CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

The continuous growth of world population, environmental change, industrialization, the arable land over the globe is decreasing every year. Therefore, the demand and requirement for hygienic food and crop yield have been growing continuously. A survey conducted by the United Nations of Food and Agriculture Organization (FAO) estimated that the population of the world is expected to reach 9.73 billion in 2050. It is expected that more cropland and water will need to meet the future food demands globally. Furthermore, other challenges such as abrupt changes in climate, lack of labor, and water scarcity spiral the pressure on agriculturists and farmers. The challenges faced by traditional agriculture and greenhouse farming need a fundamental change to develop sustainable and ecological food. Greenhouse farming is one of the best alternatives to overcome the food crisis as well as to ensure the sustainability of socio-ecological.

1.1 Basic Topics of Project

The greenhouse is a structure like a house that is covered with a plastic material or glass mainly designed to cultivate multiple crops in any season. In order to increase the quantity and quality of food, a greenhouse can regulate the growing patterns of plants accordingly. Traditional greenhouses are specifically designed with dry lands and ignore the multiple environmental variables such as temperature, humidity, among others. Usually, a smart and productive greenhouse requires environmental monitoring and control devices, to manage various weather parameters.

The IoT-enabled greenhouse farming trends. The deployment of a wireless sensor network (WSN) enables climatic data to be gathered and distributed for understanding and monitoring the internal system of the greenhouse, but there is a need to handle several other features which can cause significant destruction to the grown plants, such as, less or high-water supply, bad weather conditions, and light intensity. The eradication of these issues can be done through a precise system that helps to automate the monitoring and control methods for a smart greenhouse.

1.2 Problem Definition

- Prior greenhouse systems were not using microcontroller, The systems were not automatic. User checks the conditions and takes proper action.
- After sometime the greenhouse systems is automatic, system will work for the user and takes proper action to maintain climatic conditions. monitoring was done by live presence of user.
- The monitoring was made easy and user was able to monitor the greenhouse from his home. Here the distance from the system was limited.
- Manufacturers use Bluetooth app or system specific screen having data of green house climatic conditions. Here also the monitoring becomes bounded.

1.3 Scope of the Project

The scope of this project are as follows:

- The IoT offers an automated greenhouse environment that enables a significant association among tangible things and people.
- A user is allowed to gather real-time data processing, analysis, and meditation.
- The IoT enabled greenhouse farming to enhance productivity and decrease labor costs.
- A farmer can grow multiple crops in the appropriate season with less human effort.
- The efficient use of water and proper soil monitoring results in high crop production.
- The deployed system collects and transfers the sensed information via the IoT cloud for easy access from anywhere and at any time.

1.4 Motivation

- The purpose of a green house is to shield crops from excess clod and heat and unwanted pests. A Greenhouse makes it possible to grow certain types of crops year round.
- Within the framework of sustainable development the energy supply system is a crucial topic. Spurred on by the Kyoto Protocol the attention for the definition of energy saving programs.
- Promotion of environment-friendly technologies as renewable energy is increasing.
- Context the combination of a further decrease of emissions (CO_2, NO_X) with a faze-out.

- At the international level hydrogen as an energy carrier is one of the major topics in the discussion on pathways towards a future sustainable energy society.
- It is expected to contribute both to the reduction of air pollution, CO2-reduction and energy supply stability. The reason for this is that hydrogen offers a long term potential for energy systems with almost zero-emission level; it can be based on local and renewable energy sources.
- In the USA a 'Roadmap to hydrogen' has been developed already and on the European level recently the policy paper 'Hydrogen Energy and Fuel Cells, a vision of our future' has been presented.
- Within the Belgian energy policy the knowledge on hydrogen is rather limited and this project intends to be the first step in a scientific assessment of hydrogen in the Belgian context.

1.5 Objectives

A greenhouse is an important part of the agriculture and horticulture sectors in our country.

- It is used for growing plants faster at any season whether the climate is favorable or not.
- Automatic monitoring and controlling of the climatic parameters will directly or indirectly govern the plant growth and hence their production.
- The Automation will decrease the labor dependency and increase the profit and efficiency.
- Main aim of this project is to help farmers to grow the crops even in adverse environmental conditions like in overheat and less humidity conditions.

1.6 Organization of the Report

The report is organized into chapters as follows:

Chapter 1 - Introduction: This chapter presents a brief description about Greenhouse Monitoring Using wireless sensors network.

Chapter 2 - Review of literature: In this chapter, the works carried out by various researchers are as follow.

Chapter 3 - System requirement specification: The chapter 3 presents the specific requirement, software and hardware requirements interfaces used. It also presents a brief summary about the chapter.

Chapter 4 – System analysis: This chapter describes the existing and proposed system of greenhouse monitoring using wireless sensor network.

Chapter 5 – System Design: This chapter describes the system architecture, data flow diagram, sequence diagram and use case diagram of greenhouse monitoring using wireless sensor network.

Chapter 6 – Implementation: Chapter 6 describes the implementation of greenhouse monitoring using wireless sensor network it consist of language selection, platform selection, implemented modules and pseudocode.

Chapter 7 – Testing: This chapter describes the testing types and test cases which are made in greenhouse monitoring using wireless sensor network.

Chapter 8 – Conclusion and future enhancements: This chapter describes the conclusion and future enhancements which are may be implemented to greenhouse monitoring using wireless sensor network.

Chapter 2

LITERATURE SURVEY

1.1 Introduction

A Literature review is an objective, critical summary of reported research works relevant to a topic under consideration for research. Its purpose is to create familiarity with current thinking and research on particular topic and may justify future research into previously overlooked or under studied area.

1.2 Related Work

[1] An Intelligent IoT-Based System Design for Controlling and Monitoring Greenhouse Temperature. Ahmad F Subahi1 , Kheir Eddine Bouazza1.

Improving agricultural production can only be achieved using innovative environmentally suitable solutions and modern agricultural technologies. Using Internet of Things (IoT) technologies in greenhouse farming allows reduction of the immediate impact of external climatic conditions. In this paper, a highly scalable intelligent system controlling, and monitoring greenhouse temperature using IoT technologies is introduced. The first objective of this system is to monitor the greenhouse environment and control the internal temperature to reduce consumed energy while maintaining good conditions that improve productivity.

[2] IoT and machine learning based approach for fully automated greenhouse. H. Jaiswal, K. P. Radha, R. Singuluri, and S. A. Sampson.

With the rapid evolution of technology, automation has taken over almost all fields of operation. The change in human-computer interaction has accelerated over the years. Greenhouses have come a long way in terms of technological advances. For 100% yields, it is essential to constantly monitor the optimal parameters for plant growth. Here in this work, different parameters that impact the yield of crops like humidity, CO_2 levels, light intensity, soil moisture, temperature are being monitored, controlled and coordinated using Raspberry Pi and Arduino. Internet of Things has enabled real-time data collection from the Smart Greenhouse and visualization on ThingSpeak platform. This paper proposes a fully automated greenhouse embedded with hydroponics and vertical farming and with excellent security provisions and surveillance to become a highly advanced and diverse version of currently prevailing models.

[3] Security and Privacy for Green IoT-Based Agriculture: Review, Blockchain Solutions, and Challenges. M. A. Ferrag, L. Shu, X. Yang, A. Derhab, and L. Maglaras

This paper presents research challenges on security and privacy issues in the field of green IoT-based agriculture. We start by describing a four-tier green IoT-based agriculture architecture and summarizing the existing surveys that deal with smart agriculture. Then, we provide a classification of threat models against green IoT-based agriculture into five categories, including, attacks against privacy, authentication, confidentiality, availability, and integrity properties. Moreover, we provide a taxonomy and a side-by-side comparison of the state-of-the-art methods toward secure and privacy-preserving technologies for IoT applications and how they will be adapted for green IoT-based agriculture.

[4] Internet-of-Things (IoT)-Based Smart Agriculture: Toward Making the Fields Talk. M. Ayaz, M. Ammad-Uddin, Z. Sharif, A. Mansour, and E. M. Aggoune

The rapid emergence of the Internet-of-Things (IoT) based technologies redesigned almost every industry including ‘‘smart agriculture’’ which moved the industry from statistical to quantitative approaches. Such revolutionary changes are shaking the existing agriculture methods and creating new opportunities along a range of challenges. This article highlights the potential of wireless sensors and IoT in agriculture, as well as the challenges expected to be faced when integrating this technology with the traditional farming practices. IoT devices and communication techniques associated with wireless sensors encountered in agriculture applications are analyzed in detail.

[5] A New IoT-based Platform for Greenhouse Crop Production M. Munoz, J.L. Guzman, J.A. Sanchez, F. Rodriguez, M. Torres and M. Berenguel

This work proposes a cloud solution to build an Internet of Things (IoT) platform applied in a greenhouse crop production context. Real-time and historical data, as well as prediction models, can be accessed by RESTful (Representational State Transfer) web services developed for such a purpose. Forecasting is also provided following a Greenhouse Models as a Service (GMaaS) approach. Traditionally, such models are hardcoded in applications or are embedded in software tools to be used as Decision Support Systems

(DSS). In addition, the proposed platform allows users to register new IoT devices and their greenhouse data in the FIWARE platform, providing a cloud scale solution for the case study. RESTful services of the proposed platform are also used by a web application allowing users to interact easily with the system.

[6] Development of an Intelligent LED Lighting Control Testbed for IoT-based Smart Greenhouses. J. Jiang and M. Moallem.

The aim of our study is to develop an intelligent control system for mixing color ratios using LED lights in a greenhouse environment. To this end, different components of an experimental testbed is presented for achieving the desired light requirements for plant growth in a greenhouse environment. The proposed testbed provides a easy-to-use plant growth system with IoT-enabled control and monitoring features. To testify the features mentioned above, a feedback lighting control method to achieve a desired photosynthetic photon flux desnity (PPFD) set point is implemented. A two-week experiment was conducted on microgreen kale which was planted in the testbed and harvested at the end of the experiment. The experimental results has shown that the tested microgreen kale grew with a healthy condition in the proposed testbed and lighting environment.

[7] Design of a Novel Remote Monitoring System for Smart Greenhouses Using the Internet of Things and Deep Convolutional Neural Networks. Adel Mellit, Mohamed Benghanem, Omar Herrak and Abdelaziz Messalaoui

To support farmers and improve the quality of crops production, designing of smart greenhouses is becoming indispensable. In this paper, a novel prototype for remote monitoring of a greenhouse is designed. The prototype allows creating an adequate artificial environment inside the greenhouse (e.g., water irrigation, ventilation, light intensity, and CO2 concentration). An Android mobile application was also developed using an A6 GSM module for notifying farmers (e.g., sending a warning message in case of any anomaly) regarding the state of the plants. A low-cost camera was used to collect and send images of the plants via the webpage for possible diseases identification and classification. In this context, a deep learning convolutional neural network was developed and implemented into a Raspberry Pi 4. To supply the prototype, a small-scale photovoltaic system was built.

[8] Keeping data at the edge of smart irrigation networks: A case study in strawberry greenhouses Constantinos Marios Angelopoulos, Gabriel Filios, Sotiris Nikoletseas, Theofanis P. Raptis

Cloud-based approaches for smart irrigation have been widely used in the recent years. However, the network traffic, security and regulatory challenges, which come hand in hand with sharing the crop data with third parties outside the edge of the network, lead strawberry farmers and data owners to rely on global clouds and potentially lose control over their data, which are usually transferred to third party data centers. In this paper, we follow a three-step methodological approach in order to design, implement and validate a solution for smart strawberry irrigation in greenhouses.

[9] IoT-based adaptive network mechanism for reliable smart farm system Muhammad Rusyadi Ramlia , Philip Tobianto Daelyb , Dong-Seong Kima , Jae Min Leea

This paper presents an adaptive network mechanism for a smart farm system by using LoRaWAN and IEEE 802.11ac protocols. Generally, the internet of things (IoT) system for agriculture application is used in an environment where significant interferences can occur. These interferences can disrupt the network performance of the system. In this paper, an adaptive network mechanism is designed to improve the network performance of the system, in order to achieve a more reliable smart farm system. Specifically, the proposed adaptive network mechanism is implemented in the application layer. The system has the ability to adjust a protocol based on the network condition. For instance, the IEEE 802.11ac is suitable for transmitting data which require high data rate such as image or video.

[10] Environment Monitoring of Rose Crops Greenhouse Based on Autonomous Vehicles with a WSN and Data Analysis Paul D. Rosero-Montalvo, Vanessa C. Erazo-Chamorro, Vivian F. López-Batista, María N. Moreno-García and Diego H. Peluffo-Ordóñez

The main objective is to improve the quality of the crops while regulating the production time. To this end, a system consisting of autonomous quadruped vehicles connected with a wireless sensor network (WSN) is developed, which supports the decision-making on type of action to be carried out in a greenhouse to maintain the appropriate environmental conditions for rose cultivation. A data analysis process was carried out, aimed at designing an in-situ intelligent system able to make proper decisions regarding the cultivation.

Chapter 3

SYSTEM REQUIREMENT SPECIATION

3.1 Introduction

System requirement specifications gathered by extracting the appropriate information to implement the system. It is the elaborative conditions which the system need to attain. Moreover, the SRS delivers a complete knowledge of the system to understand what this project is going to achieve without any constraints on how to achieve this goal. This SRS not providing the information to outside characters but it hides the plan and gives little implementation details.

3.2 Functional requirements

- System should automatically read temperature.
- System should automatically read Humidity.
- System should automatically read moisture level and lighting condition.

3.3 Non-functional requirements

- **Usability:** Easy Interface for user
- **Reliability:** Theft Avoidance
- **Performance:** Should not take excessive time in taking measure
- **Supportability:** Contain easy to understand code with provisions for future enhancement.

3.4 Hardware Requirements

- Microcontroller
- Wifi
- DC Motors
- Moisture Sensor
- DHT11 Sensor

- Power Supply
- LDR
- LCD
- Relay

3.5 Software Requirements

- Embedded C
- Arduino

3.6 Preliminary Requirements

- **Sensor Selection:** Choose appropriate sensors to measure essential parameters within the greenhouse, such as temperature, humidity, light intensity, soil moisture and possibly other parameters specific to your crop or plant requirements.
- **Wireless Sensor Network Infrastructure:** Design and deploy a WSN infrastructure within the greenhouse. Determine the number of sensor nodes required based on the size of the greenhouse and the granularity of monitoring desired. Plan the placement of sensor nodes to ensure adequate coverage and minimize signal interference.
- **Power Supply:** Consider the power requirements of the sensor nodes. You can choose between battery-powered nodes or nodes that utilize alternative energy sources like solar power. Ensure that the power supply is reliable and can support the continuous operation of the sensor nodes.
- **Alerting and Control Mechanisms:** Implement a mechanism to generate alerts or notifications based on predefined thresholds or conditions. This could involve sending notifications to relevant personnel through Telegram. Consider integrating control mechanisms to automate certain processes, such as adjusting temperature or irrigation systems based on sensor readings.
- **Communication Protocols:** Select suitable communication protocols for data transmission within the WSN. Common protocols used in WSNs include Zigbee, Wi-Fi. The choice of protocol depends on factors such as range, power consumption, and data rate requirements.

3.7 System Environment

- **Sensor Nodes:** Deploy sensor nodes throughout the greenhouse to measure various environmental parameters such as temperature, humidity, light intensity and soil moisture. Select sensors based on the specific requirements of your greenhouse and crops.

- **Gateway:** Install a gateway device that acts as a bridge between the sensor nodes and the central monitoring system. The gateway collects data from the sensor nodes and relays it to the central server.
- **Communication Infrastructure:** Set up a wireless communication infrastructure to enable data transmission between the sensor nodes and the gateway. The choice of communication protocol will depend on factors like range, power consumption, and data rate requirements. Common protocols used in WSNs include Zigbee and Wi-Fi.
- **Integration and Scalability:** Consider the ability to integrate the greenhouse monitoring system with other existing greenhouse management systems or external applications. Also, design the system to be scalable, allowing for the addition of more sensor nodes or the expansion of the network as needed.

Chapter 4

SYSTEM ANALYSIS

4.1 Existing System

- Greenhouse monitoring is not automatic.
- It requires labor works.
- Mobile communication was used.
- Only some humidity levels were determined.
- The data were transmitted using Bluetooth communication protocol.

4.2 Proposed System

- Uses the WSN technology.
- More greenhouse monitoring parameters.
- An application was created to monitoring the greenhouse.

Chapter 5

SYSTEM DESIGN

5.1 Introduction

System design is the process of defining and specifying the architecture, components, modules, interfaces, and interactions of a system to fulfill specific requirements. It involves making strategic decisions about how various elements of the system will work together to achieve desired functionality, performance, and reliability. The purpose of system design is to create a blueprint or plan that guides the development and implementation of the system. It considers both functional and non-functional requirements, such as performance, scalability, security, maintainability, and usability. The design phase bridges the gap between the requirements analysis and the actual implementation of the system.

5.2 Input Design

The design should consider the specific parameters and environmental factors that need to be monitored. Here are some considerations for input design in greenhouse monitoring using WSN:

- **Identify Parameters:** Determine the parameters you want to monitor in the greenhouse. This may include temperature, humidity, light intensity, soil moisture and other relevant factors based on your specific greenhouse requirements.
- **Sensor Selection:** Choose appropriate sensors for each parameter. Consider sensors that are accurate, reliable, and compatible with WSN communication protocols. Ensure that the selected sensors are suitable for greenhouse conditions, such as temperature and humidity variations.
- **Sensor Placement:** Determine the optimal locations to place the sensors within the greenhouse. Consider placing sensors in different zones or areas to capture variations in environmental conditions. For example, place temperature and humidity sensors at different heights to capture variations across the greenhouse's vertical axis.
- **Calibration and Maintenance:** Plan for sensor calibration and maintenance procedures. Regular calibration ensures accurate readings, while maintenance involves sensor cleaning, battery replacement, and troubleshooting.

- **Power Management:** Optimize power consumption of sensor nodes to prolong their battery life. Implement power management techniques such as duty cycling, sleep scheduling, or energy harvesting solutions to maximize energy efficiency.

5.3 Output Design

Output design in the context of greenhouse monitoring using WSN (Wireless Sensor Networks) focuses on presenting the collected data and information in a meaningful and actionable format for users. The goal is to provide clear insights and facilitate decision-making based on the monitored greenhouse conditions. Here are some considerations for output design in greenhouse monitoring:

- **Real-Time Updates:** Ensure that the output design allows for real-time or near-real-time updates of the monitored data. Users should be able to view the latest readings and changes as they occur, enabling prompt responses to any critical conditions or anomalies.
- **Alerts and Notifications:** Implement a mechanism to generate alerts and notifications based on predefined thresholds or abnormal conditions. This can be in the form of visual indicators, alarms, or email/SMS notifications, notifying users when certain parameters exceed or fall below specified limits. The alerts should be clear and actionable, guiding users to take appropriate actions.
- **Integration with External Systems:** If relevant, design the output system to integrate with other greenhouse management systems or external applications. This allows users to access greenhouse data alongside other relevant information, such as weather forecasts, crop management databases, or energy management systems, to make more informed decisions.
- **Customization and User-Friendly Interface:** Design the output interface to be user-friendly and customizable. Users should have the flexibility to select the parameters they want to monitor, adjust the timeframes or intervals for data display, and personalize the interface according to their preferences.

5.4 System Architecture

System Architecture design-identifies the overall hypermedia structure for the Application. Architecture design is tied to the goals establish for a WebApp, the content to be presented, the users who will visit, and the navigation philosophy that has been established. Content architecture, focuses on the manner in which content objects and structured for presentation and navigation. WebApp architecture, addresses the manner in which the application is structure to manage user interaction, handle internal processing tasks, effect navigation, and present content. Application architecture is defined within the context of the development environment in which the application is to be implemented.

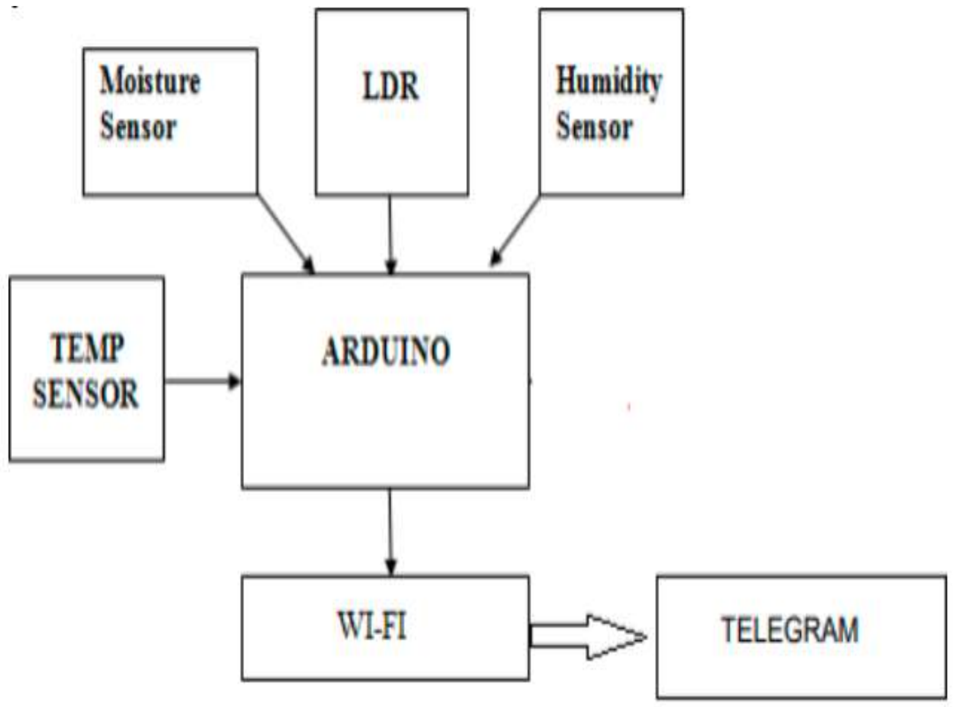


Figure 5.1: System Architecture

The above figure 5.1 is the system architecture of Greenhouse monitoring using wireless sensor network.

5.5 Data Flow Diagram

A data flow diagram (DFD) is graphic representation of the "flow" of data through an information system. A data flow diagram can also be used for the visualization of data processing (structured design). It is common practice for a designer to draw a context level DFD first which shows the interaction between the system and outside entities. DFD's show the flow of data from external entities into the system, how the data moves from one process to another, as well as its logical storage.

There are only four symbols:

- **Squares** representing external entities, which are sources and destinations of information entering and leaving the system.
- **Rounded rectangles** representing processes, in other methodologies, may be called 'Activities', 'Actions', 'Procedures', 'Subsystems' etc. which take data as input, do processing to it, and output it.
- **Arrows** representing the data flows, which can either, be electronic data or physical items. It is impossible for data to flow from data store to data store except via a process, and external entities are not allowed to access data stores directly.
- The **flat three-sided rectangle** is representing data stores should both receive information for storing and provide it for further processing.

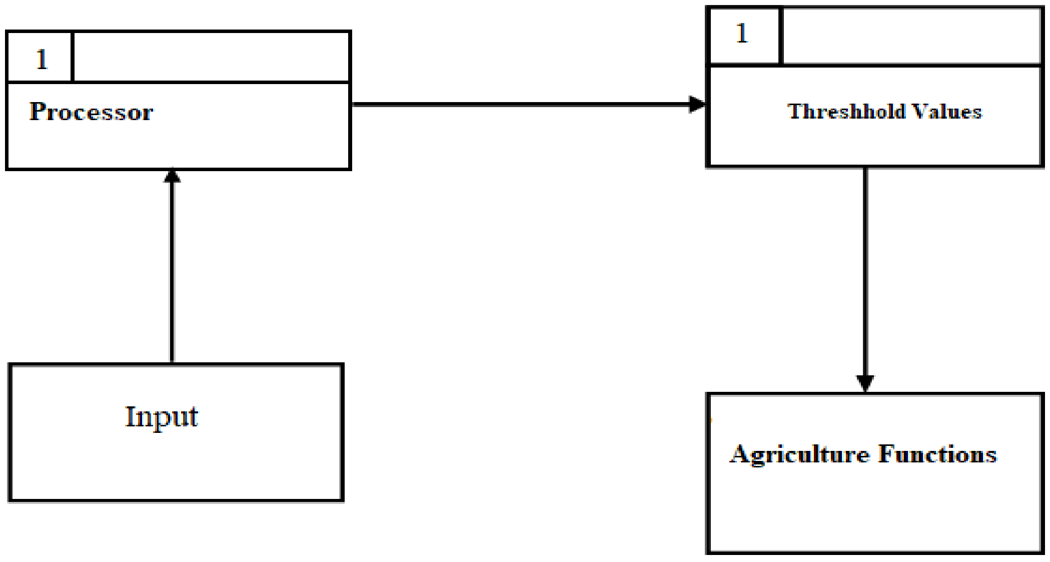


Figure 5.2: Dataflow diagram

5.6 User Case Diagram

A use case is a set of scenarios that describing an interaction between a source and a destination. A use case diagram displays the relationship among actors and use cases. The two main components of a use case diagram are use cases and actors. shows the use case diagram.

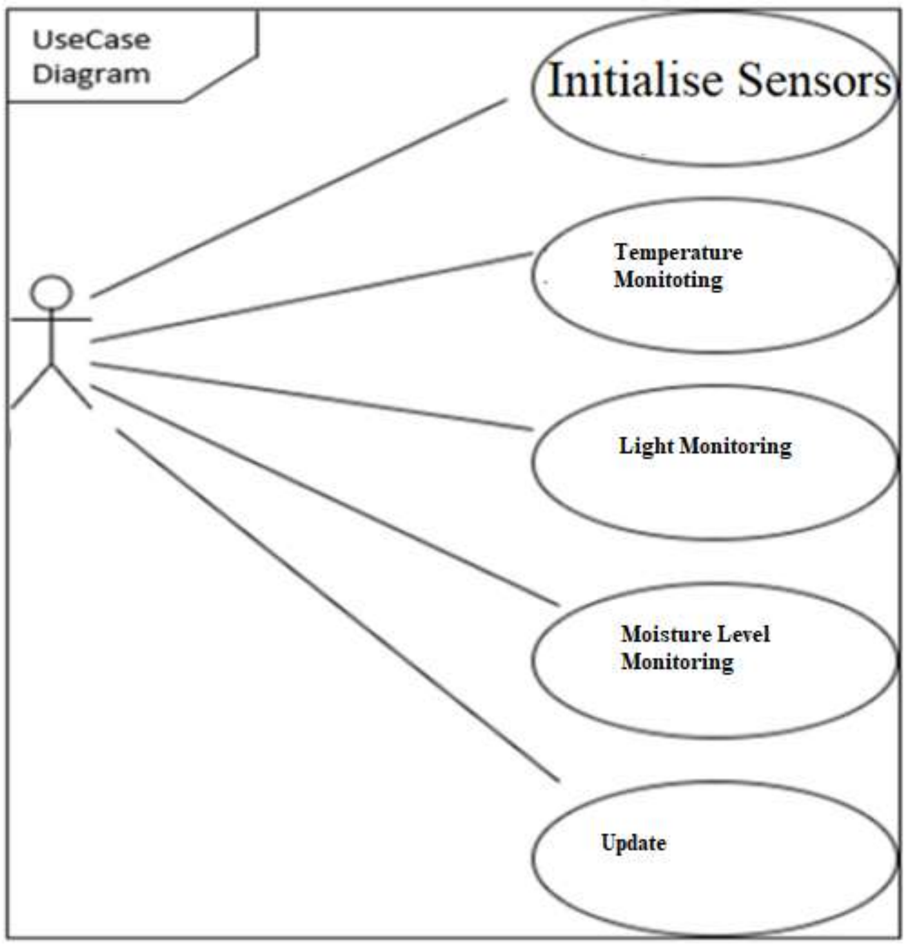


Figure 5.3: User case diagram

Chapter 6

IMPLEMENTATION

6.1 Introduction

Implementation is the process of converting a new system design into an operational one. It is the key stage in achieving a successful new system. It must therefore be carefully planned and controlled. The implementation of a system is done after the development effort is completed.

6.2 Language Selection

Embedded C

Having decided to use an 8051 process or as the basis of your embedded system, the next key decision that needs to be made is the choice of programming language. In order to identify a suitable language for embedded systems, we might begin by making the following observations:

- Computers (such as microcontroller, microprocessor or DSP chips) only accept instructions in ‘machine code’ (‘object codes’). Machine code is, by definition, in the language of the computer, rather than that of the programmer.

- All software, whether in assembly, C, C++, Java must ultimately be translated into machine code in order to be executed by the c computer.

- Embedded processors – like the 8051 – have limited processor power and very limited memory available: the language used must be efficient.

- To program embedded systems, we need low-level access to the hardware: this means, at least, being able to read from and write to particular memory locations.

- No software company remains in business for very long if it generates new code, from scratch, for every project. The language used must support the creation of flexible libraries, making it easy to re-use (well-tested) code components in a range of projects.

The language chosen should be in common use. This will ensure that you can continue to recruit experienced developers who have knowledge of the language. It will also mean that your existing developers will have access to sources of information which give examples of good design and programming practice. Even this short list immediately raises the paradox of programming language selection. From one point of view, only machine code is safe, since every other language involves a translator, and any code you create is only as safe as the code written by the manufacturers of the translator.

We can summarize C's features as follows:

- It is 'mid-level', with 'high level' features(such as support for functions and modules), and 'low- level' features (such as good access to hardware via pointers).
- It is very efficient.
- It is popular and well understood.
- Even desktop developers who have used only Java or C++ can soon understand C syntax.
- Good, well-proven compilers are available for every embedded processor (8-bit to 32-bit or more).
- Experienced staff is available.

Overall, C's strengths for embedded system development greatly outweigh its weaknesses. It may not be an ideal language for developing embedded systems, but it is unlikely that a 'perfect' language will ever be created.

6.3 Platform Selection

The **Arduino** integrated development environment (IDE) is a cross-platform application (for Microsoft Windows, macOS, and Linux) that is written in the Java programming language. It originated from the IDE for the languages Processing and Writng. It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, brace matching, and syntax highlighting, and provides simple one-click mechanisms to compile and upload programs to an Arduino board. It also contains a message area, a text console, a toolbar with buttons for common functions and a hierarchy of operation menus.

The source code for the IDE is released under the GNU General Public License, version 2. The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU tool chain, also included with the IDE distribution.

The Arduino IDE employs the program Arduino to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

6.4 Implemented Modules

Arduino UNO

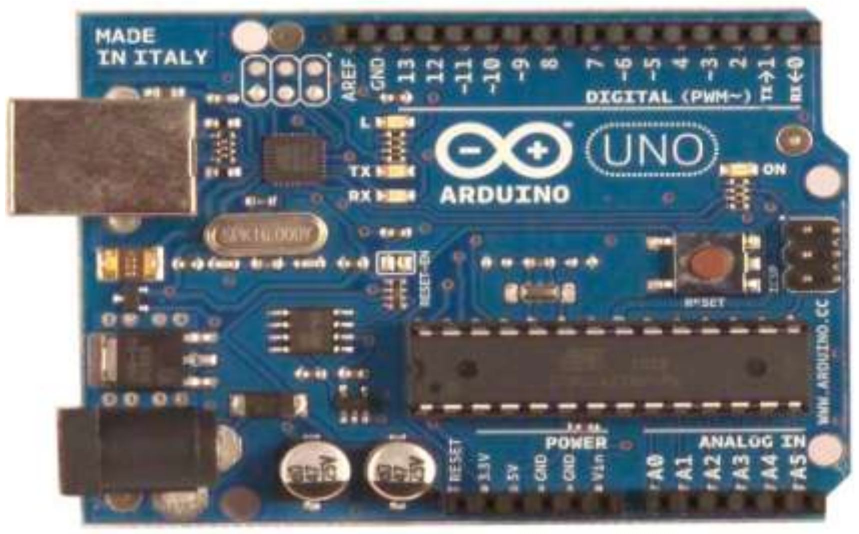


Figure 6.1 Arduino UNO

The Arduino Uno is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. 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. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to serial converter. "Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform.

Power

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC- to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center- positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector. The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5Vpin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board.

Liquid Crystal Display

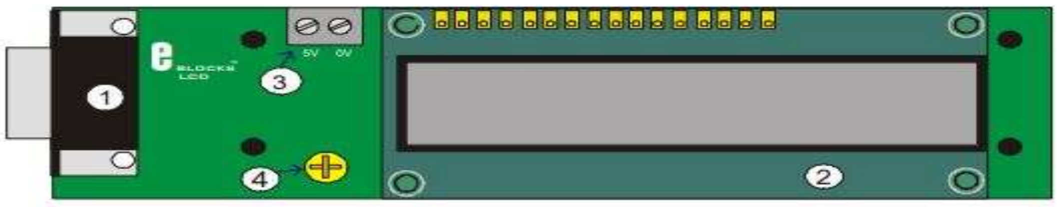


Figure 6.2 Liquid crystal display

This is an LCD Display designed for E-blocks. It is a 16 character, 2-line alphanumeric LCD display connected to a single 9-way D-type connector. This allows the device to be connected to most E-Block I/O ports. The LCD display requires data in a serial format, which is detailed in the user guide below. The display also requires a 5V power supply. Please take care not to exceed 5V, as this will cause damage to the device. The 5V is best generated from the E-blocks Multi programmer or a 5V fixed regulated power supply. The potentiometer RV1 is a contrast control that should be used to adjust the contrast of the display for the environment it is being used in.

Moisture sensor

Soil moisture sensor measures the water content in soil. It uses the property of the electrical resistance of the soil. Here, it is used to sense the moisture in field and transfer it to microcontroller in order to take controlling action of switching water pump ON/OFF. Soil Moisture Sensor measure the volumetric water content in soil Since the direct gravimetric measurement of free soil moisture requires removing, drying, and weighing of a sample, soil moisture sensors measure the volumetric water content indirectly by using some other property of the soil, such as electrical resistance, dielectric constant, or interaction with neutrons, as a proxy for the moisture content.

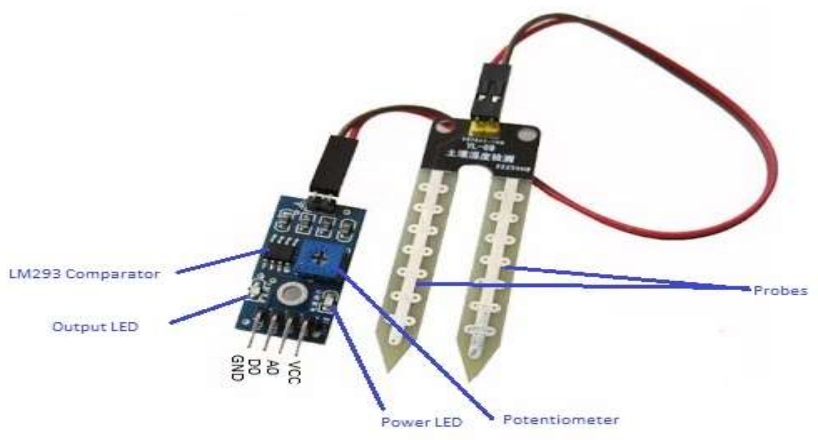


Figure 6.3 Moisture Sensor

DHT-11 Sensor

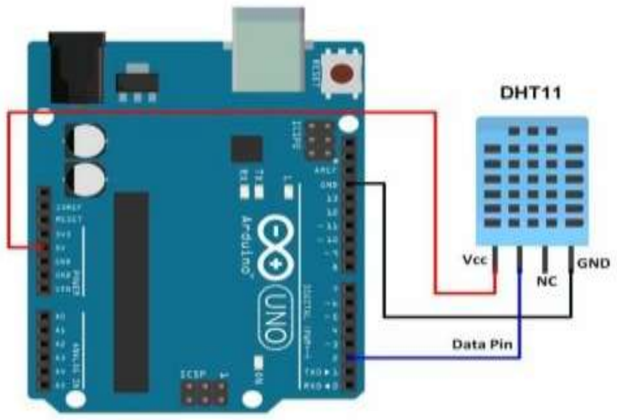


Figure 6.4 DHT-11 sensor with Arduino

DHT11 Temperature & Humidity Sensor features a temperature & humidity sensor complex with a calibrated digital signal output. By using the exclusive digital-signal-acquisition technique and temperature & humidity sensing technology, it ensures high reliability and excellent long-term stability. This sensor includes a resistive-type humidity measurement component and an NTC temperature measurement component, and connects to a high performance 8-bit microcontroller, offering excellent quality, fast response, anti-interference ability and cost-effectiveness.

Light Dependent Resistor

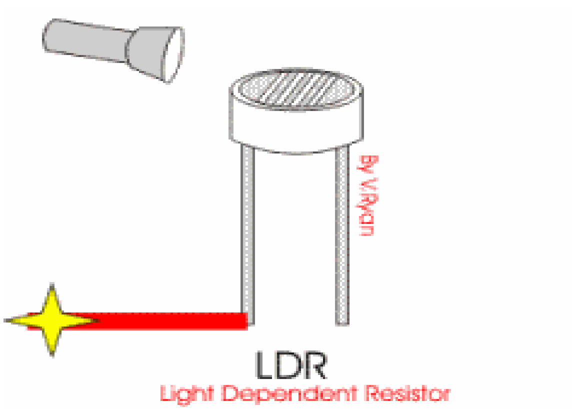


Figure 6.5 LDR Sensor

LDRs or Light Dependent Resistors are very useful especially in light/dark sensor circuits. Normally the resistance of an LDR is very high, sometimes as high as 1000 000 ohms, but when they are illuminated with light resistance drops dramatically. Electronic opto sensors are the devices that alter their electrical characteristics, in the presences of visible or invisible light. The best-known devices of this type are the light dependent resistor (LDR), the photo diode and the phototransistors.

Relay

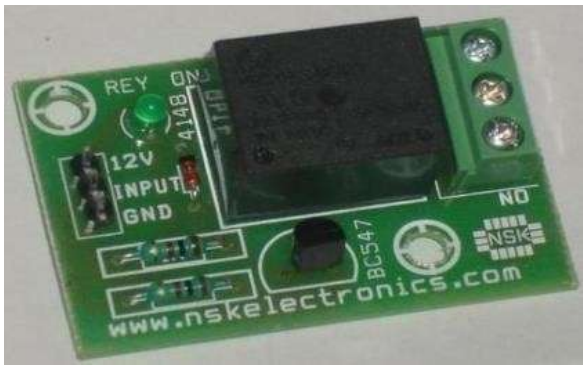


Figure 6.6: Relay module

A relay is an electrically operated switch. Current flowing through the coil of the relay creates amagnetic field which attracts a lever and changes the switch contacts. The coil current can be on or off so relays have two switch positions. Relays allow one circuit to switch a second circuit which can be completely separate from the first. For example, a low voltage battery circuit can use a relay to switch a 230V AC mains circuit. There is no electrical connection inside the relay between the two circuits. The link is magnetic and mechanical.

ESP 8266

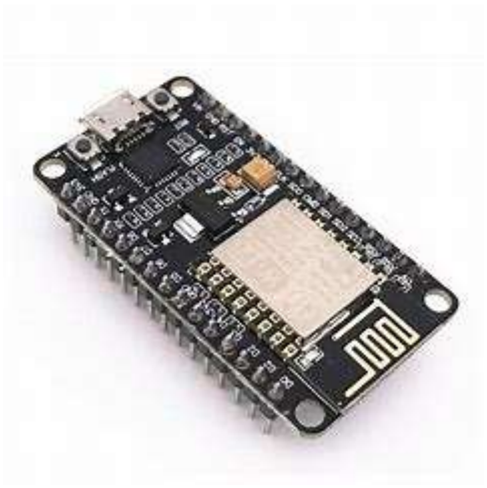


Figure 6.7:Wi-Fi Module

ESP8266 offers a complete and self-contained Wi-Fi networking solution, allowing it to either host the application or to offload all Wi-Fi networking functions from another application processor.When ESP8266 hosts the application, and when it is the only application processor in the device, it is able to boot up directly from an external flash. It has integrated cache to improve the performance of the system in such applications, and to minimize the memory requirements.

Alternately, serving as a Wi-Fi adapter, wireless internet access can be added to any microcontroller-based design with simple connectivity through UART interface or the CPU AHB bridge interface. ESP8266 on-board processing and storage capabilities allow it to be integrated with the sensors and other application specific devices through its GPIOs with minimal development up-front and minimal loading during runtime.

With its high degree of on-chip integration, which includes the antenna switch balun, power management converters, it requires minimal external circuitry, and the entire solution, including front-end module, is designed to occupy minimal PCB area.

Water Pump

Micro DC 3-6V Micro Submersible Pump Mini water pump For Fountain Garden

Mini water circulation System DIY project. This is a low cost, small size Submersible Pump Motor which can be operated from a 3 ~ 6V power supply. It can take up to 120 liters per hour with very low current consumption of 220mA. Just connect tube pipe to the motor outlet, submerge it in water and power it. Make sure that the water level is always higher than the motor. Dry run may damage the motor due to heating and it will also produce noise.

Figure 6.8: Water pump

Fan

Figure 6.9 :Fan

In Arduino and fan applications, you can sense the environment to control the fan speed. If the temperature reduces, you can also slowly reduce the fan speed to reduce the power. A few fans are simple ON/OFF fans. They are more affordable and easy to use with simple circuitry.

6.5 Pseudocode

```
#include <LiquidCrystal.h>
#include<dht.h>
```

```
#define dht_dpin 3
LiquidCrystal lcd(13, 12, 8, 9, 10, 11);
dht DHT;
int relay=2;
int relay1=16;
int relay2=4;
int moistval;
void setup()
{  pinMode(relay,OUTPUT);
   pinMode(relay1,OUTPUT);
   pinMode(relay2,OUTPUT);
   Serial.begin(9600);
   lcd.begin(16, 2);
   Serial.println("GREEN HOUSE MONOTORING....");
   lcd.clear();
   lcd.setCursor(0,0);
   lcd.print("GREEN HOUSE");
   lcd.setCursor(0,1);
   lcd.print(" MONOTORING...");
   delay(2000);
  digitalWrite(relay,LOW);
  digitalWrite(relay1,LOW);
  digitalWrite(relay2,LOW); }
void loop()
{
  MOISTURE_MONITOR();
  delay(1000);
  HUMIDITY_MEASUREMENT();
  delay(1000);
  LDR_check();
  delay(1000);
}
```

```
void HUMIDITY_MEASUREMENT(void)
{
  DHT.read11(dht_dpin);
  Serial.print("$Humidity: ");
  Serial.print(DHT.humidity);   // printing Humidity on LCD
  Serial.print(" %");
  Serial.print("#");
  delay(500);
  Serial.print("$Temperature:");
  Serial.print(DHT.temperature);   // Printing temperature on LCD
  Serial.print(" C");
  Serial.println("#");
 delay(500);
 lcd.clear();
 lcd.print("RH:");
 lcd.print(DHT.humidity);
 lcd.setCursor(0, 1);
 lcd.print("TEMP:");
 lcd.print(DHT.temperature);
 delay(500);
 if(DHT.temperature<=34)
 {
  digitalWrite(relay2,LOW);
  lcd.clear();
  lcd.print("NORMAL ");
  lcd.setCursor(0,1);
  lcd.print(" TEMPERATURE");
  Serial.println("$NORMAL TEMPERATURE#");
  delay(1000);
  Serial.print("$Temperature:");
  Serial.print(DHT.temperature);   // Printing temperature on LCD
  Serial.print(" C");
```

```
    Serial.println("#");
    delay(1000);
      }
  else if(DHT.temperature>=34)
    {
      lcd.clear();
      lcd.print("MORE ");
      lcd.setCursor(0,1);
      lcd.print(" TEMPERATURE");
      Serial.println("$MORE TEMPERATURE#");
      delay(1000);
      Serial.print("$Temperature:");
      Serial.print(DHT.temperature);   // Printing temperature on LCD
      Serial.print(" C");
      Serial.println("#");
      digitalWrite(relay2,HIGH);
      delay(1000); }}
void MOISTURE_MONITOR()
{moistval=analogRead(A1);
  Serial.println(moistval);
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("MOIST VALUE=");
  lcd.setCursor(0,1);
  lcd.print(moistval);
  delay(1000);
  if(moistval<800)
  { lcd.clear();
      lcd.print("SOIL IS WET");
      digitalWrite(relay,LOW);
      Serial.println("$SOIL IS WET#");
      delay(1000);
  }
  else{
    lcd.clear();
    lcd.print("SOIL IS DRY");
    digitalWrite(relay,HIGH);
    Serial.println("$SOIL IS DRY#");
    delay(1000);
  }
}
void LDR_check()
{
 int ldrval=analogRead(A0);
 Serial.println("ldrval="+String(ldrval));
 if(ldrval<800)
 {
 Serial.println("$NORMAL LIGHT#");
 digitalWrite(relay1,LOW);
 lcd.clear();
 lcd.print("NORMAL LIGHT");
 delay(1000);
}
else
 {
 Serial.println("$LOW LIGHT#");
 lcd.clear();
 lcd.print("LOW LIGHT");
 digitalWrite(relay1,HIGH);
 delay(1000);
}
}
```

Chapter 7

TESTING

7.1 Introduction

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. Testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

7.2 Types of Tests

- Unit Testing
- Integration Testing
- System Testing

7.2.1 Unit Testing

Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation. Unit testing is often automated but it can also be done manually. The goal of unit testing is to isolate each part of the program and show that individual parts are correct in terms of requirements and functionality.

7.2.2 Integration Testing

Integration testing is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing. Integration testing is defined as the testing of combined parts of an application to determine if they function correctly. It occurs after unit testing and before validation testing. Integration testing can be done in two ways: Bottom-up integration testing and Top-down integration testing.

7.2.3 System testing

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements.

System testing falls within the scope of black-box testing, and as such, should require no knowledge of the inner design of the code or logic.

7.3 Test Cases

Test cases are specific scenarios or conditions that are designed to validate the functionality, performance, or behavior of a system or software application. They are used to ensure that the software meets the desired requirements and performs as expected in different situations. Test cases typically consist of a set of inputs, expected results, and steps to execute.

Table 7.1 Unit Test Case 1

Test Case :	UTC-1
Name of Test:	Moisture Test
Items being tested:	Moisture Sensor and Motor Function
Sample Input:	Different Moistures Values
Expected output:	Motor should turn ON or OFF based on Moisture values
Actual output:	Motor Turned ON or OFF based on Moisture values
Remarks:	Pass

Table 7.2 Unit Test Case 2

Test Case :	UTC-2
Name of Test:	Temperature
Items being tested:	Temperature Sensor and Fan
Sample Input:	Different Temperature values
Expected output:	Fan should turn ON or OFF based on Temperature values
Actual output:	Fan turn ON or OFF based on Temperature values
Remarks:	Pass

Table 7.3 Unit Test Case 3

Test Case :	UTC-3
Name of Test:	Light Test
Items being tested:	LDR Sensor and LED Light
Sample Input:	Varying Light Source
Expected output:	LED should turn ON or OFF based on Light Source
Actual output:	LED turn ON or OFF based on given Light Source
Remarks:	Pass

Table 7.4 Integration Test Case 1

Test Case :	ITC-1
Name of Test:	Working of Processors with Sensors
Items being tested:	Processor and Sensors
Sample Input:	Power supply
Expected output:	Automatic Operation
Actual output:	The Processors and Sensors are work Automatic
Remarks:	Pass

Table 7.5 Integration Test Case 2

Test Case :	ITC-2
Name of Test:	Notification
Items being tested:	Wifi Module and Arduino
Sample Input:	Collected data from Sensors
Expected output:	User Should receive Notification on Telegram
Actual output:	The User received notification on Telegram
Remarks:	Pass

Table 7.6 System Test Case 1

Test Case :	STC-1
Name of Test:	System Testing
Items being tested:	All Modules
Sample Input:	Threshold Conditions
Expected output:	Should Operate Automatically
Actual output:	All Modules Operate Automatically based on Threshold
Remarks:	Pass

Chapter 8

CONCLUSION AND FUTURE ENHANCEMENTS

8.1 Conclusion

Setting up a greenhouse that uses the IoT monitoring and controlling system will minimize human error as well as labor cost and maximizing the production. India having the main occupation as agriculture and also the main aspect for its economy, this project will strengthen the farmers and the country financially. It is so affordable and convenient that anyone can use this system to grow the plants they want in any environmental condition. It also being eco-friendly has no damage to the environment.

8.2 Future Enhancements

This paper describes the design of a greenhouse monitoring system based on Cloud IoT. Agriculture projects even in urban areas are on a rise in recent times, in unique forms. Technological progress makes the agricultural sector grow high, which here is made by the Cloud IoT. The IoT will dramatically change the way we live our daily lives and what information is stored about us. This cloud computing is free to use anytime and anywhere as long as the computer is connected with the Internet. This monitoring system percepts different parameters inside the greenhouse using sensors, GSM, and cloud to provide the updates. The developed system can be proved profitable as it will optimize the resources in the greenhouse. The complete module is of low cost, low power operation hence, easily available to everyone. And Greenhouse monitoring using Wireless Sensor Networks has great potential for future enhancements.

Here are some possible areas of improvement and future enhancements for greenhouse monitoring using WSN:

- Integration with AI and Machine Learning.
- Use of Advanced Sensors.
- Cloud-based Monitoring and Control.
- Implementation of Energy Harvesting Techniques.

BIBLIOGRAPHY

[1] A. F. Subahi and K. E. Bouazza, ‘‘An intelligent IoT-based system design for controlling and monitoring greenhouse temperature,’’ IEEE Access, vol. 8, pp. 125488–125500, 2020.

[2] H. Jaiswal, K. P. Radha, R. Singuluri, and S. A. Sampson, ‘‘IoT and machine learning based approach for fully automated greenhouse,’’ in Proc. IEEE Bombay Sect. Signature Conf. (IBSSC), Jul. 2019, pp. 1–6.

[3] M. A. Ferrag, L. Shu, X. Yang, A. Derhab, and L. Maglaras, ‘‘Security and privacy for green IoT-based agriculture: Review, blockchain solutions, and challenges,’’ IEEE Access, vol. 8, pp. 32031–32053, 2020.

[4] M. Ayaz, M. Ammad-Uddin, Z. Sharif, A. Mansour, and E. M. Aggoune, ‘‘Internet-of-Things (IoT)-based smart agriculture: Toward making the fields talk,’’ IEEE Access, vol. 7, pp. 129551–129583, 2019.

[5] M. Muñoz, J. L. Guzman, J. A. Sánchez-Molina, F. Rodriguez, M. Torres, and M. Berenguel, ‘‘A new IoT-based platform for greenhouse crop production,’’ IEEE Internet Things J., vol. 9, no. 9, pp. 6325–6334, May 2022.

[6] J. Jiang and M. Moallem, ‘‘Development of an intelligent LED lighting control testbed for IoT-based smart greenhouses,’’ in Proc. 46th Annu. Conf. IEEE Ind. Electron. Soc. (IECON), Oct. 2020, pp. 5226–5231.

[7] A. Mellit, M. Benghanem, O. Herrak, and A. Messalaoui, ‘‘Design of a novel remote monitoring system for smart greenhouses using the Internet of Things and deep convolutional neural networks,’’ Energies, vol. 14, no. 16, p. 5045, Aug. 2021.

[8] C. M. Angelopoulos, G. Filios, S. Nikoletseas, and T. P. Raptis, ‘‘Keeping data at the edge of smart irrigation networks: A case study in strawberry greenhouses,’’ Comput. Netw., vol. 167, Feb. 2020, Art. no. 107039.

[9] M. R. Ramli, P. T. Daely, D.-S. Kim, and J. M. Lee, ‘‘IoT-based adaptive network mechanism for reliable smart farm system,’’ Comput. Electron. Agricult., vol. 170, Mar. 2020, Art. no. 105287.

[10] P. D. Rosero-Montalvo, V. C. Erazo-Chamorro, V. F. López-Batista, M. N. Moreno-García, and D. H. Peluffo-Ordóñez, ‘‘Environment monitoring of rose crops greenhouse based on autonomous vehicles with a WSN and data analysis,’’ Sensors, vol. 20, no. 20, p. 5905, Oct. 2020.

APPENDIX A – ACRONYMS

WSN – Wireless Sensor Network

FAO – Food and Agricultural Organization

SRS – System Requirements Specifications

GSM – Global System for Mobile communication

CPU – Central Processing Unit

IOT – Internet of Things

UTC – Unit Test case

ITC – Integration Test case

STC – System Test case

LCD – Liquid Crystal Display

DTH - Digital Temperature and Humidity Sensor

APPENDIX B – SNAPSHOTS

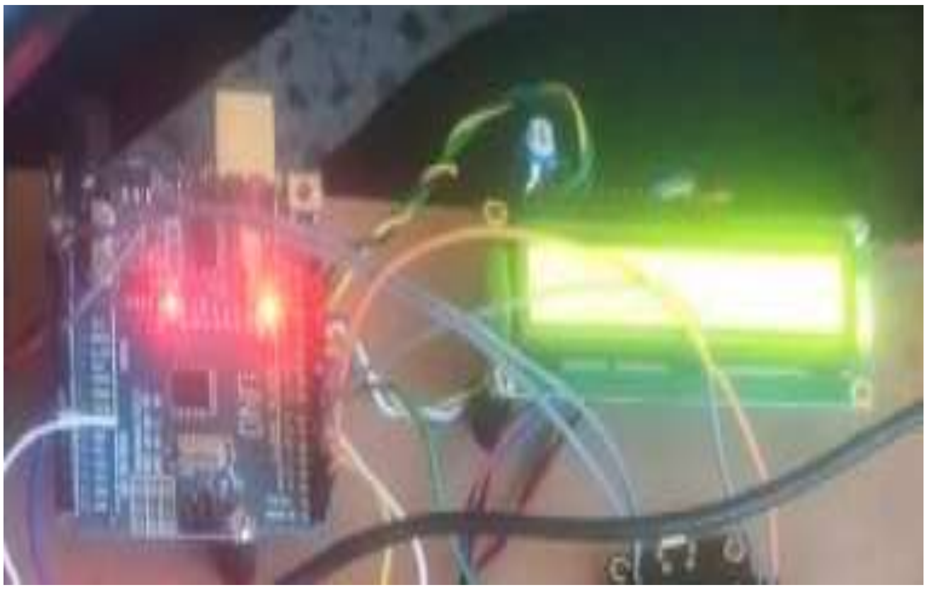


Figure B1: Arduino with LCD display

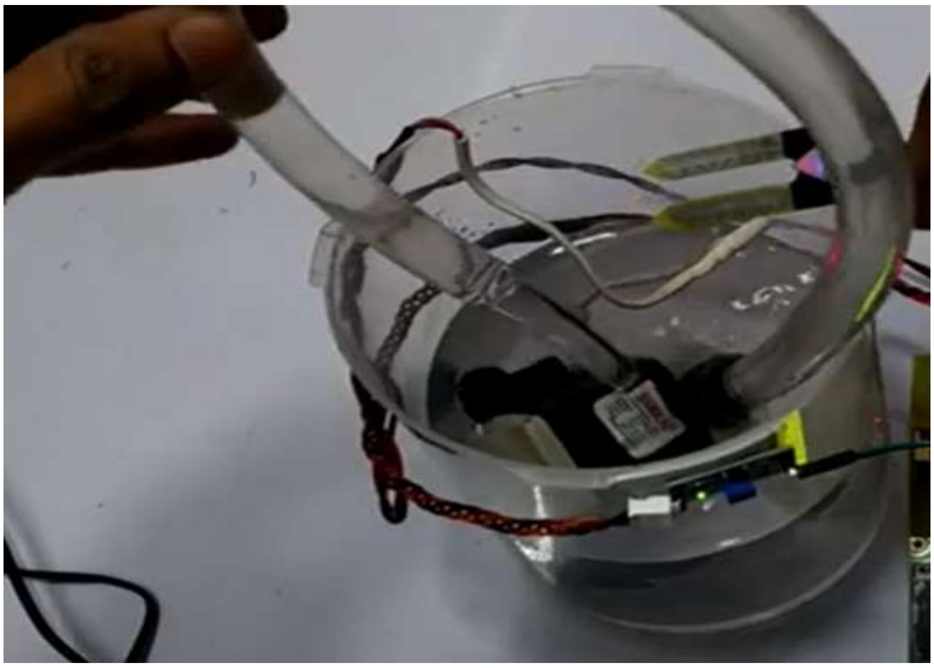


Figure B2: Working of Pump with Moisture Sensor

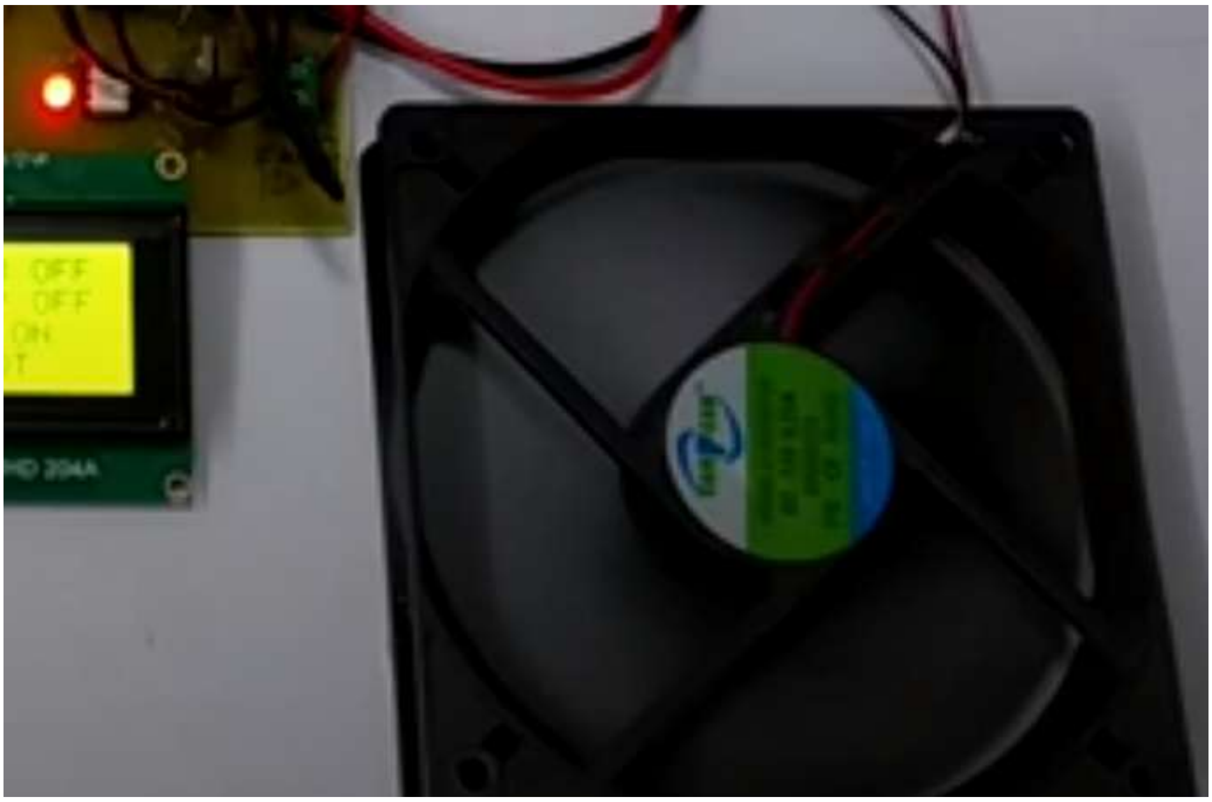


Figure B3: Working of Fan with Temperature Sensor

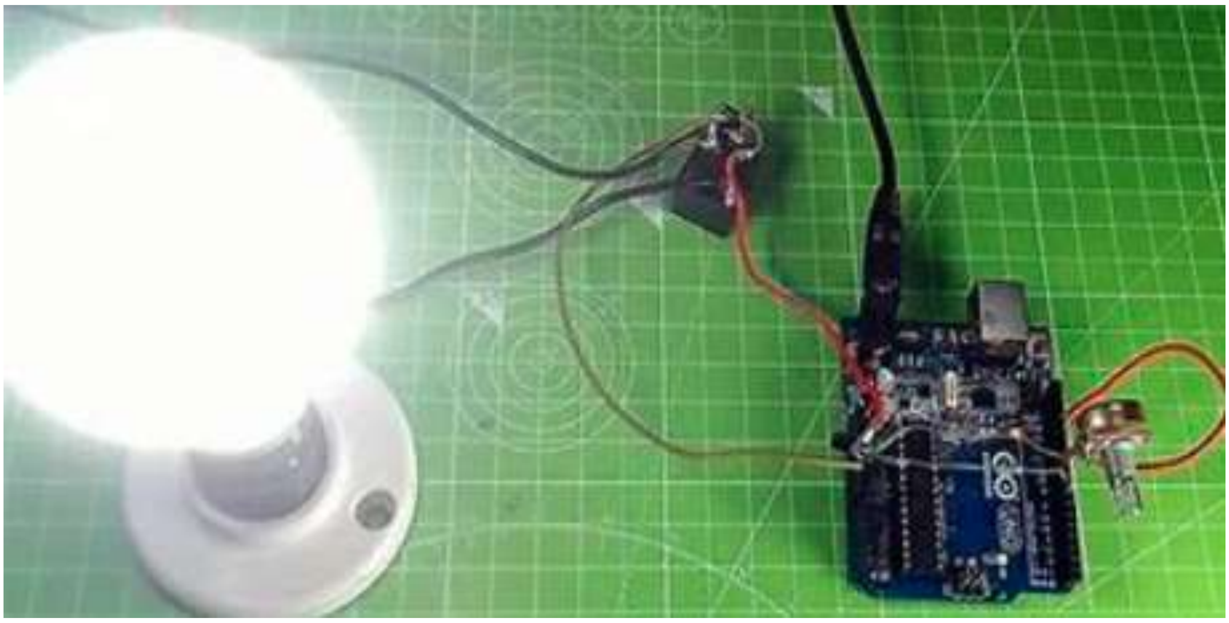
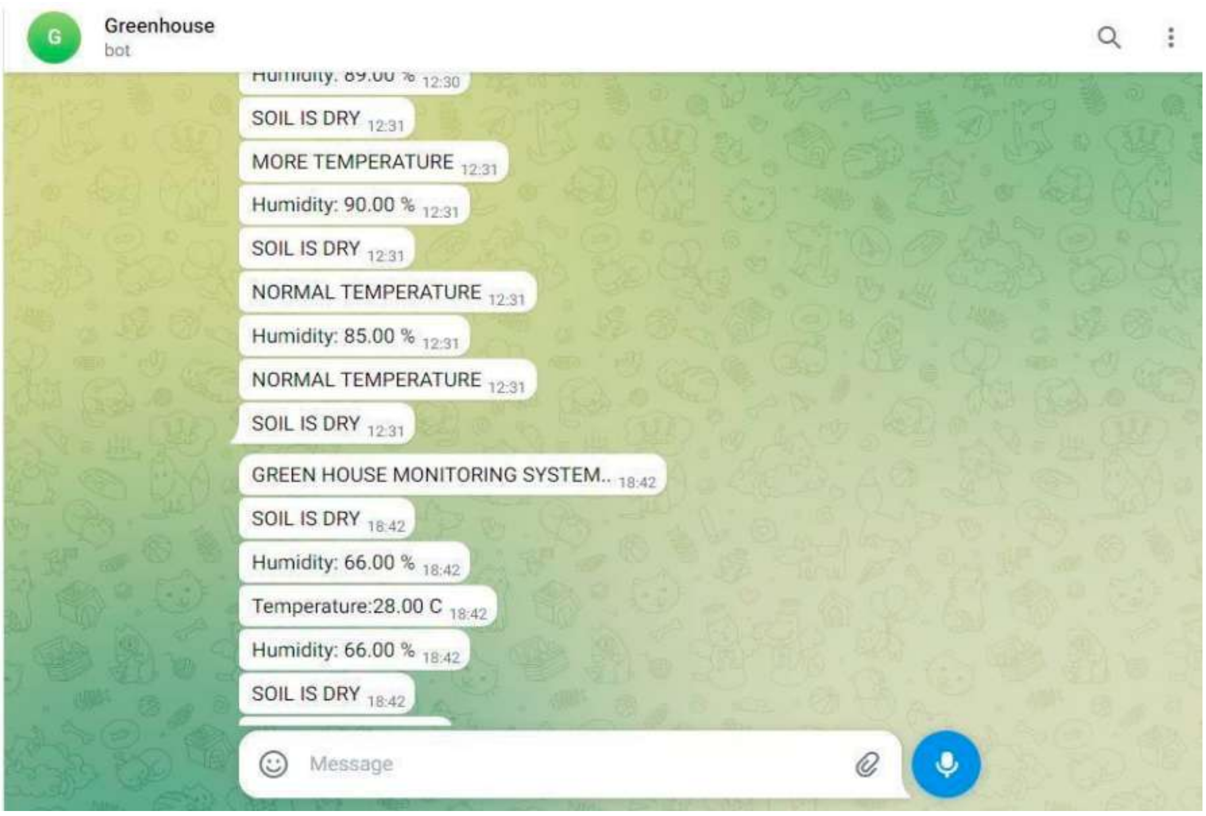


Figure B4: Working of LED with LDR Sensor



Humidity: 89.00 % 12:30

SOIL IS DRY 12:31

MORE TEMPERATURE 12:31

Humidity: 90.00 % 12:31

SOIL IS DRY 12:31

NORMAL TEMPERATURE 12:31

Humidity: 85.00 % 12:31

NORMAL TEMPERATURE 12:31

SOIL IS DRY 12:31

GREEN HOUSE MONITORING SYSTEM.. 18:42

SOIL IS DRY 18:42

Humidity: 66.00 % 18:42

Temperature:28.00 C 18:42

Humidity: 66.00 % 18:42

SOIL IS DRY 18:42

Figure B5: Telegram Notification