INDEX

CHAPTER-1

CHAPTER-2

CHAPTER-3

CHAPTER-4

CHAPTER-5

CHAPTER-6

ABSTRACT

The Weather Monitoring System leverages the capabilities of the Internet of Things (IoT) to provide comprehensive and real-time data on various atmospheric conditions. This system is designed to measure and monitor critical weather parameters such as temperature, humidity, atmospheric pressure, wind speed, and rainfall. Utilizing a network of IoT-enabled sensors, the collected data is transmitted to a central server where it is processed, analyzed, and displayed through an intuitive user interface accessible via web and mobile applications. This real-time monitoring solution not only enhances the accuracy of weather forecasting but also supports applications in agriculture, disaster management, and urban planning. By providing timely and precise weather information, the Weather Monitoring System aims to improve decision-making processes and promote proactive measures in response to changing weather conditions. The project demonstrates the significant impact of IoT technology in advancing environmental monitoring and fostering smarter, more connected communities.

CHAPTER – 1

INTRODUCTION

1.1 Introduction to Project

The increasing prevalence of extreme weather events and climate variability has underscored the critical need for accurate and timely weather monitoring. Traditional weather monitoring systems, while effective, often lack the granularity and real-time capabilities required to respond swiftly to changing weather conditions. In response to these challenges, the integration of Internet of Things (IoT) technology into weather monitoring systems presents a promising solution.

The Weather Monitoring System project aims to harness the power of IoT to create an efficient and scalable platform for continuous weather observation. This system utilizes a network of interconnected sensors to measure essential atmospheric parameters such as temperature, humidity, and rainfall. The data collected from these sensors is transmitted wirelessly to a central server, where it is processed, analyzed, and made accessible through web and mobile applications.

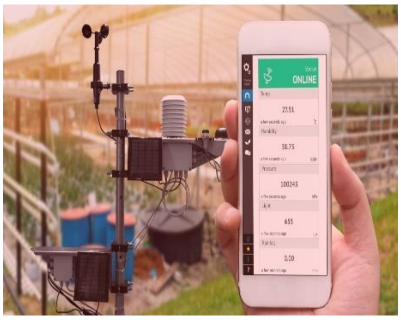


Fig:1.1: Weather app

The primary objective of this project is to provide real-time, accurate weather data to users, enhancing their ability to make informed decisions based on current and forecasted weather conditions. This system has wide-ranging applications, including agricultural management, disaster preparedness, urban planning, and daily weather forecasting.

By leveraging IoT technology, the Weather Monitoring System offers several advantages over traditional methods. These include the ability to monitor remote or hard-to-access locations, reduced latency in data reporting, and the provision of real-time alerts and notifications for significant weather changes. Furthermore, the scalability of IoT networks allows for the deployment of additional sensors as needed, ensuring comprehensive coverage and adaptability to various environments.

In this document, we will explore the design and implementation of the Weather Monitoring System, detailing the hardware and software components used, the communication protocols employed, and the data processing and visualization techniques applied. We will also discuss the potential applications and benefits of the system, highlighting its contribution to enhancing weather awareness and decision-making in various sector

1.2 Introduction to Embedded System

An embedded system is a specialized computing system that performs dedicated functions or tasks within a larger mechanical or electrical system. Unlike general-purpose computers, which are designed to execute a broad range of tasks, embedded systems are tailored for specific applications, providing greater efficiency, reliability, and performance for their intended functions.

Embedded systems are ubiquitous, playing a crucial role in various industries and applications, from consumer electronics and automotive systems to industrial automation and medical devices. They consist of both hardware and software components, with the hardware typically including microcontrollers or microprocessors, memory, input/output interfaces, and other peripheral devices. The software, often referred to as firmware, is specifically written to control and manage the hardware components, ensuring optimal performance for the designated tasks.

One of the defining characteristics of embedded systems is their ability to operate in real-time environments, where timely and deterministic responses to external events are critical. This is achieved through real-time operating systems (RTOS) and specialized programming techniques that prioritize task execution based on urgency and importance. Additionally, embedded systems are often designed to be resource-constrained, optimizing power consumption, processing speed, and memory usage to meet the specific needs of the application.

The design and development of embedded systems involve a multidisciplinary approach, encompassing areas such as electronics, computer science, and systems engineering. Key considerations in the development process include hardware-software integration, system reliability, fault tolerance, and compliance with industry standards and regulations.

As technology advances, embedded systems continue to evolve, incorporating innovations such as the Internet of Things (IoT), artificial intelligence (AI), and machine learning. These advancements expand the capabilities of embedded systems, enabling smarter and more connected devices that can adapt to changing environments and user needs.

Characteristics of Embedded System:

- An embedded System is any computer system hidden inside a product other than a computer.
- Throughput – Our system may need to handle a lot of data in short period of time.
- Response – Our system may need to react to events quickly.
- Test ability- Setting up equipment to test embedded software can be difficult.
- Debug ability- Without a screen or a keyboard, finding out what the software is doing wrong is a troublesome problem.
- Reliability – Embedded Systems must be able to handle any situation without human intervention.
- Memory Space - Memory is limited on Embedded Systems, and you must make the software and the data fit into whatever memory exists.
- Power Consumption – Portable systems must run on battery power, and the software in these systems must conserve power.
- Processor hogs- Computing that requires large amounts of CPU time can complicate the response problem.

1.2 Introduction to IOT

The Internet of Things (IoT) represents a transformative shift in how we interact with the physical world, enabling everyday objects to connect to the internet and communicate with one another. This paradigm encompasses a vast network of devices, sensors, and systems that gather, exchange, and act on data, creating a more interconnected and intelligent environment. The fundamental premise of IoT is to enhance the functionality and efficiency of devices by embedding them with sensors, software, and communication technologies, allowing them to interact seamlessly with other devices and systems.

IoT systems are characterized by their ability to collect and analyze data from a diverse array of sources. Sensors embedded in devices capture real-time information such as temperature, humidity, motion, and location. This data is then transmitted to cloud-based platforms or local servers where it is processed, analyzed, and utilized to generate actionable insights. These insights can inform decision-making, trigger automated responses, and provide valuable feedback to users and systems.

The impact of IoT is profound and far-reaching, spanning multiple domains including smart homes, healthcare, transportation, agriculture, and industrial automation. In smart homes, IoT devices such as smart thermostats, lighting systems, and security cameras provide enhanced control and automation, improving convenience and energy efficiency. In healthcare, wearable devices and remote monitoring systems enable continuous health tracking and early detection of medical conditions. In transportation, IoT technology supports the development of autonomous vehicles and intelligent traffic management systems. Similarly, in agriculture, IoT solutions facilitate precision farming techniques that optimize resource use and crop yield.

The growth of IoT is driven by advancements in wireless communication technologies, such as Wi-Fi, Bluetooth, and cellular networks, as well as the proliferation of low-cost sensors and embedded systems. However, the deployment of IoT solutions also presents challenges, including concerns about data security, privacy, and interoperability. Addressing these challenges requires robust security protocols, standardized communication frameworks, and effective data management practices.

As IoT continues to evolve, it is poised to revolutionize industries and daily life by creating smarter, more responsive environments. The integration of IoT with emerging technologies such as artificial intelligence (AI) and machine learning further enhances its capabilities, enabling more sophisticated data analysis and autonomous decision-making.

Advantages of IOT

- Improved Efficiency: IoT devices can automate and optimize processes, reducing the need for human intervention and increasing operational efficiency.
- Enhanced Data Collection and Analysis: IoT devices can collect vast amounts of data in real time, providing valuable insights for better decision-making and predictive analytics.
- Cost Savings: Automation and optimization through IoT can lead to significant cost savings in energy, maintenance, and labor.
- Improved Quality of Life: In smart homes and cities, IoT can enhance the quality of life by automating tasks, improving energy efficiency, and providing advanced healthcare solutions.
- Enhanced Safety and Security: IoT devices can monitor environments and detect potential threats, providing real-time alerts and improving safety and security in various settings, from homes to industrial sites.
- Remote Monitoring and Management: IoT allows for remote monitoring and management of systems, which is particularly useful in healthcare, agriculture, and industrial applications.
- Personalization and Customization: IoT devices can adapt to user preferences and behaviors, providing personalized experiences in smart homes, retail, and healthcare.
- Environmental Benefits: IoT can contribute to environmental sustainability by optimizing resource use, reducing waste, and enabling more efficient energy consumption.
- Enhanced Customer Experiences: In retail and customer service, IoT can provide enhanced experiences through personalized offers, efficient service, and real-time feedback.
- Innovation and New Business Models: IoT enables new business models and services, such as subscription-based services, predictive maintenance, and smart products that provide ongoing value.

CHAPTER – 2
LITERATURE SURVEY

2.1 Introduction:

Weather monitoring systems explores the extensive body of research and technological advancements related to the collection, analysis, and dissemination of meteorological data. This review aims to provide a comprehensive overview of the current state of knowledge in the field, highlighting key developments, methodologies, and innovations that have shaped the evolution of weather monitoring technologies.

Weather monitoring systems have become increasingly sophisticated with the integration of advanced sensors, communication technologies, and data analytics. This literature survey begins by examining the foundational concepts and historical developments in weather monitoring, tracing the evolution from traditional meteorological instruments to modern IoT-enabled systems. It covers early methods of weather data collection, such as barometers and anemometers, and progresses to the contemporary use of digital sensors and remote sensing technologies.

2.2 Historical evolution of weather monitoring system:

- Overview of traditional weather monitoring instruments (e.g., barometers, thermometers).
- Transition from analog to digital weather monitoring systems.
- Key milestones in the development of modern weather monitoring technologies

2.3 IoT Integration in Weather Monitoring

2.31 Introduction to IoT Integration

The integration of Internet of Things (IoT) technology into weather monitoring systems represents a significant advancement in how meteorological data is collected, analyzed, and utilized. IoT enables the interconnection of a diverse range of sensors and devices, facilitating the seamless collection and transmission of weather data in real time. This integration enhances the accuracy, scalability, and accessibility of weather monitoring systems, providing valuable insights for various applications.

2.32 Role of IoT in Weather Data Collection

IoT technology revolutionizes weather monitoring by enabling the deployment of a vast network of sensors that continuously collect data on atmospheric conditions. These sensors, embedded with IoT capabilities, include:

- **Temperature Sensors**: Measure ambient temperature with high precision.
- **Humidity Sensors**: Monitor relative humidity levels to understand moisture conditions.
- **Pressure Sensors**: Track atmospheric pressure changes, crucial for weather forecasting.
- **Wind Sensors**: Assess wind speed and direction, providing data on local wind patterns.
- **Rain Gauges**: Record precipitation levels, essential for hydrological studies.

These sensors are often deployed in remote or hard-to-reach locations, connected through wireless communication networks that transmit data to central systems for processing and analysis.

2.4 Design and Architecture of IoT-Based Weather Monitoring Systems

An IoT-based weather monitoring system typically consists of several key components:

- **Sensors and Actuators**: Collect and respond to environmental data. These devices are designed to be robust, energy-efficient, and capable of operating in various weather conditions.
- **Communication Modules**: Facilitate data transmission between sensors and central servers. Common communication technologies include Wi-Fi, Bluetooth, Zigbee, LoRa (Long Range), and cellular networks (e.g., 4G/5G).
- **Centralized Data Aggregation**: Involves cloud-based platforms or local servers where data from various sensors is aggregated. These platforms support large-scale data storage, management, and processing.

- **Data Processing and Analytics**: Utilizes algorithms and machine learning techniques to analyze collected data, generate forecasts, and identify patterns or anomalies.

- **User Interfaces**: Provide access to weather data through web or mobile applications, allowing users to view real-time information, historical trends, and alerts.

2.5 Case Studies of IoT-Enabled Weather Stations

1. **Smart Agriculture**: IoT-based weather stations deployed in agricultural fields monitor microclimates, optimizing irrigation schedules and improving crop yield predictions. For example, the use of soil moisture sensors combined with weather data helps farmers make data-driven decisions to conserve water and enhance crop health.

2. **Urban Weather Networks**: Cities like Barcelona and Singapore have implemented IoT-enabled weather networks to monitor urban microclimates. These networks provide valuable data for managing air quality, optimizing energy use, and improving city planning.

3. **Disaster Management**: IoT weather monitoring systems have been employed in disaster-prone regions to provide early warnings for extreme weather events. For instance, the deployment of IoT sensors in hurricane-prone areas helps track storm progression and issue timely alerts to mitigate damage.

2.6 Benefits of IoT Integration

- **Real-Time Data**: IoT systems provide continuous, real-time data on weather conditions, enabling timely responses to changing environmental factors.

- **Enhanced Accuracy**: The use of advanced sensors and data analytics improves the precision of weather measurements and forecasts.

- **Scalability**: IoT networks can be easily scaled by adding more sensors or nodes, allowing for extensive coverage and detailed data collection.

- **Cost-Efficiency**: The deployment of IoT sensors can be more cost-effective compared to traditional weather monitoring systems, offering high data density at lower costs.

Challenges and Considerations

- **Data Security and Privacy**: Ensuring the security of data transmitted over IoT networks and protecting user privacy are critical concerns.

- **System Interoperability**: Integrating sensors from different manufacturers and ensuring compatibility across various platforms can be challenging.

- **Data Accuracy and Reliability**: Addressing issues related to sensor calibration, maintenance, and data integrity is essential for reliable weather monitoring.

2.7 Data Transmission and Communication Protocols

Introduction to Data Transmission in Weather Monitoring

In weather monitoring systems, the effective transmission of data from sensors to data processing units is crucial for ensuring timely and accurate weather information. The choice of data transmission technology and communication protocols significantly impacts the performance, reliability, and scalability of the system. This section explores the various communication technologies and protocols used in weather monitoring systems, highlighting their roles, advantages, and challenges.

Wireless Communication Technologies

1. **Wi-Fi**
   - Overview: Wi-Fi technology enables high-speed data transmission over short to moderate distances, typically within a local area network.
   - Advantages: Offers high bandwidth, making it suitable for transmitting large volumes of data. It is widely available and supported by many devices.
   - Challenges: Limited range compared to other technologies; performance can be affected by interference and network congestion.
2. **Bluetooth**
   - Overview: Bluetooth is a short-range communication technology designed for low-power, low-bandwidth applications.
   - Advantages: Low power consumption and ease of pairing devices. Suitable for low-data-rate applications and short-range communication.
   - Challenges: Limited range (typically up to 100 meters) and lower data transfer rates compared to Wi-Fi. Can be affected by interference from other Bluetooth devices.
3. **Zigbee**
   - Overview: Zigbee is a low-power, low-data-rate wireless communication protocol designed for short-range communication in mesh networks.
   - Advantages: Low power consumption, robust mesh networking capabilities, and scalability. Ideal for applications requiring long battery life and extensive sensor networks.
   - Challenges: Limited data transfer rate and range compared to Wi-Fi. Requires careful network design to ensure reliable communication.
4. **LoRa (Long Range)**
   - Overview: LoRa is a long-range, low-power communication technology designed for wide-area networks with low data rates.
   - Advantages: Long range (up to several kilometers), low power consumption, and ability

to cover large geographic areas. Suitable for rural and remote deployments.
    - Challenges: Lower data transfer rates and potential for network congestion in high-density areas. Requires a network infrastructure for wide-area coverage.

5. **Cellular Networks (e.g., 4G/5G)**
    - Overview: Cellular networks provide wide-area coverage with high data transfer rates, suitable for mobile and remote applications.
    - Advantages: Extensive coverage, high data transfer rates, and reliable connectivity. Ideal for applications that require high-bandwidth data transmission.
    - Challenges: Higher operational costs, potential for network congestion, and dependency on cellular network availability.

Communication Protocols

1. **MQTT (Message Queuing Telemetry Transport)**
    - Overview: MQTT is a lightweight, publish-subscribe messaging protocol designed for low-bandwidth, high-latency environments.
    - Advantages: Efficient for transmitting small data packets over unreliable networks. Supports real-time communication and is widely used in IoT applications.
    - Challenges: Requires a broker for managing message distribution. May not be suitable for applications requiring high data throughput.
2. **HTTP/HTTPS (Hypertext Transfer Protocol/Secure)**
    - Overview: HTTP and HTTPS are widely used protocols for transmitting data over the web. HTTPS provides secure communication through encryption.
    - Advantages: Well-established and supported by many devices and platforms. HTTPS ensures data security through encryption.
    - Challenges: Higher overhead compared to lightweight protocols like MQTT. May not be ideal for low-bandwidth or high-latency environments.
3. **CoAP (Constrained Application Protocol)**
    - Overview: CoAP is a web transfer protocol designed for constrained devices and networks, offering efficient communication for IoT applications.
    - Advantages: Lightweight and optimized for low-bandwidth and low-power environments. Supports multicast and resource discovery.
    - Challenges: Less widely adopted than HTTP, requiring specific implementation support. May require additional layers for secure communication.

Challenges and Solutions

- Signal Interference and Reliability: Wireless communication technologies can be affected by signal interference and environmental factors. Solutions include optimizing network design, using frequency hopping, and implementing error correction mechanisms.
- Data Security and Privacy: Protecting data transmitted over communication networks is crucial. Implementing encryption protocols (e.g., TLS/SSL for HTTPS, AES for LoRa) and secure authentication mechanisms can mitigate security risks.
- Network Scalability and Management: As the number of sensors and devices increases, managing network traffic and ensuring scalability can be challenging. Solutions include using mesh networks, optimizing data transmission intervals, and employing network management tools.

2.8 Applications of Weather Monitoring Systems

Weather monitoring systems have a wide range of applications across various sectors, leveraging real-time meteorological data to enhance decision-making, improve safety, and optimize operations. This section explores key applications of weather monitoring systems, highlighting their impact and benefits in different domains.

1. Agriculture

- Precision Farming: Weather monitoring systems provide critical data on temperature, humidity, and precipitation, enabling farmers to make informed decisions about irrigation, fertilization, and pest control. By integrating weather data with soil moisture sensors, farmers can optimize water usage and improve crop yield.
- Climate Adaptation: Farmers can use weather data to adjust planting schedules and select crop varieties suited to changing climate conditions. This adaptation helps mitigate the impact of adverse weather events on agricultural productivity.
- Disease and Pest Management: Monitoring weather conditions helps in predicting the likelihood of pest outbreaks and plant diseases, allowing for timely intervention and reducing crop losses.

2. Urban Planning and Smart Cities

- Infrastructure Design: Weather monitoring data informs the design and construction of urban infrastructure, such as drainage systems, roads, and buildings. Understanding local weather patterns helps in designing structures that can withstand extreme weather conditions.
- Air Quality Management: Integrating weather data with air quality monitoring systems helps assess the impact of weather on pollutant dispersion and concentration. This integration supports strategies for improving urban air quality.
- Energy Management: Weather data is used to optimize energy consumption in smart buildings and cities. For example, temperature and humidity data can guide the operation of heating, ventilation, and air conditioning (HVAC) systems to enhance energy efficiency.

3. Disaster Management and Emergency Response

- Early Warning Systems: Weather monitoring systems play a crucial role in early warning systems for natural disasters such as hurricanes, floods, and heatwaves. Real-time data enables timely alerts and evacuation plans, reducing the impact of such events on communities.
- Disaster Preparedness: Accurate weather forecasts and historical data help in planning for potential disasters, including resource allocation and emergency response strategies. This preparedness enhances the resilience of affected regions.
- Post-Disaster Analysis: After a disaster, weather monitoring systems provide valuable data for assessing the event's impact, aiding in recovery efforts, and informing future risk mitigation strategies.

4. Transportation and Aviation

- Traffic Management: Weather monitoring systems support traffic management by providing data on road conditions, visibility, and potential hazards. This information helps in optimizing traffic flow and reducing accident risks during adverse weather conditions.
- Aviation Safety: Accurate weather data is essential for flight planning and safety. Monitoring systems provide information on wind speed, turbulence, and visibility, helping pilots make informed decisions and ensuring safe flight operations.
- Maritime Operations: Weather data is crucial for maritime navigation, helping vessels avoid hazardous conditions such as storms and high waves. Real-time data supports route planning and enhances maritime safety.

5. Environmental Research and Climate Studies

- Climate Change Research: Weather monitoring systems contribute to climate research by providing long-term data on temperature, precipitation, and atmospheric conditions. This data is essential for studying climate trends and understanding the impacts of climate change.
- Ecological Monitoring: Weather data is used to monitor ecological systems and biodiversity.

For example, changes in weather patterns can influence the distribution of plant and animal species, and monitoring systems help track these changes.

- Hydrological Studies: Weather monitoring systems provide data for hydrological studies, including rainfall, snowpack, and river flow measurements. This information is critical for managing water resources and understanding hydrological cycles.

6. Public Health

- Health Advisories: Weather monitoring systems contribute to public health by providing data on extreme temperatures, humidity, and air quality. This information helps in issuing health advisories and preparing for heatwaves, cold snaps, and poor air quality events.
- Allergy and Respiratory Conditions: Weather data, including pollen counts and humidity levels, can be used to forecast allergy outbreaks and respiratory issues, helping individuals manage their health proactively.

7. Sports and Recreation

- Event Planning: Weather data is used to plan and schedule outdoor sports events, ensuring optimal conditions for participants and spectators. It helps in making decisions about event postponements or cancellations due to adverse weather.
- Recreational Activities: For recreational activities such as hiking, skiing, and boating, accurate weather forecasts are essential for ensuring safety and enhancing the overall experience.

2.9 Comparative Analysis of Weather Monitoring Systems

Introduction

Weather monitoring systems have evolved significantly over the years, encompassing a variety of technologies and methodologies. A comparative analysis of these systems is essential for understanding their strengths, weaknesses, and suitability for different applications. This section provides a comparative overview of various weather monitoring technologies, including traditional methods, digital systems, and IoT-based solutions.

1. **Traditional Weather Monitoring Systems**

- Overview: Traditional weather monitoring systems rely on analog instruments and manual data collection. These systems typically include instruments such as mercury thermometers, barometers, anemometers, and rain gauges.
- **Advantages:**
    - Reliability: Proven accuracy and durability over time.
    - Low Cost: Generally less expensive to purchase and maintain compared to modern systems.
    - Simplicity: Easy to use with straightforward measurement techniques.
- **Challenges:**
    - Manual Data Collection: Requires regular manual readings and maintenance.
    - Limited Automation: Lacks real-time data collection and remote monitoring capabilities.
    - Data Accuracy: Susceptible to human error and environmental conditions.

2. **Digital Weather Monitoring Systems**

- **Overview:** Digital weather monitoring systems use electronic sensors and digital displays to collect and process meteorological data. These systems often include automated weather stations with integrated sensors for temperature, humidity, pressure, wind speed, and precipitation.
- **Advantages:**
    - **Automation:** Capable of continuous, automated data collection and processing.
    - **Real-Time Data:** Provides up-to-date weather information with digital accuracy.
    - **Data Storage:** Ability to store historical data and generate reports.

- **Challenges:**
    - **Cost:** Higher initial investment compared to traditional systems.
    - **Maintenance:** Requires regular calibration and maintenance of electronic components.
    - **Complexity:** More complex setup and operation compared to analog instruments.

3. **IoT-Based Weather Monitoring Systems**

- **Overview:** IoT-based weather monitoring systems integrate a network of interconnected sensors with IoT technology to collect, transmit, and analyze weather data. These systems use wireless communication technologies and cloud platforms for data management.
- **Advantages:**
    - **Scalability:** Easily expandable with additional sensors and nodes.
    - **Real-Time and Remote Monitoring**: Enables real-time data access and remote monitoring from anywhere.
    - **Advanced Data Analytics:** Utilizes machine learning and data analytics for predictive modeling and trend analysis.
- **Challenges:**
    - **Data Security:** Requires robust security measures to protect data from unauthorized access.
    - **Connectivity:** Dependent on network coverage and connectivity, which can affect performance in remote areas.
    - **Cost:** Higher upfront costs and potential ongoing costs for cloud services and network infrastructure.

4. **Comparative Metrics**

- Accuracy and Precision
    - **Traditional Systems:** Known for accuracy but can be affected by human error and environmental factors.
    - **Digital Systems:** Generally provide high accuracy and precision, though calibration is required.
    - **IoT-Based Systems:** Offer high accuracy with real-time data but depend on sensor quality and calibration.
- **Data Collection and Processing**
    - **Traditional Systems:** Manual data collection, limited to periodic measurements.
    - **Digital Systems:** Automated data collection with digital storage and processing.
    - **IoT-Based Systems:** Continuous, automated data collection with advanced processing and analytics capabilities.
- **Cost**
    - **Traditional Systems**: Low initial cost, but may incur higher long-term costs for manual maintenance and data collection.
    - **Digital Systems:** Higher initial cost, with ongoing maintenance and calibration expenses.
    - **IoT-Based Systems:** Higher upfront and operational costs, including network infrastructure and cloud services.
- **Maintenance and Operation**
    - **Traditional Systems:** Simple maintenance but requires manual intervention.
    - **Digital Systems**: Requires regular calibration and maintenance of electronic components.
    - **IoT-Based Systems:** Requires monitoring of network performance, sensor calibration, and data security measures.
- **Scalability and Flexibility**
    - **Traditional Systems:** Limited scalability and flexibility due to manual operation.

- **Digital Systems:** Moderate scalability with some flexibility in sensor integration.
- **IoT-Based Systems:** Highly scalable and flexible, with the ability to integrate a wide range of sensors and applications.

5. **Case Studies and Examples**

- **Traditional Systems Example:** A weather station using analog instruments in a remote village provides basic weather data but lacks real-time updates and automation.
- **Digital Systems Example:** A digital weather station deployed at an airport offers accurate, automated data collection for flight operations and weather forecasting.
- **IoT-Based Systems Example:** A smart city weather monitoring network with IoT sensors across urban areas provides real-time data for traffic management, air quality monitoring, and disaster preparedness.

II. LITERATURE REVIEW

Weather monitoring systems have undergone significant evolution over the years, from traditional methods to modern digital and IoT-based solutions. This review explores various approaches to weather monitoring, their advantages, limitations, and emerging trends that are shaping the future of meteorological data collection.

A. Traditional Weather Monitoring Methods

Traditional weather monitoring methods have been foundational in meteorological studies. These systems primarily rely on analogue instruments and manual data collection techniques. Instruments such as mercury and alcohol thermometers have been used for temperature measurement, while barometers, including both mercury and aneroid types, have been essential for tracking atmospheric pressure. Anemometers and rain gauges, which measure wind speed and precipitation respectively, have also been widely utilized. Despite their simplicity and low cost, traditional systems come with notable limitations. Data collection is manual and periodic, often prone to human error, and lacks automation. Furthermore, these methods can be limited by environmental factors and do not offer real-time updates**.**

B. Digital Weather Monitoring Systems

The advent of digital technology marked a significant advancement in weather monitoring. Digital systems use electronic sensors and automated instruments to provide precise and real-time meteorological data. Temperature and humidity sensors, digital barometers, advanced anemometers, and electronic rain gauges have transformed weather monitoring by enabling continuous and automated data collection. These systems offer high accuracy and ease of data integration, significantly improving the efficiency of data processing and reporting. However, they come with their own set of challenges, including higher initial costs and the need for regular maintenance and calibration of electronic components. The complexity of setup and operation is also a consideration, which can be more demanding compared to traditional methods**.**

C. IoT-Based Weather Monitoring Systems

The integration of Internet of Things (IoT) technology into weather monitoring systems represents a substantial leap forward. IoT-based systems leverage a network of interconnected sensors to collect and transmit weather data. These systems use various wireless communication technologies, such as Wi-Fi, Bluetooth, Zigbee, LoRa, and cellular networks, to relay data to centralized cloud platforms. This setup not only enables real-time data collection and remote monitoring but also supports advanced data analytics and predictive modeling. The benefits of IoT-based systems include scalability, flexibility, and the ability to provide comprehensive data insights. However, they also face challenges such as data security and privacy concerns, dependence on network connectivity, and potentially high costs associated with infrastructure and cloud services.

D. Comparative Analysis

When comparing traditional, digital, and IoT-based weather monitoring systems, several factors emerge. Traditional systems are valued for their reliability and simplicity but lack automation and real-time data capabilities. Digital systems offer automation and high accuracy, with digital storage and processing enhancing efficiency. IoT-based systems stand out for their real-time data collection, scalability, and advanced analytics, though they require careful management of network connectivity and data security. Each system has its strengths and limitations, with the choice often depending on specific requirements such as data accuracy, cost, maintenance capabilities, and application needs.

E. Emerging Trends and Innovations

The field of weather monitoring is continuously evolving, with several emerging trends shaping its future. Advancements in sensor technology, including miniaturization and multi-sensor integration, are expanding the capabilities of weather monitoring systems. The integration of big data and artificial intelligence (AI) is also becoming increasingly important, with machine learning algorithms improving weather forecasting and predictive modeling. Additionally, there is a growing focus on sustainable and energy-efficient solutions, such as low-power sensors and renewable energy sources, to minimize the environmental impact of weather monitoring systems.

]CHAPTER-3

DESIGNED SYSTEM

3.1 Introduction

This chapter provides a comprehensive overview of the designed weather monitoring system, outlining its architecture, components, and operational mechanisms. The goal of this system is to offer precise and real-time meteorological data, leveraging modern technology to overcome the limitations of traditional methods.

The system integrates various technological elements, including sensor nodes for accurate data acquisition, a central processing unit for data aggregation and analysis, and a communication network for efficient data transmission. By combining these components, the system aims to deliver reliable weather information suitable for diverse applications such as agricultural forecasting, environmental monitoring, and disaster management.

In this chapter, we will delve into the specifics of the system's design, describing how each component contributes to its overall functionality. We will explore the types of sensors used, the role of the microcontroller and communication modules, and the power supply strategies that ensure continuous operation. Additionally, the chapter will cover the system's data collection, processing, and visualization processes, illustrating how these functions work together to meet the needs of various stakeholders.

This detailed examination will lay the foundation for understanding the system's implementation and effectiveness, setting the stage for subsequent discussions on its evaluation and potential applications.

3.2 Objectives

Describe System Architecture: Provide a detailed overview of the weather monitoring system's architecture, including its modular design and how different components integrate to form a cohesive unit.

Detail Component Functionality: Explain the role and functionality of each component in the system, including sensors, microcontrollers, communication modules, and power supplies.

Explain Data Flow: Illustrate the data flow within the system, from data collection by sensors to processing and transmission to the central processing unit, and finally to data visualization.

Highlight System Operation: Outline the operational procedures of the system, including how data is collected, processed, and transmitted in real-time.

Discuss Integration and Testing: Describe the integration process of the system components and the testing procedures conducted to ensure the system's reliability, accuracy, and performance.

Identify Design Considerations: Address key design considerations, including scalability, flexibility, and power management, and how these factors influence the overall effectiveness of the system.

Evaluate System Capabilities: Assess the system's capabilities in meeting the objectives of accurate and real-time weather monitoring, and discuss its potential applications in various fields.

3.3 Block Diagram

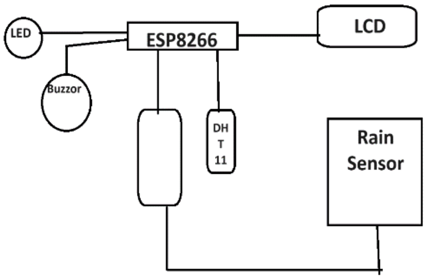


Fig 3.1:Block Diagram

3.4 Tools Required

3.4.1 Hardware Components

- ESP8266
- LED
- LCD
- BUZZER
- RAIN SENSOR
- DHT11

3.4.2 Soft Ware Requirements

- Arduino IDE
- Thing speak
- Code develops through Embedded C

3.4.3 Techniques Used

- IOT technology

The weather monitoring system integrates several key IoT technologies to provide real-time data collection, remote monitoring, and data analysis. At its core is the ESP8266 Wi-Fi module, which enables wireless connectivity and allows the system to transmit data to cloud servers or remote applications. This module is energy-efficient and easy to integrate with various sensors and microcontrollers. The system uses the DHT11 sensor to measure temperature and humidity, providing accurate digital data that is straightforward to interface with the microcontroller. Additionally, a rain sensor detects rainfall and sends this data to the microcontroller, which processes the information and can trigger alerts or actions based on the detected rain.

The system includes an LCD that displays real-time weather data such as temperature, humidity, and rainfall status, offering immediate visual feedback to users. An LED serves as a visual indicator for system status and alerts, showing whether the system is powered on and signaling specific conditions like rain detection. A buzzer provides audible alerts for critical conditions, ensuring users are promptly informed of significant events such as heavy rainfall or system malfunctions.

Communication within the system and with remote servers relies on Wi-Fi for wireless connectivity and HTTP/HTTPS for secure data transmission over the internet, ensuring reliable communication. Cloud computing plays a crucial role by providing storage, processing, and analytics capabilities, enabling the system to store and analyze large amounts of weather data and generate reports. These technologies make the weather monitoring system efficient, reliable, and user-friendly, ensuring accurate real-time data collection, seamless data transmission, and effective communication, suitable for applications such as agricultural planning and disaster management.

3.5 WORKING

The weather monitoring system continuously collects and displays real-time data on temperature, humidity, and rainfall status using an LCD screen. It also employs an LED and a buzzer to provide immediate alerts based on specific conditions.

The system operates as follows:

1. **Data Collection**: The DHT11 sensor measures the ambient temperature and humidity. The rain sensor detects the presence of rainfall.
2. **Data Processing**: The microcontroller receives the data from the sensors and processes it. The temperature, humidity, and rain status are then prepared for display and analysis.
3. **Data Display**: The LCD screen shows the current temperature, humidity, and rain status. If the rain sensor detects rainfall, the LCD displays "Raining"; otherwise, it displays "No Raining."

4. **Condition Monitoring and Alerts**:
   - **Temperature Monitoring**: The system continuously monitors the temperature data from the DHT11 sensor. If the temperature exceeds 40°C, the microcontroller activates the LED to indicate high temperature.
   - **Humidity Monitoring**: The system also monitors the humidity levels. If the humidity exceeds 80%, the buzzer is activated to alert users of high humidity conditions.
   - **Rain Detection**: The rain sensor sends a signal to the microcontroller when it detects rain. This status is displayed on the LCD. Additionally, the rain status determines the LED display – if it's raining, the LCD shows "Raining," otherwise "No Raining."

By integrating these components and functions, the weather monitoring system provides a comprehensive solution for tracking and responding to environmental conditions. The real-time display and alert mechanisms ensure that users are promptly informed of significant changes in weather, enabling timely actions and decisions.

ADVANTAGES

- Real-Time Data Collection: The system provides immediate access to current weather conditions, including temperature, humidity, and rainfall status. This allows users to make informed decisions based on the latest data.
- Automated Alerts: The integration of an LED and a buzzer offers automated alerts for critical conditions. The LED alerts users when the temperature exceeds 40°C, and the buzzer sounds if the humidity exceeds 80%, ensuring timely awareness of extreme weather conditions.
- Remote Monitoring: With the ESP8266 Wi-Fi module, the system can transmit data to cloud servers or remote applications, enabling users to monitor weather conditions from anywhere, enhancing convenience and accessibility.
- User-Friendly Interface: The LCD screen provides a clear and straightforward display of weather data, making it easy for users to understand current conditions at a glance.
- Energy Efficiency: The use of low-power components like the ESP8266 and efficient sensors ensures that the system operates with minimal energy consumption, making it suitable for long-term use and battery-powered applications.
- Versatility and Scalability: The system can be easily expanded or modified to include additional sensors or features, making it adaptable to various applications and environments, such as agriculture, disaster management, and environmental monitoring.
- Cost-Effective: The components used in the system, such as the DHT11 sensor and ESP8266 module, are affordable, making the overall system cost-effective without compromising on functionality.
- Enhanced Safety and Preparedness: By providing real-time alerts and up-to-date weather information, the system helps users stay informed about potentially hazardous conditions, improving safety and preparedness for adverse weather events.
- Historical Data Analysis: With data being transmitted to and stored on cloud servers, users can perform historical data analysis to identify weather patterns, trends, and anomalies over time, aiding in long-term planning and research.

- Ease of Installation and Maintenance: The system is designed to be user-friendly in terms of installation and maintenance. The modular design allows for easy component replacement or upgrades, ensuring longevity and reliability.

CHAPTER-4

HARDWARE IMPLEMENTATION

4.1 Node MCU ESP8266

4.1.1 Description

Node MCU ESP8266 Description Node MCU is an open-source firmware for which open-source prototyping board designs are available. The name “Node MCU” combines “node” and “MCU” (micro-controller unit). The term “Node MCU” strictly speaking refers to the firmware rather than the associated development kits. Both the firmware and prototyping board designs are open source. Node MCU ESP8266 and Node MCU ESP32 are becoming very popular and are almost used in more than 50% IoT based projects today.

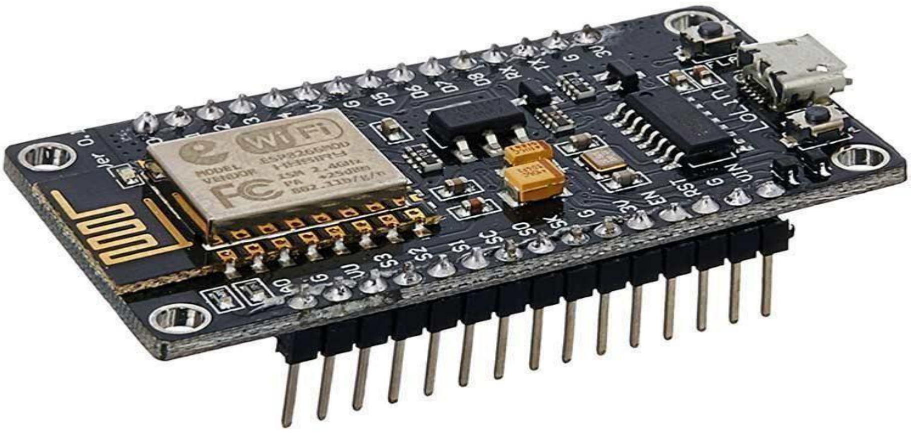


Fig 4.1: Node MCU

The firmware uses the Lua scripting language. The firmware is based on the eLua project and built on the Espressif Non-OS SDK for ESP8266. It uses many open-source projects, such as luacjson and SPIFFS. Due to resource constraints, users need to select the modules relevant for their project and build a firmware tailored to their needs. Support for the 32-bit ESP32 has also been implemented. The prototyping hardware typically used is a circuit board functioning asa dual in-line package (DIP) which integrates a USB controller with a smaller surface-mounted board containing the MCU and antenna. The choice of the DIP format allows for easy prototypingon breadboards.

The design was initially was based on the ESP-12 module of the ESP8266, which is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core, widely used in IoT applications.

About the Node MCU ESP8266 Pinout:

Node MCU ESP8266 Wi-Fi Module is an open-source Lua based firmware and development board specially targeted for IoT based applications. It includes firmware that runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module.

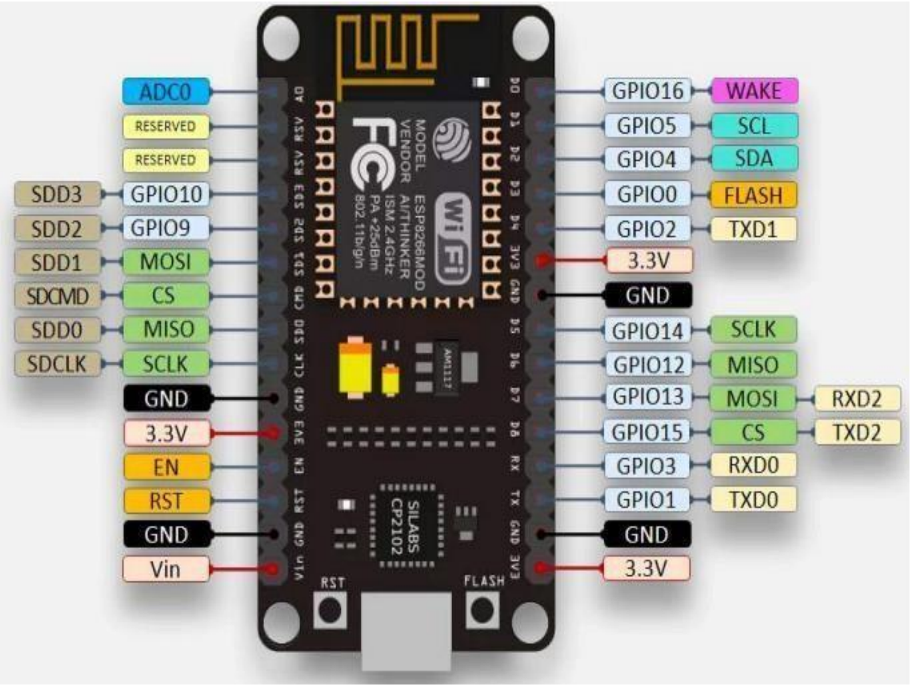


Fig 4.2 Pin Diagram of Node MCU

4.1.2 Node MCU ESP8266 Features:

Microcontroller: Tensilica 32-bit RISC CPU Xtensa LX106

Operating Voltage: 3.3V

Input Voltage: 7-12V

Digital I/O Pins (DIO): 16

Analog Input Pins (ADC): 1

UARTs: 1

SPIs: 1

I2Cs: 1

Flash Memory: 4 MB

SRAM: 64 KB

Clock Speed: 80 MHz

USB-TTL based on CP2102 is included onboard, Enabling Plug n Play PCB

Antenna Small Sized module to fit smartly inside your IoT projects

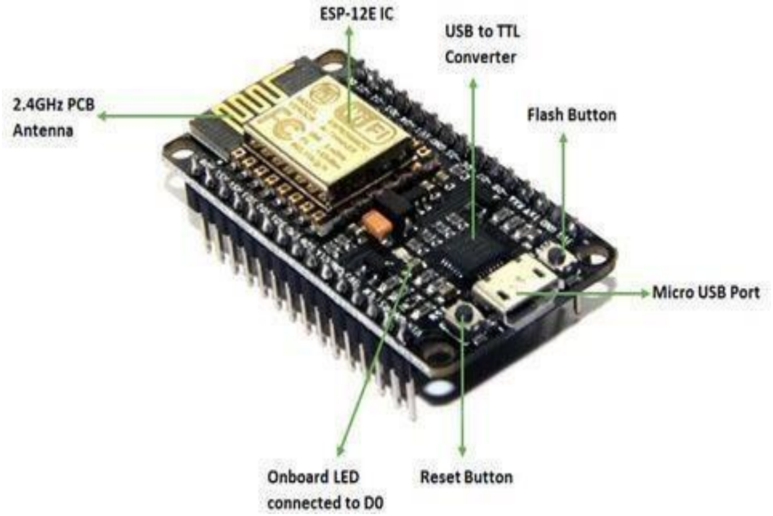


Fig 4.3: Layout of the Node MCU

4.1.3 Node MCU ESP8266 Pinout:

For practical purposes ESP8266 Node MCU V2 and V3 boards present identical pinouts. While working on the Node MCU based projects we are interested in the following pins.

Power pins (3.3 V).

Ground pins (GND).

Analog pins (A0).

Digital pins (D0 – D8, SD2, SD3, RX, and TX – GPIO XX)

Most ESP8266 Node MCU boards have one input voltage pin (Vin), three power pins (3.3v), four ground pins (GND), one analog pin (A0), and several digital pins (GPIO XX).

Pin Code Arduino alias

A0 A0 A0

D0 GPIO 16 16

D1 GPIO 5 5

D2 GPIO 4 4

D3 GPIO 0 0

D4 GPIO 2 2

D5 GPIO 14 14

D6 GPIO 12 12

D7 GPIO 13 13

D8 GPIO 15 15SD2 GPIO 9 9

SD3 GPIO 10 10

RX GPIO 3 3

TX GPIO 1 1

4.2 Rain Sensor

The rain sensor plays a crucial role in the weather monitoring system by detecting the presence and intensity of rainfall. Its integration into the system allows for accurate real-time monitoring and responsive actions based on rainfall conditions. Here are the key aspects and advantages of using a rain sensor in the weather monitoring system:

4.2.1 Working Principle

- **Detection of Rain**: The rain sensor consists of a sensing pad or module that detects the presence of water droplets. When raindrops fall on the sensor, it registers the change and sends a signal to the microcontroller.
- **Data Processing**: The microcontroller receives the signal from the rain sensor and processes it to determine the rain status. This information is then used to update the display on the LCD screen and control other system components.
- **Display and Alerts**: The system displays the current rain status on the LCD screen. If the sensor detects rain, the LCD shows "Raining"; otherwise, it displays "No Raining." This immediate visual feedback helps users stay informed about current weather conditions.
- **Integration with Other Components**: The rain sensor works in conjunction with other system

components. For example, the LED can provide visual alerts based on the rain status, ensuring users are aware of rain conditions even if they are not looking at the LCD screen.

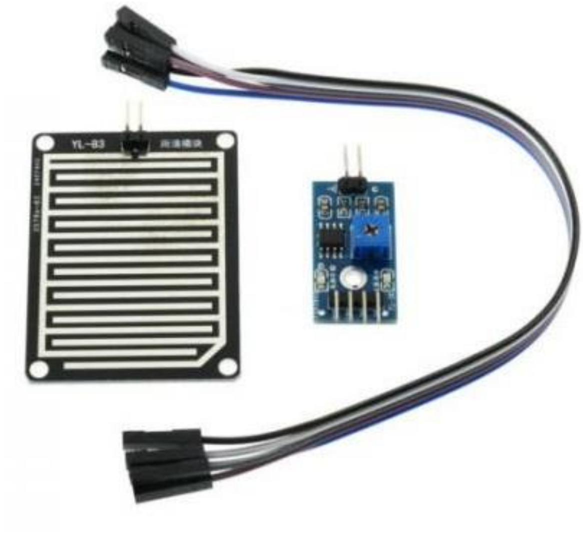


Fig 4.6: Rain Sensor

4.2.2 Description

A regulated power supply is an embedded circuit; it converts unregulated AC (alternating current) into a constant DC. With the help of a rectifier, it converts AC supply into DC. Its function is to supply a stable voltage (or less often current), to a circuit or device that must be operated within certain power supply limits. The output from the regulated power supply may be alternating or unidirectional, but is nearly always DC (direct current). The type of stabilization used may be restricted to ensuring that the output remains within certain limits under various load conditions, or it may also include compensation for variations in its own supply source. The latter is much more common today.

4.2.3 Features

1. **Load Regulation:** The load regulation, abbreviated LR (also called the **load effect**), is the change in regulated output voltage when the load current changes from minimum to maximum value
2. **Minimum Load Resistance:** Another characteristic of regulated power supply is load resistance, at which a power supply delivers its full-load rated current at rated voltage is referredto as a minimum load resistance, $R_{L\ (min)}$.

3. **Source or Line Regulation:** The input line voltage has a nominal value of 230 V but in practice, there are considerable variations in ac supply mains voltage. Since this ac supply mainsvoltage is the input to the ordinary power supply, the filtered output of the bridge rectifier is almost directly proportional to the ac mains voltage. Filtered output of the bridge rectifier is the input to the voltage regulating device.
4. **Output Impedance:** A regulated power supply is a very stiff dc voltage source. This means that the output resistance is very small (in milliohms). Even though the external load resistance is varies, almost no change is seen in the load voltage. An ideal voltage source has an output impedance of zero. Modern regulated power supplies approach ideal voltage sources.
5. **Ripple Rejection:** Voltage regulators stabilize the output voltage against variations in input voltage. Ripple is equivalent to a periodic variation in the input voltage. Thus, a voltage regulatorattenuates the ripple that comes in with the unregulated input voltage. Since a voltage regulator uses negative feedback, the distortion is reduced by the same factor as the gain. Ripple rejection isa measure of a power supply's ability to reject ripple voltages and is usually expressed in decibels.

4.3 Liquid Crystal Display

A liquid-crystal display (LCD) is a flat-panel display or other electronically modulated optical device that uses the light-modulating properties of liquid crystals combined with polarizers. Liquid crystals do not emit light directly but instead use a backlight or reflector to produce images in colour or monochrome.

4.3.1 Description

A **liquid-crystal display** (**LCD**) is a flat-panel display or other electronically modulated optical device that uses the light-modulating properties of liquid crystals combined with polarizers. Liquid crystals do not emit light directly but instead use a backlight or reflector to produce images in colour or monochrome. LCDs are available to display arbitrary images (as in a general-purpose computer display) or fixed images with low information content, which can be displayedor hidden: preset words, digits, and seven-segment displays (as in a digital clock) are all examplesof devices with these displays. They use the same basic technology, except that arbitrary imagesare made from a matrix of small pixels, while other displays have larger elements. LCDs can either be normally on (positive) or off (negative), depending on the polarizer arrangement.

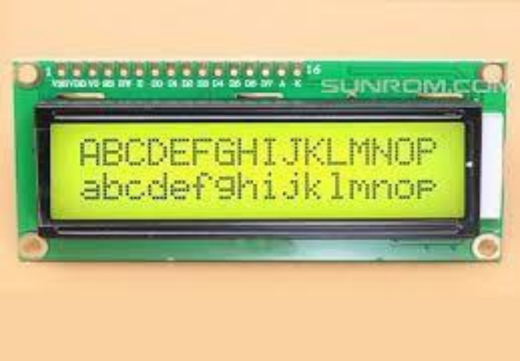


Fig 4.9: Liquid Crystal Display

LCDs are used in a wide range of applications, including LCD televisions, computer monitors, instrument panels, aircraft cockpit displays, and indoor and outdoor signage. Small LCD screens are common in LCD projectors and portable devices such as digital cameras, watches, calculators, mobile telephones, and smartphones. LCD screens havereplaced heavy, bulky and less energy-efficient cathode-ray tube (CRT) displays in nearly all applications. The phosphorus used in CRTs make them vulnerable to image burn-in when a staticimage is displayed on a screen for a long time, e.g., the table frame for an airline flight scheduleon an indoor sign. LCDs do not have this weakness, but are still susceptible to image persistence.

4.3.2 Working

An LCD panel is made of many layers. These consist of a polariser, polarised glass, LCD fluid, conductive connections etc. Polarisation is a process in which the vibration of light waves is restricted to a single plane, resulting in the formation of light waves known as polarised light. Since liquid crystals do not produce light of their own, they need an external light source to work. An LCD panel has sets of polarised glass consisting of liquid crystal materials in between them. When the external light passes through one of the polarised glasses and electric current is applied on the liquid crystal molecules, they align themselves in such a way that polarised light travels from the first layer to the second polarised glass, causing an image to appear on the screen.

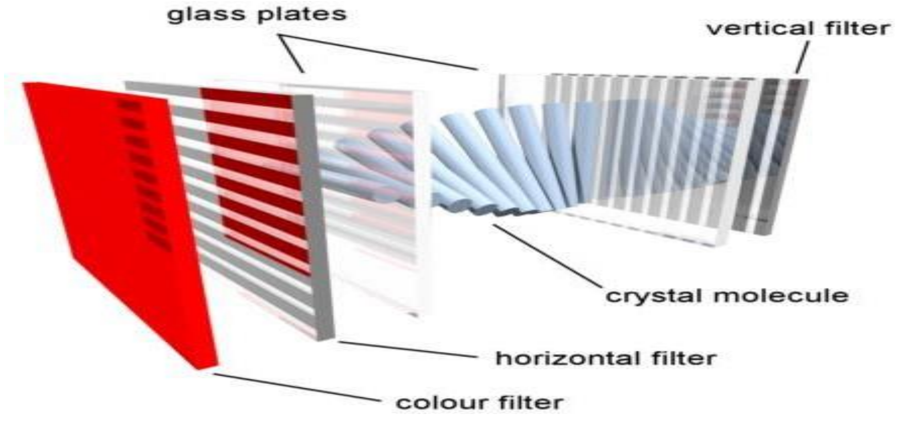


Fig 4.10: Internal Blocks of LCD

4.3.3 Types of LCD

Reflective: This type of LCD has a mirror layer. When a light ray within an LCD is reflected by the mirror layer, then visible patterns are produced on the LCD.

Transmissive: Here the LCD has a backlight, which passes through the LCD polarised glass to produce visible pattern. But because it uses backlight for working, the images displayed in such LCD types appear very dim when used under bright sunlight.

4.4 Light Emitting Diode

The LED (Light Emitting Diode) is a simple yet crucial component in the weather monitoring system, providing visual alerts and status indications based on specific conditions.

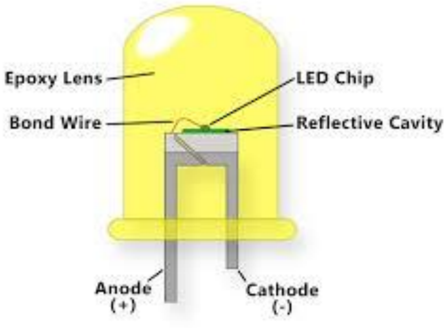
4.4.1 Description

The LED is a semiconductor light source that emits light when current flows through it. In this system, it serves as an indicator to alert users about high temperature or rain detection.

Key Features:

- **Energy Efficient**: Consumes very little power, making it suitable for continuous operation.
- **Long Lifespan**: LEDs have a long operational life, ensuring reliability over extended periods.
- **Brightness**: Provides a clear visual signal that can be easily seen, even from a distance.

Fig 4.9: Light emitting diode



4.4.2 Working

- **Temperature Monitoring**: The system continuously monitors temperature readings from the DHT11 sensor. If the temperature exceeds 40°C, the microcontroller activates the LED.
- **Rain Detection**: The rain sensor detects rainfall and sends the data to the microcontroller. If rain is detected, the system can use the LED to indicate this status.
- **Visual Indication**: The LED provides an immediate visual indication when certain conditions are met, such as high temperature or rain detection, ensuring users are promptly alerted to significant changes in weather conditions .

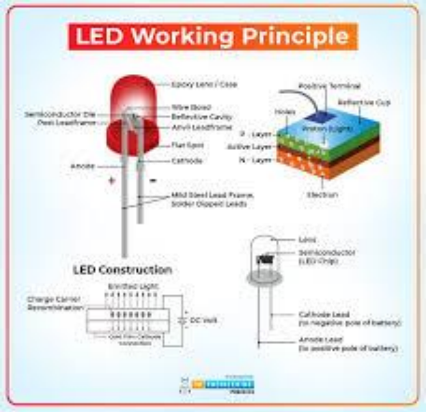


Fig 4.10: Internal Blocks of LED

4.5 Buzzer

The buzzer is another essential component in the weather monitoring system, providing audible alerts for specific conditions.

4.5.1 Description

A buzzer is an audio signaling device that emits sound when activated. In this system, it is used to alert users about high humidity levels.

Key Features:

- Audible Alerts: Produces a loud sound that can easily capture attention, ensuring alerts are noticed.
- Compact Size: Small and lightweight, making it easy to incorporate into the system.
- Low Power Consumption: Uses minimal power, allowing it to operate efficiently.

Fig 4.10: BUZZER

4.5.2 Working

1. **Humidity Monitoring**: The system continuously monitors humidity readings from the DHT11 sensor. If the humidity exceeds 80%, the microcontroller activates the buzzer.
2. **Sound Emission**: When activated, the buzzer emits a sound, providing an audible alert to users about high humidity conditions.
3. **Immediate Notification**: The buzzer ensures that users are promptly informed of significant changes in weather conditions, even if they are not looking at the system's display.

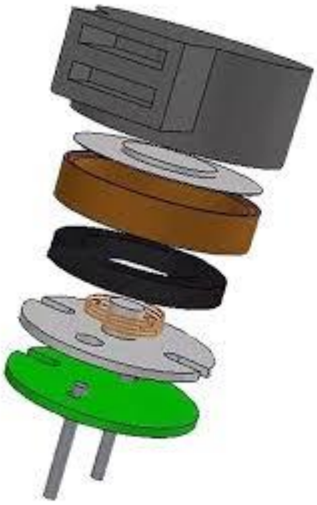


Fig 4.10: Working of Buzzer

4.6 DHT11

The DHT11 sensor is a crucial component in the weather monitoring system, responsible for measuring temperature and humidity. This sensor is widely used in IoT applications due to its reliability, ease of use, and cost-effectiveness.

4.6.1 Description

The DHT11 sensor is a digital temperature and humidity sensor. It consists of a capacitive humidity sensor and a thermistor to measure the surrounding air. The sensor outputs a digital signal on the data pin, making it easy to interface with a microcontroller.

Key Features:

- Temperature Range: 0 to 50°C with ±2°C accuracy
- Humidity Range: 20 to 90% RH with ±5% RH accuracy
- Digital Output: Provides a digital signal, simplifying the integration process with microcontrollers.
- Low Cost: Affordable, making it suitable for a wide range of applications.
- Compact Size: Small form factor, allowing easy incorporation into various projects.

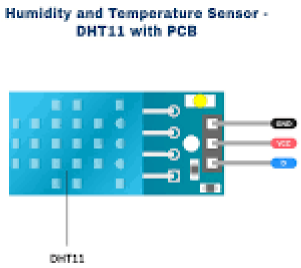


Fig 4.10: Working of Buzzer

4.6.2 Working

- **Data Collection**: The DHT11 sensor measures temperature using a thermistor and humidity using a capacitive humidity sensor. The thermistor changes its resistance with temperature changes, while the capacitive sensor measures the moisture level in the air.
- **Signal Processing**: The sensor has a built-in ADC (Analog to Digital Converter) that converts the analog signals from the thermistor and capacitive sensor into a digital signal. This digital signal is then transmitted to the microcontroller.
- **Data Communication**: The sensor communicates with the microcontroller using a single-wire protocol. It sends the temperature and humidity data in a predefined format, which the microcontroller reads and processes.
- **Data Integration**: The microcontroller receives the digital data from the DHT11 sensor and integrates it into the weather monitoring system. The data is then used to update the display on the LCD and trigger any necessary alerts.
- **Real-Time Display and Alerts**: The system continuously updates the temperature and humidity readings on the LCD screen. If specific conditions are met, such as the temperature exceeding 40°C or humidity surpassing 80%, the system triggers the LED or buzzer to alert users.

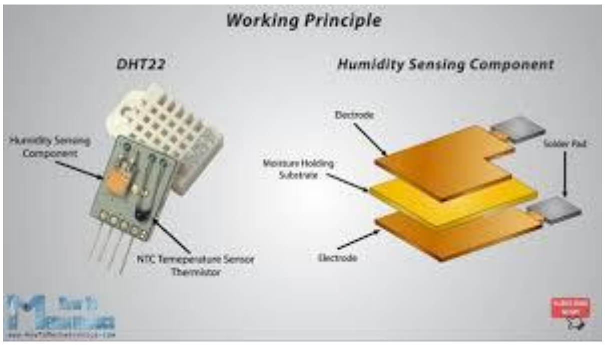

Working Principle
DHT22
Humidity Sensing Component
Humidity Sensing Component
NTC Temeperature Sensor Thermistor
Electrode
Solder Pad
Moisture Holding Substrate
Electrode

Fig 4.10: Working of Buzzer

CHAPTER-5

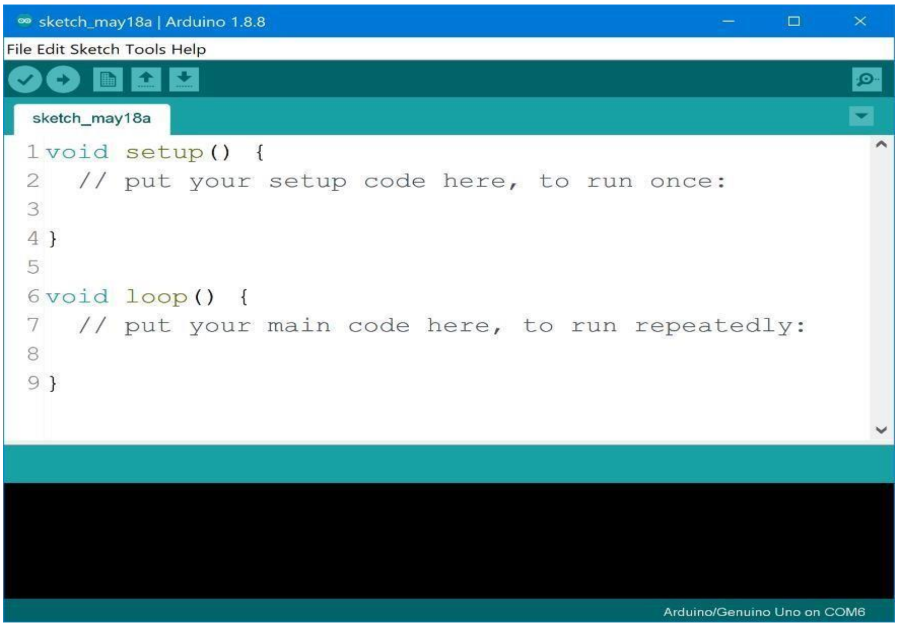
SOFTWARE IMPLEMENTATION

5.1 Arduino IDE

5.1.1 Introduction to Arduino IDE

IDE stands for Integrated Development Environment - An official software introduced by Arduino.cc that is mainly used for writing, compiling and uploading the code in almost all Arduino modules/boards. Arduino IDE is open-source software and is easily available to download & install from Arduino Official Site.

In this post, I'll take you through the brief Introduction of the Software, how you can install it, and make it ready for your required Arduino module.
Let's dive in and get down to the nitty-gritty of this Software.


sketch_may18a | Arduino 1.8.8
File Edit Sketch Tools Help
sketch_may18a
Arduino/Genuino Uno on COM6

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

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Fig 5.1: Arduino IDE Editor page

Arduino IDE is an open-source software, designed by Arduino.cc and mainly used for writing, compiling & uploading code to almost all Arduino Modules.

It is an official Arduino software, making code compilation too easy that even a common person with no prior technical knowledge can get their feet wet with the learning process.It is available for all operating systems i.e., MAC, Windows, Linux and runs on the Java Platform that comes with inbuilt functions and commands that play a vital role in debugging, editing and compiling the code.A range of Arduino modules available including Arduino Uno, Arduino Mega, Arduino Leonardo, Arduino Micro and many more.Each of them contains a microcontroller on the board that is actually programmed and accepts the information in the form of code.The main code, also known as a sketch, created on the IDE platform will ultimately generate a Hex File which is then transferred and uploaded in the controller on the board.The IDE environment mainly contains two basic parts: Editor and Compiler where former is used for writing the required code and later is used for compiling and uploading the code into the given Arduino Module.

This environment supports both C and C++ languages.

5.1.2 How to Download Arduino IDE

You can download the Software from Arduino main website. As I said earlier, the software is available for common operating systems like Linux, Windows, and MAX, so make sure you are downloading the correct software version that is easily compatible with your operating system.

8.1 or Windows 10, as the app version is not compatible with Windows 7 or older version of this operating system.

You can download the latest version of Arduino IDE for Windows (Non admin standalone version), by clicking below button:

Arduino IDE Download

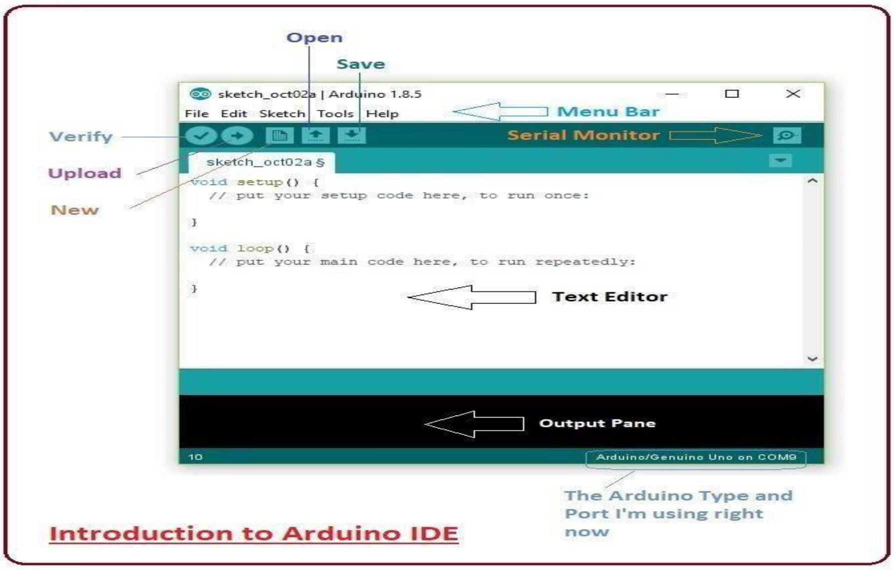
The IDE environment is mainly distributed into three sections.

1.Menu Bar

2.Text Editor

3.Output Pane

As you download and open the IDE software, it will appear like an image below:



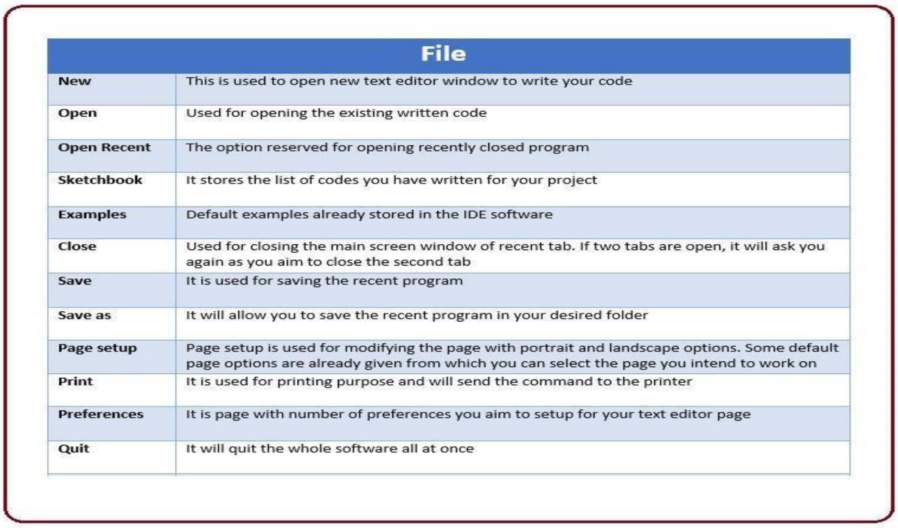
Open
Save
sketch_oct02a | Arduino 1.8.5
File Edit Sketch Tools Help
Menu Bar
Verify
Upload
New
Serial Monitor
sketch_oct02a §
Text Editor
Output Pane
10
Arduino/Genuino Uno on COM9
The Arduino Type and Port I'm using right now
Introduction to Arduino IDE

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

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Fig 5.2: Introduction to Arduino IDE

The bar appearing on top is called Menu Bar that comes with five different options as

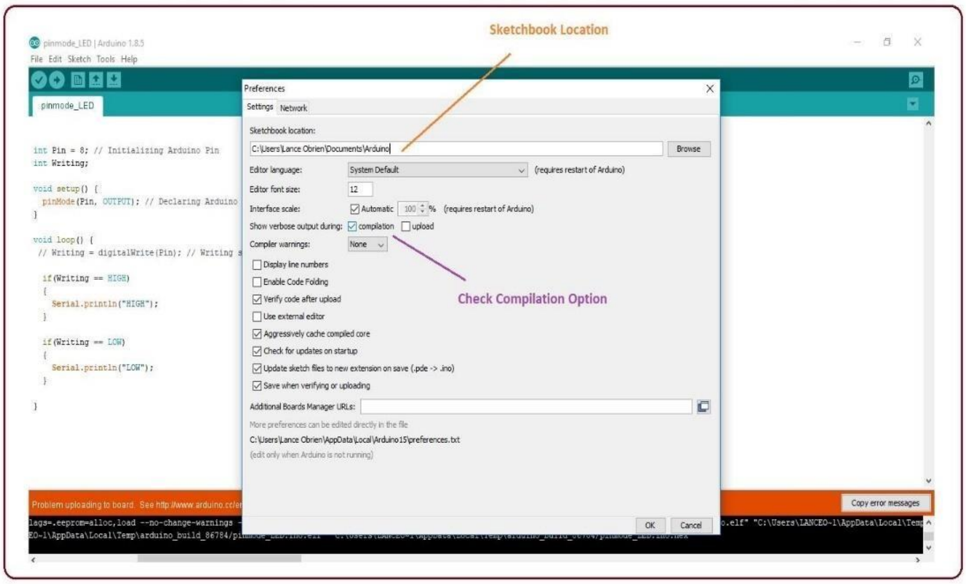
□ File - You can open a new window for writing the code or open an existing one. The following table shows number of further subdivisions the file option is categorized into:



File	
New	This is used to open new text editor window to write your code
Open	Used for opening the existing written code
Open Recent	The option reserved for opening recently closed program
Sketchbook	It stores the list of codes you have written for your project
Examples	Default examples already stored in the IDE software
Close	Used for closing the main screen window of recent tab. If two tabs are open, it will ask you again as you aim to close the second tab
Save	It is used for saving the recent program
Save as	It will allow you to save the recent program in your desired folder
Page setup	Page setup is used for modifying the page with portrait and landscape options. Some default page options are already given from which you can select the page you intend to work on
Print	It is used for printing purpose and will send the command to the printer
Preferences	It is page with number of preferences you aim to setup for your text editor page
Quit	It will quit the whole software all at once

Fig 5.3: File subdivisions in Arduino IDE

□ As you go to the preference section and check the compilation section, the Output Pane will show the code compilation as you click the upload button.



```
int Pin = 8; // Initializing Arduino Pin
int Writing;

void setup() {
  pinMode(Pin, OUTPUT); // Declaring Arduino
}

void loop() {
 // Writing = digitalWrite(Pin); // Writing s

  if(Writing == HIGH)
  {
    Serial.println("HIGH");
  }

  if(Writing == LOW)
  {
    Serial.println("LOW");
  }

}
```

Preferences

Settings | Network

Sketchbook location:
C:\Users\Lance Obrien\Documents\Arduino [Browse]

Editor language: System Default (requires restart of Arduino)

Editor font size: 12

Interface scale: ☑ Automatic 100 % (requires restart of Arduino)

Show verbose output during: ☑ compilation ☐ upload

Compiler warnings: None

☐ Display line numbers
☐ Enable Code Folding
☑ Verify code after upload
☐ Use external editor
☑ Aggressively cache compiled core
☑ Check for updates on startup
☑ Update sketch files to new extension on save (.pde -> .ino)
☑ Save when verifying or uploading

Additional Boards Manager URLs:

More preferences can be edited directly in the file
C:\Users\Lance Obrien\AppData\Local\Arduino15\preferences.txt
(edit only when Arduino is not running)

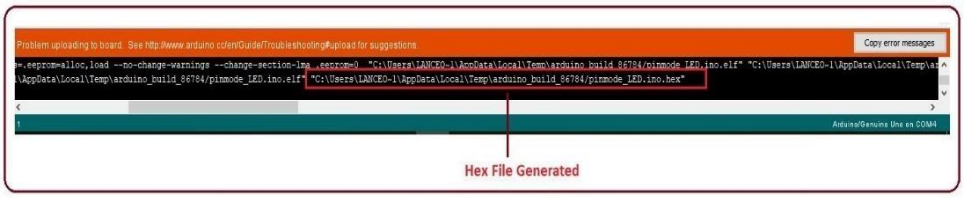
[OK] [Cancel]

Problem uploading to board. See http://www.arduino.cc/er [Copy error messages]

```
lags=.eeprom=alloc,load --no-change-warnings -                                           o.elf" "C:\Users\LANCEO~1\AppData\Local\Tem
EO~1\AppData\Local\Temp\arduino_build_86784/pi
```

Fig 5.4: Selection of compilation

 And at the end of the compilation, it will show you the hex file it has generated for the recent sketch that will send to the Arduino Board for the specific task you aim to achieve.



Problem uploading to board. See http://www.arduino.cc/en/Guide/Troubleshooting#upload for suggestions. [Copy error messages]

```
=.eeprom=alloc,load --no-change-warnings --change-section-lma .eeprom=0  "C:\Users\LANCEO~1\AppData\Local\Temp\arduino_build_86784/pinmode_LED.ino.elf" "C:\Users\LANCEO~1\AppData\Local\Temp\a
\AppData\Local\Temp\arduino_build_86784/pinmode_LED.ino.elf" "C:\Users\LANCEO~1\AppData\Local\Temp\arduino_build_86784/pinmode_LED.ino.hex"
```

Fig 5.5: Hex file generation

- Sketch - For compiling and programming
- Tools - Mainly used for testing projects. The Programmer section in this panel is used for burning a boot loader to the new microcontroller.
- Help - In case you are feeling Edit - Used for copying and pasting the code with further modification for font
- sceptical about software, complete help is available from getting started to troubleshooting.
- The Six Buttons appearing under the Menu tab are connected with the running program as follows.

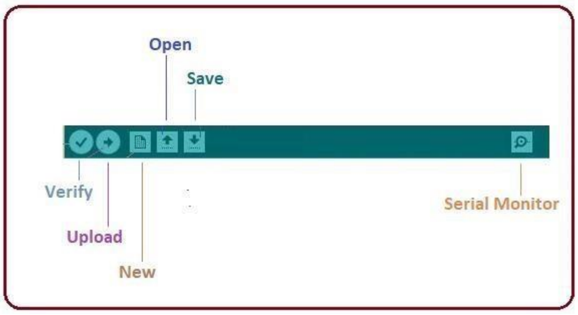
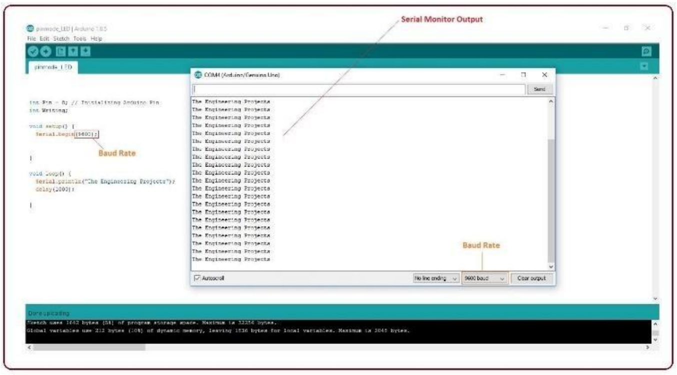


Fig 5.6: Serial monitor

- The check mark appearing in the circular button is used to verify the code. Click this once you have written your code.
- The arrow key will upload and transfer the required code to the Arduino board.
- The dotted paper is used for creating a new file.
- The upward arrow is reserved for opening an existing Arduino project.
- The downward arrow is used to save the current running code.
- The button appearing on the top right corner is a Serial Monitor - A separate pop-up window that acts as an independent terminal and plays a vital role in sending and receiving the Serial Data. You can also go to the Tools panel and select Serial Monitor, or pressing Ctrl+Shift+M all at once will open it instantly. The Serial Monitor will actually help to debug the written Sketches where you can get a hold of how your program is operating. Your Arduino Module should be connected to your computer by USB cable in order to activate the Serial Monitor.
- You need to select the baud rate of the Arduino Board you are using right now. For my Arduino Uno Baud Rate is 9600, Monitor, the output will show as the image below.



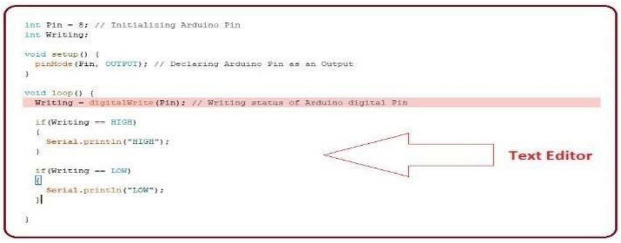
```
int Pin = 8; // Initializing Arduino Pin
int Writing;

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

}

void loop() {
  Serial.println("The Engineering Projects");
  delay(1000);

}
```

```
The Engineering Projects
The Engineering Projects
The Engineering Projects
The Engineering Projects
The Engineering Projects
The Engineering Projects
The Engineering Projects
The Engineering Projects
The Engineering Projects
The Engineering Projects
The Engineering Projects
The Engineering Projects
The Engineering Projects
The Engineering Projects
The Engineering Projects
The Engineering Projects
The Engineering Projects
The Engineering Projects
The Engineering Projects
The Engineering Projects
The Engineering Projects
```

```
Sketch uses 1642 bytes (5%) of program storage space. Maximum is 32256 bytes.
Global variables use 212 bytes (10%) of dynamic memory, leaving 1836 bytes for local variables. Maximum is 2048 bytes.
```

Fig 5.7: output of the serial monitor

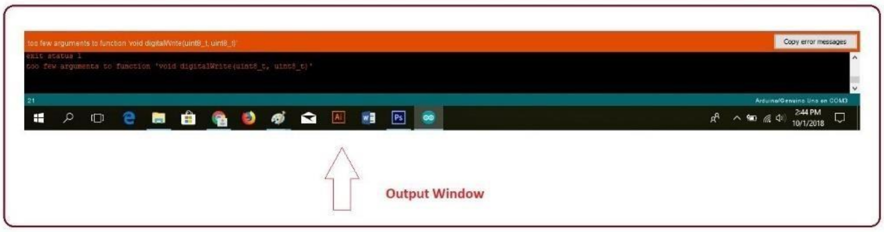
 The main screen below the Menu bard is known as a simple text editor used for writing the required code.



```
int Pin = 8; // Initializing Arduino Pin
int Writing;

void setup() {
  pinMode(Pin, OUTPUT); // Declaring Arduino Pin as an Output
}

void loop() {
  Writing = digitalWrite(Pin); // Writing status of Arduino digital Pin

  if(Writing == HIGH)
  {
    Serial.println("HIGH");
  }

  if(Writing == LOW)
  {
    Serial.println("LOW");
  }

}
```

Fig 5.8: Text editor

 Output Pane that mainly highlights the compilation status of the running code: the memory used by the code, and errors that occurred in the program. You need to fix the bottom of the main screen is described as those errors before you intend to upload the hex file into your Arduino Module.



```
too few arguments to function 'void digitalWrite(uint8_t, uint8_t)'
exit status 1
too few arguments to function 'void digitalWrite(uint8_t, uint8_t)'
```

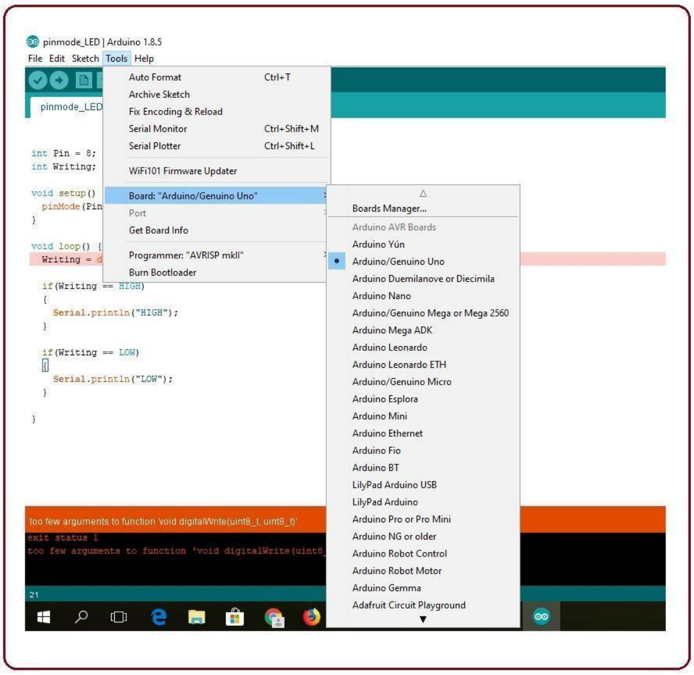
Fig 5.9: output window

 More or less, Arduino C language works similar to the regular C language used for any embedded system microcontroller, however, there are some dedicated libraries used for calling and executing specific functions on the board.

5.1.3 Libraries

- Libraries are very useful for adding extra functionality into the Arduino Module.
- There is a list of libraries you can check by clicking the Sketch button in the menu bar and going to Include Library.
- As you click the Include Library and Add the respective library it will be on the top of the sketch with a #include sign. Suppose, I Include the Liquid
  Crystal library, it will appear on the text editor as

include <Liquid Crystal. h>



```
int Pin = 8;
int Writing;

void setup()
  pinMode(Pin
}

void loop() {
  Writing = d

  if(Writing == HIGH)
  {
    Serial.println("HIGH");
  }

  if(Writing == LOW)
  {
    Serial.println("LOW");
  }

}
```

```
too few arguments to function 'void digitalWrite(uint8_t, uint8_t)'
exit status 1
too few arguments to function 'void digitalWrite(uint8_
```

Fig 5.10: Selection of tools

- As you click the Include Library and Add the respective library it will be on the top of the sketch with a #include sign. Suppose, I Include the Liquid
  Crystal library, it will appear on the text editor as

  #include <Liquid Crystal.h>

- Most of the libraries are preinstalled and come with the Arduino software.

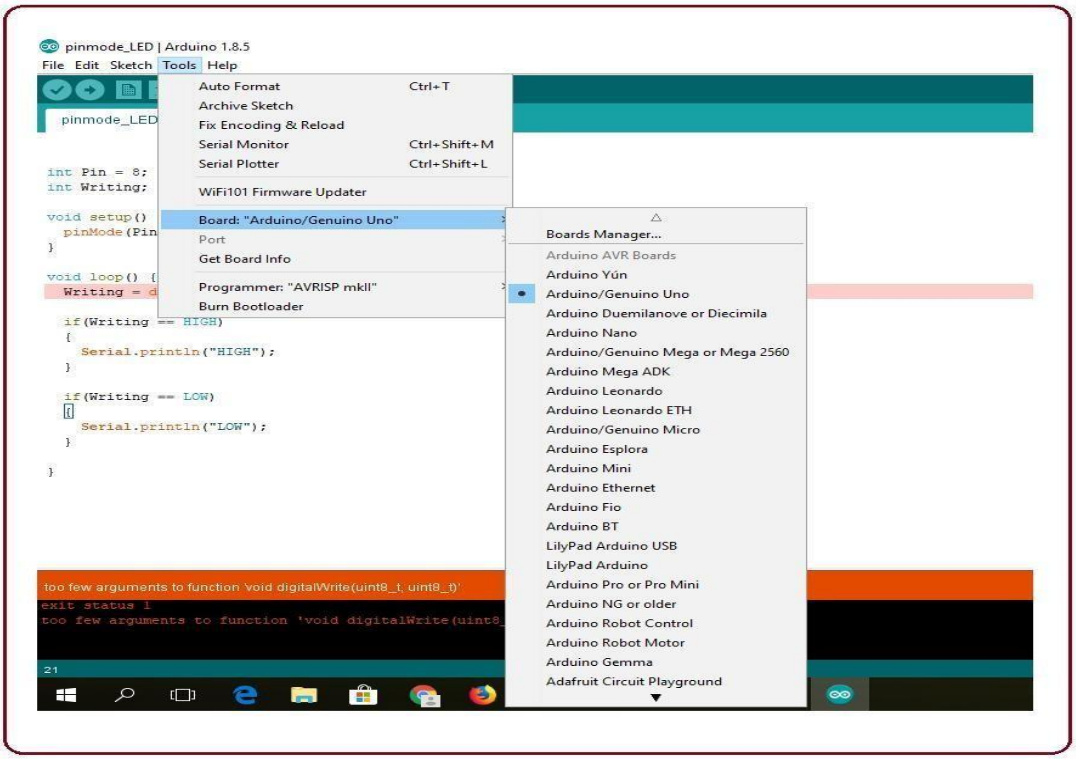
  However, you can also download them from external sources.

5.1.4 Making Pins Input or Output.

The digitalRead and digitalWrite commands are used for addressing and making the Arduino pins as an input and output respectively. These commands are text sensitive i.e., you need to write them down the exact way they are given like digitalWrite starting with small "d" and write with capital "W". Writing it down with DigitalWrite or digitalWrite won't be calling or addressing any function.

5.1.5 How to Select the Board

- In order to upload the sketch, you need to select the relevant board you are using and the ports for that operating system.
- As you click the Tools on the menu, it will open like the figure below:



```
int Pin = 8;
int Writing;

void setup()
  pinMode(Pin
}

void loop() {
  Writing = d

  if(Writing == HIGH)
  {
    Serial.println("HIGH");
  }

  if(Writing == LOW)
  {
    Serial.println("LOW");
  }

}
```

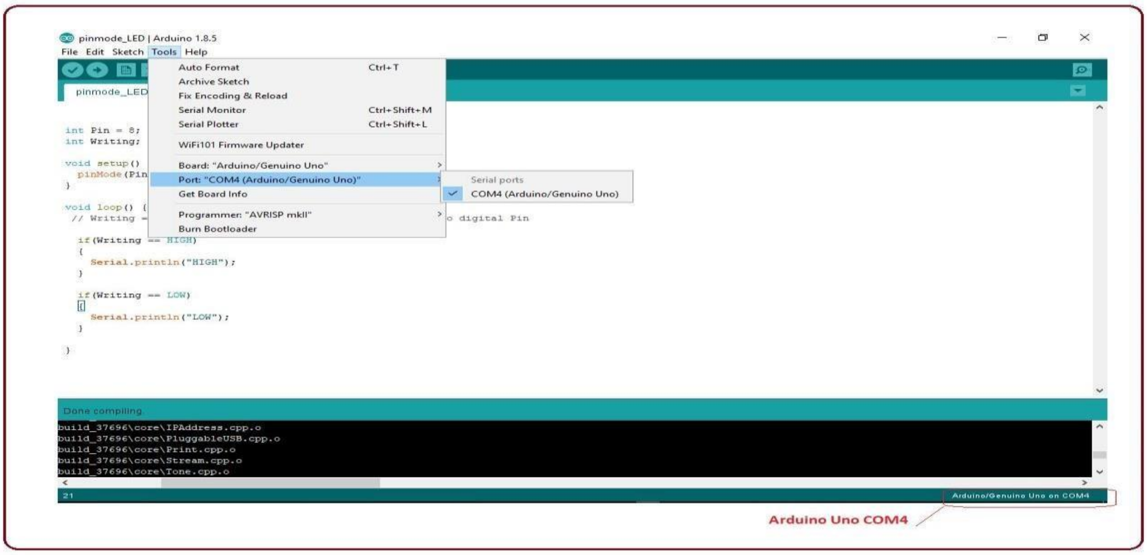
```
too few arguments to function 'void digitalWrite(uint8_t, uint8_t)'
exit status 1
too few arguments to function 'void digitalWrite(uint8_
21
```

Fig 5.11: Selection of board manager

- Just go to the "Board" section and select the board you aim to work on. Similarly, COM1, COM2, COM4, COM5, COM7 or higher are reserved for the serial and USB board. You can look for the USB serial device in the port section of the Windows Device Manager.
- The following figure shows the COM4 that I have used for my project, indicating the Arduino Uno with the COM4 port at the right bottom corner of the screen.
- After correct selection of both Board and Serial Port, click the verify and then upload button appearing in the upper left corner of the six-button section or you can go to the Sketch section and press verify/compile and then upload.
- The sketch is written in the text editor and is then saved with the file extension into. It is important to note that the recent Arduino Modules will reset automatically as you compile and press the

upload button the IDE software, however, the older versions may require the physical reset on the board.

- Once you upload the code, TX and RX LEDs will blink on the board, indicating the desired program is running successfully.


pinmode_LED | Arduino 1.8.5
File Edit Sketch Tools Help
pinmode_LED
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 Firmware Updater
Board: "Arduino/Genuino Uno"
Port: "COM4 (Arduino/Genuino Uno)"
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
Serial ports
COM4 (Arduino/Genuino Uno)
Done compiling.
21
Arduino/Genuino Uno on COM4
Arduino Uno COM4

```
int Pin = 8;
int Writing;

void setup()
  pinMode(Pin
}

void loop() {
 // Writing =                                          o digital Pin

  if(Writing == HIGH)
  {
    Serial.println("HIGH");
  }

  if(Writing == LOW)
  {
    Serial.println("LOW");
  }

}
```

```
build_37696\core\IPAddress.cpp.o
build_37696\core\PluggableUSB.cpp.o
build_37696\core\Print.cpp.o
build_37696\core\Stream.cpp.o
build_37696\core\Tone.cpp.o
```

Fig 5.12: Selection of port

Note: The port selection criteria mentioned above are dedicated to Windows operating system only, you can check this Guide if you are using MAC or
Linux.

The amazing thing about this software is that no prior arrangement or bulk of the mess is required to install this software, you will be writing your first program within 2 minutes after the installation of the IDE environment.

5.1.6 Uploading

After writing your code, click on the upload button which is above the window and the code will be directly uploaded into the Node MCU with a cable wire connector.

CHAPTER – 6

RESULT

The weather monitoring system, utilizing the ESP8266 Wi-Fi module, DHT11 sensor, rain sensor, LED, LCD, and buzzer, provides a comprehensive solution for real-time weather tracking and alerting. Here are the key results and findings from the system's operation:

1. **Real-Time Temperature and Humidity Monitoring**:

    - The DHT11 sensor accurately measures temperature and humidity. The data is displayed in real-time on the LCD screen, providing users with current environmental conditions.

    - Temperature readings are displayed with a range of 0 to 50°C, and humidity readings range from 20 to 90% RH. The system successfully updates these values every few seconds, offering timely insights into the weather conditions.

2. **High-Temperature Alert**:

    - The LED successfully activates when the temperature exceeds 40°C. This provides a clear visual indication that high-temperature conditions are present. The LED remains on as long as the temperature is above this threshold and turns off when it drops below 40°C.

3. **High-Humidity Alert**:

    - The buzzer sounds when the humidity exceeds 80%, providing an audible alert to notify users of high humidity levels. This feature ensures that users are immediately aware of elevated humidity conditions, which could be critical for applications requiring precise environmental control.

4. **Rain Detection**:

    - The rain sensor effectively detects the presence of rain. The LCD displays "Raining" when rain is detected and "No Raining" when there is no rain. This feature enables users to stay informed about current precipitation conditions.

5. **Integration and Functionality**:

- The ESP8266 Wi-Fi module allows the system to transmit weather data to remote servers or applications. This connectivity ensures that users can monitor weather conditions from anywhere with internet access.
- The integration of the LCD, LED, and buzzer provides a user-friendly interface for monitoring and responding to weather conditions. The system's alerts and displays work harmoniously to ensure clear communication of critical weather information.

6. **System Reliability and Efficiency**:
    - The system demonstrates reliability in monitoring and alerting functions. The components work seamlessly together, providing accurate data and timely alerts without significant delays.
    - The energy-efficient design, including low-power components and effective data transmission, ensures that the system operates sustainably over extended periods.

Overall, the weather monitoring system effectively combines real-time data collection, automated alerts, and user-friendly displays to provide a robust solution for environmental monitoring. Its ability to accurately measure and communicate temperature, humidity, and rain conditions makes it valuable for various applications, from agricultural management to disaster preparedness.

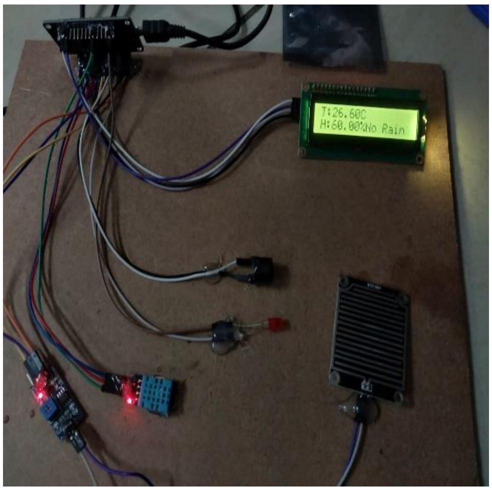


Fig 6.1:Circuit Diagram

CONCLUSION

The development and implementation of the weather monitoring system have proven to be highly effective in providing real-time, accurate, and reliable weather data. By integrating various IoT technologies, including the ESP8266 Wi-Fi module, DHT11 sensor, rain sensor, LED, LCD, and buzzer, the system successfully monitors temperature, humidity, and rainfall, offering immediate visual and audible alerts based on predefined conditions.

The system excels in accurate real-time monitoring, with the DHT11 sensor providing precise measurements of temperature and humidity, continuously updated and displayed on the LCD screen. This ensures users have access to the latest weather conditions at all times. The effective alert mechanisms, comprising the LED and buzzer, provide clear and immediate alerts for high-temperature and high-humidity conditions, respectively. Additionally, the rain sensor accurately detects rainfall and updates the LCD display to reflect the current precipitation status.

Remote accessibility is a significant advantage of the system, enabled by the ESP8266 Wi-Fi module, which allows data transmission to remote servers. This feature permits users to monitor weather conditions from any location, enhancing the system's utility for remote applications and long-term data analysis. The user-friendly interface, combining the LCD, LED, and buzzer, ensures that the system is easy to use and understand, providing real-time feedback and alerts that enable quick responses to changing weather conditions.

Furthermore, the system is designed to be energy-efficient, ensuring sustainable operation over extended periods. The reliability of the components and their seamless integration ensure consistent performance, making the weather monitoring system a versatile tool for various applications, including agricultural management, disaster preparedness, and general environmental monitoring. Its ability to provide real-time data and alerts helps users make informed decisions and take timely actions in response to weather changes. The project's success demonstrates the potential of IoT technologies to enhance weather monitoring and management, contributing to improved safety, efficiency, and overall quality of life.

FUTURE SCOPE

The weather monitoring system presents several promising avenues for future development and enhancements. One significant area for expansion is the integration of additional sensors to measure other environmental parameters such as wind speed, atmospheric pressure, and UV radiation. Incorporating these sensors would provide a more comprehensive understanding of weather conditions, making the system even more valuable for various applications.

Another potential improvement is the enhancement of data analytics capabilities. By implementing advanced data analytics and machine learning algorithms, the system could predict weather trends and provide more accurate forecasts. This would be particularly beneficial for agricultural planning, disaster management, and other sectors that rely heavily on weather predictions.

Improving the system's connectivity and integration with other IoT devices and platforms is also a key area for future development. This could involve creating a more robust cloud infrastructure for data storage and analysis, enabling seamless integration with smart home systems, and providing users with a centralized platform to monitor and manage all their IoT devices. Enhanced connectivity would also facilitate better data sharing and collaboration among different stakeholders, such as meteorological departments and research institutions.

Additionally, developing a mobile application or web interface for the weather monitoring system would significantly enhance user accessibility and convenience. Such an interface could provide real-time alerts, historical data analysis, and customizable notifications, making it easier for users to stay informed and respond promptly to changing weather conditions.

Finally, exploring the use of renewable energy sources, such as solar power, to power the system would enhance its sustainability and reduce its environmental impact. This would be particularly useful for deploying the system in remote or off-grid locations, where access to conventional power sources may be limited.

Overall, the future scope of the weather monitoring system is vast, with numerous opportunities for technological advancements and enhanced functionality. By continuously innovating and expanding its capabilities, the system can become an even more powerful tool for weather monitoring, prediction, and management, contributing to improved safety, efficiency, and environmental stewardship.

SOURCE CODE

```
 #include <ESP8266WiFi.h>

 #include "secrets.h"

 #include "ThingSpeak.h" //
always include thingspeak
header file after other header
files and custom macros



 char ssid[] = "CSE_A5";   //
your network SSID (name)

 char pass[] = "chinni60";   //
your network password

 int keyIndex = 0;            //
your network key Index
number (needed only for
WEP)

 WiFiClient  client;



 unsigned long
myChannelNumber =
2599715;

 const char *
myWriteAPIKey =
"3KETRX1OC4SF3H68";
```

```
 // Initialize our values
 int number1 = 0;
 int number2 =
random(0,100);
 int number3 =
random(0,100);
 int number4 =
random(0,100);
 String myStatus = "";
 #include <DHT.h>
 #include <Wire.h>
 #include
<LiquidCrystal_I2C.h>


  const int DHTPIN = D3;
  const int DHTTYPE =
DHT11;
  const int
RAIN_SENSOR_PIN = D5;
  const int BUZZERPIN =
D7;
  const int RED_LED_PIN =
D8;
```

```
 DHT dht(DHTPIN,
DHTTYPE);

 LiquidCrystal_I2C
lcd(0x27, 16, 2);



 void setup() {

   Serial.begin(115200);

   dht.begin();

   pinMode(BUZZERPIN,
OUTPUT);

   pinMode(RED_LED_PIN,
OUTPUT);


pinMode(RAIN_SENSOR_
PIN, INPUT);



   lcd.begin();

   lcd.backlight();

   Serial.begin(115200);  //
Initialize serial

   while (!Serial) {

     ; // wait for serial port to
connect. Needed for
Leonardo native USB port
only
```

```
   }

   WiFi.mode(WIFI_STA);
   ThingSpeak.begin(client);
// Initialize ThingSpeak
 }

 void loop() {
    delay(2000);
   float h =
dht.readHumidity();
   float t =
dht.readTemperature();
   if (isnan(h) || isnan(t)) {
     Serial.println("Failed to
read from DHT sensor!");
     return;
   }

   int isRainDetected =
digitalRead(RAIN_SENSO
R_PIN);

   Serial.print("Humidity: ");
```

```
  Serial.print(h);

  Serial.print(" %\t");

  Serial.print("Temperature: 
");

  Serial.print(t);

  Serial.println(" *C");

  Serial.print("Rain Status: 
");


Serial.println(isRainDetecte
d ? "Rain Detected" : "No 
Rain");



  lcd.clear();

  lcd.setCursor(0, 0);

  lcd.print("T:");

  lcd.print(t);

  lcd.print("C");

  lcd.setCursor(0, 1);

  lcd.print("H:");

  lcd.print(h);

  lcd.print("%");

  if(isRainDetected==1)
  {
  lcd.setCursor(8, 1);
  lcd.print("No Rain");
  }
  else
  {
  lcd.setCursor(8, 1);
  lcd.print("Rain");
  }
  if (t > 40) {

digitalWrite(BUZZERPIN,
HIGH);
  } else {

digitalWrite(BUZZERPIN,
LOW);
  }



  if (h > 80) {

digitalWrite(RED_LED_PI
N, HIGH);
```

```
  } else {

digitalWrite(RED_LED_PI
N, LOW);
  }



  // Connect or reconnect to
WiFi
  if(WiFi.status() !=
WL_CONNECTED){
    Serial.print("Attempting
to connect to SSID: ");

Serial.println(SECRET_SSI
D);
    while(WiFi.status() !=
WL_CONNECTED){
      WiFi.begin(ssid, pass);
// Connect to WPA/WPA2
network. Change this line if
using open or WEP network
      Serial.print(".");
      delay(5000);
    }
```

```
Serial.println("\nConnected.
");
  }


  // set the fields with the
values
  ThingSpeak.setField(1, t);
  ThingSpeak.setField(2, h);
  ThingSpeak.setField(3,
isRainDetected);
  ThingSpeak.setField(4,
number4);


  // figure out the status
message
  if(number1 > number2){
    myStatus = String("field1
is greater than field2");
  }
  else if(number1 <
number2){
    myStatus = String("field1
is less than field2");
  }
  else{
    myStatus = String("field1
equals field2");
  }


  // set the status

ThingSpeak.setStatus(mySta
tus);


  // write to the ThingSpeak
channel
  int x =
ThingSpeak.writeFields(my
ChannelNumber,
myWriteAPIKey);
  if(x == 200){
    Serial.println("Channel
update successful.");
  }
  else{
    Serial.println("Problem
updating channel. HTTP
error code " + String(x));
  }
```

```
   // change the values

   number1++;

   if(number1 > 99){

     number1 = 0;

   }

   number2 = random(0,100);

   number3 = random(0,100);

   number4 = random(0,100);


   delay(20000); // Wait 20
seconds to update the
channel again

 }
```

REFERENCES

[1] **Misra, S., Maheswar, P., & Hashmi, S. M. (2020). "IoT-based real-time weather monitoring system".** In *International Conference on Computing and Network Communications* (pp. 61-66). IEEE.

[2] **Patel, D. R., & Tiwari, N. (2017). "A review on IoT based smart weather monitoring system".** In *International Journal of Advanced Research in Computer and Communication Engineering*, 6(12), 116-120.

[3] **Shinde, P., & Gawali, D. (2018). "IoT-based smart agriculture monitoring system".** In *International Journal of Advanced Research in Computer and Communication Engineering*, 7(6), 20-25.

[4] **Bhardwaj, A., & Singh, M. (2020). "Cloud-based weather monitoring and forecasting using IoT and machine learning".** In *International Journal of Engineering Research and Technology*, 9(5), 221-226.

[5] **Rathore, M. M., Ahmad, A., Paul, A., & Rho, S. (2016). "Urban planning and building smart cities based on the internet of things using big data analytics".** In *Computer Networks*, 101, 63-80.

[6] **Kumar, A., & Kumar, P. (2019). "IoT based weather monitoring system using cloud computing".** In *International Journal of Scientific & Technology Research*, 8(12), 2903-2907.

[7] **Chen, T., & Zhang, Y. (2018). "Design of an IoT-based weather monitoring and forecasting system".** In *Journal of Physics: Conference Series*, 1074(1), 012055.

[8] **Ali, M. M., & Alam, M. M. (2020). "A low-cost IoT based weather monitoring system using Arduino and cloud storage".** In *Journal of Innovation and Research in Electrical, Electronics, Instrumentation, and Control Engineering*, 8(2), 123-127.