Automatic controlling system of pump based on temperature and moisture conditions along with Monitoring using IOT

A Project Work

Submitted in partial fulfilment of Requirements for the award of the

Degree of

BACHELOR OF TECHNOLOGY

in

ELECTRONICS AND COMMUNICATION ENGINEERING

by

N. YASASWINI (208T1A0482)
P. SOWMYA SRI SYAMALA (208T1A0496)
P. DIVYA SRI SARITHA (208T1A0489)
U. BHAVANI SWAMY (208T1A04B8)

Under the Esteemed guidance of

Dr. K. SRINIVASA RAO
Professor, Dept. Of ECE

Department of Electronics and Communication Engineering

DHANEKULA INSTITUTE OF ENGINEERING &TECHNOLOGY
GANGURU, VIJAYAWADA -521 139
Affiliated to JNTUK, Kakinada & Approved by AICTE, New Delhi
Certified by ISO 9001-2015, Accredited by NBA
APRIL-2024

DHANEKULA INSTITUTE OF ENGINEERING & TECHNOLOGY

GANGURU, VIJAYAWADA -521 139

Affiliated to JNTUK, Kakinada & Approved by AICTE, New Delhi

Certified by ISO 9001-2015, Accredited by NBA

Department of Electronics & Communication Engineering

CERTIFICATE

This is to certify that the project work entitled **"Automatic system of pump based on temperature and moisture conditions along with Monitoring using IOT"** is a bonafide record of project work done jointly by N.YASASWINI (**208T1A0482**), P.SOWMYA SRI SYAMALA (**208T1A0496**), P.DIVYA SRI SARITHA (**208T1A0489**),U.BHAVANI SWAMY (**208T1A04B8**), under my guidance and supervision and is submitted in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Electronics & Communication Engineering by **Jawaharlal Nehru Technological University, Kakinada during the academic year 2023-2024**.

PROJECT GUIDE	HEAD OF THE DEPARTMENT
Dr. K. Srinivasa Rao	Dr. M. Vamshi Krishna
Professor	Professor & H.O.D
Department of ECE	Department of ECE

External Examiner

ACKNOWLEDGEMENT

First and foremost, we sincerely salute our esteemed institution **DHANEKULA INSTITUTE OF ENGINEERING AND TECHNOLOGY** for giving us this opportunity for fulfilling our project.

We express our sincere thanks to our beloved Principal **Dr. Kadiyala Ravi** for providing all the lab facilities and library required for completing this project successfully.

This project work would not have been possible without the support of many people. We wish to express our gratitude to **Dr. M. Vamshi Krishna**, Ph.D., H.O.D. ECE Department for constant support and valuable knowledge in the successful completion of this project.

We are glad to express our deep sense of gratitude to **Dr. K. Srinivasa Rao**, Professor, for abundant help and offered invaluable assistance, support, and guidance throughout this work.

We thank one and all who have rendered help directly or indirectly in the completion of this project successfully.

PROJECT ASSOCIATES:

N. YASASWINI (208T1A0482)

P. SOWMYA SRI SYAMALA (208T1A0496)

P. DIVYA SRI SARITHA (208T1A0489)

U. BHAVANI SWAMY (208T1A04B8)

DECLARATION

I declare that this project report titled "**Automatic controlling system of pump based on temperature and moisture conditions along with Monitoring using IOT"** is submitted in partial fulfilment of the degree of **B. Tech in Electronics and Communication Engineering** is a record of original work carried out by us under the supervision of **Dr.K. Srinivasa Rao** and has not formed the basis for the award of any other degree or diploma, in this or any other Institution or University. In keeping with the ethical practice in reporting scientific information, due acknowledgments have been made wherever the findings of others have been cited.

N. YASASWINI
(208T1A0482)

P. SOWMYA SRI SYAMALA
(208T1A0496)

P. DIVYA SRI SARITHA
(208T1A0489)

U. BHAVANI SWAMY
(208T1A04B8)

Ganguru

TABLE OF CONTENTS

DHANEKULA INSTITUTE OF ENGINEERING AND TECHNOLOGY

Department of Electronics & Communications Engineering

VISION-MISSION-PEOs

VISION/MISSION/PEOs

Institute Vision	Pioneering Professional Education through Quality
Institute Mission	• Providing Quality Education through state-of-art infrastructure, laboratories, and committed staff. • Moulding Students as proficient, competent, and socially responsible engineering personnel with ingenious intellect. • Involving faculty members and students in research and development works for the betterment of society.
Department Vision	Be a model in the arena of Electronics and Communication Engineering Education &amp; Research to Elevate Rural Community
Department Mission	• Imparting professional education endowed with ethics and human values to transform students to be competent and committed electronics engineers. • Adopting best pedagogical methods to maximize knowledge transfer. • Having adequate mechanisms to enhance understanding of theoretical concepts through practice. • Establishing an environment conducive for lifelong learning and entrepreneurship development. • To train as effective innovators and deploy new technologies for service of society.
Program Educational Objectives (PEO's)	PEO1: Shall have professional competency in electronics and communications with strong foundation in science, mathematics and basic engineering. PEO2: Shall design, analyze and synthesize electronic circuits and simulate using modern tools. PEO3: Shall Discover practical applications and design innovative circuits for Lifelong learning PEO4: Shall have effective communication skills and practice the ethics consistent with a sense of social responsibility.

DHANEKULA INSTITUTE OF ENGINEERING AND TECHNOLOGY

Department of Electronics & Communications Engineering

POs/PSOs

PROGRAM OUTCOMES

1	**Engineering Knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and engineering programs.
2	**Problem analysis**: Identify, formulate, review research literature, and analyze complex Engineering problems reaching substantiated conclusions using first principles of Mathematics, natural sciences, and engineering sciences
3	**Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, and societal considerations.
4	**Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis, and synthesis of the information to provide valid conclusions.
5	**Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6	**The engineer and society**: Apply to reason informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues, and the consequent responsibilities relevant to the professional engineering practice.
7	**Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8	**Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9	**Individual and teamwork**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10	**Communication**: Communicate effectively on complex engineering activities with the Engineering community and with society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11	**Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12	**Lite-long learning**: Recognize the need for and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSO's)

PSO1: Make use of specified software tool for designing and developing VLSI and Embedded Systems.

PSO2: Innovate and Design application specific electronic circuits for modern wireless communication.

Project vs. POs Mapping

Project title	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
Implementation of Automated Koha Library Management System using Cloud Platform	2	3	3	3	3	2	3	-	3	3	3	3

3-High 2-Medium 1-Low

Justification of Mapping of Project with Program Outcomes:

1. The knowledge of mathematics, science, engineering fundamentals, and engineering Programs are strongly correlated to all course outcomes.
2. Application of Ethical principles is not correlated to all course outcomes.

Project vs. PSOs Mapping

Project Title	PSO1	PSO2
Implementation of Automated Koha Library Management System using Cloud Platform	2	3

3-High 2-Medium 1-Low

Justification of Mapping of Project with Program Specific Outcomes:

This project is related to Communications area, which helps to expertise in the Corresponding area by applying engineering fundamentals are correlated to all course outcomes.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ABBREVATION	DESCRIPTION
DHT	Digital Humidity and Temperature
LCD	Liquid Crystal Display
IDLE	Integrated Development and Learning Environment

ABSTRACT

The aim of the project “Automatic controlling system of pump based on temperature and moisture conditions along with Monitoring using IOT” is to use IoT technology to offer an effective irrigation solution for agriculture. The primary controller of the system is an Arduino board, and sensors like the DHT11 and soil moisture sensors are used to measure critical variables including temperature, humidity, and soil moisture. The sensor values are displayed on an LCD screen for farmers to easily monitor. To automate the irrigation operation, a relay and pump are also incorporated into the system.

In order to make sure that crops receive water when necessary, the relay triggers the pump when the temperature rises beyond a predetermined threshold and the soil moisture sensor detects dry conditions. The irrigation system's connectivity and operation are improved with the inclusion of Python code, which sends Moisture and Motor status to Gmail. This enables farmers to use their smartphones to monitor the irrigation system. Farmers can obtain up-to-date information about crop conditions and environmental factors affecting them in real-time by uploading sensor data.

As per the central water commission, in India the total consumption of water is around 84% for agriculture. Out of 84% around 70% of irrigation water is wasted due to the poor management of the farmer. It is observed that many crops are water intensive crops like rice, paddy and few required very less amount of water for growth like lentils. To avoid wastage of water a smart irrigation system is required.

This proposed system gives the adequate water to the plants for their growth using an intelligent system. It also required Arduino Uno, Relay module and pump. This system precisely monitors, controls and fulfills the requirement of water. The proposed system is tested on the field and it gives satisfactory results. It reduces around 80% of wastage water as compared to the conventional method. It also saves electricity for farmers. In future, Internet of Things (IoT) is used for transmitting system data to the registered mobile phone of the farmer using a cloud database.

CHAPTER 1

INTRODUCTION

1.1. Introduction

A sustainable irrigation system for farmers is what the project " Automatic controlling system of pump based on temperature and moisture conditions along with the monitoring using IOT" aims to solve Conventional irrigation techniques frequently require human intervention and can cause crops to be either overwatered or underwatered, wasting resources and producing lower yields. This idea offers a creative solution that uses Internet of Things technology to automate and improve the irrigation process in response to these difficulties.

The system intends to give farmers real-time insights into soil conditions and water requirements by integrating sensors, controls, and data transfer capabilities, ultimately enabling more accurate and efficient irrigation management. Arduino microcontrollers, which act as the core processing units for obtaining and processing data from a variety of sensors, are the fundamental component of the suggested system. In order to gather important environmental data, these sensors—which include DHT11 sensors for temperature and humidity monitoring and soil moisture sensors—are placed strategically across the agricultural areas. The device can intelligently decide when and how much water to supply the crops by continuously monitoring these data. By proactively conserving water and fostering healthy crop growth and plant development, this proactive irrigation strategy maximizes agricultural output and sustainability Agriculture is undoubtedly the backbone of India as most of the population is involved in agricultural production. India is currently the most populous country in the world and is still increasing rapidly which also creates a need for increase in production of agricultural products. As the population increases in proportion the need of more agricultural produce also increases which means ultimately more fresh water for irrigation. Wastage of fresh water has been found recurring whenever the irrigation is unplanned. This creates the necessity to develop new and better irrigation plans without pressuring the farmers. It has been 15 years that the farmers have started to embrace the technology and started using software to keep track of all the data they require during agriculture to maximize the profit and minimize the loss. In today's scenario information is the most valuable thing, and agriculture is becoming a bigger chunk of it rapidly, in this the farmers have to collect data from many sensors and farming machinery parts.

1.2. Importance Of Automatic controlling system of pump based on temperature and moisture conditions along with Monitoring using IOT

Through IoT integration, the system enables automated control of the pump based on predefined parameters. By analyzing sensor data, including humidity levels and solar panel activity, the system autonomously adjusts irrigation operations, reducing manual intervention and improving overall efficiency.

IoT-enabled temperature and humidity sensors provide real-time monitoring and control of environmental conditions. These sensors collect data and transmit it to a centralized system, allowing users to remotely access the information.

1.2.1. This include:

1. Sensing and Control System:

- Temperature and Moisture Sensors: Install sensors in the soil to measure temperature and moisture levels accurately.
- Microcontroller: Utilize a microcontroller (such as Arduino or Raspberry Pi) to process sensor data and control the pump accordingly.
- pump Control: Develop algorithms that determine when to activate the pump based on predefined temperature and moisture thresholds. For example, if the soil moisture is below a certain level and the temperature is within a specific range, activate the pump to irrigate the soil.

2. IoT Integration:

- Connectivity: Connect the microcontroller to the internet using Wi-Fi or other communication protocols.
- Cloud Platform: Send sensor data to a cloud platform (like AWS IoT, Google Cloud IoT, or Azure IoT) for storage and analysis.
- Mobile App/Website: Develop a user interface (app or website) where users can monitor real-time data, set custom thresholds, and receive alerts.

3. Public Awareness:

- Education Campaigns: Start educational initiatives to increase the general public's understanding of the significance of effective water use in agriculture.
- Workshops and Seminars: Provide farmers' and gardeners with training on how to operate the system efficiently through workshops and seminars.
- Community Involvement: Promote community involvement by planning activities and exhibits that highlight the advantages of technology.

4. Health Innovation:

- Environmental Impact: Emphasize how efficient water usage can positively impact the environment by conserving resources and reducing water wastage.
- Crop Health: Highlight how maintaining optimal moisture levels can improve crop health and yield, ultimately benefiting both farmers and consumers.
- Remote Monitoring: Enable remote monitoring of soil conditions, allowing farmers to detect issues early and take preventive measures, thus reducing the need for harmful pesticides and fertilizers

5. Data Analytics and Insights:

- Data Analysis: Use historical data collected from the IoT platform to identify patterns, optimize watering schedules, and improve overall system efficiency.
- Predictive Analytics: Implement predictive models to anticipate future watering needs based on weather forecasts and historical trends.
- Feedback Loop: Continuously gather feedback from users to refine the system and address any issues or concerns.

1.3. Features of Automatic controlling system of pump based on temperature and moisture conditions along with Monitoring using IOT

1.Real-time Monitoring: Continuous monitoring of temperature and moisture levels in the soil using sensors to provide real-time data.

2.Threshold Alerts: Set customizable thresholds for temperature and moisture levels. Receive alerts via SMS, email, or push notifications when conditions deviate from the desired range.

3.Remote Control: Ability to remotely control the pump system through a mobile app or web interface. Users can manually activate or deactivate the pump based on real-time data or preset schedules.

4.Data Logging and Analysis: Collect and store historical data on temperature, moisture, and pump usage. Analyze trends over time to optimize watering schedules and resource usage.

5.Predictive Analytics: Utilize machine learning algorithms to predict future moisture requirements based on historical data, weather forecasts, and plant types.

6.Energy Efficiency: Implement energy-saving features such as variable speed pumps or solar-powered systems to reduce energy consumption.

7.Integration with Weather Data: Integrate weather forecasts and local weather data to adjust watering schedules dynamically based on upcoming weather conditions.

8.User-friendly Interface: Develop a user-friendly interface for both desktop and mobile devices, allowing users to easily visualize sensor data, set preferences, and receive notifications.

9.Security: Implement robust security measures to protect data privacy and prevent unauthorized access to the system.

10.Scalability: Design the system to be scalable, allowing for easy expansion to accommodate larger areas or additional sensors as needed.

11.Compatibility: Ensure compatibility with different types of pumps and irrigation systems to accommodate various agricultural setups.

12.Documentation and Support: Provide comprehensive documentation and customer support to assist users with setup, troubleshooting, and maintenance.

1.3.1. Characteristics of Automatic controlling system of pump based on temperature and moisture conditions along with Monitoring using IOT

1.Sensing Capabilities:

Utilizes temperature and moisture sensors to accurately measure environmental conditions in the soil.

2.Automation:

Automatically activates or deactivates the pump based on predefined thresholds for temperature and moisture levels.

3.Real-time Monitoring:

Provides real-time data on temperature, moisture, and pump status, allowing users to monitor conditions remotely.

4.Customizable Thresholds:

Allows users to set customizable thresholds for temperature and moisture levels, with the ability to adjust settings as needed.

5.Alert System:

Sends alerts to users via various channels (SMS, email, push notifications) when conditions deviate from the desired range.

6.Remote Control:

Enables users to remotely control the pump system through a mobile app or web interface, providing flexibility and convenience.

7.Data Logging:

Logs historical data on temperature, moisture, and pump usage, facilitating analysis and optimization of watering schedules.

8.Predictive Analytics:

Uses historical data and machine learning algorithms to predict future moisture requirements and optimize watering schedules.

9.Energy Efficiency:

Incorporates energy-saving features such as variable speed pumps or solar-powered systems to minimize energy consumption.

1.3.2. Significance of Automatic controlling system of pump based on temperature and moisture conditions along with Monitoring using IOT

1. Water Conservation: By automatically activating the pump only, when necessary, based on soil moisture levels, the system helps conserve water resources by avoiding over-irrigation. This is particularly crucial in regions facing water scarcity or drought conditions.
2. Optimized Crop Growth: Maintaining optimal moisture levels in the soil promotes healthy plant growth and development. The system ensures that crops receive the right amount of water at the right time, leading to improved yields and quality of produce.
3. Resource Efficiency: By utilizing IoT sensors and data analytics, the system optimizes water usage and energy consumption, leading to more efficient resource utilization in agriculture.
4. Labor Savings: Automating the pump control process reduces the need for manual monitoring and intervention, saving farmers time and labor. They can remotely monitor and manage irrigation systems, freeing up time for other essential tasks.
5. Environmental Impact: Efficient irrigation practices supported by the system help minimize runoff and leaching of fertilizers and pesticides, reducing environmental pollution and preserving soil and water quality.
6. Risk Mitigation: The system provides early detection of moisture stress or temperature extremes, allowing farmers to take proactive measures to mitigate risks such as crop wilting, disease outbreaks, or frost damage.
7. Data-Driven Decision Making: IoT-enabled monitoring generates valuable data on soil conditions, crop performance, and environmental factors. This data can be analyzed to gain insights into agricultural processes, optimize irrigation strategies, and improve overall farm management practices.
8. Scalability and Adaptability: The system can be scaled up or down to accommodate different farm sizes and crop types. It can also be adapted to various environmental conditions and geographic locations, making it versatile and applicable across diverse agriculture.
9. Public Awareness and Education: Implementing modern agricultural technologies like IoT-based irrigation systems raises public awareness about sustainable farming practices and the importance of adopting innovative solutions to address global challenges such as food security and climate change.

10. Economic Benefits: Improved crop yields, reduced water and energy costs, and enhanced operational efficiency contribute to the economic viability of farming operations, benefiting both farmers and consumers in the long run.

1.3.3. Advantages of an Automatic controlling system of pump based on temperature and moisture conditions along with Monitoring using IOT

1.Water Conservation: Optimizes water usage by delivering the precise amount of water needed based on real-time soil moisture conditions, reducing water wastage and promoting sustainable irrigation practices.

2.Energy Efficiency: Reduces energy consumption by operating the pump only when necessary, resulting in lower electricity costs and reduced carbon emissions, especially if coupled with energy-efficient pump technologies or renewable energy sources.

3.Improved Crop Health and Yield: Maintains optimal soil moisture levels, which is essential for promoting healthy root development, nutrient uptake, and overall plant growth. This leads to improved crop yields, quality, and resilience to environmental stresses.

4.Labor Savings: Automates the irrigation process, eliminating the need for manual monitoring and intervention. Farmers can remotely monitor and control irrigation systems, saving time, labor, and resources.

5.Remote Monitoring and Control: Enables farmers to monitor soil conditions and manage irrigation systems remotely via mobile apps or web interfaces, providing flexibility and convenience, especially for large-scale or remotely located farms.

6.Predictive Maintenance: Provides early detection of potential issues such as pump failures or irrigation system malfunctions, allowing for timely maintenance and minimizing downtime, thereby ensuring continuous operation and crop productivity.

7.Data-driven Decision Making: Collects and analyzes data on soil moisture, temperature, and other environmental factors, enabling farmers to make informed decisions about irrigation scheduling, crop management, and resource allocation for optimized farm productivity.

8.Scalability and Adaptability: Can be scaled up or down to accommodate different farm sizes, crop types, and environmental conditions.

The system can also be tailored to specific crop water requirements, soil types, and local climate patterns.

9.Environmental Sustainability: Reduces the environmental impact of agriculture by minimizing water usage, runoff, and leaching of fertilizers and pesticides into water bodies, contributing to soil and water conservation and ecosystem health.

10.Integration with Precision Agriculture: Integrates with other precision agriculture technologies such as GPS-guided tractors, drones, and crop monitoring systems to create a holistic approach to farm management, optimizing resource use and maximizing productivity.

11.Cost Savings: Reduces water and energy costs, lowers operational expenses associated with manual labor and maintenance, and increases crop yields and revenue potential, resulting in overall cost savings and improved profitability for farmers.

12.Public Awareness and Reputation: Demonstrates a commitment to sustainable farming practices and environmental stewardship, enhancing the public perception of the farm operation and potentially attracting environmentally conscious consumers and investors.

1.3.4. Disadvantages of an Automatic controlling system of pump based on temperature and moisture conditions along with Monitoring using IOT

1.Initial Cost: Implementing IoT-based systems involves upfront costs for purchasing sensors, controllers, communication devices, and software.

The initial investment may be a barrier for small-scale farmers or those with limited financial resources.

2.Complexity: Setting up and configuring an IoT-enabled system requires technical expertise in sensor installation, network setup, data management, and software integration.

Farmers without adequate knowledge or training may struggle to implement and maintain it effectively

3.Reliability and Maintenance: IoT devices, sensors, and communication networks are susceptible to malfunctions, connectivity issues, and software bugs.

Regular maintenance, software updates, and troubleshooting are necessary to ensure the reliability and performance of the system.

4.Data Security and Privacy Concerns: Transmitting sensitive agricultural data over the internet raises concerns about data security and privacy.

Unauthorized access, data breaches, or cyberattacks could compromise the confidentiality and integrity of farm-related information.

5.Dependency on Technology: Overreliance on IoT technology for farm management may leave farmers vulnerable to disruptions caused by power outages, network failures, or technical glitches. Without backup systems or manual alternatives, critical farming operations could be affected.

6.Compatibility Issues: Integrating IoT devices from different manufacturers or vendors may result in compatibility issues, interoperability challenges, or data integration problems. Standardization efforts are necessary to ensure seamless connectivity and interoperability across diverse systems.

7.Data Overload: IoT sensors generate vast amounts of data, which can overwhelm farmers with irrelevant or redundant information.

Without proper data management and analytics tools, extracting actionable insights from the data may be difficult, leading to information overload.

8.Limited Connectivity in Remote Areas: Farmers in remote or rural areas with poor internet connectivity may struggle to implement IoT-based systems effectively.

Limited access to high-speed internet or cellular networks could hinder real-time monitoring and remote control capabilities.

9.Environment Factors: Environmental factors such as extreme weather conditions, physical obstructions, or electromagnetic interference can disrupt wireless communication signals and degrade sensor performance, affecting the reliability and accuracy of data collection.

10.Regulatory Compliance: Adhering to regulatory requirements related to data privacy, environmental regulations, and agricultural practices may pose challenges for farmers using IoT technologies.

Compliance with legal and regulatory frameworks adds complexity and administrative burdens to farm operations.

11.Skills Gap: The adoption of IoT-based technologies requires farmers to acquire new skills in data analysis, technology management, and digital literacy.

Bridging the skills gap and providing training and support are essential for successful implementation and adoption of IoT solutions.

1.3.5. Conclusion:

In conclusion, the implementation of an automatic controlling system of a pump based on temperature and moisture conditions, integrated with IoT monitoring, represents a significant advancement in modern agriculture. This technology offers a plethora of benefits, including water conservation, energy efficiency, improved crop health and yield, labor savings, and data-driven decision-making. By leveraging real-time data and automation, farmers can optimize irrigation practices, reduce resource wastage, and enhance farm productivity in a sustainable manner.

However, it's essential to acknowledge the potential disadvantages associated with IoT-based systems, such as initial costs, complexity, reliability concerns, and data security issues. Overcoming these challenges requires investment in technology infrastructure, skills development, and regulatory compliance.

Despite these challenges, the potential of automatic controlling systems of pumps based on temperature and moisture conditions, along with IoT monitoring, to revolutionize agriculture and contribute to food security, environmental sustainability, and economic prosperity is undeniable. With continued innovation, collaboration, and adoption of best practices, IoT-enabled agriculture holds promise for addressing the evolving needs of farmers, consumers, and the planet.

CHAPTER 2

LITERATURE SURVEY

Existing literature on the design and development of Automatic controlling system of pump based on temperature and moisture conditions along with Monitoring using IOT, as well as other factors relevant to the study's aims, has been carefully analyzed in detail, with both print and electronic materials were included.

IOT Based intelligent irrigation support system for smart farming applications

Neha Kailash Nawandar and Vishalentils Satpute

The IoT-based intelligent irrigation support system holds tremendous potential for revolutionizing agriculture and advancing smart farming applications. By integrating IoT sensors, data analytics, and automation, this system enables farmers to make informed decisions, optimize water usage, and enhance crop productivity.

The benefits of this technology include water conservation, energy efficiency, improved crop yields, and sustainable farming practices.

However, further research and development are necessary to refine the system's capabilities, address implementation challenges, and meet the evolving needs of the agricultural industry. Overall, the IoT-based intelligent irrigation support system represents a significant step forward in modernizing agriculture and promoting environmental sustainability in farming operations.

The IoT-based intelligent irrigation support system offers immense potential to revolutionize agriculture by promoting water conservation, enhancing crop productivity, and reducing environmental impact.

By harnessing the power of IoT technology, farmers can make informed decisions and optimize irrigation practices for sustainable and profitable farming operations. Further research and development are needed to refine the system's capabilities and address the evolving needs of the agricultural industry.

This paper presents an irrigation management system with sensor data fetching and compression, compressed data Transfer, data processing and decision making and active invoke capabilities. A Network of sensor capabilities.

IOT Based Smart Irrigation System

Srishti Rawal

The IoT-based smart irrigation system offers a transformative solution for optimizing water usage in agriculture. By leveraging IoT sensors, data analytics, and automation, this system enables real-time monitoring, precise control over irrigation processes, and data-driven decision-making.

The benefits include improved crop yields, water conservation, energy efficiency, and sustainable farming practices. As a result, the IoT-based smart irrigation system holds great promise for revolutionizing agriculture and addressing the challenges of water scarcity and environmental sustainability in farming operations.

M. A. Baharudin

“Smart home garden Irrigation system using raspberry pi” in 2017 IEEE Malaysia international conference on communications (MICC),28-30 Nov.2017, the P uteri pacific, Johor B ahru, Malaysia the development and implementation of the smart home garden irrigation system using Raspberry Pi offer a promising solution for efficient and automated gardening practices. By leveraging Raspberry Pi technology, this system provides homeowners with the ability to monitor and control their garden irrigation remotely, thereby optimizing water usage and promoting plant health.

The benefits of this system include convenience, water conservation, and improved plant growth, as users can adjust watering schedules based on real-time data and environmental conditions. Additionally, the integration of smart sensors and automation capabilities enhances the overall efficiency of garden maintenance, reducing the need for manual intervention.

While the smart home garden irrigation system using Raspberry Pi demonstrates significant potential, further research and development are needed to enhance its functionality, reliability, and user-friendliness. Additionally, addressing challenges such as power management, sensor accuracy, and connectivity issues will be crucial for widespread adoption and successful implementation in various home garden settings. Overall, the smart home garden irrigation system represents a valuable application of IoT technology in promoting sustainable gardening practices and enhancing the homeowner's gardening experience. With continued innovation and refinement, this system has the potential to revolutionize home gardening and contribute to environmental conservation efforts.

Manju Kaushik, Rajeev ratan

“Development of a Novel approach to enhance the performance of real time intelligent telemetry based irrigation system” in 2018 5th international conference on signal processing and integrated networks (SPIN).

The development of a novel approach to enhance the performance of real-time intelligent telemetry-based irrigation systems marks a significant step forward in optimizing water management practices in agriculture. By leveraging advanced telemetry technology, this approach offers farmers the ability to monitor and control irrigation processes in real-time, leading to improved efficiency, water conservation, and crop yields.

The key findings from this study highlight the effectiveness of the proposed approach in enhancing the performance of irrigation systems by integrating intelligent algorithms, predictive analytics, and real-time data processing capabilities. Through continuous monitoring of environmental parameters and soil conditions, coupled with adaptive control strategies, the system can dynamically adjust irrigation schedules to meet the specific needs of crops while minimizing water wastage.

The benefits of this novel approach include increased water savings, reduced operational costs, and enhanced agricultural productivity. By optimizing irrigation practices based on real-time data and predictive modeling, farmers can achieve better crop outcomes while simultaneously conserving water resources and promoting sustainability in agriculture. However, further research and development efforts are necessary to refine the proposed approach, address scalability challenges, and validate its performance across different agricultural contexts and environmental conditions. Additionally, considerations for interoperability, data security, and user interface design will be essential for widespread adoption and seamless integration into existing irrigation infrastructure.

Overall, the development of this novel approach represents a significant contribution to the field of precision agriculture and holds promise for revolutionizing irrigation practices in the pursuit of sustainable and efficient water management in agriculture.

The main objective of the project is to monitor the air eminence in industrial and urban areas. The proposed outline includes a set of gas sensors (CO, and NO2) that are positioned on masses and structure of a IOT (Internet of things) and a dominant server to support both short-range realtime incident management and a continuing deliberate planning. In this Arduino platform is used to communicate the data simply and quickly. WSN (Wireless sensor network) acts as the trans receiver. This provides a real-time low rate monitoring system over the use of low rate, low information rate, and little control wireless communication technology. The projected monitoring system can be transferred to or shared by different applications. Through IOT we can able to visualize the values from the globe. The problem in this paper is they haven't calibrated the sensor and not even converted the sensor output value into PPM. As per the guidelines by UN Data, 0-50 is SAFE value and 51-100 is moderate. Delhi is the most polluted city in the world recorded 350PPM. While using two sensors, as both sensors have internal heat element, it draws more power (P= VxI), so though the both sensors are turned ON, its output voltage levels varies and shows unpredicted values due to insufficient drive. So, we used a 9V battery and a 7805 family REGULATOR for the CO sensor MQ7. For MQ135 we have given the power from Arduino only.

Arduino Based Weather Monitoring System

Karthik Krishnamurthi, Suraj Thapa, Lokesh Kothari, Arun Prakash Department of Computer Science, Christ University, Bangalore, India.

This Paper makes use of 3 sensors to measure the weather/environment factors such as temperature, humidity, light intensity, dew point and heat index. The values read from the sensors are processed by the Arduino micro-controller and stored in a text file which can be processed upon to derive analysis. The readings are also displayed on a board LCD for quick viewing.

CHAPTER 3

PLATFORMS USED-Arduino IDE and Python

3.1.Platforms used

The IOT also helps monitoring the data in real time. The smart irrigation system is developed in this project is based on Arduino UNO and we have used to write code in Arduino IDE and for the Gmail Notification we have used Python IDLE to write the code.

3.2.Arduino IDE

HOW TO PROGRAM ARDUINO:

Once the circuit has been created on the breadboard, you’ll need to upload the program (known as a sketch) to the Arduino. The sketch is a set of instructions that tells the board what functions it needs to perform. An Arduino board can only hold and perform one sketch at a time. The software used to create Arduino sketches is called the IDE which stands for Integrated Development Environment. The software is free to download.

INSTALLING THE ARDUINO IDE:

Installing the Arduino IDE is easy on most operating systems and the IDE can be downloaded from the Arduino website. When the IDE downloads, run the executable and follow the onscreen instructions to install the software. After the IDE installs you should have a desktop icon for loading the Arduino IDE.

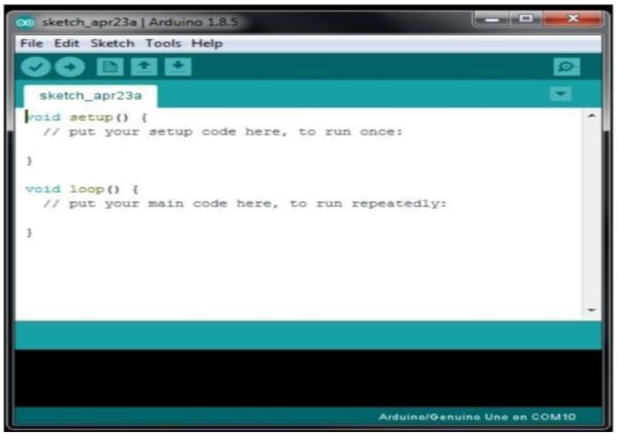
At this point, you need to connect your Arduino Uno to the PC using a USB cable. While most modern operating systems should recognize the Arduino as a serial port automatically.

CONFIGURING THE ARDUINO IDE:

The IDE loaded you should be presented with a similar window as shown below. Every Arduino sketch has two main parts to the program A minimal Arduino C/C++ sketch, as seen by the Arduino IDE programmer, consist of only two functions:

setup (): This function is called once when a sketch starts after power-up or reset. It is used to initialize variables, input and output pin modes, and other libraries needed in the sketch.

loop (): After setup () has been called, function loop () is executed repeatedly in the main program. It controls the board until the board is powered off or is reset.


sketch_apr23a | Arduino 1.8.5
File Edit Sketch Tools Help
sketch_apr23a
Arduino/Genuino Uno on COM10

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

}

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

}
```

Fig:3.1. Arduino IDE

"Before we can load an example project to test the Arduino, we need to configure the IDE to use the Arduino that is connected to the computer. To do this, navigate to Tools > Port and then look for the COM port that is your Arduino.

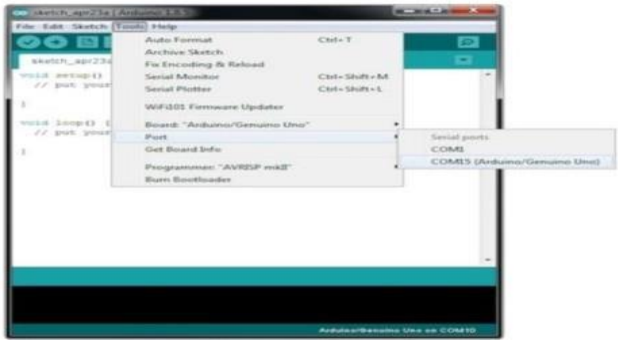

sketch_apr23a | Arduino 1.8.5
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 Firmware Updater
Board: "Arduino/Genuino Uno"
Port
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
Serial ports
COM1
COM15 (Arduino/Genuino Uno)
Arduino/Genuino Uno on COM10

Fig: 3.2. Arduino Port Selection

With the port selected you will also need to tell the IDE what board you are using. Since this example uses the Arduino Uno you will need to tell the IDE that you are using the Uno. To do this, navigate to Tools > Board and then select “Arduino / Genuino Uno”.

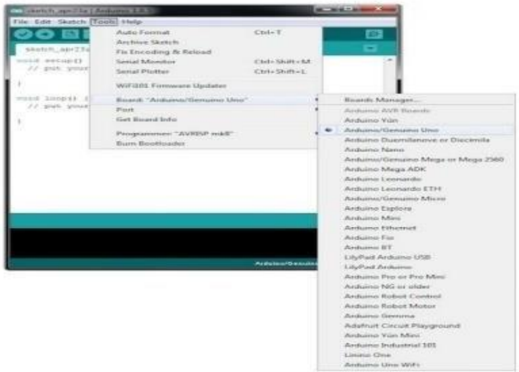


Fig:3.3. Arduino Board Selection

Most Arduino boards contain a light-emitting diode (LED) and a load resistor connected between pin 13 and ground, which is a convenient feature for many tests and program functions. A typical program for a beginning Arduino programmer blinks a LED repeatedly.

3.3. Use of Arduino IDE

The Arduino Software (IDE) makes it easy to write code and upload it to the board offline. We recommend it for users with poor or no internet connection. This software can be used with any Arduino board. There are currently two versions of the Arduino IDE, one is the IDE 1.x.x and the other is IDE 2.x. The IDE 2.x is new major release that is faster and even more powerful to the IDE 1.x.x. In addition to a more modern editor and a more responsive interface it includes advanced features to help users with their coding and debugging.

The following steps can guide you with using the offline IDE (you can choose either IDE 1.x.x or IDE 2.x):

1. Download and install the Arduino Software IDE: Arduino IDE 1.x.x (Windows, Mac OS, Linux, Portable IDE for Windows and Linux, ChromeOS). Arduino IDE 2.x
2. Connect your Arduino board to your device.
3. Open the Arduino Software (IDE).

The Arduino Integrated Development Environment - or Arduino Software (IDE) - connects to the Arduino boards to upload programs and communicate with them. Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension. No.

3.4. Python IDLE

Every Python installation comes with an **Integrated Development and Learning Environment**, which you'll see shortened to IDLE or even IDE. These are a class of applications that help you write code more efficiently. While there are many IDEs for you to choose from, Python IDLE is very bare-bones, which makes it the perfect tool for a beginning programmer.

Python IDLE comes included in Python installations on Windows and Mac. If you're a Linux user, then you should be able to find and download Python IDLE using your package manager. Once you've installed it, you can then use Python IDLE as an interactive interpreter or as a file editor.

Introduction to Python IDLE:

Python IDLE is a simple yet powerful development environment bundled with the Python programming language. It's designed to provide an easy-to-use interface for writing, testing, and debugging Python code. Here's a brief introduction to Python IDLE:

Interactive Shell:

One of the key features of Python IDLE is its interactive shell. When you open IDLE, you're greeted with a Python shell prompt, where you can type Python code directly and execute it instantly. This is great for experimenting with Python syntax, testing small code snippets, and learning Python interactively.

Code Editor:

IDLE also offers a code editor where you can write and save Python scripts. The editor provides syntax highlighting, indentation guides, and automatic indentation to help you write clean and readable code. You can open multiple editor windows to work on different scripts simultaneously.

Running Scripts:

In IDLE, you can run Python scripts directly from the editor window. Once you've written your code, you can execute it by pressing the F5 key or by selecting "Run Module" from the "Run" menu. IDLE will execute the script and display the output in the interactive shell.

Integrated Debugger:

IDLE includes a built-in debugger that helps you identify and fix errors in your Python code. You can set breakpoints, step through code, inspect variables, and watch expressions to understand how your code is executing. This is especially useful for debugging complex scripts and troubleshooting issues.

Help and Documentation:

IDLE provides access to Python's extensive documentation directly within the environment. You can use the "Help" menu to access Python's documentation, search for specific topics, and browse through modules and functions. This is a handy feature for quickly looking up information while coding.

Customization Options:

Python IDLE allows you to customize various aspects of the interface to suit your preferences. You can change the font size, configure syntax highlighting colours, adjust indentation settings, and more. Customizing the environment can help improve your coding experience and make IDLE feel more personalized.

Lightweight and Beginner-Friendly:

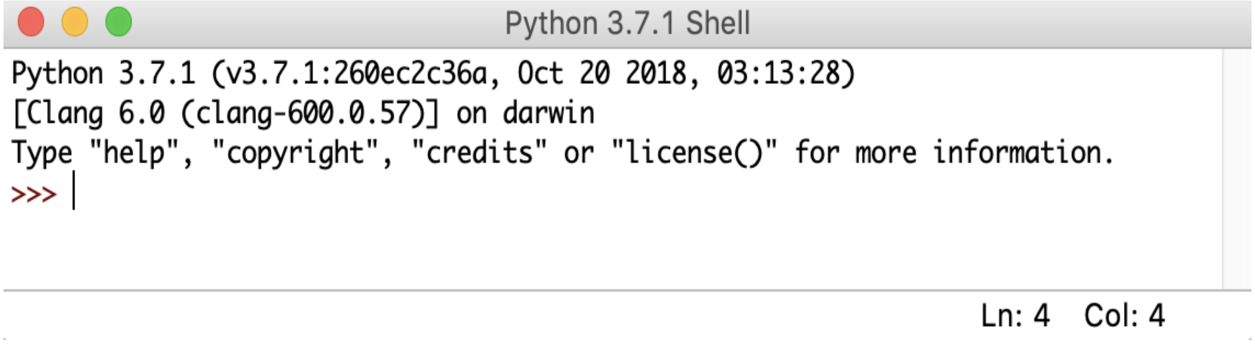
One of the advantages of Python IDLE is its simplicity and ease of use. It's lightweight, fast, and suitable for beginners who are just getting started with Python programming. IDLE provides a straightforward environment for learning Python concepts and experimenting with code without overwhelming users with complex features.

Platform Independence:

IDLE is available on multiple platforms, including Windows, macOS, and Linux, making it accessible to a wide range of users. Whether you're using a Windows PC, a Mac, or a Linux machine, you can use IDLE to write and run Python code with consistency.

How to Use the Python IDLE Shell

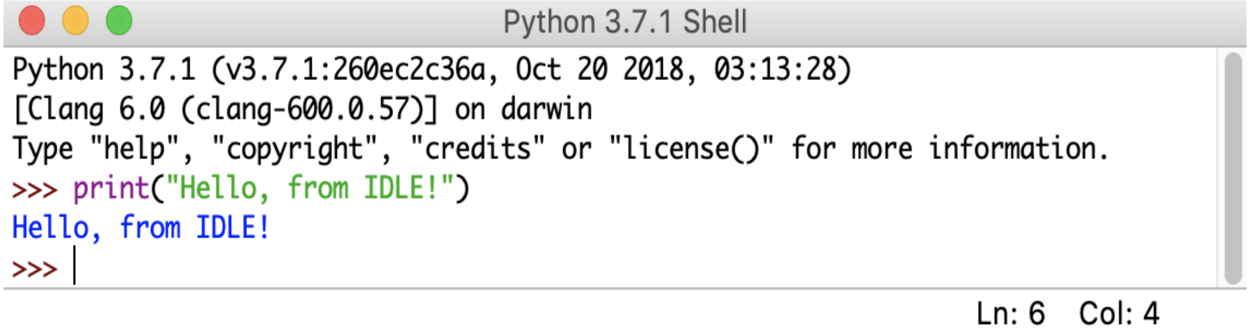
The shell is the default mode of operation for Python IDLE. When you click on the icon to open the program, the shell is the first thing that you see:



```
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 03:13:28)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license()" for more information.
>>> 
```

Fig: 3.4. Blank Python Interpreter Window

This is a blank Python interpreter window. You can use it to start interacting with Python immediately. You can test it out with a short line of code:



```
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 03:13:28)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license()" for more information.
>>> print("Hello, from IDLE!")
Hello, from IDLE!
>>> 
```

Fig: 3.5. Python IDLE

Here, you used print () to output the string "Hello, from IDLE!" to your screen. This is the most basic way to interact with Python IDLE. You type in commands one at a time and Python responds with the result of each command.

Next, take a look at the menu bar. You’ll see a few options for using the shell:

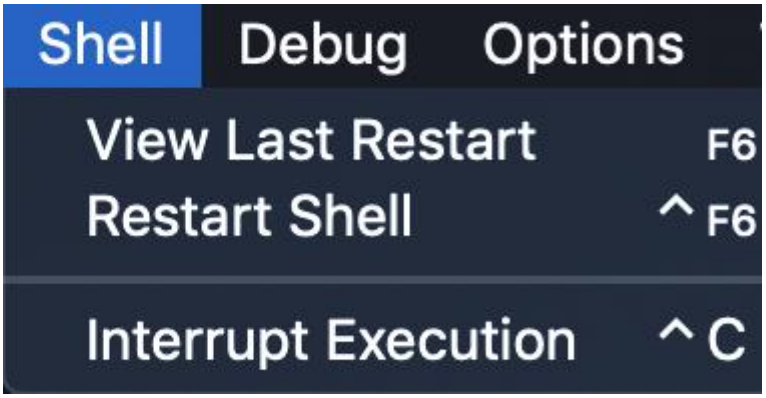
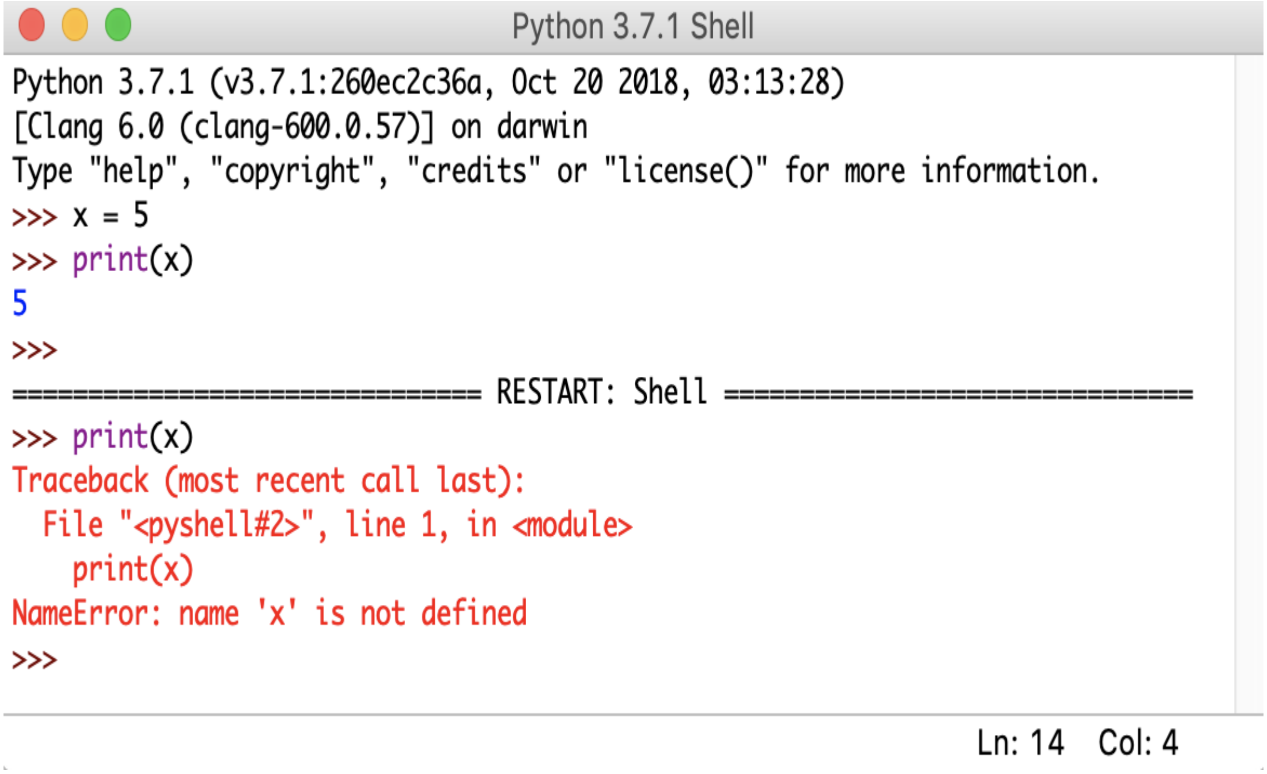


Fig:3.6. Python IDLE Features

You can restart the shell from this menu. If you select that option, then you'll clear the state of the shell. It will act as though you've started a fresh instance of Python IDLE. The shell will forget about everything from its previous state:

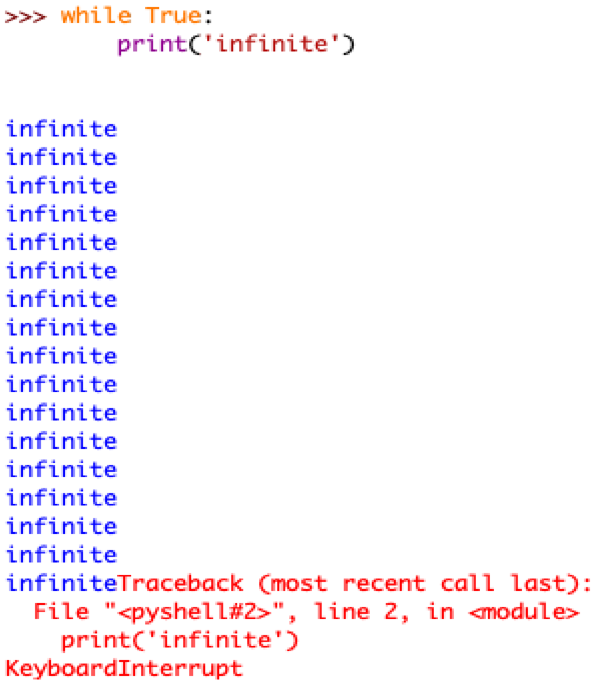


```
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 03:13:28)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license()" for more information.
>>> x = 5
>>> print(x)
5
>>>
=============================== RESTART: Shell ===============================
>>> print(x)
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    print(x)
NameError: name 'x' is not defined
>>>
```

Fig: 3.7. Restart: Shell

In the image above, you first declare a variable, x = 5. When you call print(x), the shell shows the correct output, which is the number 5. However, when you restart the shell and try to call print(x) again, you can see that the shell prints a traceback. This is an error message that says the variable x is not defined. The shell has forgotten about everything that came before it was restarted.

You can also interrupt the execution of the shell from this menu. This will stop any program or statement that's running in the shell at the time of interruption. Take a look at what happens when you send a keyboard interrupt to the shell:



```
>>> while True:
        print('infinite')


infinite
infinite
infinite
infinite
infinite
infinite
infinite
infinite
infinite
infinite
infinite
infinite
infinite
infinite
infinite
infinite
infiniteTraceback (most recent call last):
  File "<pyshell#2>", line 2, in <module>
    print('infinite')
KeyboardInterrupt
```

Fig: 3.8. Program Interrupt

A Keyboard Interrupt error message is displayed in red text at the bottom of your window. The program received the interrupt and has stopped executing.

3.4.1. Python IDLE Features

1. Object Oriented Language
2. Easy to understand
3. Free and open source
4. GUI programming support
5. High level language

Object Oriented Language:

Python is an object-oriented programming language, which means it allows you to define classes and objects to structure your code in a way that reflects the real-world entities and relationships. Here's an introduction to object-oriented programming (OOP) in Python:

Classes and Objects:

Class: A class is a blueprint for creating objects. It defines the properties (attributes) and behaviors (methods) that objects of the class will have. In Python, you define a class using the class keyword.

Object (Instance): An object is an instance of a class. It represents a specific instance of the class, with its own unique state (values of attributes) and behavior (methods). You create objects of a class using the class name followed by parentheses.

Encapsulation: Encapsulation is the bundling of data (attributes) and methods that operate on that data within a single unit (class). It allows you to control access to the data and ensure that the object's internal state remains consistent.

Inheritance: Inheritance allows a class (subclass) to inherit properties and methods from another class (superclass). It promotes code reusability and establishes a relationship between classes.

Polymorphism: Polymorphism allows objects of different classes to be treated as objects of a common superclass. It enables a single interface (method) to be used for different types of objects.

Abstraction: Abstraction is the concept of hiding complex implementation details and showing only the essential features of an object. It allows you to focus on what an object does rather than how it does it.

Python's support for object-oriented programming makes it a versatile language for developing large and complex applications, enabling code organization, reuse, and maintainability.

Free And Open Source

Absolutely, Python is not only free and open source but also widely supported across various platforms. Here's what that means:

Free:

No Cost: Python is completely free to download, use, and distribute. You don't need to pay anything to use Python, whether for personal or commercial purposes.

No Restrictions: There are no restrictions on how you use Python. You can use it for any purpose, modify it as needed, and even sell your software built with Python without paying any licensing fees.

Open Source:

Community-Driven: Python is developed and maintained by a large community of developers worldwide. Anyone can contribute to its development, report bugs, suggest improvements, and participate in discussions.

Source Code Access: The entire source code of Python is freely available for anyone to view, modify, and distribute under the Python Software Foundation License (PSFL) or a compatible license.

Transparency: Being open source means that Python's development process is transparent. You can see how the language evolves over time, understand the decisions made by its developers, and contribute your own ideas.

Platform Independence:

Cross-Platform: Python is designed to be cross-platform, meaning it runs on various operating systems such as Windows, macOS, and Linux. This ensures that your Python code can be executed on different platforms without modification.

Benefits of Free and Open Source:

Accessibility: Being free and open source makes Python accessible to everyone. You can download Python from its official website and start coding right away without any barriers.

Community Support: The large and active Python community provides extensive support, resources, and documentation to help you learn and solve problems. There are numerous online forums, tutorials, and libraries available to assist Python developers.

Customizability: Since Python is open source, you can customize it to suit your specific needs. If you encounter a bug or need a new feature, you can modify the source code yourself or request changes from the community.

Trustworthiness: Open source software is often considered more trustworthy and secure because its source code is open for inspection by anyone. This transparency allows users to verify that the software behaves as expected and to identify and fix security vulnerabilities.

GUI Programming Support:

Python provides several libraries and frameworks for GUI (Graphical User Interface) programming, allowing developers to create desktop applications with rich graphical interfaces. Here are some popular options:

- Tkinter:

Description: Tkinter is the standard GUI toolkit for Python. It comes bundled with Python, so you don't need to install anything extra to use it.

Pros:

Easy to use, especially for beginners.

Lightweight and comes pre-installed with Python.

Good for simple GUI applications.

Cons:

Limited styling options compared to modern GUI frameworks.

Less flexibility and fewer advanced features compared to some other options.

- PyQt / PySide:

Description: PyQt and PySide are Python bindings for the Qt framework. PyQt is developed by Riverbank Computing, while PySide is developed by the Qt for Python project.

Pros:

Powerful and feature-rich.

Supports modern GUI development with advanced features.

Good for building complex applications.

Cons:

Requires installation of additional libraries ('PyQt' or' PySide').

May have licensing considerations for commercial use.

- Kivy:

Description: Kivy is an open-source Python library for developing multi-touch applications. It's particularly well-suited for building cross-platform applications.

Pros:

Cross-platform (Windows, macOS, Linux, Android, iOS).

Supports multi-touch and gestures.

Flexible and customizable.

Cons:

Less commonly used than Tkinter or PyQt.

Can be more challenging for beginners.

- wxPython:

Description: wxPython is a Python wrapper for the wxWidgets C++ library. It provides native-looking GUI applications for various platforms.

Pros:

Native look and feel across different platforms.

Mature and stable.

Cons:

More complex than Tkinter for simple applications.

Requires additional installation (wxPython).

High Level Language:

A high-level programming language, like Python, is designed to be easily understood by humans and to abstract away many low-level details. Here's what makes Python a high-level language:

Readability:

Clear and Simple Syntax: Python's syntax is designed to be simple and readable, making it easier for programmers to understand and write code.

Rich Standard Library:

Extensive Standard Library: Python comes with a large standard library that provides built-in modules and functions for performing various tasks, such as file I/O, networking, and data manipulation, without needing to install additional packages.

Dynamic Typing: Python is dynamically typed, meaning you don't need to declare the data type of variables explicitly. Variables can hold values of any type, and their type is determined dynamically at runtime.

Interpreted:

Interpreted Language: Python is an interpreted language, which means that code is executed line by line by the Python interpreter. This allows for quick development and testing cycles without the need for compilation.

Cross-Platform:

Platform Independence: Python code can run on various platforms, including Windows, macOS, and Linux, without modification, making it highly portable.

High-Level Constructs:

Built-in Data Structures: Python provides high-level data structures such as lists, dictionaries, and sets, which make it easy to manipulate collections of data.

Object-Oriented Programming (OOP): Python supports OOP concepts like classes, objects, inheritance, and polymorphism, enabling developers to organize and structure code in a more intuitive way.

Functional Programming: Python supports functional programming techniques like lambda functions, map, filter, and reduce, which allow for concise and expressive code.

Rapid Development:

Quick Prototyping: Python's simplicity and ease of use make it ideal for rapid prototyping and development of applications.

Less Code: Python often requires less code compared to other languages, thanks to its high-level constructs and expressive syntax, which can lead to increased productivity.

Community and Ecosystem:

Large Community: Python has a large and active community of developers who contribute libraries, frameworks, and resources, providing solutions to a wide range of problems.

Vibrant Ecosystem: The Python ecosystem includes thousands of third-party libraries and frameworks for various purposes, making it easy to find tools and solutions for almost any task.

3.4.2. Python IDLE works with

6. Web Development
7. Machine Learning
8. Data Analysis
9. Scientific Developer

3.5. Advantages of Arduino IDE

The five key advantages of the Arduino programming language are:

1. Easy to Learn and Use: The Arduino programming language is based on C++, with a simple and straightforward syntax that is easy to pick up even for beginners.
2. Versatility: Arduino boards can be used in a wide variety of projects, such as robotics, home automation, and IoT applications.
3. Widely Supported: The Arduino community is large and active, with a wealth of resources and tutorials available to help users learn and troubleshoot.
4. Cost-Effective: Arduino boards and components are relatively inexpensive, making them accessible to hobbyists and students.
5. Open Source: The Arduino software and hardware are open source, which means that users have access to the source code and can modify it to suit their needs.

3.6. Disadvantages of Arduino IDE

1. Sketches and shields can be difficult to modify.
2. No debugger included for checking scripts.

3.7. Advantages of Python IDLE

Python IDLE (Integrated Development and Learning Environment) offers several advantages for developers and learners:

1.User-Friendly Interface: Python IDLE provides a simple and intuitive interface that is easy to navigate, making it ideal for beginners and experienced developers alike.

2.Interactive Shell: IDLE includes an interactive shell where users can execute Python code line by line, making it convenient for testing and debugging code snippets.

3.Code Editor: IDLE features a built-in code editor with syntax highlighting, auto-indentation, and code completion, which helps improve code readability and productivity.

4.Integrated Debugger: IDLE includes a debugger tool that allows users to set breakpoints, step through code execution, and inspect variables, facilitating the debugging process.

5.Multiplatform Support: Python IDLE is available for multiple platforms, including Windows, macOS, and Linux, ensuring consistency across different operating systems.

6.Integrated Documentation: IDLE provides access to Python's extensive documentation directly within the IDE, allowing users to quickly look up function definitions, module documentation, and language references.

7.Extension and Customization**:** IDLE supports extensions and customizations through plugins and configuration options, allowing users to tailor the IDE to their specific needs and preferences.

8.Educational Tools: Python IDLE is commonly used in educational settings for teaching programming concepts and Python language fundamentals due to its simplicity and accessibility.

9.Integration with Python Libraries**:** IDLE seamlessly integrates with Python's standard library and third-party libraries, enabling users to leverage a wide range of tools and resources for their projects.

10.Community Support: Python IDLE benefits from a large and active community of users and developers who provide support, tutorials, and resources, making it easier for users to learn and troubleshoot issues.

3.8. Disadvantages of Python IDLE

1.Limited Features: Compared to other integrated development environments (IDEs) like PyCharm or Visual Studio Code, Python IDLE lacks advanced features such as integrated version control, code refactoring tools, and advanced debugging capabilities.

2.Performance: Python IDLE may experience performance issues when working with large codebases or performing computationally intensive tasks, as it may not be as optimized as other IDEs for handling such scenarios.

3.While Python IDLE supports some level of customization through plugins and configuration options, it may not offer as much flexibility or extensibility as other IDEs, limiting users' ability to tailor the environment to their specific needs.

4.User Interface: Some users find the user interface of Python IDLE to be simplistic or outdated compared to modern IDEs, which may affect usability and productivity, especially for developers accustomed to more feature-rich environments.

5.Integration with External Tools: Python IDLE may not seamlessly integrate with external tools and libraries as some other IDEs do, which could be a drawback for developers who rely on specific tools or workflows for their projects.

6.Documentation and Support: While Python IDLE benefits from the broader Python community's support, it may not have as extensive documentation or community resources available compared to more widely used IDEs, which could make it challenging for users to find help or troubleshoot issues.

7.Platform Dependency: Although Python IDLE is available on multiple platforms, including Windows, macOS, and Linux, users may encounter platform-specific issues or inconsistencies in functionality, which could affect cross-platform development workflows.

8.Learning Curve: While Python IDLE is designed to be beginner-friendly, users transitioning from other IDEs may find it lacking in certain features or workflows they are accustomed to, which could result in a learning curve or productivity slowdown during the adjustment period.

3.9. Conclusion

In conclusion, both the Arduino IDE and Python IDLE serve as valuable tools for programming and development, each with its own strengths and limitations. The Arduino IDE offers a user-friendly environment specifically tailored for programming Arduino microcontrollers. It provides a simple interface, extensive libraries, and built-in tools for compiling and uploading code to Arduino boards. The Arduino IDE is particularly well-suited for beginners and hobbyists who are getting started with electronics and embedded systems.

On the other hand, Python IDLE provides an integrated development and learning environment for programming in the Python language. It offers features such as an interactive shell, code editor, and debugger, making it suitable for a wide range of Python development tasks, from While both IDEs excel in their respective domains, they also have their disadvantages.

The Arduino IDE may lack advanced features and customization options found in other development environments, while Python IDLE may not offer the same level of performance or integration with external tools as more advanced Python IDEs. Ultimately, the choice between Arduino IDE and Python IDLE depends on the specific requirements of the project and the user's familiarity and comfort with each environment.

For Arduino-based projects, the Arduino IDE is the natural choice, while for Python development, Python IDLE offers a convenient and accessible option. However, developers may also choose to explore other IDEs and development tools to find the best fit for their needs.

CHAPTER 4

SYSTEM MODEL

4.1. Existing Model

The existing irrigation methods typically employed by farmers often rely on manual observation and intuition to determine when and how much water to apply to crops. Farmers may use traditional techniques such as visual inspection of soil moisture or relying on predetermined schedules for irrigation. However, these methods lack precision and may result in either overwatering or underwatering, leading to inefficient water usage and suboptimal crop growth. Additionally, the manual nature of these methods requires significant labor and time investment, limiting scalability and productivity. Furthermore, traditional irrigation practices may not take into account variations in soil conditions or weather patterns, leading to inconsistencies in water application and potential crop stress. Overall, the reliance on conventional irrigation methods poses challenges in achieving efficient resource utilization and maximizing agricultural productivity**.**

Scalability and productivity are further restricted by the manual nature of these procedures, which necessitates a large labor and time investment. It's also possible that conventional irrigation techniques don't account for changes in weather patterns or soil conditions, which could result in uneven water application and crop stress. All things considered, using traditional irrigation techniques presents difficulties in attaining effective resource use.

Required Components:

1. Arduino UNO
2. Soil Moisture Sensor
3. DHT11 Sensor
4. Relay
5. DC Motor Pump
6. LCD
7. Power Supply

4.2.Circuit Diagram :

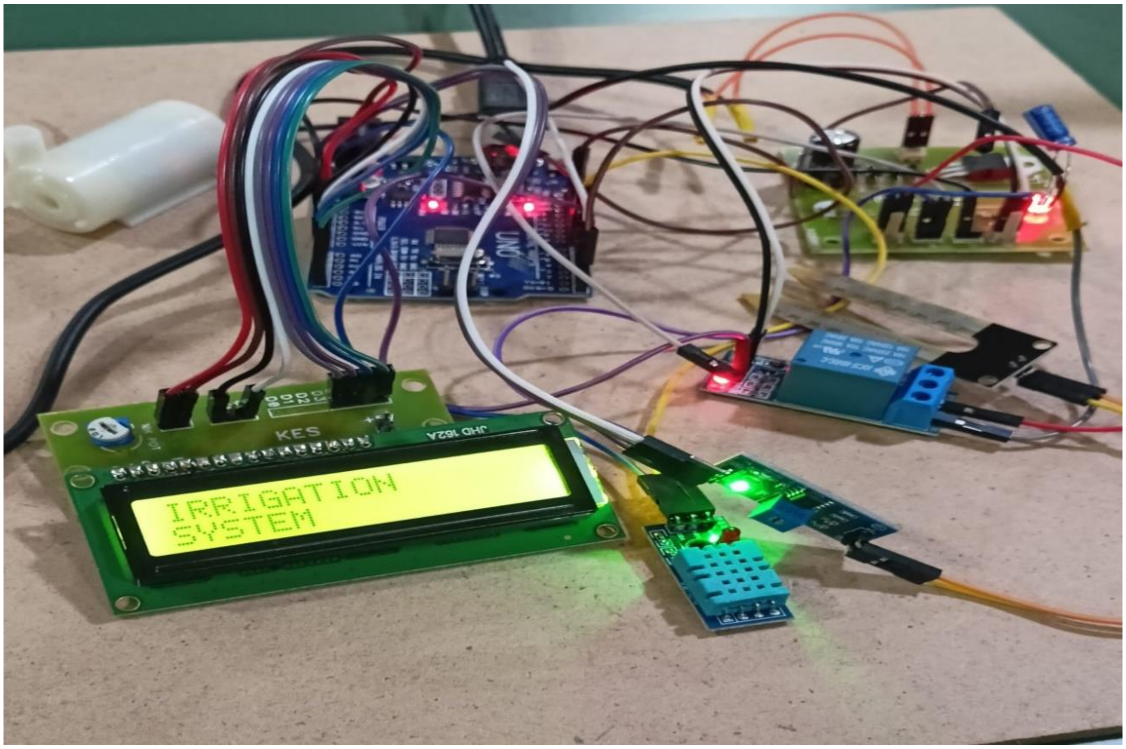


Fig.4.1. Circuit diagram of Proposed Method

4.3. Working:

The project you're describing, "Automatic controlling system of pump based on temperature and moisture conditions along with Monitoring using IoT," is essentially an automated irrigation or water management system designed to optimize water usage based on environmental conditions. This type of system can be crucial for applications like agriculture, gardening, or maintaining green spaces in urban environments. Here’s a breakdown of how such a project could work, including its main components and functionalities.

This sensor mainly utilizes capacitance to gauge the water content of the soil (dielectric permittivity). The working of this sensor can be done by inserting this sensor into the earth and the status of the water content in the soil can be reported in the form of a percent.

This sensor makes it perfect to execute experiments within science courses like environmental science, agricultural science, biology, soil science, botany, and horticulture.

```
#include "DHT.h"
#define DHTPIN 8
#define DHTTYPE DHT11
#include<LiquidCrystal.h>
#include<SoftwareSerial.h>
SoftwareSerial ard(9,10);
LiquidCrystal LCD (A0,A1,A2,A3,A4,A5);
DHT dht(DHTPIN, DHTTYPE);
```

'DHTPIN' is defined as pin8, which means the DHT sensor's data pin is connected to digital pin 8 on the Arduino.

DHTTYPE specifies the DHT sensor, in this case, a DHT11.

LiquidCrystal LCD(A0,A1,A2,A3,A4,A5);line initializes the LCD and specifies the Arduino pins connected to the LCD's RS,En,D4,D5,D6 and D7 pins,respectively.

SoftwareSerial ard(9,10);initializes a software serial port using pin9 for RX and pin 10 for TX

```
int sensor_pin=2;
int relay=3;
String str;
```

This line declares an integer variable named sensor_pin and it is connected to the digital pin 2.

Relay module is connected to the digital pin3

```
void setup () {
pinMode(sensor_pin,INPUT);
pinMode(relay,OUTPUT);
Serial.begin(9600);
ard.begin(115200);
digitalWrite(relay,HIGH);
```

Set the digital pin a input

Set the digital pin as output

```
dht.begin();
LCD.begin(16,2);
LCD.clear();
LCD.setCursor(0,0);
```

```
 LCD.print("IRRIGATION");
 LCD.setCursor(0,1);
 LCD.print("SYSTEM");
 delay (2000);
```

the code initializes a DHT sensor, initializes an LCD display, clears the display, prints "IRRIGATION" on the first line and "SYSTEM" on the second line of the LCD, and then waits for 2 seconds before proceeding with further instructions.

```
void loop ()
{
 float h = dht.readHumidity();
 float t = dht.readTemperature();
 if (isnan(h) || isnan(t)) {
```

These lines read the humidity and temperature from a DHT sensor (assuming dht is an instance of the DHT sensor object).

The 'readHumidity()' function returns the humidity value as a float.

The 'readTemperature()' function returns the temperature value as a float.

The isnan() function checks if a value is "not a number" (NaN). It's used to check if the sensor readings are valid.

If either the humidity or temperature readings are NaN, it means there was an error reading the sensor data.

```
  LCD.clear();
  LCD.setCursor(0,0);
  LCD.print("T:");
  LCD.setCursor(2,0);
  LCD.print(t);
  LCD.setCursor(8,0);
  LCD.print("M:");
  LCD.setCursor(10,0);
  LCD.print(sensor_data);
  LCD.setCursor(12,0);
  LCD.print("H:");
```

```
 LCD.setCursor(14,0);
 LCD.print(h);
 delay (2000);
 str=String(sensor_data) +String (“!”) +String(t)+String ("@”) +String(h);
 ard.println(str);
 if(t>34&&sensor_data==1)
 {
```

LCD.clear(): Clears the LCD display, removing any previous text.

LCD.setCursor(x, y): Sets the cursor position where x is the column and y is the row.

LCD.print(...): Prints text or numbers at the current cursor position.

```
 else if(t>34&&sensor_data==0)
 {
digitalWrite(relay,LOW);
  // LCD.clear();
  LCD.setCursor(0,1);
  LCD.print("Motor ON");
  delay (5000);
```

digitalWrite(relay, LOW); This line turns on the relay which I assume is connected to control a motor. Since relay was set to OUTPUT in your setup, writing LOW likely means activating the device connected to it, based on your setup (common in active-low relay setups)

```
digitalWrite(relay,HIGH);
  // LCD.clear();
  LCD.setCursor(0,1);
  LCD.print("Motor OFF");
  delay (1000);
  }
  else if(t<=34&&sensor_data==0)
  {
digitalWrite(relay,HIGH);
 // LCD.clear();
 LCD.setCursor(0,1);
LCD.print("Motor OFF");
 delay (1000);
 }
 }
```

Motor Control: Both conditions you've mentioned (if and else if) set the relay to HIGH, implying the motor is being turned off. This seems to be the default state when conditions are not met to activate the motor.

4.4. Proposed Model

The project's suggested technique involves setting in place a smart irrigation system that is IoT-based. An Arduino microcontroller is integrated with a number of sensors, including DHT11 and soil moisture sensors, in this system. These sensors enable more precise and accurate irrigation control by providing real-time data on temperature, humidity, and soil moisture levels. The Python code that communicates the moisture and motor status to gmail improve the irrigation system's connectivity and functionality. This makes it possible for farmers to keep an eye on their irrigation system using smartphones.

4.5. Block Diagram:

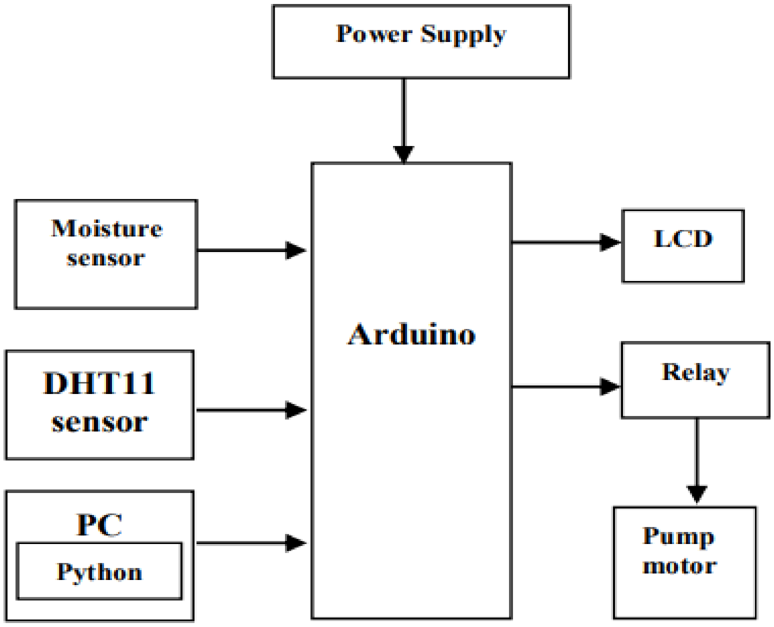


Fig.4.2. Block Diagram of Automatic controlling system of pump based on temperature and moisture conditions along with Monitoring using IOT

4.6. Hardware Description

The hardware of the Air quality monitoring display board using IoT include

1. Arduino UNO
2. Soil Moisture Sensor
3. DHT11 Sensor
4. Relay
5. DC Motor Pump
6. LCD
7. Power Supply

4.6.1. Arduino UNO

The Arduino UNO is the best board to get started with electronics and coding. If this is your first experience tinkering with the platform, the UNO is the most robust board you can start playing with. The UNO is the most used and documented board of the whole Arduino family. Arduino UNO is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. You can tinker with your UNO without worrying too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again.

Replaceable chip:

The ATmega328P can easily be replaced, as it is not soldered to the board

EEPROM:

The ATmega328P also features 1kb of EEPROM, a memory which is not erased when powered off.

Battery Connector:

The Arduino UNO features a barrel plug connector, that works great with a standard 9V battery.

Software & Cloud:

The following software tools allow you to program your board both online and offline

1) Arduino IDE

2) Arduino CLI

3) Web Editor Hardware:

The hardware listed below is compatible with this product:

1) 4 Relays Shield

2) Motor shield Rev3

3) 9 Axis Motion Shield

Suggested Libraries:

- Wire: This Library allows you to communicate with I2C/TWI devices
- SPI: The SPI library allows you to communicate with SPI devices, with the Arduino as the controller device.
- Servo: The Servo library allows an Arduino board to control RC (hobby)servo motors.

Board	Name	Arduino UNOR3
	SKU	A000066
Microcontroller	ATmega328P	
USB Connecter	USB-B	
Pins	Built in LED Pin	13
	Digital I/O Pins	14
	Analog input pins	6
	PWM pins	6
Communication	UART	Yes
	12C	Yes
	SPI	Yes

Power	I/O Voltage	5V
	Input voltage (nominal)	7-12V
	DC current per I/O pin	20mA
	Power supply connector	Barrel plug
Clock speed	Main Processor	ATmega328P 16MHZ
	USB serial processor	ATmega16U2 16MHZ
Memory	ATmega328P	2KB SRAM,32KB FLASH,1KBEEPROM
Dimensions	Weight	25g
	Length	68.6mm
	Width	53.4mm
Board	Name	Arduino UNOR3
	SKU	A000066
Microcontroller	ATmega328P	
USB Connector	USB-B	
Pins	Built-in LED pin	13
	Digital I/O pins	14
	Analog input pins	6
	PWM pins	6
Communication	UART	YES
	12C	YES
	SPI	YES
Power	I/O Voltage	5V
	Input Voltage	7-12V
	DC current per I/O pins	20mA
	Power connector supply	Barrel plug
Clock speed	Main processor	ATmega328P 16MHz
	USB-Serial processor	ATmega16U2 16MHz

Memory	ATmega328P	2KB SRAM,32KB FLASH,1KBEEPROM
Dimensions	Weight	25g
	Width	53.4mm
	Length	68.6mm

Tabel 4.1. Arduino Uno Specifications

PIN DESCRIPTION:

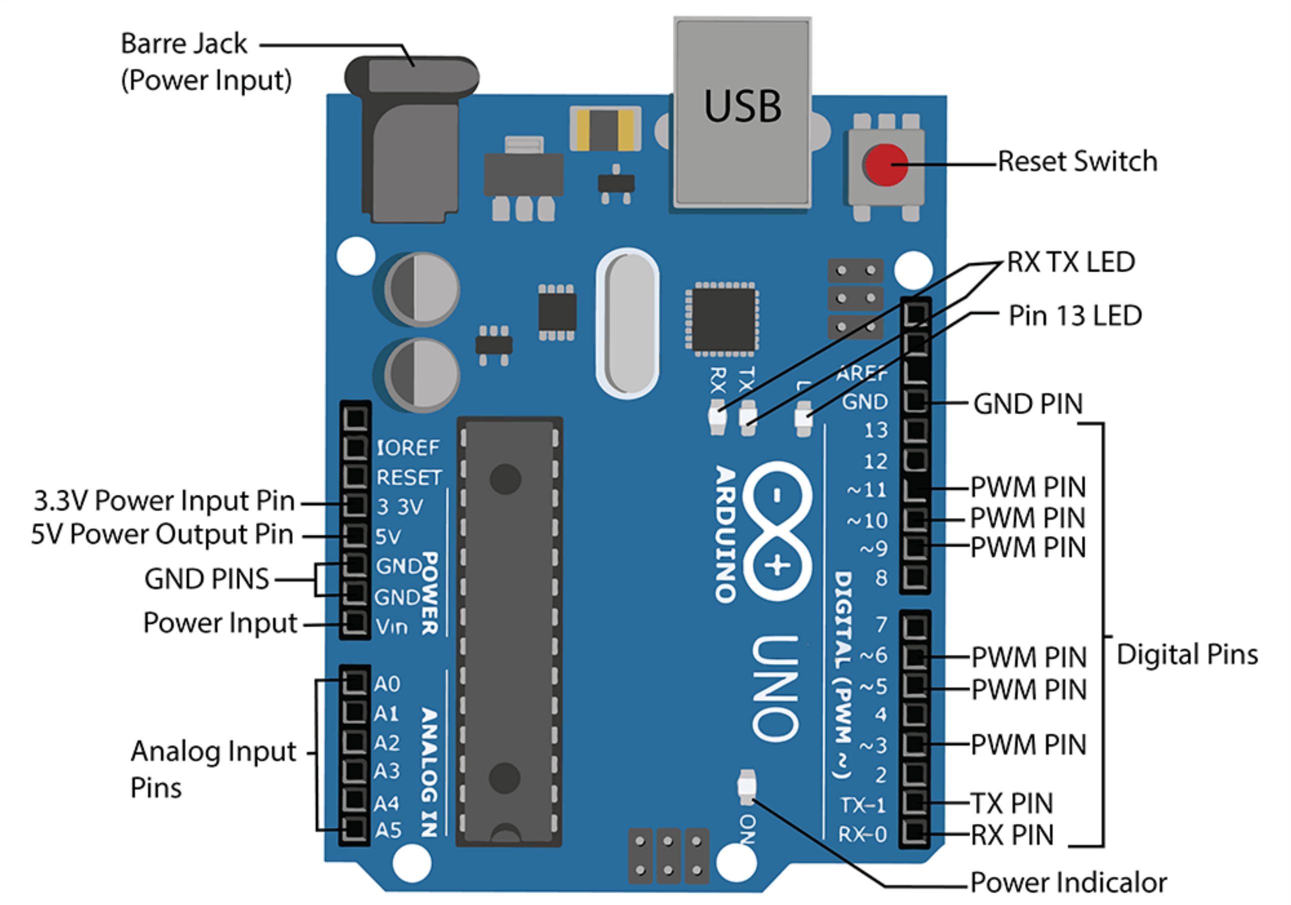


Fig.4.3. Arduino UNO pin description

4.6.2. SOIL MOISTURE SENSOR

Fig: 4.4. Soil Moisture Sensor

In order to measure the water content (or moisture level) in the soil, soil moisture sensors are extensively employed in environmental monitoring, landscaping, and agriculture. By adjusting irrigation schedules, these sensors can help save water and promote better plant development. Soil moisture sensors might differ in terms of complexity, cost, accuracy, and applicability based on their design and technology. Here is a thorough explanation of the various kinds of common soil moisture sensors and how they work:

Their Interface with Microcontroller Systems

Configuration and Wiring: The majority of soil moisture sensors (either resistive or capacitive) for amateur use include analog outputs that may be connected straight to the analog input pins of microcontrollers such as Arduino. Additionally, they frequently have digital output choices that activate at a particular moisture threshold.

Analyzing Information: An ADC (Analog-to-Digital Converter) is used by microcontrollers to transform analog signals from sensors that measure moisture content into digital values that may be handled or shown.

Application in Code: For example, in Arduino, you might use analog Read(sensor Pin), where sensor Pin is the pin that the sensor is connected to, to read the moisture level. As a result, you would receive a figure that would normally range from 0 (totally dry) to 1023 (entirely wet).

Applications:

Soil moisture sensors find applications in:

Agricultural Technology: Enhancing irrigation efficiency by providing data to adjust watering schedules based on real soil moisture levels.

Environmental Monitoring: Tracking soil moisture trends to predict drought conditions or monitor ecosystem health.

Smart Home Gardening: Integrating with home automation systems to maintain the health of garden plants automatically.

Pin Description:

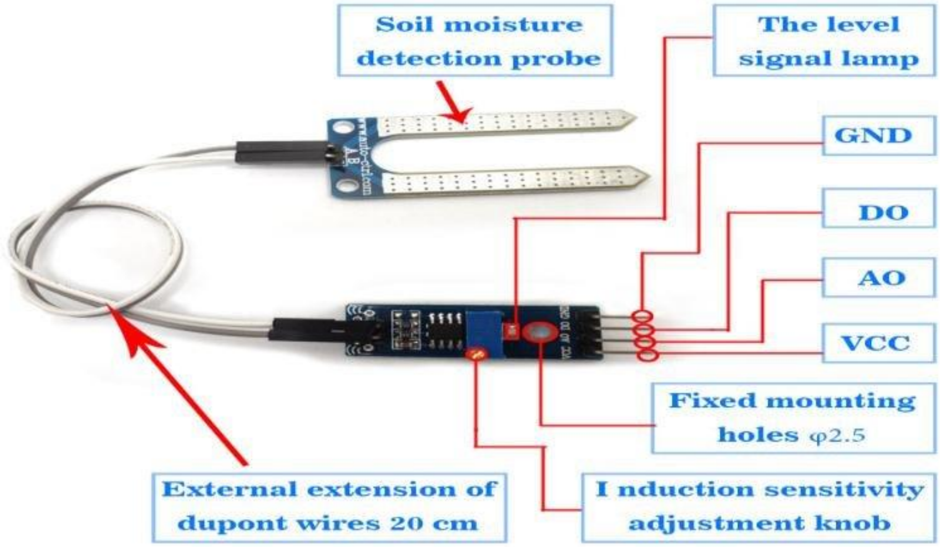

Soil moisture detection probe
The level signal lamp
GND
DO
AO
VCC
Fixed mounting holes φ2.5
External extension of dupont wires 20 cm
I nduction sensitivity adjustment knob

Fig: 4.5. Pin Description of Soil Moisture Sensor

Soil moisture sensors typically have multiple pins, each serving a specific purpose. Here's a description of the common pins found on soil moisture sensors:

VCC or VDD (Voltage Supply):
This pin is used to supply power to the sensor. It's usually connected to a positive voltage source (such as +5V or +3.3V).

Typically, this pin is connected to the positive terminal of the power supply.

GND (Ground):
This pin is the ground connection for the sensor.

It's connected to the negative terminal of the power supply or the ground of the microcontroller or system.

Analog Output (AO):

This pin provides an analog voltage output that represents the soil moisture level.

The voltage output typically varies based on the soil moisture level detected by the sensor.

This pin is usually connected to an analog input pin of a microcontroller or ADC (Analog-to-Digital Converter) module.

Digital Output (DO):

Some soil moisture sensors also provide a digital output to indicate whether the soil moisture level is above or below a certain threshold.

When the soil moisture is above the set threshold, this pin outputs a high voltage (usually VCC).

When the soil moisture is below the threshold, it outputs a low voltage (usually GND).

This pin can be connected directly to a digital input pin of a microcontroller for digital soil moisture level detection.

Temperature Compensation (T):

Some advanced soil moisture sensors include a temperature compensation pin to enhance accuracy by adjusting the readings based on temperature.

This pin might be connected to a temperature sensor or thermistor, or it may accept an external temperature input.

Temperature compensation helps to correct soil moisture readings affected by temperature variations.

Reference Voltage (REF):

In some soil moisture sensors, there's a reference voltage pin used for calibration or reference purposes.

This pin might be used to set a reference voltage for comparison with the output voltage.

It's typically connected to a reference voltage source, often VCC.

Not Connected (NC):

Some pins may be labeled as NC, indicating that they are not connected to anything internally.

These pins are not used in the typical operation of the sensor and are left unconnected.

Features:

1. Sensor Type

Capacitive: Measures the dielectric permittivity of the soil, which changes with moisture level.

These sensors are durable and less prone to corrosion.

Resistive: Measures electrical resistance between two probes, which decreases as moisture increases. These are generally simpler and less expensive but can corrode over time.

Time Domain Reflectometry (TDR): Uses the time of return of a reflected signal to measure soil moisture with high accuracy, suitable for precise scientific applications.

2. Measurement Accuracy

High accuracy is crucial for applications like research and high-precision farming. Sensors vary in their precision, and some may require calibration to specific soil types.

3. Durability and Lifespan

Designed to withstand environmental elements like soil acidity, temperature extremes, and moisture conditions. Capacitive sensors, in particular, offer enhanced durability due to their non-corrosive nature.

4. Ease of Installation

Sensors are typically easy to install, often just needing to be inserted into the soil at the appropriate depth. Some sensors come with additional mounting equipment or enclosures for protection.

5. Data Output and Connectivity

Output can be analog (voltage related to moisture level) or digital (serial communication like I2C, SPI, or RS-485).

Many modern sensors include wireless connectivity options such as Wi-Fi, Bluetooth, or LoRaWAN for integration into smart systems or IoT networks.

6. Integration Capabilities

Can often be integrated with other systems like automated irrigation controls, greenhouse management systems, or broader agricultural monitoring platforms.

7. Power Requirements

Designed to operate at low power, making them suitable for remote and wireless applications where power is a limitation.

Some sensors are equipped with solar panels or options for battery operation to enhance their usability in field conditions.

8. Real-Time Monitoring

Advanced models support real-time data collection and alerting capabilities, enabling immediate responses to changes in soil moisture levels.

9. Scalability

Systems can range from single sensors suitable for home gardens to multiple sensors networked across extensive agricultural fields for comprehensive moisture mapping.

10. Environmental Impact

Typically made with environmentally safe materials and designed to operate with minimal environmental disturbance.

11. Cost-Effectiveness

While basic models are quite affordable, more advanced sensors with higher accuracy and features like wireless communication tend to be more expensive but offer long-term savings through more efficient water usage and crop management.

These features make soil moisture sensors an indispensable tool in modern agriculture and environmental management, providing vital data needed to optimize water usage, enhance crop yields, and minimize environmental impact.

4.6.3. DHT 11 Sensor

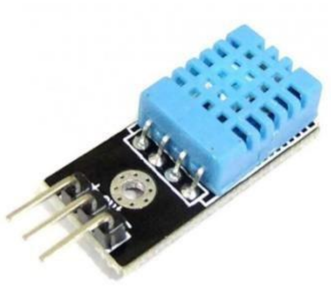


Fig: 4.6. DHT 11 sensor

The DHT11 is a basic, ultra low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air and spits out a digital signal on the data pin (no analog input pins needed).

It's fairly simple to use but requires careful timing to grab data. The only real downside of this sensor is you can only get new data from it once every 2 seconds, so when using our library, sensor readings can be up to 2 seconds old.

The DHT sensors are made of two parts, a capacitive humidity sensor and a thermistor. There is also a very basic chip inside that does some analog to digital conversion and spits out a digital signal with the temperature and humidity. The digital signal is fairly easy to read using any microcontroller.

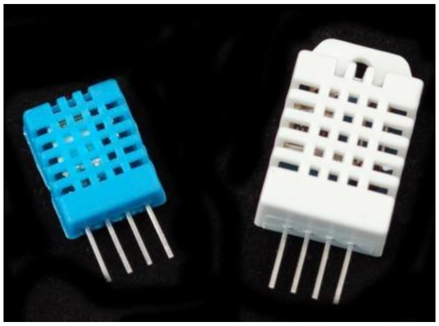


Fig: 4.7. Humidity sensor and thermistor

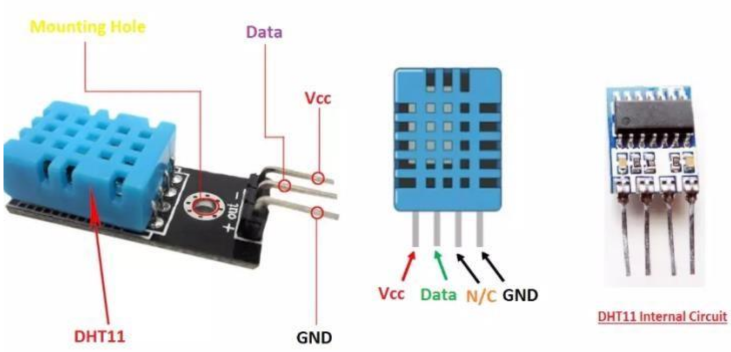


Fig: 4.8. DHT 11 parts

DHT 11characteristics

1. Ultra low cost
2. 3 to 5V power and I/O
3. 2.5mA max current use during conversion (while requesting data)
4. Good for 20-80% humidity readings with 5% accuracy
5. Good for 0-50°C temperature readings ±2°C accuracy
6. No more than 1 Hz sampling rate (once every second)
7. Body size 15.5mm x 12mm x 5.5mm

8. 4 pins with 0.1" spacing

It is an embedded sensor used to measure temperature & humidity in the surroundings and gives calibrated digital output. It can measure temperature in the range of 0°C to 50°C with ±2°C accuracy. Its humidity range is from 20% to 80% with ±5% accuracy. It is a small, low cost and easy to interface embedded sensor.

Working Principle

DHT11 sensor consists of a capacitive humidity sensing element and a thermistor for sensing temperature. The humidity sensing capacitor has two electrodes with a moisture holding substrate as a dielectric between them. Change in the capacitance value occurs with the change in humidity levels. The IC measure, process this changed resistance values and change them into digital form.

For measuring temperature this sensor uses a Negative Temperature coefficient thermistor, which causes a decrease in its resistance value with increase in temperature. To get larger resistance value even for the smallest change in temperature, this sensor is usually made up of semiconductor ceramics or polymers. This sensor is used in various applications such as measuring humidity and temperature values in heating, ventilation and air conditioning systems. Weather stations also use these sensors to predict weather conditions. The humidity sensor is used as a preventive measure in homes where people are affected by humidity. Offices, cars, museums, greenhouses and industries use this sensor for measuring humidity values and as a safety measure.

SNO	**Parameter**	**Value**
1	Measures	Humidity & Temperature
2	Sensors	Capacitive Humidity & Thermistor
3	Humidity Range	20% to 80% with (+/-)5% accuracy
4	Package	4 pins in a single row
5	Operating Voltage	3.0V to 5.5V
6	Operating Current	0.3mA(measuring),60uA(idle)

7	Response time	6s-15s
8	Sampling Frequency	1HZ
9	Dimensions	27mm x 59mm x 13.5mm

Tabel.4.2. Features and Specifications

Applications:

Offices, cars, museums, greenhouses and industries use this sensor for measuring humidity values and as a safety measure. Temperature and humidity sensors are among the most commonly used environmental sensors. Humidity sensors are also sometimes referred to as hygrometers. These devices are used to provide the actual humidity condition within the air at any given point or in any given place.

4.6.4. Relay

Relays are switches that open and close circuits electromechanically or electronically. Relays control one electrical circuit by opening and closing contacts in another circuit.

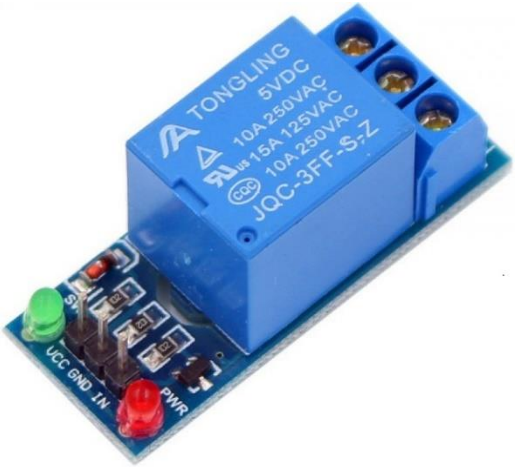


Fig: 4.9. Relay

Relays are integral components in smart irrigation systems, facilitating the automation and control necessary for efficient water management. Here’s how relays are typically used in such systems:

Features:

1. Controlling Water Pumps:

Relays enable the remote or automated control of water pumps in an irrigation system. When the system's controller (such as a microcontroller or a dedicated irrigation controller) detects the need for watering based on timers or sensor input (like soil moisture levels), it sends a signal to the relay. The relay then activates, turning on the water pump to deliver water to the crops or plants. This helps in managing water use effectively without manual intervention.

2. Activating Solenoid Valves:

Solenoid valves control the flow of water in different zones of an irrigation system. Relays are used to switch these valves on and off. By integrating relays, a smart irrigation system can precisely control which areas receive water and for how long, based on specific watering needs of different plant types or soil conditions. This zonal control helps in optimizing water usage and ensures that water is not wasted.

3. Interfacing with Sensors and Other Devices:

Smart irrigation systems often include various sensors such as soil moisture sensors, temperature sensors, and rain sensors. Although relays do not directly interact with these sensors, they act upon the data provided by them through the controller. For example, if a rain sensor detects sufficient moisture, the irrigation controller can use relays to stop all watering activities, thus conserving water.

4. Power Management:

Relays can manage the distribution and switching of power within the irrigation system, especially in larger setups where multiple devices (like multiple pumps and an array of solenoid valves) need controlled power for operation. This is crucial in preventing overloads and ensuring that only necessary components are powered at any given time, enhancing the energy efficiency of the system.

5. Integration with Smart Home Systems:

In residential applications, smart irrigation systems can be integrated with home automation systems. Relays enable this integration by allowing the irrigation system to be controlled through the home automation's interface, which might include smartphone apps or voice control systems. This provides users with convenience and flexibility in managing their garden or lawn irrigation.

6. Safety and Protection:

Relays also play a critical role in safety and protective measures in irrigation systems. They can be used to cut off power in case of a fault, such as a leak detection or pump failure, to prevent water wastage or electrical hazards.

Pin Description:

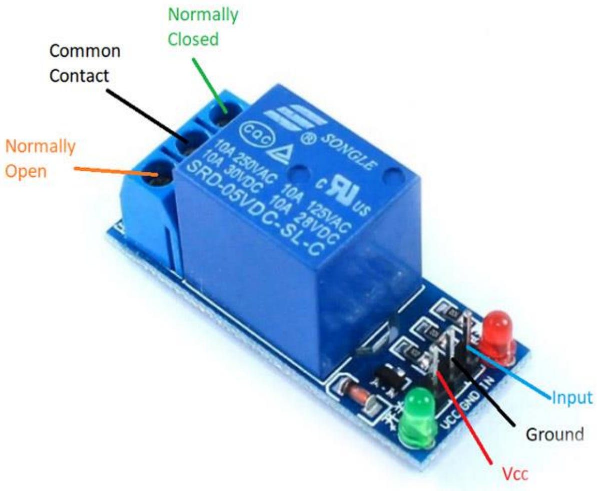


Fig: 4.10. Pin description of Relay

1.Coil Pins: These are the pins connected to the coil of the relay.

Coil+ (C+): This pin connects to one end of the coil.

Coil- (C-): This pin connects to the other end of the coil.

2.Contact Pins:_These are the pins connected to the switch contacts.

Normally Open (NO): This pin is disconnected when the relay is not energized (de-energized).

Common (COM): This is the common pin to which one end of both the NO and NC contacts are connected.

Normally Closed (NC): This pin is connected when the relay is not energized (de-energized).

Benefits in Smart Irrigation:

Efficiency: Reduces water usage by allowing precise control over irrigation based on real-time environmental data.

Convenience: Automates the watering process, eliminating the need for manual intervention and making it easier to manage large or complex gardens and agricultural fields.

Adaptability: Can be easily programmed to adapt to different watering schedules and requirements as per seasonal changes or specific crop needs.

In summary, relays are fundamental for the automated control systems in smart irrigation, enabling not only the efficient use of water resources but also integrating with broader smart home or agricultural management systems for enhanced control and monitoring.

4.6.5. DC Motor Pump:

DC driven pumps transfer fluid in a number of ways by using direct current from a motor, battery, or solar power. Usually, motorized pumps require 6, 12, 24, or 32 volts of DC electricity to function. Photovoltaic (PV) panels with solar cells—which generate direct current when exposed to sunlight—are used in solar-powered DC pumps.

Fig: 4.11. DC Motor Pump

Basic Structure:

DC Motor: The driving force behind the pump, a DC motor operates on direct current electricity. It typically consists of a rotor (armature), stator, commutator, and brushes. The motor's speed can be easily controlled by varying the voltage applied to it, which is a significant advantage in applications requiring variable flow rates.

Pump Mechanism: Attached directly to the motor, the pump mechanism can vary depending on the application and might include types such as centrifugal pumps, diaphragm pumps, or gear pumps. Each type has a different method of moving fluid, chosen based on the required pressure, flow rate, and nature of the fluid.

Functionality:

Operation: When the DC motor is powered, the electrical energy is converted into mechanical energy, causing the motor shaft to rotate. This rotation is transferred to the pump mechanism, which then moves the fluid through the system by creating variations in pressure or by direct displacement.

Control: The flow rate and pressure output of the pump can be controlled by adjusting the voltage supplied to the DC motor, making it highly adaptable to different operating conditions.

Features:

DC motor pumps are commonly used in automatic irrigation systems due to their efficiency, controllability, and suitability for off-grid applications. Here are some key features and considerations:

Efficiency: DC motor pumps are generally more energy-efficient than their AC counterparts. They can be designed to operate at variable speeds, allowing for better matching of pump output to irrigation system demand.

Variable Speed Control: DC motor pumps can be easily controlled to vary the flow rate and pressure of water, enabling precise irrigation according to the needs of different plants, soil types, and weather conditions.

Direct Operation from DC Power Sources: DC motor pumps can be directly powered by batteries, solar panels, or other DC power sources, making them suitable for remote or off-grid locations where AC power is unavailable or expensive to install.

High Starting Torque: DC motor pumps typically offer high starting torque, allowing them to quickly reach operating speed even under load. This is beneficial for priming the pump and maintaining consistent water pressure.

Compact Size and Lightweight: DC motor pumps are often more compact and lightweight compared to AC pumps of similar capacity, making them easier to install and transport.

Low Maintenance: DC motor pumps generally require less maintenance than AC pumps due to their simpler design and fewer moving parts. This reduces downtime and operating costs over the pump's lifespan.

Corrosion Resistance: For irrigation systems that use water from sources with high mineral content or potential for corrosion, DC motor pumps can be constructed with corrosion-resistant materials like stainless steel or engineered plastics.

Overload Protection: Many DC motor pumps are equipped with built-in overload protection features, such as thermal sensors or current-limiting circuits, to prevent damage in case of overheating or overloading.

Noise Level: DC motor pumps tend to operate more quietly than AC pumps, which can be advantageous in residential or noise-sensitive areas.

Remote Monitoring and Control: Some advanced DC motor pumps offer remote monitoring and control capabilities, allowing users to adjust pump settings or receive alerts about pump status via mobile apps or web interfaces.

Long Lifespan: With proper maintenance, DC motor pumps can have a long service life, providing reliable water delivery for many years.

4.6.6. LCD DISPLAY

Fig: 4.12. LCD Display

Nowadays, we always use the devices which are made up of LCDs such as CD players, DVD players, digital watches, computers, etc. These are commonly used in the screen industries to replace the utilization of CRTs. Cathode Ray Tubes use huge power when compared with LCDs, and CRTs heavier as well as bigger.

These devices are thinner as well power consumption is extremely less. The LCD 16×2 working principle is, it blocks the light rather than dissipate. This article discusses an overview of LCD 16X2, pin configuration and its working.

The term LCD stands for liquid crystal display. It is one kind of electronic display module used in an extensive range of applications like various circuits & devices like mobile phones, calculators, computers, TV sets, etc.

These displays are mainly preferred for multisegment light-emitting diodes and seven segments. The main benefits of using this module are inexpensive; simply programmable, animations, and there are no limitations for displaying custom characters, special and even animations, etc.

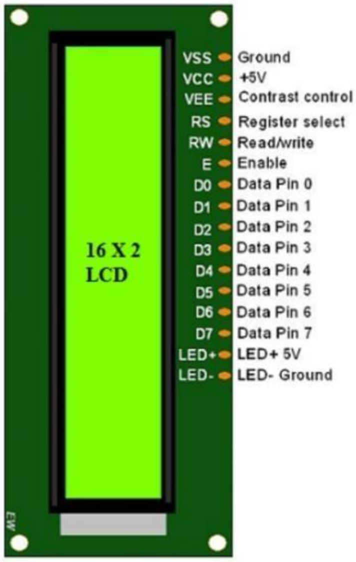


Fig: 4.13. Pin Description

Pin1 (Ground/Source Pin): This is a GND pin of display, used to connect the GND terminal of the microcontroller unit or power source.

Pin2 (VCC/Source Pin): This is the voltage supply pin of the display, used to connect the supply pin of the power source.

Pin3 (V0/VEE/Control Pin): This pin regulates the difference of the display, used to connect a changeable POT that can supply 0 to 5V.

Pin4 (Register Select/Control Pin): This pin toggles among command or data register, used to connect a microcontroller unit pin and obtains either 0 or 1 (0 = data mode , and 1 = command mode).

Pin5 (Read/Write/Control Pin): This pin toggles the display among the read or writes operation, and it is connected to a microcontroller unit pin to get either 0 or 1 (0 = Write Operation, and 1 = Read Operation).

Pin 6 (Enable/Control Pin): This pin should be held high to execute Read/Write process, and it is connected to the microcontroller unit & constantly held high.

Pins 7-14 (Data Pins): These pins are used to send data to the display. These pins are connected in two-wire modes like 4-wire mode and 8-wire mode. In 4-wire mode, only four pins are connected to the microcontroller unit like 0 to 3, whereas in 8-wire mode, 8-pins are connected to microcontroller unit like 0 to 7.

Pin15 (+ve pin of the LED): This pin is connected to +5V

Pin 16 (-ve pin of the LED): This pin is connected to GND.

Features of LCD16x2

The features of this LCD mainly include the following.

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

Registers of LCD

1. A 16×2 LCD has two registers like data register and command register. The RS (register select) is mainly used to change from one register to another. When the register set is ‘0’,

then it is known as command register. Similarly, when the register set is ‘1’, then it is known as data register.

2. Command Register: The main function of the command register is to store the instructions of command which are given to the display. So that predefined tasks can be performed such as clearing the display, initializing, set the cursor place, and display control. Here commands processing can occur within the register.
3. Data Register: The main function of the data register is to store the information which is to be exhibited on the LCD screen. Here, the ASCII value of the character is the information which is to be exhibited on the screen of LCD. Whenever we send the information to LCD, it transmits to the data register, and then the process will be starting there. When register set =1, then the data register will be selected.
4. Registers of LCD:A 16×2 LCD has two registers like data register and command register. The RS (register select) is mainly used to change from one register to another. When the register set is ‘0’, then it is known as command register. Similarly, when the register set is ‘1’, then it is known as data register.
5. Command Register: The main function of the command register is to store the instructions of command which are given to the display. So that predefined tasks can be performed such as clearing the display, initializing, set the cursor place, and display control. Here commands processing can occur within the register.
6. Data Register: The main function of the data register is to store the information which is to be exhibited on the LCD screen. Here, the ASCII value of the character is the information which is to be exhibited on the screen of LCD. Whenever we send the information to LCD, it transmits to the data register, and then the process will be starting there. When register set =1, then the data register will be selected.

16×2 LCD Commands

The commands of LCD 16X2 include the following.

1. For Hex Code-01, the LCD command will be the clear LCD screen
2. For Hex Code-02, the LCD command will be returning home
3. For Hex Code-04, the LCD command will be decrement cursor
4. For Hex Code-06, the LCD command will be Increment cursor

5. For Hex Code-05, the LCD command will be Shift display right
6. For Hex Code-07, the LCD command will be Shift display left
7. For Hex Code-08, the LCD command will be Display off, cursor off
8. For Hex Code-0A, the LCD command will be cursor on and display off
9. For Hex Code-0C, the LCD command will be cursor off, display on
10. For Hex Code-0E, the LCD command will be cursor blinking, Display on
11. For Hex Code-0F, the LCD command will be cursor blinking, Display on
12. For Hex Code-10, the LCD command will be Shift cursor position to left
13. For Hex Code-14, the LCD command will be Shift cursor position to the right
14. For Hex Code-18, the LCD command will be Shift the entire display to the left
15. For Hex Code-1C, the LCD command will be Shift the entire display to the right
16. For Hex Code-80, the LCD command will be Force cursor to the beginning (1st line)
17. For Hex Code-C0, the LCD command will be Force cursor to the beginning (2nd line)
18. For Hex Code-38, the LCD command will be 2 lines and 5×7 matrix

SPECIFICATIONS OF 16X2 LCD DISPLAY MODULE:

1. Operating Voltage: 4.7V to 5.3V
2. Operating Current 1mA (without backlight)
3. Can display (16x2) 32 Alphanumeric Characters
4. Custom Characters Support
5. Works in both 8-bit and 4-bit Mode

As you can see the module has (from right) two power pins Vss and Vcc to power the LCD. Typically Vss should be connected to ground and Vcc to 5V, but the LCD can also operate from voltage between 4.7V to 5.3V. Next, we have the control pins namely Contrast (VEE), Register Select (RS), Read/Write (R/W) and Enable (E). The Contrast pin is used to set the contrast (visibility) of the characters, normally it is connected to a 10k potentiometer so that the contrast can be adjusted. The Read/Write pin will be grounded in most cases because we will only be writing characters to the LCD and not read anything from it. The Register Select (RS) and Enable pin (E) pin are the control pins of the LCD and will be connected to the digital pins GPIO pins of the microcontroller.

Pin Description

Pin no	Pin Symbol	Pin Name	Connection Description
1	VSS	Ground	Connected to Ground
2	VCC	Power	Connected to Power (Typically 5V)
3	VEE	Contrast	Connected to potentiometer 10k to control contrast
4	RS	Register select	Connected to Microcontroller
5	R/W	Read/Write	Connected to Ground
6	E	Enable	Connected to Microcontroller
7	DB0	Data Pin 0	Connected to Microcontroller based on 4-bit or 8-bit working mode
8	DB1	Data Pin 1	
9	DB2	Data Pin 2	
10	DB3	Data Pin 3	
11	DB4	Data Pin 4	
12	DB5	Data Pin 5	
13	DB6	Data Pin 6	
14	DB7	Data Pin 7	
15	LED+	LED Positive	Connected to 5V through 100 ohm Resistor
16	LED-	LED Negative	Connected to Ground

Tabel.4.3. pin specifications of 16X2 LCD Display

4.6.7. Power Supply:

The Arduino board can be powered by a solar panel, plug-in adapter, wall wart, or 7V to 12V power supply. For the pump motor, you'll need an additional 12V battery, power supply, or solar panel. An analog output that the moisture sensor produces can be readily interfaced with an Arduino board.

Does irrigation need power supply?

The decision is made for you, whether you do or do not have power available to the area you want to irrigate. If you have no options for a power supply, you will have to go for a DC Controller. If you have the option of power, then you can have an AC controller.

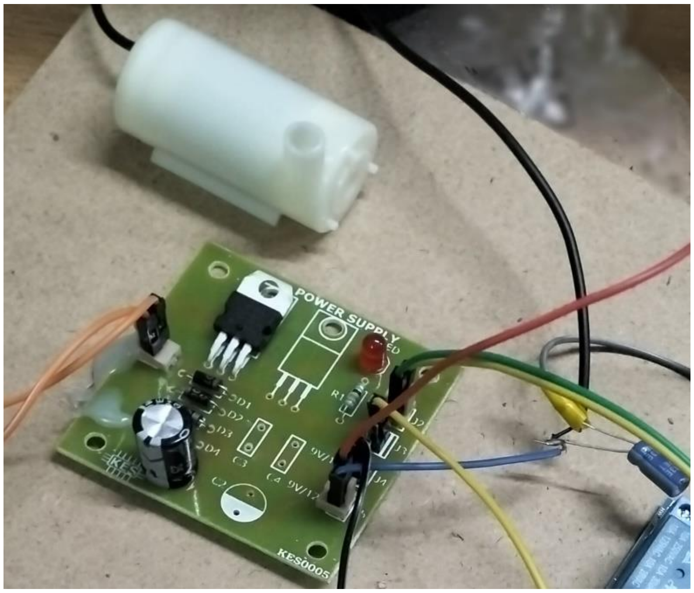


Fig: 4.14. Power Supply

For an automatic irrigation system, the power supply needed depends on several factors such as the size of the system, the type of components used, and whether it's being powered by mains electricity or battery.

Here's a breakdown of the power supply considerations:

Voltage and Current: Determine the voltage and current requirements of all the components in the irrigation system. This includes the controller, solenoid valves, sensors, and any other accessories.

Controller Power Supply: Most irrigation controllers operate on low voltage, typically 12V or 24V. They may be powered by a plug-in adapter, batteries, or directly from a mains power source. Choose a power supply with the appropriate voltage and current rating for your controller.

Solenoid Valves: Solenoid valves are the devices that control the flow of water in the irrigation system. They are usually operated using low voltage (commonly 12V or 24V) to prevent electrical hazards. The power supply should be able to provide enough power to activate all the solenoid valves simultaneously.

Backup Power: If the system is designed to operate during power outages, consider using a backup power source such as batteries or a generator. Make sure the backup power source can provide sufficient power to run the system for the required duration.

Weatherproofing: Ensure that the power supply and all electrical connections are adequately weatherproofed if they are exposed to the elements.

Energy Efficiency: Consider using energy-efficient components to reduce power consumption, especially if the system is powered by batteries or solar panels.

Safety: Always prioritize safety when dealing with electrical systems. Use appropriate safety measures such as circuit breakers, ground fault circuit interrupters (GFCIs), and waterproof enclosures.

Remote Access: If the system includes remote access or control features, ensure that the power supply can support any additional requirements such as Wi-Fi connectivity or cellular modems.

Block Diagram of Power Supply:

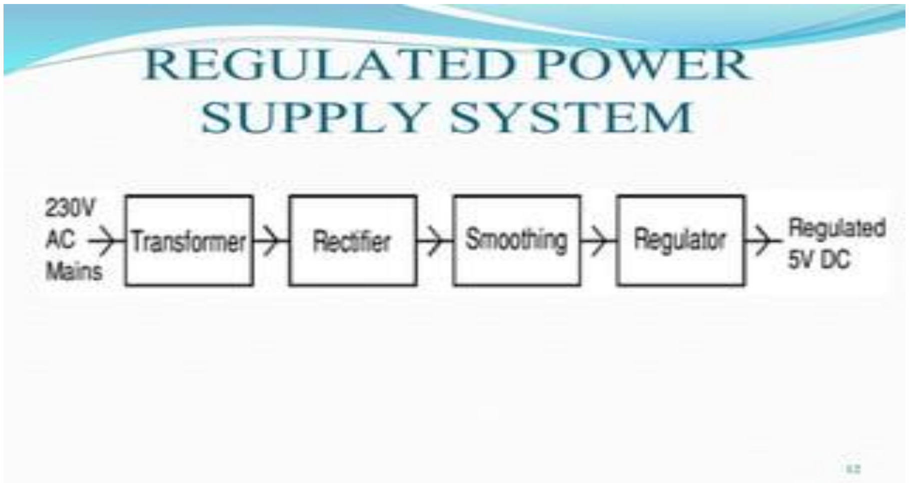


Fig: 4.15. Block Diagram of Power Supply

This block diagram illustrates the flow of power and control signals within an automatic irrigation system, from the power source to the various components responsible for managing water distribution.

TRANSFORMER:

Voltage Transformation: One of the primary functions of a transformer is to change the voltage level of an alternating current (AC) signal. This is accomplished through electromagnetic induction.

Step-Up or Step-Down: Transformers can step up (increase) or step down (decrease) the input voltage level to match the requirements of the load or the system. In an irrigation system, the voltage might need to be stepped down from the standard mains voltage to a lower level suitable for the system components, typically 12V or 24V.

Safety: Transforming voltage is important for safety reasons. Lower voltage systems are generally safer to work with, reducing the risk of electric shock. In an irrigation system, using lower voltages is particularly important if the system is installed in damp or wet environments.

Isolation: Transformers provide electrical isolation between the input and output circuits. This isolation protects the load (the irrigation system components) from voltage spikes, noise, and faults that may occur in the mains supply.

Galvanic Isolation: Transformers provide galvanic isolation, meaning there is no direct electrical connection between the input and output windings. This isolation prevents ground loops and other unwanted interactions between different parts of the system.

Impedance Matching: In some cases, transformers are used to match impedance between the power source and the load. This helps to maximize power transfer efficiency.

Filtering: Transformers also provide some degree of filtering. They can help attenuate certain types of noise and interference, improving the quality of the power supplied to the system.

Power Distribution: Transformers can be used to distribute power across different parts of a system. For example, in an irrigation system, a single transformer might power multiple solenoid valves or sensors.

RECTIFIER:

AC to DC Conversion: The primary function of a rectifier is to convert alternating current (AC) into direct current (DC). In most countries, household power is supplied as AC, typically at a frequency of 50 Hz or 60 Hz. However, many electronic devices, including irrigation system components, operate on DC power.

Diode Bridge Rectification: Most rectifiers in power supplies use a configuration called a diode bridge rectifier. This arrangement allows current to flow in only one direction, effectively converting the -ve half of AC waveform into positive DC voltage.

Smoothing: While the output of a rectifier is technically DC, it's not pure DC; it's a pulsating voltage with significant ripple. To smooth this out, a filter capacitor is often connected across the output of the rectifier. This capacitor charges up during the peaks of the AC waveform and discharges during the troughs, effectively reducing the ripple and providing a more stable DC

Output Voltage Regulation: Some rectifiers also include voltage regulation circuits to maintain a consistent output voltage despite fluctuations in the input voltage or load variations. This ensures that the connected devices receive a steady and reliable power supply.

Protection Diodes: Rectifiers may include protection diodes to prevent damage from reverse polarity connections. These diodes block current flow if the input voltage is applied in reverse, protecting the circuitry from potential damage.

SMOOTHING:

Reducing Ripple Voltage: After rectification, the output of the power supply is DC but not perfectly smooth. It contains fluctuations known as ripple voltage, caused by the alternating nature of the input AC voltage and the charging and discharging cycles of the capacitor. The smoothing capacitor effectively smooths out this ripple, resulting in a more stable DC voltage.

Stabilizing Output Voltage: By reducing ripple voltage, the smoothing capacitor stabilizes the output voltage. This is essential for ensuring that the voltage supplied to the load remains relatively constant over time, regardless of variations in the input voltage or changes in the load current.

Improving Regulation: Smoothing capacitors improve the regulation of the power supply. Voltage regulation refers to the ability of the power supply to maintain a constant output voltage despite changes in the input voltage or load current. By smoothing out the ripple, the capacitor helps to maintain a more consistent output voltage, contributing to better regulation.

Filtering High-Frequency Noise: Smoothing capacitors also act as filters, attenuating high-frequency noise present in the power supply output. This noise can be generated by switching components in the power supply circuitry or induced from external sources. The capacitor shunts high-frequency noise to ground, preventing it from reaching the load and ensuring a cleaner power supply output.

Energy Storage: The smoothing capacitor stores energy during the peaks of the rectified waveform and releases it during the troughs. This helps to fill in the gaps between the peaks, resulting in a smoother output voltage.

Improving Transient Response: Smoothing capacitors improve the transient response of the power supply. Transients are sudden changes in load current or input voltage. The capacitor helps to supply additional current during transient events, minimizing voltage fluctuations and ensuring the stability of the power supply output.

REGULATOR:

Stabilizing Output Voltage: The primary function of a voltage regulator is to stabilize the output voltage of the power supply. It ensures that the output voltage remains constant within a specified range, even when the input voltage varies or the load current changes.

Load Regulation: Voltage regulators maintain a stable output voltage despite changes in the load current drawn by the connected devices. This prevents voltage droop or fluctuations that could affect the proper functioning of electronic components.

Line Regulation: Voltage regulators compensate for fluctuations in the input voltage (line voltage). This is particularly important in systems powered by fluctuating mains voltage or batteries with varying charge levels.

Voltage Reference: Voltage regulators use a stable voltage reference to compare against the output voltage. This reference voltage is typically very precise, ensuring accurate regulation of the output voltage.

Error Amplifier: Voltage regulators employ an error amplifier to compare the output voltage to the reference voltage. Any difference between the two voltages (error) is amplified and used to adjust the output voltage.

Feedback Loop: The amplified error signal is fed back into the voltage regulator circuit to control the output voltage. This feedback loop continuously adjusts the regulator to maintain the desired output voltage.

Protection Features: Many voltage regulators include protection features such as overvoltage protection (OVP), undervoltage protection (UVP), and thermal shutdown to safeguard against overloads, short circuits, or excessive temperatures.

Transient Response: Voltage regulators ensure a fast and stable response to sudden changes in load current or input voltage. This prevents voltage spikes or drops during transient events, maintaining stability in the power supply output.

Efficiency: While stabilizing the output voltage, voltage regulators operate with high efficiency, minimizing power losses and heat dissipation.

Types of Voltage Regulators:

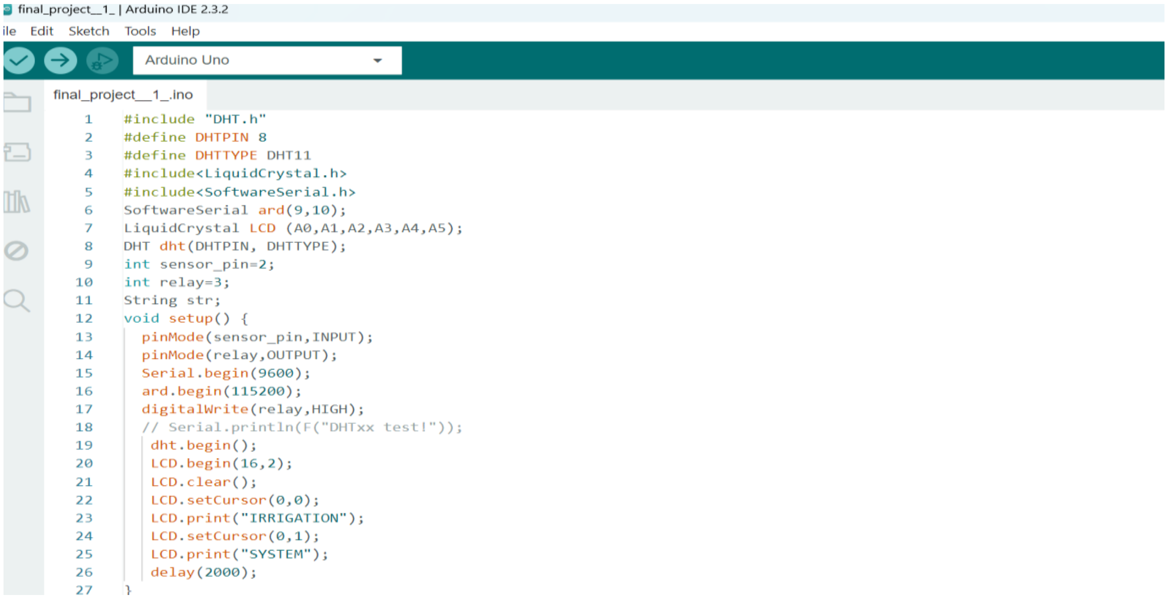
i. Linear Regulators: Provide a simple and low-noise solution for voltage regulation but are less efficient for large voltage differences.

ii. Switching Regulators: Offer higher efficiency by converting the input voltage to a high-frequency AC signal, then rectifying and filtering it back to DC. These are more complex but provide better efficiency, especially for larger voltage differences.

CHAPTER 5

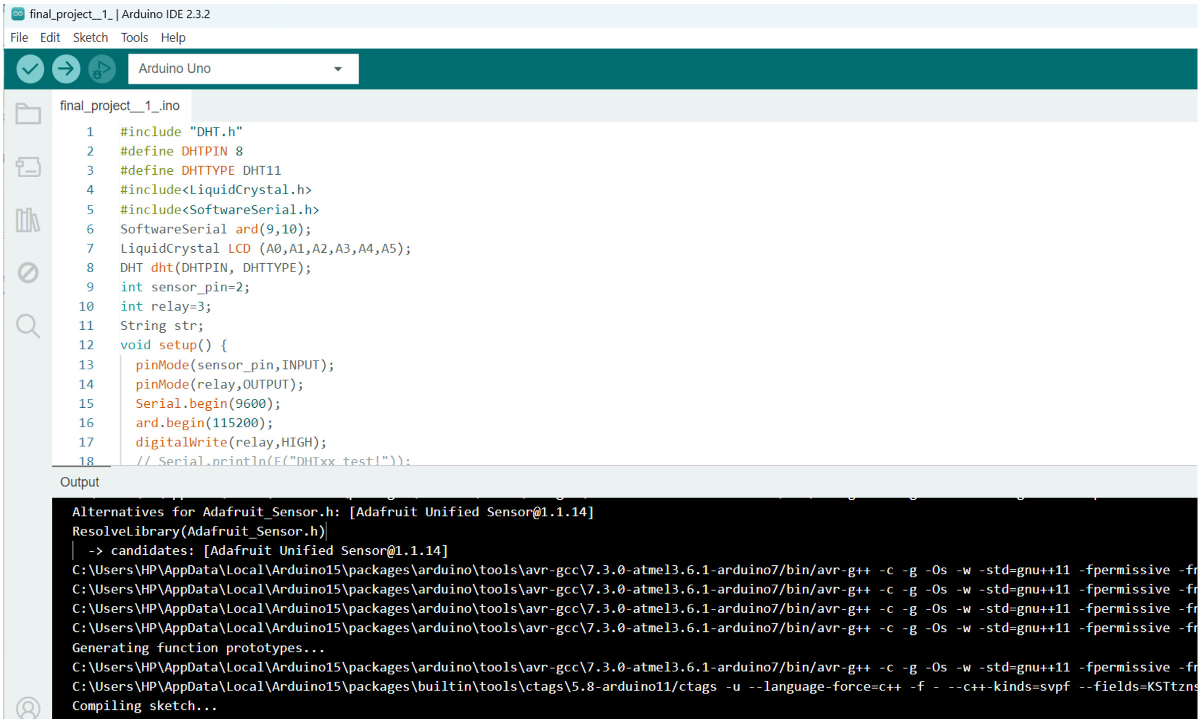
RESULTS

5.1. RESULTS / OUTPUTS


final_project__1_ | Arduino IDE 2.3.2
ile Edit Sketch Tools Help
Arduino Uno
final_project__1_.ino

```
#include "DHT.h"
#define DHTPIN 8
#define DHTTYPE DHT11
#include<LiquidCrystal.h>
#include<SoftwareSerial.h>
SoftwareSerial ard(9,10);
LiquidCrystal LCD (A0,A1,A2,A3,A4,A5);
DHT dht(DHTPIN, DHTTYPE);
int sensor_pin=2;
int relay=3;
String str;
void setup() {
  pinMode(sensor_pin,INPUT);
  pinMode(relay,OUTPUT);
  Serial.begin(9600);
  ard.begin(115200);
  digitalWrite(relay,HIGH);
  // Serial.println(F("DHTxx test!"));
   dht.begin();
   LCD.begin(16,2);
   LCD.clear();
   LCD.setCursor(0,0);
   LCD.print("IRRIGATION");
   LCD.setCursor(0,1);
   LCD.print("SYSTEM");
   delay(2000);
}
```

Fig.5.1. Serial Monitor


final_project__1_ | Arduino IDE 2.3.2
File Edit Sketch Tools Help
Arduino Uno
final_project__1_.ino
Output

```
#include "DHT.h"
#define DHTPIN 8
#define DHTTYPE DHT11
#include<LiquidCrystal.h>
#include<SoftwareSerial.h>
SoftwareSerial ard(9,10);
LiquidCrystal LCD (A0,A1,A2,A3,A4,A5);
DHT dht(DHTPIN, DHTTYPE);
int sensor_pin=2;
int relay=3;
String str;
void setup() {
  pinMode(sensor_pin,INPUT);
  pinMode(relay,OUTPUT);
  Serial.begin(9600);
  ard.begin(115200);
  digitalWrite(relay,HIGH);
  // Serial.println(F("DHTxx test!"));
```

```
Alternatives for Adafruit_Sensor.h: [Adafruit Unified Sensor@1.1.14]
ResolveLibrary(Adafruit_Sensor.h)
 │ -> candidates: [Adafruit Unified Sensor@1.1.14]
C:\Users\HP\AppData\Local\Arduino15\packages\arduino\tools\avr-gcc\7.3.0-atmel3.6.1-arduino7/bin/avr-g++ -c -g -Os -w -std=gnu++11 -fpermissive -fr
C:\Users\HP\AppData\Local\Arduino15\packages\arduino\tools\avr-gcc\7.3.0-atmel3.6.1-arduino7/bin/avr-g++ -c -g -Os -w -std=gnu++11 -fpermissive -fr
C:\Users\HP\AppData\Local\Arduino15\packages\arduino\tools\avr-gcc\7.3.0-atmel3.6.1-arduino7/bin/avr-g++ -c -g -Os -w -std=gnu++11 -fpermissive -fr
C:\Users\HP\AppData\Local\Arduino15\packages\arduino\tools\avr-gcc\7.3.0-atmel3.6.1-arduino7/bin/avr-g++ -c -g -Os -w -std=gnu++11 -fpermissive -fr
Generating function prototypes...
C:\Users\HP\AppData\Local\Arduino15\packages\arduino\tools\avr-gcc\7.3.0-atmel3.6.1-arduino7/bin/avr-g++ -c -g -Os -w -std=gnu++11 -fpermissive -fr
C:\Users\HP\AppData\Local\Arduino15\packages\builtin\tools\ctags\5.8-arduino11/ctags -u --language-force=c++ -f - --c++-kinds=svpf --fields=KSTtzns
Compiling sketch...
```

Fig.5.2 Execution of code

After achieving the correct results connect the Arduino uno and 16x2 lcd display to the Arduino uno and dump the code to get the required data.

When the code dump into the Arduino Board, the values i.e., temperature, moisture, and humidity will be displayed on LCD Display. And the MOTOR ON and OFF condition also monitored.

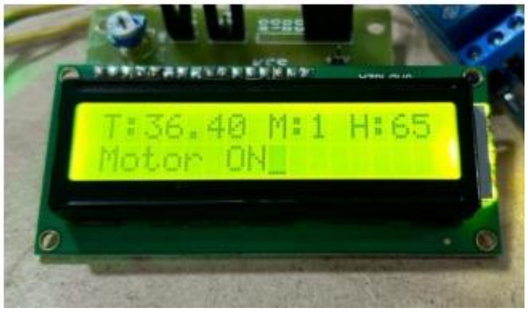


Fig: 5.3. ON condition displayed on LCD

Turn on the motor if the temperature is higher than 34C and the soil moisture content is 1 (dry). Also, the motor will turn on if the temperature rises over 34C and the soil moisture content is 0 (wet).

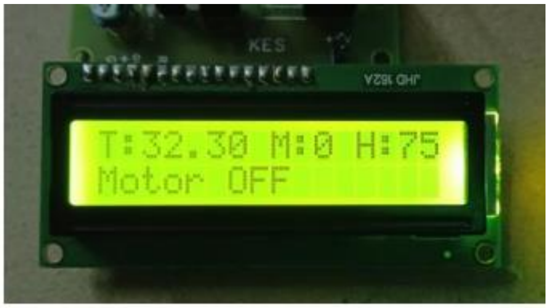
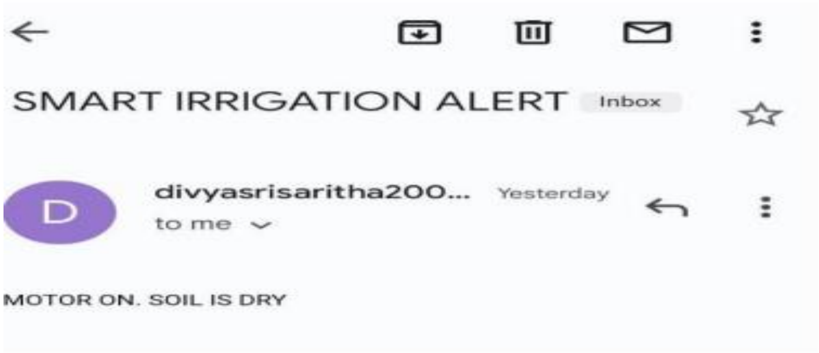


Fig: 5.4. OFF condition displayed on LCD

The motor will be turned off if the temperature drops below 34°C and the soil is wet (0°). Moreover, the motor turns on if the temperature drops below 34°C and the soil moisture content is 1 (i.e., dry).

5.2. IMPLEMENTATION:

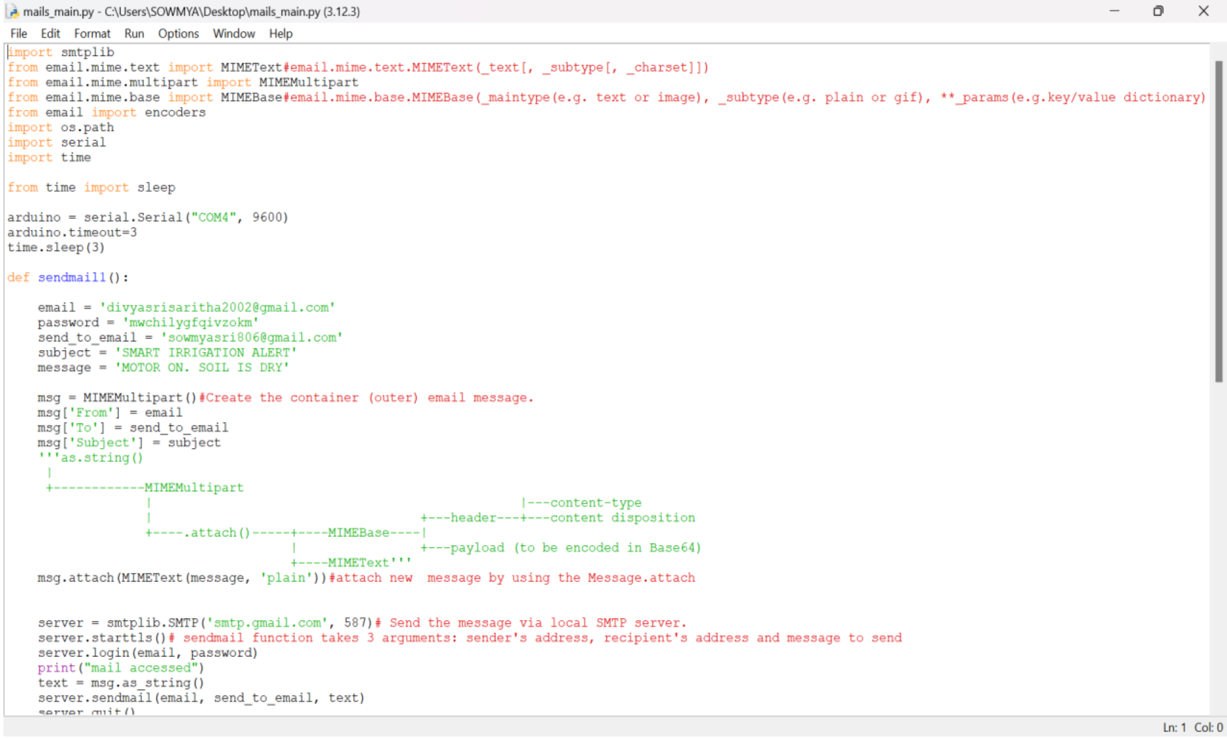
Gmail Notification



MOTOR ON. SOIL IS DRY

Fig: 5.5. Gmail Notification

Farmers will receive a Gmail notification about Moisture and motor status using Python code. Python programming is used for irrigation system to improve its connectivity and functionality. The code communicates the moisture and motor status to Arduino DHT11 sensor Relay Moisture LCD sensor Pump motor Power Supply PC Python Gmail. This allows farmers to keep an eye on the irrigation system using their smartphones.

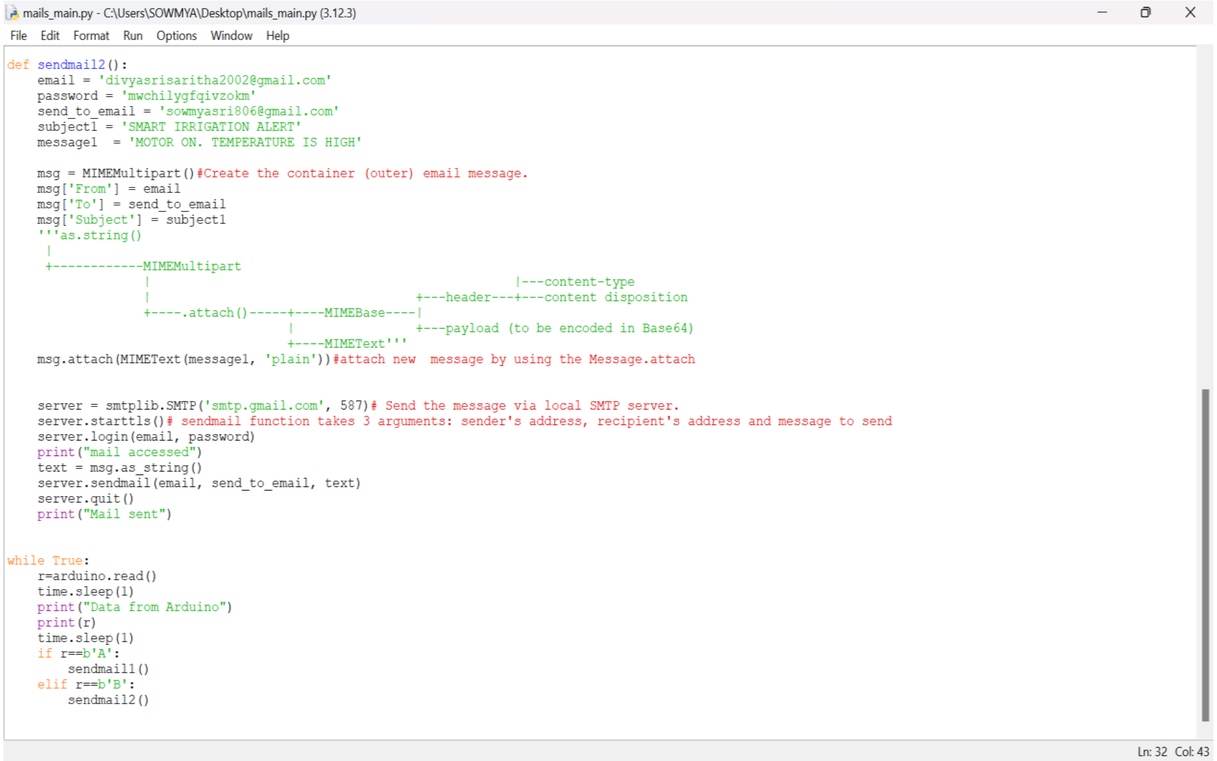

mails_main.py - C:\Users\SOWMYA\Desktop\mails_main.py (3.12.3)
File Edit Format Run Options Window Help
Ln: 1 Col: 0

```
import smtplib
from email.mime.text import MIMEText#email.mime.text.MIMEText(_text[, _subtype[, _charset]])
from email.mime.multipart import MIMEMultipart
from email.mime.base import MIMEBase#email.mime.base.MIMEBase(_maintype(e.g. text or image), _subtype(e.g. plain or gif), **_params(e.g.key/value dictionary)
from email import encoders
import os.path
import serial
import time

from time import sleep

arduino = serial.Serial("COM4", 9600)
arduino.timeout=3
time.sleep(3)

def sendmail1():

    email = 'divyasrisaritha2002@gmail.com'
    password = 'mwchilygfqivzokm'
    send_to_email = 'sowmyasri806@gmail.com'
    subject = 'SMART IRRIGATION ALERT'
    message = 'MOTOR ON. SOIL IS DRY'

    msg = MIMEMultipart()#Create the container (outer) email message.
    msg['From'] = email
    msg['To'] = send_to_email
    msg['Subject'] = subject
    '''as.string()
     |
     +------------MIMEMultipart
                  |                                                |---content-type
                  |                                   +---header---+---content disposition
                  +----.attach()-----+----MIMEBase----|
                                     |                +---payload (to be encoded in Base64)
                                     +----MIMEText'''
    msg.attach(MIMEText(message, 'plain'))#attach new  message by using the Message.attach


    server = smtplib.SMTP('smtp.gmail.com', 587)# Send the message via local SMTP server.
    server.starttls()# sendmail function takes 3 arguments: sender's address, recipient's address and message to send
    server.login(email, password)
    print("mail accessed")
    text = msg.as_string()
    server.sendmail(email, send_to_email, text)
    server.quit()
```

Secure Credentials: Avoid hardcoding sensitive information such as passwords directly in the code. Use environment variables or configuration files that are not included in the source code repository.

Error Handling: Proper error handling has been added to inform you if the email was not sent successfully.

Testing: Test the function in a controlled environment to ensure it behaves as expected without spamming unintended recipients.


mails_main.py - C:\Users\SOWMYA\Desktop\mails_main.py (3.12.3)
File Edit Format Run Options Window Help
Ln: 32 Col: 43

```python
def sendmail2():
    email = 'divyasrisaritha2002@gmail.com'
    password = 'mwchilygfqivzokm'
    send_to_email = 'sowmyasri806@gmail.com'
    subject1 = 'SMART IRRIGATION ALERT'
    message1  = 'MOTOR ON. TEMPERATURE IS HIGH'

    msg = MIMEMultipart()#Create the container (outer) email message.
    msg['From'] = email
    msg['To'] = send_to_email
    msg['Subject'] = subject1
    '''as.string()
     |
     +------------MIMEMultipart
                  |                                                |---content-type
                  |                                   +---header---+---content disposition
                  +----.attach()-----+----MIMEBase----|
                                     |                +---payload (to be encoded in Base64)
                                     +----MIMEText'''
    msg.attach(MIMEText(message1, 'plain'))#attach new  message by using the Message.attach


    server = smtplib.SMTP('smtp.gmail.com', 587)# Send the message via local SMTP server.
    server.starttls()# sendmail function takes 3 arguments: sender's address, recipient's address and message to send
    server.login(email, password)
    print("mail accessed")
    text = msg.as_string()
    server.sendmail(email, send_to_email, text)
    server.quit()
    print("Mail sent")


while True:
    r=arduino.read()
    time.sleep(1)
    print("Data from Arduino")
    print(r)
    time.sleep(1)
    if r==b'A':
        sendmail1()
    elif r==b'B':
        sendmail2()
```

Testing: Test the function in a controlled environment first to make sure it functions as expected.

Environment Variables: Set the EMAIL_PASSWORD environment variable in your system before running the script, or use a .env file and load it with libraries like dotenv.

CHAPTER 6

CONCLUSION & FUTURE SCOPE

6.1. CONCLUSION

In conclusion, the Internet of Things-based intelligent irrigation system offers a revolutionary approach to contemporary agriculture, tackling major issues that confront farmers all over the world. Through the utilization of cutting-edge technologies like automation, data analytics, and sensor networks, the system provides agricultural irrigation management with previously unheard-of levels of sustainability, efficiency, and precision. Farmers are able to optimize water use, reduce resource waste, and increase crop output by using real-time monitoring of soil moisture, weather, and crop health. Moreover, farming methods are made more convenient and flexible by the system's capacity to remotely monitor and control irrigation operations.

Using Python code, a Gmail alert regarding the moisture and motor status would be sent to farmers. Irrigation systems with Python programming have better functionality and connection. The Arduino DHT11 sensor Relay Moisture LCD sensor Pump motor Power Supply PC Python Gmail receives notification of the moisture and motor declare via program. This enables farmers to use their smartphones to monitor the irrigation system.

Turn on the motor if the temperature rises above 34C and the soil moisture level is 1 (dry). Additionally, the motor will activate if the temperature goes above 34C and the soil moisture content is 0 (wet). The motor will turn off if the temperature drops below 34°C and the soil is wet (0°). Furthermore, the motor turns on when the temperature drops below 34°C and the soil moisture level is 1 (dry).

6.2. FUTURE SCOPE

Future automatic pump control systems based on moisture and temperature conditions have a wide and bright future, especially when combined with IoT (Internet of Things) technology. These technologies can be used in many different fields, such as smart city infrastructure, industrial processes, and agriculture. Some of the main areas for upcoming advancements and possible inventions.

Using a compound annual growth rate (CAGR) of 13.8% from 2020 to 2025, research indicates that the global market for IoT-based pump monitoring solutions will be valued at $6.3 billion.

This makes sense because pump monitoring systems based on the Internet of Things can reduce energy use by up to 30%.

REFERENCES

1. Navarro-Hellín, H., Martínez-del-Rincon, J., Domingo-Miguel, R., Soto-Valles, F. andTorresSánchez, R., 2016. A decision support system for managing irrigation in agriculture. Computers and Electronics in Agriculture, 124, pp.121-131.
2. S.Shobana, B. Sanjana Pandey, Padmashri.R, U.Triveni Iot Based Smart Irrigation System Using Soil Moisture Sensor And Esp8266nodemcu International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online) Vol. 9, Issue 2, pp: (52-58), Month: April - June 2021
3. Kavianand, G., V. M. Nivas, R. Kiruthika, and S. Lalitha. "Smart drip irrigation system for sustainable agriculture." In Technological Innovations in ICT for Agriculture and Rural Development (TIAR), 2016 IEEE, pp. 19-22. IEEE, 2016.
4. Wu, Hao, Fangpeng Chen, Hanfeng Hu, Qi Liu, and Sai Ji. "A Secure System Framework for an Agricultural IoT Application." In International Conference on Computer Science and its Applications, pp. 332-341. Springer Singapore, 2016.
5. Campos, Isidro, Claudio Balbontín, Jose González-Piqueras, Maria P. González-Dugo, Christopher MU Neale, and Alfonso Calera. "Combining a water balance model with evapotranspiration measurements to estimate total available soil water in irrigated and rainfed vineyards." Agricultural Water Management 165 (2016): 141- 152.
6. Rajalakshmi, P., and S. Devi Mahalakshmi. "IOT based crop-field monitoring and irrigation automation." In Intelligent Systems and Control (ISCO), 2016 10th International Conference on, pp. 1-6. IEEE, 2016.
7. Roussel, N., F. Frappart, G. Ramillien, J. Darrozes, F. Baup, and C. Ha. "Detection of soil moisture content changes by using a single geodetic antenna: The case of an agricultural plot." In 2015 IEEE, International Geoscience and Remote Sensing Symposium (IGARSS), pp. 2008-2011. IEEE, 2015
8. Abrisqueta, Isabel, Wenceslao Conejero, Mercedes Valdés-Vela, Juan Vera, Ma Fernanda Ortuño, and Ma Carmen Ruiz-Sánchez. "Stem water potential estimation of drip-irrigated

early-maturing peach trees under Mediterranean conditions." Computers and Electronics in Agriculture 114 (2015): 7-13.

9. Giusti, Elisabetta, and Stefano Marsili-Libelli. "A Fuzzy Decision Support System for irrigation and water conservation in agriculture." Environmental Modelling & Software 63 (2015): 73-86.
10. Soulis, Konstantinos X., Stamatios Elmaloglou, and Nicholas Dercas. "Investigating the effects of soil moisture sensors positioning and accuracy on soil moisture based drip irrigation scheduling systems." Agricultural Water Management 148 (2015): 258-268
11. Navarro-Hellín, H., R. Torres-Sánchez, F. SotoValles, C. AlbaladejoPérez, J. A. López-Riquelme, and R. Domingo-Miguel. "A wireless sensors architecture for efficient irrigation water management." Agricultural Water Management 151 (2015): 64-74.
12. Ruiz-Canales, A., and M. Ferrández-Villena. "New proposals in the automation and remote control of water management in agriculture: Agromotic systems." Agricultural Water Management 151 (2015): 1- 3.
13. Upadhyay, Priyanka, Gurpreet Matharu, and Naveen Garg. "Modeling agility in Internet of Things (IoT) architecture." In Information Systems Design and Intelligent Applications, pp. 779-786. Springer India, 2015.
14. Guo, Tiantian, and Weizhu Zhong. "Design and implementation of the span greenhouse agriculture Internet of Things system." In Fluid Power and Mechatronics (FPM), 2015 International Conference on, pp. 398-401. IEEE, 2015.
15. Hari Ram, V. Vijay, H. Vishal, S. Dhanalakshmi, and P. Meenakshi Vidya. "Regulation of water in agriculture field using Internet Of Things." In Technological Innovation in ICT for Agriculture and Rural Development (TIAR), 2015 IEEE, pp. 112-115. IEEE, 2015.
16. Alnaimi, Firas B. Ismail, Yee Chaw Chu, and Khairul Salleh Mohamed Sahari. "Hybrid renewable power system for agriculture irrigation system." In System Integration (SII), 2014 IEEE/SICE International Symposium on, pp. 736-742. IEEE, 2014.
17. Bing, Fu. "Research on the agriculture intelligent system based on IOT." In 2012 International Conference on Image Analysis and Signal Processing, pp. 1-4. IEEE, 2012.
18. Li, S., 2012, August. Application of the Internet of Things Technology in Precision Agriculture Irrigation Systems. In Computer Science & Service System (CSSS), 2012 International Conference on (pp. 1009-1013). IEEE.

19. Zhou, Hong, BingWu Liu, and PingPing Dong. "The Technology System Framework of the Internet of Things and Its Application Research in Agriculture." In International Conference on Computer and Computing Technologies in Agriculture, pp. 293-300. Springer Berlin Heidelberg, 2011
20. Maton, L., D. Leenhardt, M. Goulard, and J-E. Bergez. "Assessing the irrigation strategies over a wide geographical area from structural data about farming systems." Agricultural systems 86, no. 3 (2005): 293- 311.
21. 21. Zwart, Sander J., and Wim GM Bastiaanssen. "Review of measured crop water productivity valuesfor irrigated wheat, rice, cotton and maize." Agricultural Water Management 69, no. 2 (2004): 115-133.

APPENDIX

SOFTWARE CODE

The code that should be developed is given below:

```
#include "DHT.h"
#define DHTPIN 8
#define DHTTYPE DHT11
#include<LiquidCrystal.h>
#include<SoftwareSerial.h>
SoftwareSerial ard(9,10);
LiquidCrystal LCD (A0,A1,A2,A3,A4,A5);
DHT dht(DHTPIN, DHTTYPE);
int sensor_pin=2;
int relay=3;
String str;
void setup() {
pinMode(sensor_pin,INPUT);
pinMode(relay,OUTPUT);
```

```
Serial.begin(9600);
ard.begin(115200);
digitalWrite(relay,HIGH);
// Serial.println(F("DHTxx test!"));
dht.begin();
 LCD.begin(16,2);
 LCD.clear();
 LCD.setCursor(0,0);
 LCD.print("IRRIGATION");
 LCD.setCursor(0,1);
 LCD.print("SYSTEM");
 delay(2000);
 }
 void loop()
 {
  float h = dht.readHumidity();
  float t = dht.readTemperature();
  if (isnan(h) || isnan(t)) {
  // Serial.println(F("Failed to read from DHT sensor!"));
  return;
  }
  // Serial.println(F("Humidity: "));
 // Serial.println(h);
 // Serial.println(F("Temperature: "));
 // Serial.println(t);
 int sensor_data=digitalRead(sensor_pin);
 // Serial.println(sensor_data);
```

```
LCD.clear();
LCD.setCursor(0,0);
LCD.print("T:");
LCD.setCursor(2,0);
LCD.print(t);
LCD.setCursor(8,0);
LCD.print("M:");
LCD.setCursor(10,0);
LCD.print(sensor_data);
 LCD.setCursor(12,0);
 LCD.print("H:");
 LCD.setCursor(14,0);
 LCD.print(h);
 delay(2000);
 str=String(sensor_data)+String("!")+String(t)+String("@")+String(h);
 ard.println(str);
 if(t>34&&sensor_data==1)
 {
 digitalWrite(relay,LOW);
 // LCD.clear();
 LCD.setCursor(0,1);
 LCD.print("Motor ON");
 Serial.println('B');
 delay(5000);
 digitalWrite(relay,HIGH);
 // LCD.clear();
 LCD.setCursor(0,1);
 LCD.print("Motor OFF");
 delay(1000);
 }
 else if(t<=34&&sensor_data==1)
 {
 digitalWrite(relay,LOW);
 // LCD.clear();
 LCD.setCursor(0,1);
 LCD.print("Motor ON");
 Serial.println('A');
 delay(3000);
 digitalWrite(relay,HIGH);
 // LCD.clear();
 LCD.setCursor(0,1);
 LCD.print("Motor OFF");
 delay(1000);
 }
 else if(t>34&&sensor_data==0)
 {
 digitalWrite(relay,LOW);
 // LCD.clear();
 LCD.setCursor(0,1);
 LCD.print("Motor ON");
 delay(5000);
 digitalWrite(relay,HIGH);
 // LCD.clear();
 LCD.setCursor(0,1);
 LCD.print("Motor OFF");
 delay(1000);
 }
 else if(t<=34&&sensor_data==0)
 {
 digitalWrite(relay,HIGH);
 // LCD.clear();
 LCD.setCursor(0,1);
 LCD.print("Motor OFF");
 delay(1000);
 }
 }
```