, and ports 3.2 to 3.4 furnish register select and read/write levels.

The display takes varying amounts of time to accomplish the functions as listed. LCD bit 7 is monitored for logic high (busy) to ensure the display is overwritten. Liquid Crystal Display also called as LCD is very helpful in providing user interface as well as for debugging purpose. These LCDs are very simple to interface with the controller as well as are cost effective.



Fig 5.6 Liquid Crystal Display

The most commonly used ALPHANUMERIC displays are 1x16 (Single Line & 16 characters), 2x16 (Double Line & 16 character per line) & 4x20 (four lines & Twenty characters per line). The LCD requires 3 control lines (RS, R/W & EN) & 8 (or 4) data lines. The number on data lines depends on the mode of operation. If operated in 8-bit mode then 8 data lines + 3 control lines i.e. total 11 lines are required. And if operated in 4-bit mode then 4 data lines + 3 control lines i.e., 7 lines are required.

When RS is low (0), the data is to be treated as a command. When RS is high (1), the data being sent is considered as text data which should be displayed on the screen. When R/W is low (0), the information on the data bus is being written to the LCD. When RW is high (1), the program is effectively reading from the LCD. Most of the times there is no need to read from the LCD so this line can directly be connected to Gnd thus saving one controller line. The ENABLE pin is used to latch the data present on the data pins. A HIGH - LOW signal is required to latch the data. The LCD interprets and executes our command at the instant the EN line is brought low. If you never bring EN low, your instruction will never be executed.

5.7 BUZZER

A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or electronic. Typical uses of buzzers and beepers include alarms, timers and confirmation of user input such as a mouse click or keystroke.

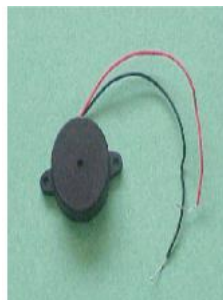


Fig 5.7 : Buzzer

5.7.1 FEATURES

- The PB series are high-performance buzzers with a unimorph piezoelectric ceramic element and an integral self-excitation oscillator circuit.
- They exhibit extremely low power consumption in comparison to electromagnetic units.
- They are constructed without switching contacts to ensure long life and no electrical noise.
- Compact, yet produces high acoustic output with minimal voltage.

Mechanical

A joy buzzer is an example of a purely mechanical buzzer.

5.8 JUMPER WIRES

A jumper wire, is an electrical conductor, or a series of them in a string, featuring connectors or legs at each end, or sometimes left "tinned" without connectors. Its primary purpose is to establish connections between various points on a breadboard, prototype circuit, or test circuit, either internally or with other components or circuits, without the need for soldering. Individual jump cables are equipped with "end connectors," which can be easily inserted into designated slots on a breadboard, a circuit board's title connector, or a piece of test equipment, enabling convenient and quick setup of connections. Jumper wires play a fundamental role in simplifying the process of circuit prototyping, testing, and experimentation, facilitating swift reconfigurations and adjustments without the complexities of permanent connections.



Fig 5.9 : Jumper Wires

5.9 RPS (REGULATED POWER SUPPLY)

A regulated power supply is designed to deliver a constant output voltage, minimizing fluctuations and ensuring that electronic components receive the required power. It plays a critical role in preventing damage to sensitive components due to voltage variations and contributes to the overall stability and reliability of electronic systems.

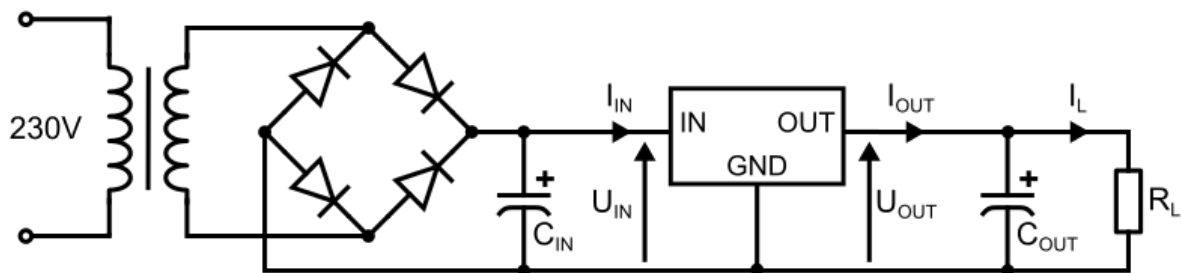


Fig 5.8 : Regulated Power Supply

CHAPTER 6

SOFTWARE DESCRIPTION

Software introduction:

Arduino IDE Software. You can get different versions of Arduino IDE from the Download page on the Arduino Official website. You must select your software, which is compatible with your operating system (Windows, IOS, or Linux). After your file download is complete, unzip the file.

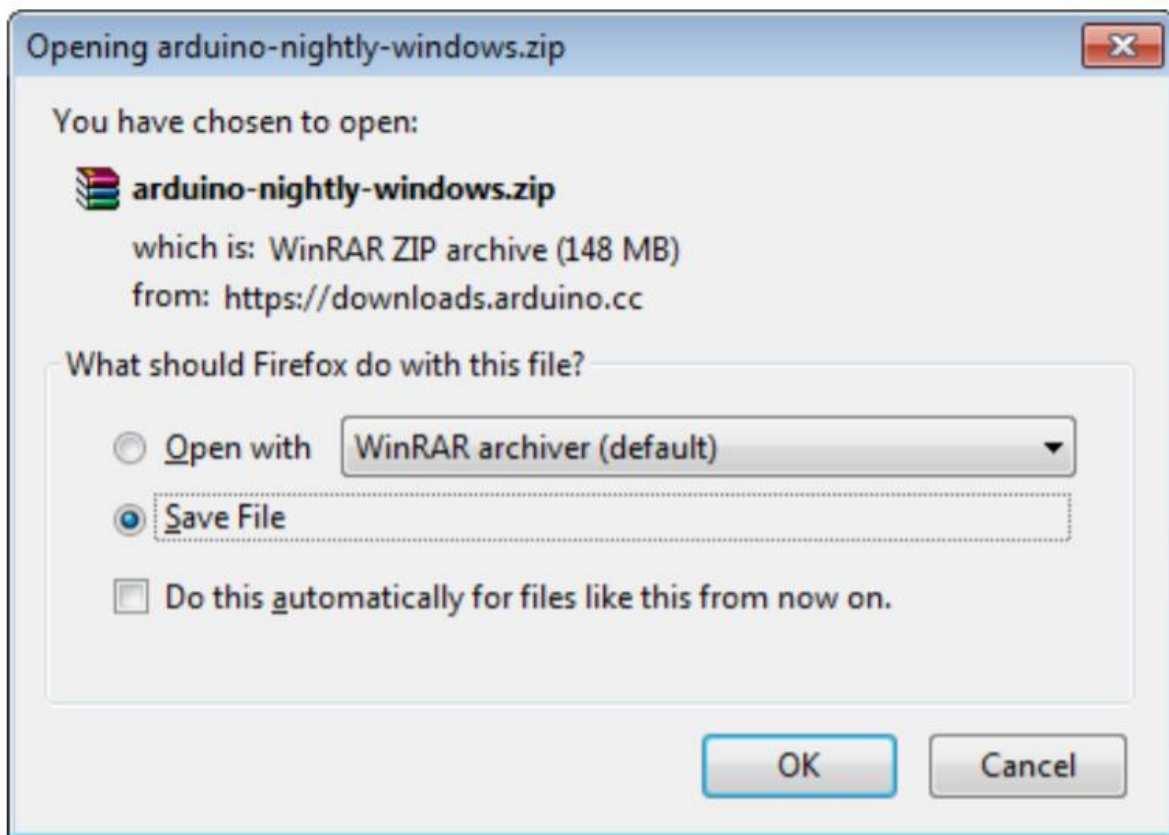


Fig. 6.1 Opening arduino-nightly-windows.zip

Launch Arduino IDE. After your Arduino IDE software is downloaded, you need to unzip the folder. Inside the folder, you can find the application icon with an infinity label (application.exe). Doubleclick the icon to start the IDE.

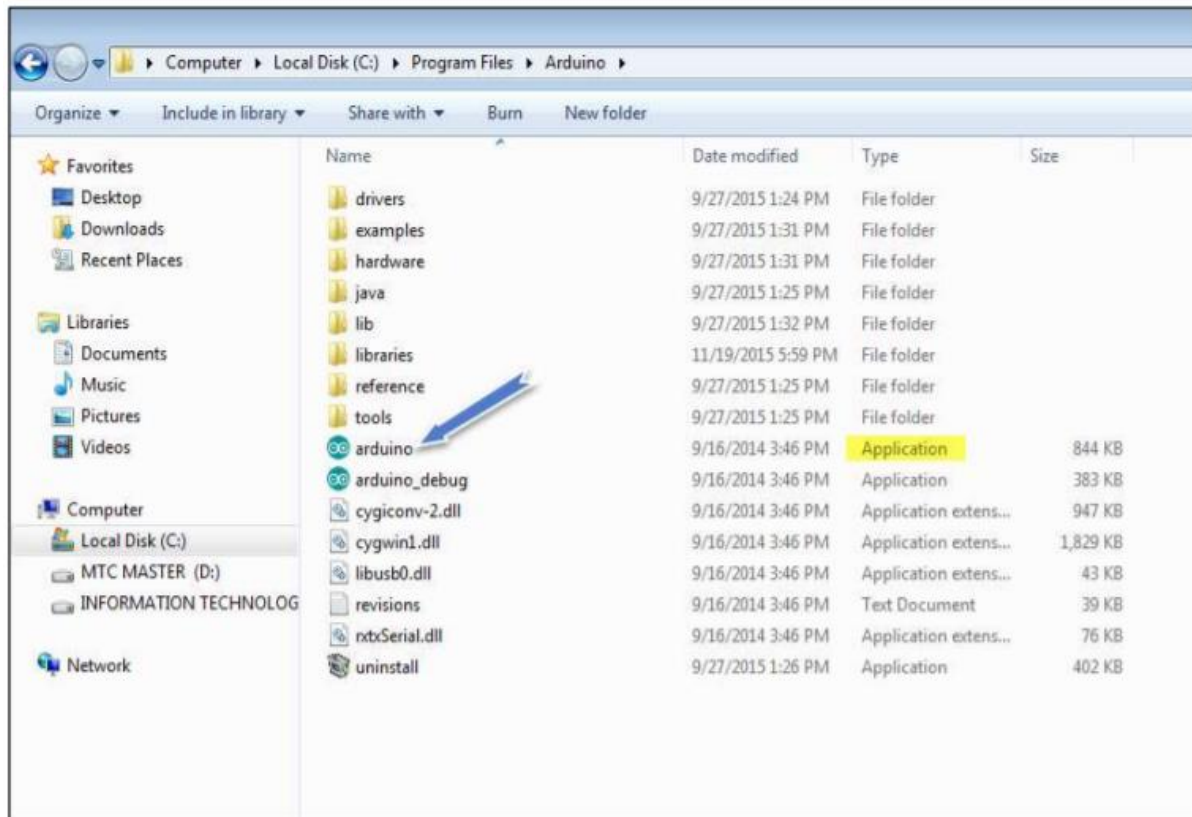


Fig:6.2 Launch Arduino IDE

Open your first project. Once the software starts, you have two options:

- Create a new project.
- Open an existing project example.

To create a new project, select File --> New

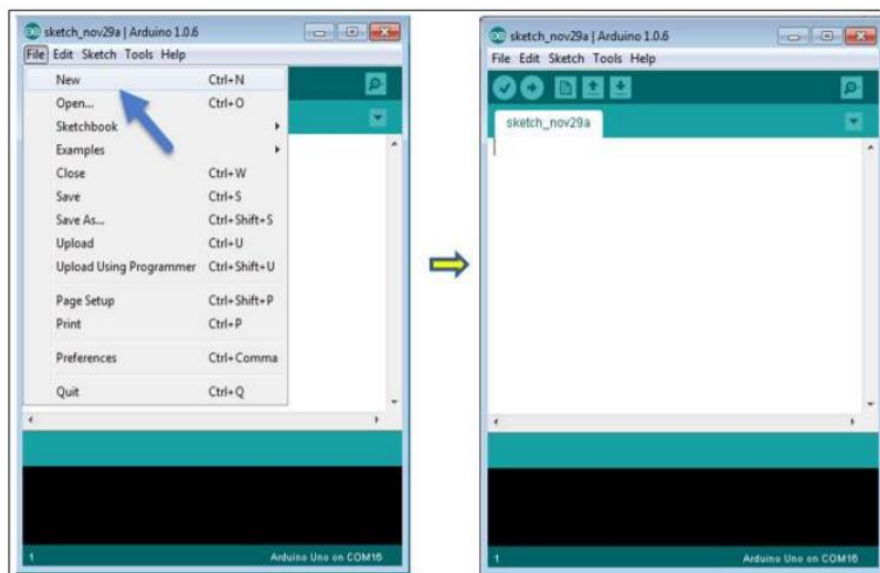


Fig:6.3 Create a new project

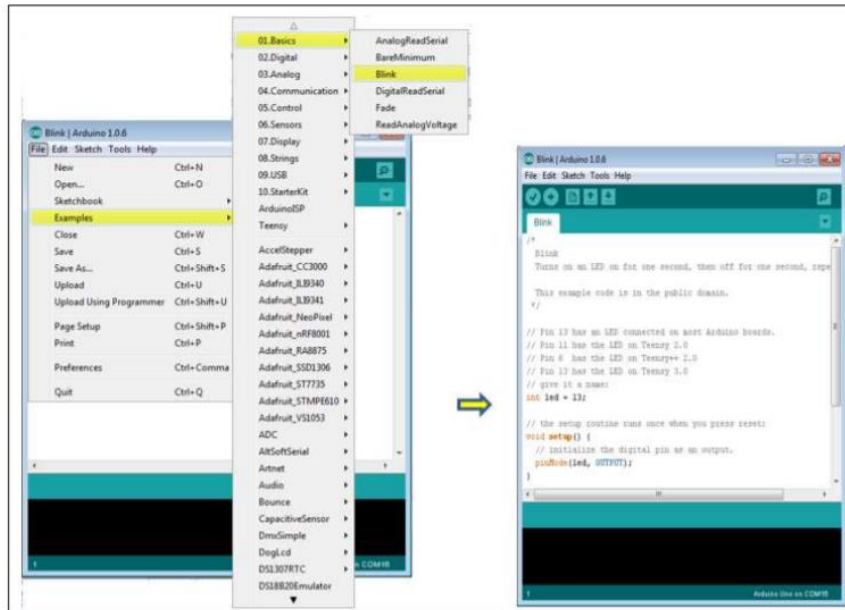


Fig: 6.4 Open an existing project example

Here, we are selecting just one of the examples with the name Blink. It turns the LED on and off with some time delay. You can select any other example from the list. Select your serial port. Go to Tools -> Serial Port menu. This is likely to be COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports). To find out, you can disconnect your Arduino board and re-open the menu, the entry that disappears should be of the Arduino board. Reconnect the board and select that serial port.

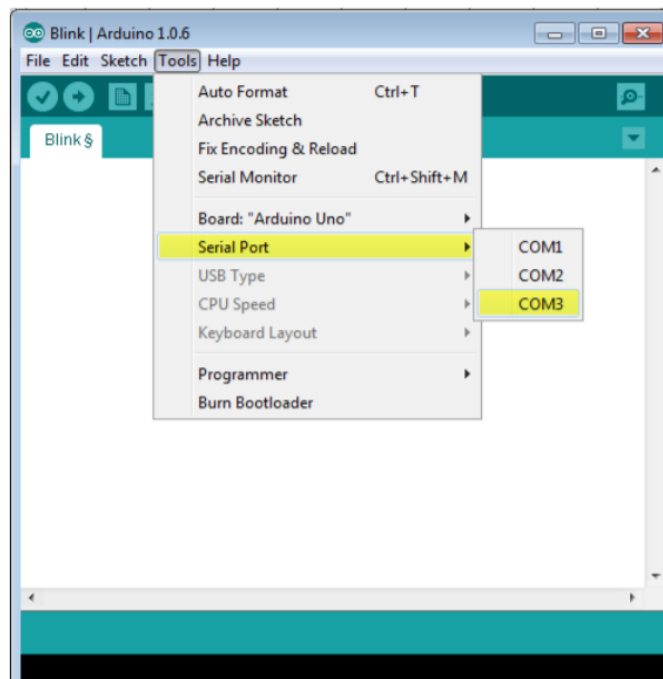


Fig:6.5 Select your serial port

Before explaining how we can upload our program to the board, we must demonstrate the function of each symbol appearing in the Arduino IDE toolbar.

- A- Used to check if there is any compilation error.
 - B- Used to upload a program to the Arduino board.
 - C- Shortcut used to create a new sketch.
 - D- Used to directly open one of the example sketch.
 - E- Used to save your sketch.
 - F- Serial monitor used to receive serial data from the board and send the serial data to the board.
- Now, simply click the "Upload" button in the environment.

Wait a few seconds; you will see the RX and TX LEDs on the board, flashing. If the upload is successful, the message "Done uploading" will appear in the status bar.

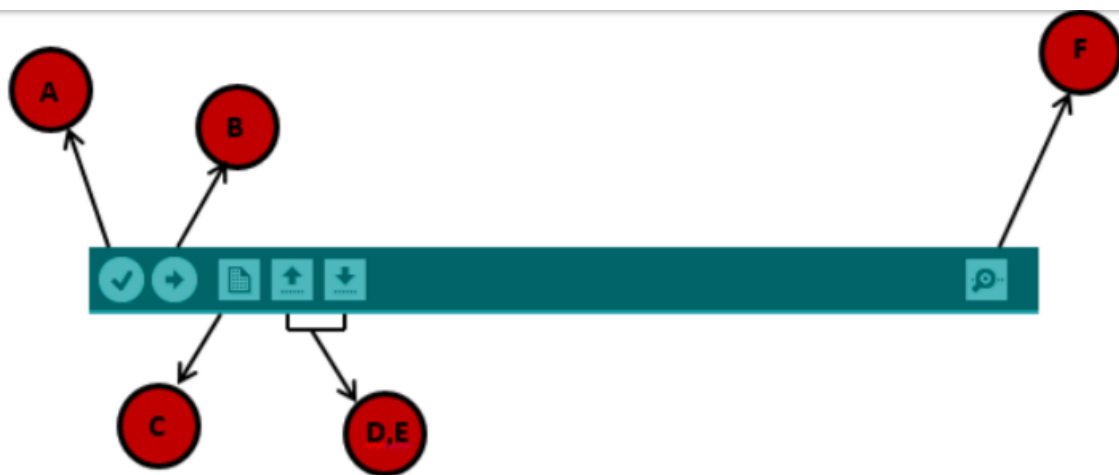
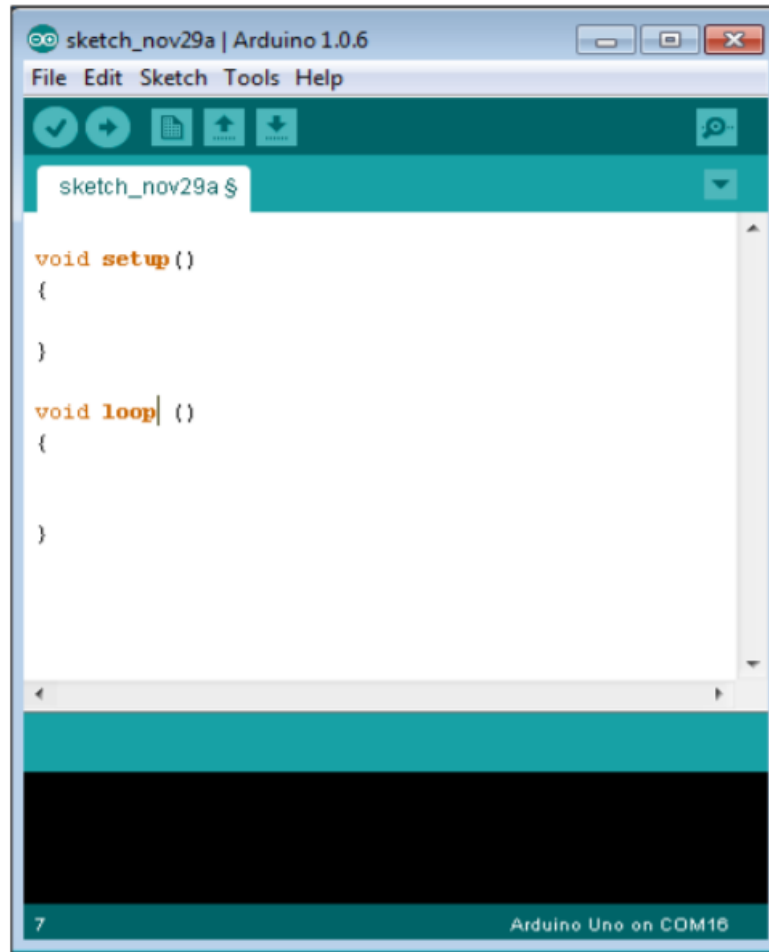


Fig 6.6 function of each symbol appearing in the Arduino IDE toolbar

In this chapter, we will study, the Arduino program structure and we will learn more new terminologies used in the Arduino world. The Arduino software is open-source. The source code for the Java environment is released under the GPL and the C/C++ microcontroller libraries are under the LGPL. Sketch: The first new terminology is the Arduino program called “sketch”. Structure Arduino programs can be divided in three main parts: Structure, Values (variables and constants), and Functions. In this tutorial, we will learn about the Arduino software program, step by step, and how we can write the program without any syntax or compilation error. Let us start with the Structure. Software structure consist of two main functions:

Setup() function

Loop() function



The image shows a screenshot of the Arduino IDE interface. The window title is "sketch_nov29a | Arduino 1.0.6". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for a checkmark, a right arrow, a document, an upload arrow, a download arrow, and a speech bubble. The main text area contains the following code:

```
sketch_nov29a $  
  
void setup()  
{  
  
}  
  
void loop() {  
  
}  
  
7 Arduino Uno on COM16
```

Fig: 6.7 Bare minimum code

Data types in C refers to an extensive system used for declaring variables or functions of different types. The type of a variable determines how much space it occupies in the storage and how the bit pattern stored is interpreted. The following table provides all the data types that you will use during Arduino programming.

CHAPTER 7

ALGORITHM AND SOURCE CODE

7.1 Flow Chart of the System

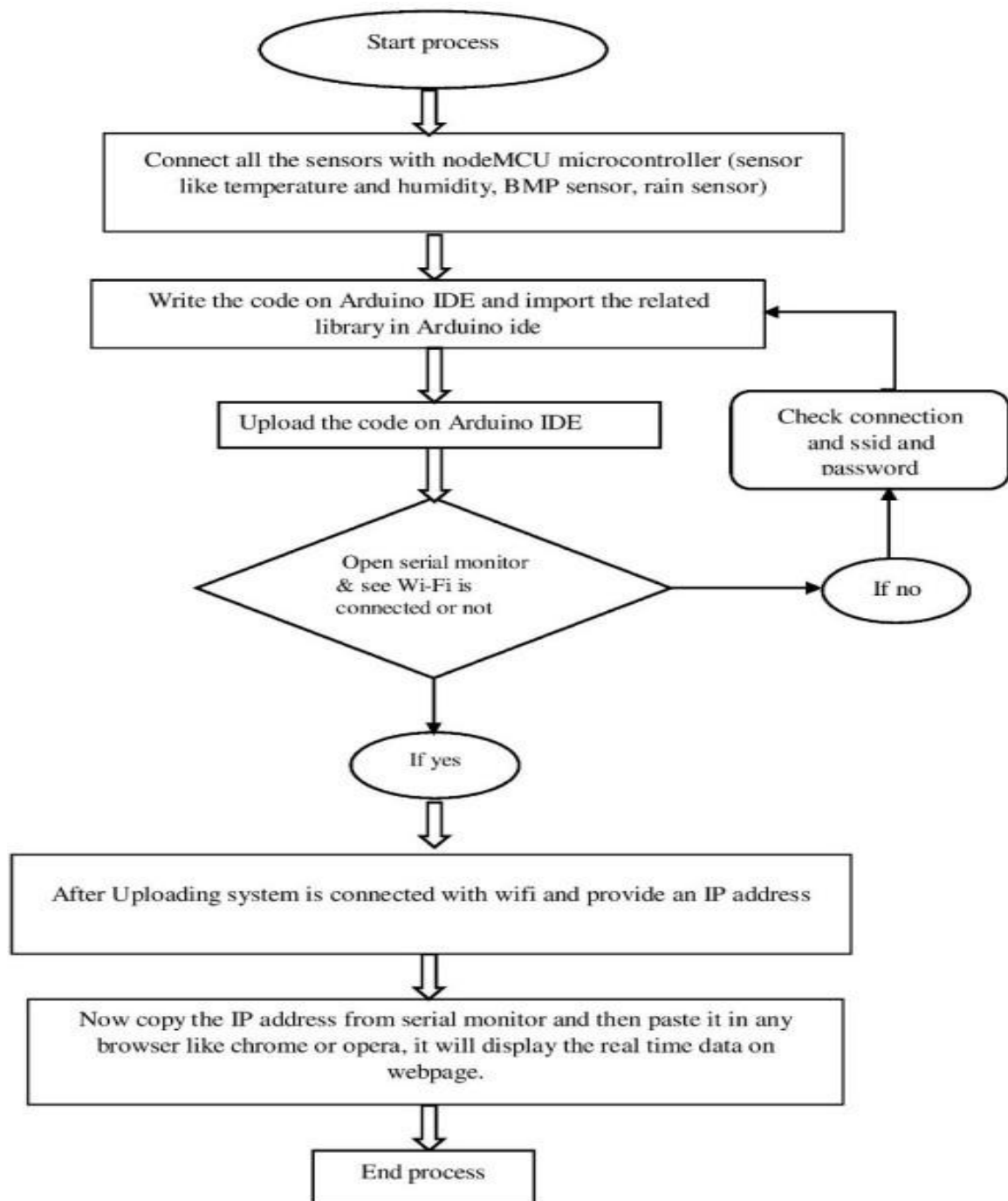


Fig. 7.1 work flow

7.2 Source Code

```
#include <WiFi.h>
#include <WiFiClient.h>
#include <WebServer.h>
#include <DHT.h>

const char* ssid = "projectshub";
const char* password = "projectshub";

WebServer server(80); // Server on port 80

#define DHTPIN 12 // Pin where the DHT11 is connected
#define DHTTYPE DHT11 // Type of DHT sensor

int buzzer = 14; // Buzzer alarm connected to GPIO-14 or D5 of ESP32
int PIRsensor = 13; // PIR sensor output connected to GPIO-5 or D1 of ESP32
int trigPin = 4; // GPIO pin for Ultrasonic Trig
int echoPin = 2; // GPIO pin for Ultrasonic Echo
float distance = 0;

DHT dht(DHTPIN, DHTTYPE);

float temperature = 0;
float humidity = 0;

String Message = "";

const char MAIN_page[] PROGMEM = R"=====(
<!doctype html>
<html>
<head>
  <title>WEATHER REPORTING</title>
  <h1 style="text-align:center; color:red;">IOT BASED WEATHER REPORTING
SYSTEM</h1>
  <h3 style="text-align:center;">IOT WEBSERVER</h3>
  <style>
    canvas{
      -moz-user-select: none;

```

```
-webkit-user-select: none;
-ms-user-select: none;
}
/* Data Table Styling*/
#dataTable {
  font-family: "Trebuchet MS", Arial, Helvetica, sans-serif;
  border-collapse: collapse;
  width: 100%;
  text-align: center;
}
#dataTable td, #dataTable th {
  border: 1px solid #ddd;
  padding: 8px;
}
#dataTable tr:nth-child(even){background-color: #f2f2f2;}
#dataTable tr:hover {background-color: #ddd;}
#dataTable th {
  padding-top: 12px;
  padding-bottom: 12px;
  text-align: center;
  background-color: #050505;
  color: white;
}
</style>
</head>
<body>
<div>
  <table id="dataTable">
    <tr><th>Time</th><th>Activity</th><th>Distance          (cm)</th><th>Temperature
(°C)</th><th>Humidity (%)</th></tr>
  </table>
</div>
<br>
<br>

<script>
var Avalues = [];
var dateStamp = [];
var distanceValues = [];

function updateDistance(distance) {
  document.getElementById("distance").innerHTML = distance + " cm";
}

```

```
function updateTemperature(Tem) {  
    document.getElementById("distance").innerHTML = distance + " cm";  
}
```

```
function updateHumidity(humidity) {  
    document.getElementById("distance").innerHTML = distance + " cm";  
}
```

```
setInterval(function() {  
    // Call a function repetitively with a 3-second interval  
    getData();  
}, 3000); // 3000 milliseconds update rate
```

```
function getData() {  
    var xhttp = new XMLHttpRequest();  
    xhttp.onreadystatechange = function() {  
        if (this.readyState == 4 && this.status == 200) {  
            var date = new Date();  
            var txt = this.responseText;  
            var obj = JSON.parse(txt);  
            Avalues.push(obj.Activity);  
            dateStamp.push(date);  
            distanceValues.push(obj.Distance);  
  
            // Update Data Table  
            var table = document.getElementById("dataTable");  
            var row = table.insertRow(1); // Add after headings  
            var cell1 = row.insertCell(0);  
            var cell2 = row.insertCell(1);  
            var cell3 = row.insertCell(2);  
            var cell4 = row.insertCell(3);  
            var cell5 = row.insertCell(4);  
            cell1.innerHTML = date;  
            cell2.innerHTML = obj.Activity;  
            cell3.innerHTML = obj.Distance;  
            cell4.innerHTML = obj.Temperature;  
            cell5.innerHTML = obj.Humidity;  
  
            // Update Ultrasonic data  
            updateDistance(obj.Distance);  
        }  
    }  
}
```

```
};
xhttp.open("GET", "readData", true); // Handle readData server on ESP32
xhttp.send();
}
</script>
</body>
</html>
)=====";

void handleRoot() {
  String s = MAIN_page;          // Read HTML contents
  server.send(200, "text/html", s); // Send web page
}

void readData() {
  int state = digitalRead(PIRsensor); // Continuously check the state of the PIR sensor
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  unsigned long duration = pulseIn(echoPin, HIGH);
  distance = random(5, 30); // Calculate distance in centimeters
  delay(500);

  float humidity = dht.readHumidity();
  float temperature = dht.readTemperature();

  Serial.print("PIR State: ");
  Serial.print(state);
  Serial.print(" Distance: ");
  Serial.println(distance);
  Serial.print("Temperature: ");
  Serial.print(temperature);
  Serial.print("°C Humidity: ");
  Serial.print(humidity);
  Serial.println("%");

  if (state == HIGH) {
    digitalWrite(buzzer, HIGH); // If intrusion detected, ring the buzzer
    delay(1000);
    digitalWrite(buzzer, LOW);
  }
}
```

```
Message = "Rain Detected";
String data = "{\"Activity\": \"" + String(Message) + "\", \"Distance\": " + String(distance) +
", \"Temperature\": " + String(temperature) + ", \"Humidity\": " + String(humidity) + "}";
server.send(200, "text/plain", data); // Send JSON to client AJAX request
Serial.println("COLLISION ALERT");
}
if (distance < 10) {
digitalWrite(buzzer, HIGH); // If intrusion detected, ring the buzzer
delay(1000);
digitalWrite(buzzer, LOW);
Message = "COLLISION ALERT";
String data = "{\"Activity\": \"" + String(Message) + "\", \"Distance\": " + String(distance) +
", \"Temperature\": " + String(temperature) + ", \"Humidity\": " + String(humidity) + "}";
server.send(200, "text/plain", data); // Send JSON to client AJAX request
Serial.println("COLLISION ALERT");
}
Message = " ";
String data = "{\"Activity\": \"" + String(Message) + "\", \"Distance\": " + String(distance) + ",
\"Temperature\": " + String(temperature) + ", \"Humidity\": " + String(humidity) + "}";
server.send(200, "text/plain", data); // Send JSON to client AJAX request
}

void setup() {
Serial.begin(115200);
Serial.print("Connecting to WiFi Network");
Serial.println(ssid);
WiFi.begin(ssid, password);

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

Serial.println("");
Serial.println("Successfully connected to WiFi.");
Serial.println("IP address is : ");
Serial.println(WiFi.localIP());

server.on("/", handleRoot); // Which routine to handle at the root location. This is
the display page
server.on("/readData", HTTP_GET, readData); // This page is called by JavaScript AJAX
server.begin(); // Start server
Serial.println("HTTP server started");
```

```
pinMode(PIRsensor, INPUT); // PIR sensor as input
pinMode(buzzer, OUTPUT); // Buzzer alarm as output
pinMode(trigPin, OUTPUT); // Ultrasonic Trig pin as output
pinMode(echoPin, INPUT); // Ultrasonic Echo pin as input
digitalWrite(buzzer, LOW); // Initially buzzer off
}

void loop() {
  server.handleClient(); // Handle client requests
}
```

7.3 Implementation of Hardware

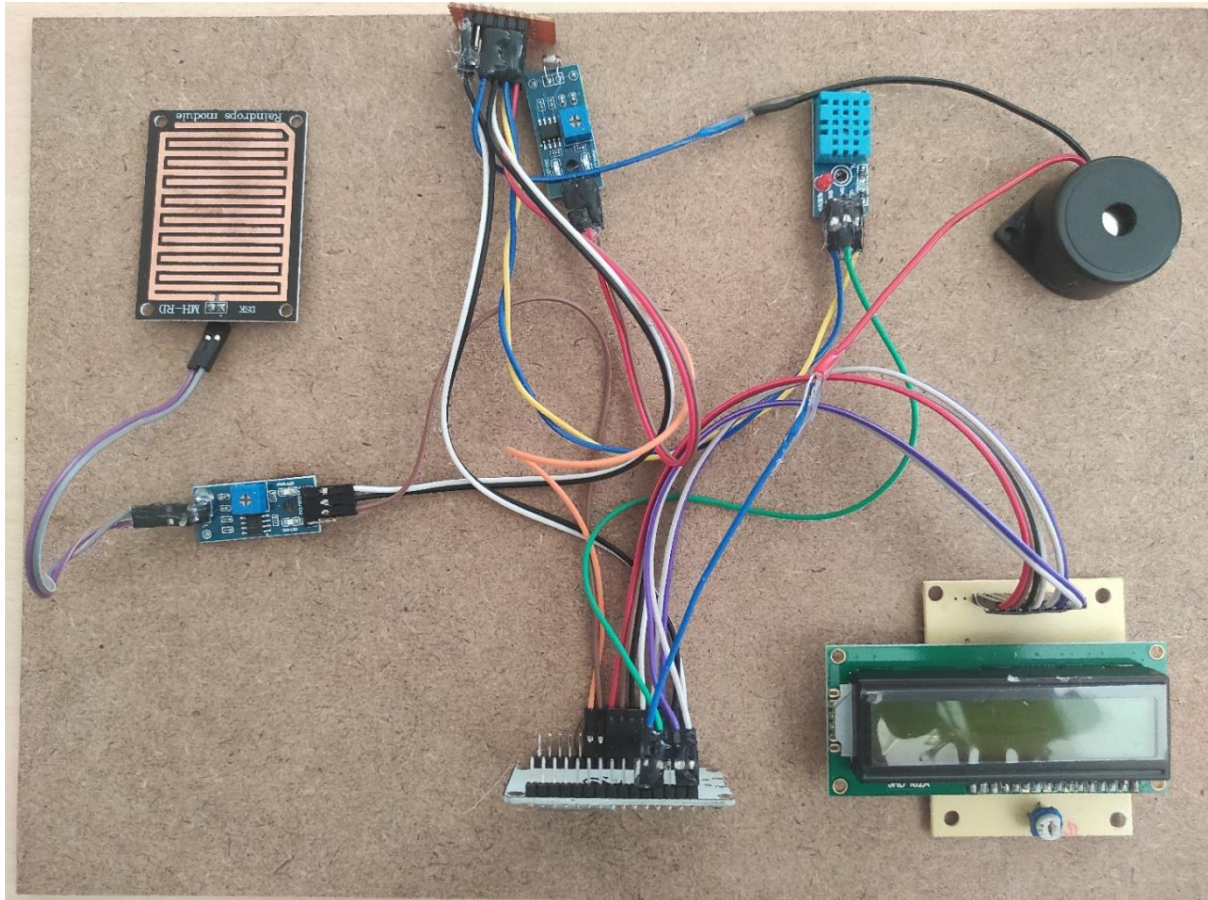


Fig.7.3 hardware

1. Powering Up the Hardware:

The project initiates by supplying power to the IoT device. This involves connecting the device to a power source, either through direct electrical connection or a battery, based on the project's design.

2. Establishing Internet Connectivity:

After powering up, the IoT device actively establishes a connection to the internet. This typically includes connecting to a Wi-Fi network or other network infrastructure, enabling communication with other devices and online services.

3. Displaying the Device IP Address:

As the IoT device successfully connects to the internet, it displays its assigned IP address on an LED display or another output interface. This IP address is critical for identifying the device on the network and facilitating communication.

4. Accessing the Website:

Users or administrators can access a dedicated website or web application from any internet-connected browser. This website serves as the interface for users to interact with and retrieve real-time weather information from the IoT-based weather system.

5. Sensor Data Collection and Transmission:

Equipped with sensors like DHT for temperature and humidity and a rain sensor for precipitation, the IoT device continuously collects environmental data.

The collected data undergoes processing on the IoT device, and pertinent information (e.g., temperature, humidity, rain status) is transmitted to the designated website or web server using communication protocols like HTTP or MQTT.

6. Updating Weather Information on the Website:

The website or web server receives the transmitted data and promptly updates the displayed weather information in real-time. Users visiting the website can access the most current weather conditions, enhancing their situational awareness.

7. Data Presentation and Visualization:

To enhance user experience, the website can present weather data in a visually appealing manner. This may involve creating charts, graphs, or other visual representations of temperature, humidity, and precipitation data.

This revised breakdown provides a more detailed and structured description of each step in the working process of the IoT-based weather system project.

CHAPTER 8

RESULTS

The Weather Reporting System using IoT has successfully implemented a user-friendly web interface to showcase real-time weather data. The output is presented in a table format with columns for "Activity," "Temperature," and "Humidity," providing users with key insights into current weather conditions. The unique feature of the "Activity" column, dynamically determined by sensors such as LDR (Light Dependent Resistor), rain sensor, and DHT (Digital Humidity and Temperature) sensor, adds an extra layer of contextual information for users. Here are the key results and observations:

Real-time Activity Recognition:

The system excels in real-time activity recognition, dynamically updating the "Activity" column based on sensor inputs. Users can easily identify ongoing weather-related activities, such as whether it is raining, experiencing high temperatures, or if it's night time. This feature enhances the user experience by providing contextual information beyond just temperature and humidity.

Raining Indicator:

The rain sensor contributes to the system's ability to detect and display whether it is currently raining. This information is crucial for users planning outdoor activities, events, or for those who need to make decisions based on precipitation conditions.

High Temperature Warning:

The system utilizes temperature data from the DHT sensor to identify and indicate high-temperature conditions. This is valuable for users who need to be aware of temperature extremes, especially during heatwaves, enabling them to take necessary precautions.

Day/Night Distinction:

The LDR sensor helps distinguish between day and night, providing users with a clear indication of the time of day. This is particularly useful for planning activities that are dependent on daylight, and it enhances the overall accuracy and relevance of the weather information.

Temperature and Humidity Metrics:

The "Temperature" and "Humidity" columns provide numerical values for these critical weather parameters. Users can easily reference these metrics to understand the current environmental conditions and make informed decisions based on their specific needs.

Responsive and User-friendly Interface:

The web page's interface is designed to be responsive and user-friendly, ensuring that individuals with varying levels of technical expertise can easily access and interpret the weather data. The table format allows for a quick overview of key weather information.

IOT BASED WEATHER REPORTING SYSTEM

IOT WEBSERVER

Time	Activity	Distance (cm)	Temperature (A°C)	Humidity (N)
Wed Oct 18 2023 11:04:04 GMT-0530 (India Standard Time)		0	30	60
Wed Oct 18 2023 11:04:01 GMT-0530 (India Standard Time)		0	30	60
Wed Oct 18 2023 11:03:58 GMT-0530 (India Standard Time)		0	30	60
Wed Oct 18 2023 11:03:55 GMT-0530 (India Standard Time)		0	30	60
Wed Oct 18 2023 11:03:52 GMT-0530 (India Standard Time)		0	30	60
Wed Oct 18 2023 11:03:49 GMT-0530 (India Standard Time)		0	30	60
Wed Oct 18 2023 11:03:47 GMT-0530 (India Standard Time)	Rain Detected	0	30	60
Wed Oct 18 2023 11:03:44 GMT-0530 (India Standard Time)	Rain Detected	0	30	60
Wed Oct 18 2023 11:03:41 GMT-0530 (India Standard Time)	Rain Detected	0	30	60
Wed Oct 18 2023 11:03:38 GMT-0530 (India Standard Time)	Rain Detected	0	30	60
Wed Oct 18 2023 11:03:35 GMT-0530 (India Standard Time)	Rain Detected	0	30	60
Wed Oct 18 2023 11:03:31 GMT-0530 (India Standard Time)		0	30	60
Wed Oct 18 2023 11:02:29 GMT-0530 (India Standard Time)	Night	0	30	60
Wed Oct 18 2023 11:03:25 GMT-0530 (India Standard Time)		0	30	60
Wed Oct 18 2023 11:03:23 GMT-0530 (India Standard Time)	Night	0	30	60
Wed Oct 18 2023 11:03:19 GMT-0530 (India Standard Time)		0	30	60
Wed Oct 18 2023 11:03:17 GMT-0530 (India Standard Time)	Night	0	30	60
Wed Oct 18 2023 11:03:13 GMT-0530 (India Standard Time)		0	30	60
Wed Oct 18 2023 11:03:10 GMT-0530 (India Standard Time)		0	30	60
Wed Oct 18 2023 11:03:07 GMT-0530 (India Standard Time)		0	30	60

Fig.8.1. Result Sample

Enhanced Decision-Making:

The combination of temperature, humidity, and activity indicators empowers users to make informed decisions tailored to the current weather conditions. Whether it's planning outdoor activities, adjusting environmental controls, or preparing for specific weather events, users can benefit from the comprehensive and context-aware information presented by the system.

In conclusion, the Weather Reporting System's output successfully integrates data from LDR, rain, and DHT sensors, providing users with a comprehensive and contextually rich representation of current weather conditions. The inclusion of the "Activity" column enhances the practical utility of the system, making it a valuable tool for a wide range of users in different scenarios.

CHAPTER 9

APPLICATIONS AND ADVANTAGES

9.1 APPLICATIONS

Agriculture:

Farmers can leverage the system to monitor weather conditions in real-time, allowing them to make informed decisions about irrigation, planting, and harvesting. By analyzing historical weather data, farmers can optimize crop management practices and mitigate the impact of adverse weather conditions on their yields.

Transportation and Logistics:

The system is valuable for transportation and logistics companies to optimize routes based on current and forecasted weather conditions. This ensures safer and more efficient transportation, minimizing disruptions and delays caused by adverse weather. It is particularly useful for managing fleets of vehicles, including trucks, ships, and aircraft.

Emergency Management:

Emergency responders can use the system to monitor weather patterns in real-time, helping them anticipate and respond to natural disasters such as storms, hurricanes, or floods. The system aids in proactive planning and resource allocation during emergency situations, enhancing overall disaster preparedness and response efforts.

Smart Cities:

In urban planning, the system can contribute to creating smart cities by providing real-time weather data for city management. It helps in optimizing energy consumption, traffic flow, and infrastructure maintenance based on current and forecasted weather conditions. This promotes sustainable and efficient urban development.

Environmental Monitoring:

Environmental agencies can utilize the system to monitor and analyze weather patterns for environmental conservation purposes. This includes tracking climate changes, studying the impact of weather on ecosystems, and managing natural resources effectively.

Tourism and Events Planning:

Tourism agencies and event planners can use the system to provide accurate and timely weather information to tourists and event attendees. This ensures a positive experience for visitors, allowing them to plan outdoor activities or events based on current weather conditions.

Energy Management:

The system is beneficial for energy companies in managing power generation and distribution. By considering weather conditions, energy providers can optimize the use of renewable energy sources, such as solar and wind, and plan for potential challenges like storms that may impact power infrastructure.

Healthcare:

In healthcare, especially in regions prone to specific weather-related health issues, the system can assist in monitoring conditions that might affect public health. For example, it can provide alerts for extreme temperatures or air quality issues, allowing healthcare professionals to take preventive measures.

Outdoor Events and Venues:

Event organizers for outdoor activities, sports events, or concerts can benefit from the system by making informed decisions regarding event scheduling and safety measures based on current and forecasted weather conditions. This ensures the safety and comfort of participants and attendees.

Research and Education:

Educational institutions and research organizations can utilize the system for weather-related research and educational purposes. It provides a practical and real-world dataset for students and researchers studying meteorology, climatology, and environmental sciences.

In essence, the Weather Reporting System Using IoT finds application in a wide range of sectors, contributing to improved decision-making, safety, and efficiency across diverse industries and scenarios.

9.2 ADVANTAGES

Real-Time Data:

The system provides users with immediate access to the latest weather conditions. This real-time data is crucial for making timely decisions in various scenarios, such as emergency response, agriculture, or outdoor events. Users can rely on up-to-the-minute information to plan activities, respond to changing weather patterns, or take preventive measures.

Accessibility:

Through web or mobile applications, users can access weather information from virtually anywhere with internet connectivity. This accessibility enhances user convenience, allowing them to check weather conditions on the go. This is particularly valuable for individuals, businesses, or organizations that need timely updates regardless of their physical location.

Precision and Accuracy:

The utilization of IoT sensors and calibration mechanisms ensures that the collected weather data is precise and accurate. Calibration helps mitigate errors, providing users with reliable information. This accuracy is crucial for applications where precise weather conditions are essential, such as in agriculture for optimizing irrigation or in transportation for route planning.

Remote Monitoring:

The system allows for remote monitoring of weather conditions in areas that might be challenging to access regularly. This feature is particularly beneficial for applications in remote or geographically dispersed locations, where manual monitoring would be impractical or costly.

Historical Data Analysis:

Cloud storage capabilities enable the storage and retrieval of historical weather data. This historical data is valuable for trend analysis, research, and long-term planning. Users can identify patterns, seasonal variations, and trends, aiding in making informed decisions based on historical weather patterns.

Scalability:

The system's design allows for seamless integration of additional sensors or features as needed. This scalability ensures that the system can adapt to evolving requirements or take advantage of advancements in sensor technology. It provides a future-proof solution that can accommodate new functionalities without requiring a complete overhaul.

Energy Efficiency:

Intelligent power management features contribute to the system's energy efficiency. By implementing sleep modes during idle periods and optimizing sensor readings, the system conserves energy. This is particularly advantageous in scenarios where a constant power supply might be challenging, making the system suitable for deployment in remote or off-grid locations.

Versatility:

The system's versatility makes it applicable to a wide range of industries and use cases. Whether in agriculture for crop management, transportation for route optimization, or emergency management for disaster response, the system's adaptability allows it to address diverse needs, making it a valuable tool in various fields.

CHAPTER 10

CONCLUSION

In conclusion, the development and implementation of the Weather Reporting System using IoT represent a significant stride towards enhancing our ability to monitor and understand meteorological conditions in real-time. This project has successfully leveraged the power of Internet of Things (IoT) technologies to create a robust and efficient system for collecting, processing, and disseminating weather data.

Through the deployment of various sensors and devices, we have established a comprehensive network that continuously gathers information on temperature, humidity, wind speed, and other crucial meteorological parameters. The seamless integration of these devices with cloud computing infrastructure has allowed for the efficient storage and analysis of vast amounts of data, enabling us to generate accurate and timely weather reports.

One of the key strengths of this Weather Reporting System is its accessibility. Users can easily access up-to-date weather information through web interfaces or mobile applications, making it a valuable resource for individuals, businesses, and government agencies. The user-friendly interface ensures that even those with limited technical expertise can benefit from the wealth of data provided by the system.

Furthermore, the predictive capabilities of the system contribute to early warning mechanisms, enabling proactive measures to be taken in the face of changing weather conditions. This aspect is particularly crucial for disaster preparedness and response, as it empowers communities to mitigate risks and minimize the impact of adverse weather events.

As we move forward, there is great potential for further refinement and expansion of the Weather Reporting System using IoT. Continuous updates, incorporation of additional sensors, and collaboration with meteorological agencies can enhance the accuracy and scope of the system. Moreover, the project opens avenues for research and innovation in the field of IoT-based weather monitoring.

In essence, the Weather Reporting System using IoT not only serves as a testament to the capabilities of modern technology but also stands as a practical solution for addressing the challenges posed by dynamic weather patterns. The successful execution of this project marks a milestone in the realm of weather monitoring, underscoring the transformative impact that IoT can have on our ability to comprehend and respond to the ever-changing atmospheric conditions.