**Automated IoT-Based Irrigation System**

**Pauline Wairimu Wang’ombe**

**Student ID: 135859**

**Group: 4.2B**

**An Information System Project 2 documentation submitted to the School of Computing and Engineering Sciences in partial fulfilment of the requirements for the award of the bachelor’s degree in business information technology of   
Strathmore University**

**School of Computing and Engineering Sciences**

**Strathmore University**

**Nairobi, Kenya.**

**December 2023**

# Declaration

I declare that this work has not been previously submitted and approved for the award of a Bachelor’s degree by this or any other University. To the best of my knowledge and belief, the documentation contains no material previously published or written by another person except where due reference is made in the documentation itself.

Student’s signature:

………………………………….………….. [*Signature*]

………………………………….………….. [*Date*]

**Approval**

The Information System Project 2 documentation of Pauline Wairimu Wang’ombe was reviewed and approved *(for examination)* by:

Supervisor’s signature:

………………………………….………….. [*Signature*]

………………………………….………….. [*Date*]

**Abstract**

Agriculture plays a vital role in ensuring food security and sustainable development. However, conventional irrigation practices often result in inefficient water usage, leading to water scarcity, increased operational costs, and low crop yields. Water has become a scarce resource in present times therefore there is need for it to be used very efficiently to enable irrigation. Most farmers irrigate their crops manually by use of a water pump. If they forget to switch it off, water is wasted. The aim of this project was to develop an Automated IoT-based irrigation system that revolutionizes irrigation by addressing water efficiency, monitoring, and control issues. The system has a humidity and temperature sensor to detect the humidity and temperature around the plant. A soil moisture sensor detects the soil moisture of the plant so when the moisture is below a minimum value, water is supplied from a water tank using an ultrasonic sensor. The farmer can see the temperature, data, and soil moisture content of their plant on a web-based dashboard which allows them to automatically control their water pump. The tools that have been used are sensors, a microcontroller, firebase, node-red and Arduino IDE and the system has been tested through unit testing and integration testing.

**Keywords*:*** *IoT, smart irrigation, automation*

**Table of Contents**

Declaration ii

Chapter 1: Introduction 1

1.1 Background 1

1.2 Problem Statement 2

1.3 Aim 3

1.4 Specific Objectives 3

1.5 Research Questions 3

1.6 Justification 3

1.7 Scope and Limitations 4

Chapter 2: Literature Review 4

2.1 Introduction 4

2.2 Current methods of irrigation 4

2.2.1 Surface Irrigation 4

2.2.2 Drip Irrigation 5

2.2.3 Sprinkler Irrigation 5

2.3 Gaps in current methods of irrigation 5

2.3.1 Applying water uniformly ignores crop-specific needs and reduces potential yield………………………………………………………………………………………5

2.3.2 Inefficient use of water leads to oversaturation of water in crops 5

2.3.3 The use of Bluetooth in Wireless Sensor Networks (WSN) causes communication problems due to loss of signals over wide distances 6

2.3.4 Crop vitality is endangered by the absence of tank water monitoring in agricultural practices 6

2.4 Current smart irrigation IoT systems 6

2.4.1 Crop X Agronomic Farm Management System 6

2.4.2 Semios 7

2.4.3 Soil Scout 7

2.5 How proposed system extends the current systems in place 7

2.5.1 Conceptual Framework 8

Chapter 3: System Development Methodology 9

3.1 Introduction 9

3.2 Rapid Application Development (RAD) 9

3.2.1 Define Requirements 9

3.2.2 Prototype 10

3.2.3 Rapid Construction and Feedback Gathering 10

3.2.4 Implementation 10

3.3 Method to be used to Gather the Functional and Non-Functional Requirements 11

3.4 List of Design Diagrams that will be drawn in Chapter 4 11

3.5 List of Development Tools that will be used 12

3.6 Method to be used to test the developed system 12

3.7 Domain of Execution 12

3.8 Proposed Modules and System Architecture 13

Chapter 4: System Development Methodology 14

4.1 Introduction 14

4.2 System Requirements 14

4.2.1 Functional Requirements 14

4.2.2 Non-Functional Requirements 14

4.3 System Analysis Diagrams 15

4.3.1 Use Case Diagram 15

4.3.2 Sequence Diagram 19

4.3.3 Entity Relationship Diagram (ERD) 20

4.3.4 Class Diagram 21

4.3.5 Database Schema 22

Chapter 5: System Implementation and Testing 24

5.1 Introduction 24

5.2 Description of the Implementation Environment 24

5.2.1 Hardware Specifications 24

5.2.2 Software Specifications 27

5.3 Description of Testing 28

5.3.1 IoT 28

5.3.2 Testing Paradigm 30

Chapter 6: Conclusion, Recommendations and Future Works 32

6.1 Conclusion 32

6.2 Recommendations 32

6.3 Future Works 33

Appendix 1: Gantt Chart 40

Appendix 2: Marking Guide 41

Appendix 3: Plagiarism Report…………………………………………………………….…42

Appendix 4: GitHub Analytics and Link……………………………….……………………43

Appendix 5: User Manual…………………………………………………………………….44

**List of Figures**

Figure 2. 1: Conceptual Framework 8

Figure 3. 1: System Architecture……………………………………………………………..13

Figure 4. 1: System Use Case Diagram 16

Figure 4. 2: System Sequence Diagram 20

Figure 4. 3: System Entity Relationship Diagram 21

Figure 4. 4: System Class Diagram 22

Figure 4. 5: System Database Schema 23

[Appx 5. 1: Setup of the system 44](#_Toc152245221)

[Appx 5. 2: Setup of the system testing dry soil 45](#_Toc152245222)

[Appx 5. 3: Dashboard displaying irrigation pump is on and soil moisture is low 46](#_Toc152245223)

[Appx 5. 4: Setup of the system testing wet soil 47](#_Toc152245224)

[Appx 5. 5: Dashboard displaying irrigation pump is off and soil moisture is high 48](#_Toc152245225)

[Appx 5. 6: Setup of the system testing that the water level is below 20% 49](#_Toc152245226)

[Appx 5. 7: Dashboard displaying tank pump is on and tank water level is low 50](#_Toc152245227)

[Appx 5. 8: Setup of the system testing that the water level is above 60% 51](#_Toc152245228)

[Appx 5. 9: Dashboard displaying tank pump is off and tank water level is full 52](#_Toc152245229)

**List of Tables**

Table 4. 1: Use case description of temperature and humidity sensor 16

Table 4. 2: Use case description of soil moisture sensor 17

Table 4. 3: Use case description of ultrasonic sensor 17

Table 4. 4: Use case description of Firebase module 18

Table 4. 5: Use case description of NodeMCU ESP8266 18

Table 4. 6: Use case description of changing threshold values for farmer 18

Table 4. 7: Use case description of checking plant data and water tank data for farmer 19

**List of Abbreviations**

**A** Analog

**A0** Analog Value

**AC**  Alternating Current

**ADC**  Analog-to-Digital Conversion

**AWG**  American Wire Gauges

**D**  Digital

**DC**  Direct Current

**D0**  Digital Value

**DIO**  Digital Input/Output

**ERD**  Entity Relationship Diagram

**GND** Ground

**GPIO**  General-purpose Input/Output

**GSM** Global System for Mobile Communications

**IDE**  Integrated Development Environment

**IN**  Input

**IoT**  Internet of Things

**LED**  Light Emitting Diode

**MQTT** Messaging Queuing Telemetry Transport

**mA**  Milliampere

**NoSQL**  Not Without SQL

**OOAD**  Object Oriented Analysis and Design

**RAD**  Rapid Application Development

**SMS**  Short Messaging Service

**UI**  User Interface

**UML**  Unified Modelling Language

**V**  Voltage

**VCC**  Voltage Common Collector

**Wi-Fi** Wireless Fidelity

**WSN**  Wireless Sensor Networks

**Acknowledgements**

This project is dedicated to my family whom I sincerely thank for supporting me throughout the project. To my supervisor, Dr. Nelson Ochieng, thank you for your invaluable wisdom and guidance throughout the project. To the team at the IoT lab at @iLabAfrica, thank you for helping me understand IoT better. To my friends, thank you for encouraging me to finish the project. And special thanks to the Almighty God for blessing me with the ability to finish the project.

# Introduction

## Background

The Internet of Things (IoT) describes the network of physical objects— “things”—that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the internet (Oracle, 2020). A “thing” can be a car that has built-in sensors to tell the driver when the tire pressure is low, it can be a person who has a pacesetter, a farm animal with a chip to tell the farmer if the animal is unwell among many other things (Gillis, 2022). IoT enables everyday objects to be connected to the internet via embedded services which enables fast communication as well as making the physical world meet with the digital world.

IoT is used in many industries such as health, traffic monitoring, agriculture, fleet management, hospitality, wearables among others. In the agriculture sector, IoT has been used to automate greenhouses through products such as Farmapp and Growlink which enable the farmer to get real time information on conditions such as lighting, temperature, soil condition and humidity. It has also been used in precision farming in which IoT sensors are used to collect metrics on the plants such as lighting, soil condition, pest infections which helps farmers estimate the perfect amount of water, pesticides that the crop needs, CropX builds IoT sensors that measure soil moisture, temperature, and electric conductivity (*IoT in Agriculture*, 2023). According to the Merriam -Webster Dictionary, Irrigation is the watering of land by artificial means to foster plant growth (Merriam-Webster, 2023). Many farmers still rely on manual ways of irrigating their plants such as drip irrigation. The challenges of the traditional methods of irrigation are that the farmer has no way of knowing if the plant is receiving sufficient water or if the plant has depleted the water that it received by checking the moisture content in the soil. A lot of water and electricity is wasted if the farmer forgets to turn off their tap and with the current drought, water is a very precious commodity. Most tanks do not come with a way of checking the current water level which is frustrating since the farmer often finds themselves caught unaware by an empty tank when they want to irrigate their plants. When a water pump sucks in air due to low water levels, it can result in damage to the pump and in turn affect the entire water system which is expensive to repair or replace. The temperature and humidity surrounding a plant affect how it grows but most farmers do not have a way of knowing the perfect balance of these two conditions to give their plants the best chances of growth. This solution enables a farmer to access a web application through which they can see the temperature, humidity, water level in their tank and soil moisture content of their plants. The farmer can irrigate their plants remotely and can change the threshold values of irrigation.

The system uses a humidity and temperature sensor to detect the water vapor content and temperature around the plant. It also uses a soil moisture sensor to detect the soil moisture of the plant. The ultrasonic sensor detects when the water is below a certain threshold in the tank and refills it saving the farmer valuable time.

## Problem Statement

The current methods of irrigation have made farmer’s lives easier as compared to traditional methods of irrigation however does they still have their own challenges. Drip irrigation does not have a way of regulating the water according to the type of crop and growth stage. Instead, the same amount of water is used for all crops which means that some crops could be under watered leading to low yields. (Eshete et al., 2020). The proposed system enables the farmer to control the amount of water by use of a soil moisture sensor that relays information to the microcontroller which controls the flow of water to the plants.

A challenge facing Wireless Sensor Network (WSN) irrigation systems is that they rely on Bluetooth as the communication protocol between the sensor and the web application which suffers from problems of communication due to wide distances, energy consumption and connectivity. The proposed system uses Wi-Fi rather than Bluetooth to enable farther distances of communication between the sensors and the web application (Roy et al., 2021). Another challenge is inefficient use of water which results in the plants being oversaturated with water. This affects their growth and causes waterlogging of the soil which severely affects its productivity (Stockle, 2010). The proposed system enables the farmer to change the threshold values of irrigation in case it is a dry season, and they want to conserve even more water than the system already does by tweaking the values in the web-based dashboard. Farmers tend to water their plants without knowing the water level in their tanks which is risky since the water can get depleted during a very crucial time.(Gobhinath et al., 2019). The proposed system uses an ultrasonic sensor that detects how much water is left in the tank and refills the tank.

## Aim

The aim of this project is to develop an Automated IoT-based irrigation system that shows a farmer data on a web application about their plant and enables them to irrigate their crops remotely and refill their water tanks remotely.

## Specific Objectives

1. To investigate the challenges in the current smart methods of irrigation.
2. To investigate the challenges faced by farmers with regards to farming.
3. To design a web-based dashboard that will show the farmer the data relayed from the sensors.
4. To develop an IoT irrigation system.
5. To test the system.

## Research Questions

1. What are the challenges in the current smart methods of irrigation?
2. What are the challenges faced by farmers with regards to farming?
3. How can an IoT irrigation system be designed and developed to show the farmer the data relayed from the sensors through a web-based dashboard?
4. How can the system be tested?

## Justification

This solution solves the problem of farmers having to manually irrigate their land. It also solves the problem of farmers over or under irrigating their land since they can see how much water their crops need (Eshete et al., 2020). Farmers no longer have to worry about spoiling their water pumps or being caught unaware by an empty water tank since can see the level of water in their tanks (Gobhinath et al., 2019). Farmers can see the conditions that are favourable to their crops through a web application and do not have to worry about forgetting to irrigate their crops or not switching off the water since the system automatically does that for them.

## Scope and Limitations

This system was focused on helping farmers irrigate their land remotely through use of an IoT irrigation system that relays the data from the temperature, humidity, soil moisture and ultrasonic sensor in a web application. It was implemented by using a NodeMCU ESP8266 microcontroller and temperature, humidity, soil moisture and ultrasonic sensors. The delimitations are that the system does not cover the ability to spray weeds remotely and it does not send an SMS to the farmer.

# Literature Review

## Introduction

This chapter will be reviewing the current methods of irrigation that farmers are using and the gaps that exist in each of these existing methods and why they are not effective. It will also look at the smart irrigation IoT systems that are in place and how they work. Additionally, it will review how the proposed system extends the current systems and the different microcontrollers available and how they work.

## Current methods of irrigation

### Surface Irrigation

It refers to systems that carry water to crops by using gravity over the surface of a field (Taghvaeian, 2017). It is among the first engineering innovations of man, and dates to more than 6000 years ago. It has three major categories: Basin, border, and furrow systems. In basin systems, water is directed rapidly to the entire basin and allowed to penetrate while furrow systems are mainly used to irrigate row crops and vegetables. Surface irrigation systems have

Some advantages such as being inexpensive, and less sensitive to weather variables and water quality (Taghvaeian, 2017).

### Drip Irrigation

It is also known as trickle irrigation or micro-irrigation. It is a low-pressure system designed for precise water delivery. It uses a system of pipes, tubes, emitters, and sprinklers. Water is applied in low amounts but over a long period of time. It happens mostly at the ground level. A cover at each end of the pipe or tube prevents water from leaking out of the emitter points and permits water to pressurize the whole length of the pipe (AGRIVI, 2022). Some advantages of drip irrigation include low evaporation losses, having a fast rate of vegetative growth, the land does not need to be levelled and no soil erosion takes place (Jayant et al., 2022).

### Sprinkler Irrigation

It is a method of applying water which is like rainfall. A system of pipes is used to distribute water, typically via pumping. Sprinklers are used to spray it into the air, where it fragments into tiny water drops that land on the ground (C. Brouwer). Sprinklers, the pump supply system, and the working environment must be planned to provide for a consistent application of water. Some advantages of sprinkler irrigation are it helps to increase yield, it reduces soil compaction, it is suitable for undulating land, and it reduces labour cost (C.Brouwer, 2023).

## Gaps in current methods of irrigation

### Applying water uniformly ignores crop-specific needs and reduces potential yield

Applying water uniformly to crops seems very rational but is not always helpful as in the case of the Godino irrigation scheme in Ethiopia. Farmers complained that they were not able to acquire the full water demand needed for a good yield of potatoes. This is due to the same amount of water being applied regardless of crop type and growth stage (Eshete et al., 2020). Soil and crop development tend to differ which is why when water is applied uniformly, it can lead to under irrigation or over irrigation (Daccache et al., 2015).

### Inefficient use of water leads to oversaturation of water in crops

Poorly planned or maintained irrigation systems can cause problems such as excessive water being applied to crops. Not only is it a waste of water which is a scarce resource considering global warming, but it can also reduce yields and quality of the crops as well as increasing the threat of nutrient leaching (Lincoln Zotarelli, 2019). This is especially true in sandy soils which is why the system was structured to use thresholds from the sensors to carry out irrigation. It is not only crops that are affected by too much water. The soil suffers too and becomes a very good habitat for pathogens, tuber rotting ones specifically. It can also cause foliar blights and wilts in tubers which limits the potential for a good yield (C. Shock, 2006).

### The use of Bluetooth in Wireless Sensor Networks (WSN) causes communication problems due to loss of signals over wide distances

Wireless Sensor Networks (WSN) are a game changer in agriculture due to enabling the use of sensors such as soil moisture sensor, temperature sensor et cetera which in turn allows precision agriculture to flourish. However, they have challenges such as the use of Bluetooth which suffers from attenuation of signals over long distances which affects the inter-node communication (Ojha et al., 2015). Additionally, WSN’s tend to be power consuming and can be unreliable. When placed in agricultural fields, they easily lose communication as they consider it to be a harsh environment (Mafuta et al., 2013).

### Crop vitality is endangered by the absence of tank water monitoring in agricultural practices

Farmers tend to operate on a “by feel” method of farming in which they are unaware of the specific figures of the parameters that make their crops grow such as temperature, water, humidity et cetera. Considering this, they are also unaware of the water level in their tanks. In some irrigation schemes in India, it was found that knowing the water level in a tank is very useful to calculate the area of crops to be irrigated (Burton, 2016). Furthermore, in an irrigation scheme in India, the only way to check the level of the water is manually since there is no system to show the level of water in the tank (Nehlin, 2016)

## Current smart irrigation IoT systems

### Crop X Agronomic Farm Management System

The user-friendly CropX software allows for the management of several farms and fields from a single account by synthesizing data from the ground up. The creation of predictive agronomic insights and guidance given to users on a straightforward, yet effective dashboard accessed through desktop or mobile device is powered by soil sensors, satellites, farm equipment, and a wide variety of data sources. Some of the data they provide include soil depths, weather, topography, soil type among others. Sensors are used to capture data which helps the system give the farmer suggestions on what a plant needs to thrive (LinkedIn, 2013).They have the most sophisticated adaptive irrigation solution in the world, which automatically optimizes watering and provides farms with huge increases in crop yields as well as water and energy cost savings . Additionally, they use machine learning to take the data from the sensors and use it to predict suggestions to the farmer on what a plant needs to survive. (Crop X Agronomic Farm Management System, 2013).

### Semios

Semios is a scalable data analytics platform that aids in the prediction, detection, and prevention of pest and disease pressure for growers of tree fruit and tree nut crops. The Semios analytics engine uses a variety of data and information sources, including a strong, wireless network of in-canopy sensors detecting weather, soil, and insect pest activity on every customer farm (LinkedIn, 2021). The system allows the farmer to get accurate information about the water content in their soil at different soil depths and allows them to check if the plants are getting enough water as well as planning what time to irrigate their plants by using in-canopy sensors and their irrigation planner tool (Semios, 2010).

### Soil Scout

Soil Scout is the only soil sensor solution that can be freely deployed in the field. The system is pre-configured and starts working as soon as you power up the base station. The tiny sensor will run underground for 20 years without maintenance. It does not interfere with field work and is safe from wheel pressure and wildlife, providing excellent cost efficiency for years to come. The intelligent placement of sensors allows for real-time monitoring of actual differences, enabling optimal treatment of each area individually. When irrigating, data before/during/after enables both rapid response and long-term capability to maintain optimal soil moisture and save irrigation costs (Soil Scout, 2013).

## How proposed system extends the current systems in place

The proposed system carries out automatic irrigation and the data about the plant is displayed on a web-based dashboard. The water tank is automatically refilled when the water level is low, and the system stops filling the tank when a certain threshold is met but does not carry out prediction analytics about the data.

### Conceptual Framework

A diagram of a machine

Description automatically generated

Figure 2. 1: Conceptual Framework

The sensors take the measurements of temperature, humidity, soil moisture content and water level and then relay it to the NodeMCU ESP8266 microcontroller which sends the data via Wi-Fi to Google Firebase which is a database. The database stores the data and the data is displayed on the web application dashboard. The relay module controls the water pump enabling it to supply water or switch off water.

# System Development Methodology

## Introduction

An information system's development process is organized, planned, and managed using a system development methodology (slideshare, 2016). The system applied the Object-Oriented Analysis and Design (OOAD) approach.

Object-Oriented Analysis and Design (OOAD) is meant to help in developing systems that have to change rapidly due to dynamic business environments (w3computing.com, 2014). It analyses the problem and identifies all objects and their interactions as well as developing the system based on the objects (ScienceDirect, 2022).

It was used because it has the component of reusability which saves time and effort when developing the system, it is scalable which means that it can handle changes from users and business requirements over time, it can be maintained easily and it is flexible (“Object Oriented Analysis and Design,” 2020).

## Rapid Application Development (RAD)

The system used Rapid Application Development (RAD) which is a strategy for adaptive software development that places less of a focus on detailed design and instead relies on prototype and immediate feedback (kissflow, 2023). It uses minimum planning but focuses more on rapid prototyping and to ensure quicker product delivery, the functional components are created concurrently as prototypes and then combined to create the final product (tutorialspoint, 2013). It provides the advantage of quicker delivery, fewer errors, quicker adjustments, and better adoption of new technologies (kissflow, 2023).

### Define Requirements

All the people involved in the system such as the users, developers come together and discuss the system’s requirements but on a broad level which enables it to be flexible as time goes by. It is a short phase due to the nature of rapid application development of having changing requirements. It is mostly done to understand the project at length (*Rapid Application Development (RAD) Guide | Definition, Steps*, 2023). The system’s requirements were:

1. Purchasing a temperature and humidity sensor, soil moisture sensor and ultrasonic sensor, a microcontroller, relays, pumps, and a breadboard.
2. Testing each sensor to ensure that it works correctly.
3. Testing all the components together to see if they work correctly together.
4. Writing the logic in the code that will enable the system to automatically irrigate the plants or refill the water tank.
5. Sending the data from the sensors to the database through firebase.
6. Displaying the data from the sensors on a dashboard in node-red.
7. Testing the final system to ensure it can irrigate the plants and refill the water tank.

### Prototype

Different prototypes with different features and functions are made quickly and shown to the clients who evaluate it and choose what to keep and what to discard. They mostly contain the main features (kissflow, 2023). It is usually repeated when needed as the project progresses (Chien, 2020). The prototypes are adapted so that testing is possible at each stage (kissflow, 2023). The system’s prototypes include:

1. The sensors individually working and transmitting data to the Arduino IDE.
2. The sensors together with the microcontroller working and transmitting data to the Arduino IDE.
3. The sensors sending code to the database which is firebase.
4. The data being displayed on a web-based dashboard on node-red.
5. The final system irrigating the plants and refilling the water tank.

### Rapid Construction and Feedback Gathering

In this stage, coding, testing, and integration is done which turns the prototype into a working system. It can be repeated as often as is needed. fast application development tools or low-code tools are used.

Feedback is obtained from users about the interface and functionality, and they also suggest changes and new ideas that can solve problems as they are found. This helps the developer continue prototyping or move on to the last step (Chien, 2020).

### Implementation

Components are transferred to a real-world production setting where extensive testing is done to find product flaws. Full-scale testing is also done in this stage. Documentation is written and other maintenance tasks are completed before the user can be given the system (Chien, 2020).

Once a system has undergone a thorough evaluation for elements like stability and longevity, it is prepared for delivery (*Rapid Application Development (RAD) Guide | Definition, Steps*, 2023).

## Method to be used to Gather the Functional and Non-Functional Requirements

Document Analysis in which relevant documentation such as academic papers, reports and manuals were analysed and the academic papers were obtained from Google Scholar. Industry standard and regulations contain guidelines and regulations related to smart irrigation systems in which requirements were drawn from. Expert consultation from domain experts such as IoT specialists who can provide insights into necessary requirements was carried out.

## List of Design Diagrams that will be drawn in Chapter 4

A use case diagram states the behaviour that is expected of the system and can be implemented both textually and visually. It helps in designing the system from the user’s point of view (Visual Paradigm, 2019). It states the steps taken by the user to complete an activity and is useful to identify, clarify and organize system requirements (Brush, 2022).

A database schema specifies the logical restrictions, including table names, fields, data types, and the connections between these entities, that govern how data is arranged in a relational database (IBM, 2023). It includes a description of the database that can be illustrated using a schema diagram (tutorialspoint, 2014).

Entity Relationship Diagram (ERD) is a kind of structural diagram used in the design of databases. An ERD has a variety of symbols and connectors that represent two key pieces of information: the key players within the system's scope and their interrelationships. Entities in ERD are business objects like persons or roles, tangible business items, intangible business objects etc. "Relationship" refers to how these components of the system relate to one another (Visual Paradigm, 2019).

A sequence diagram is a diagram created using the Unified Modelling Language (UML) that shows the flow of messages sent and received by objects during an interaction. A group of objects that are represented by lifelines and the messages they exchange over the course of an interaction make up a sequence diagram. The order in which messages are transferred between objects is depicted in a sequence diagram. Sequence diagrams can also display the communication channels between various items (*IBM Documentation*, 2021).

## List of Development Tools that will be used

The system uses Arduino and the microcontroller used is NodeMCU ESP8266 because it has a built in Wi-Fi adapter which will make it easy to connect to the internet. The sensors used are a humidity and temperature sensor, soil moisture sensor and an ultrasonic sensor which measures the level of water in a tank since it is a distance sensor. The programming language used is C++ and it is designed to run on Arduino-compatible microcontroller boards. The sensors send data to a web-based dashboard on node-red through the MQTT messaging protocol.

## Method to be used to test the developed system

Software testing with a focus on individual software system units or components is known as unit testing. Unit testing checks that each piece of software operates as intended and complies with specifications (“Unit Testing | Software Testing,” 2019). It is done in isolation to make sure that a unit doesn't rely on any outside programs or processes (TechTarget, 2022). The system needed unit testing to ensure that the various sensors are working correctly and sending data accurately.

In Integration testing, units or individual software components are tested collectively during this testing. The goal is to identify flaws when integrated components or units interact (java point, 2020). The system needed integration testing to check if sensors, microcontrollers, and the web-based dashboard are working well together.

## Domain of Execution

The system was run on a web-based dashboard because it offers cross-platform compatibility thus the farmer will not have to worry about having to install a specific application. It enabled the farmer to see data about their plant and change the threshold values of irrigation. They were also able to see whether their water tank was refilled with water when it became empty.

## Proposed Modules and System Architecture

A diagram of a software system

Description automatically generated

Figure 3. 1: System Architecture

Sensors are essential for gathering information about the environment, including soil moisture, temperature, humidity and ultrasonic. The use of these sensors can help assess the requirement for irrigation by providing real-time data about the farm conditions.

Data exchange between irrigation system components and a central control unit or cloud-based platform is made possible by the communication module. It can send and receive commands using wireless communication protocols including Wi-Fi especially through the MQTT protocol.

The control unit is the irrigation system's mind. It collects information from the sensors, processes it, and then decides according to predetermined rules or user-defined parameters. When and how much water should be applied to the crops is decided by the control unit. In this case it is the NodeMCU ESP8266.

An interface for users to interact with the irrigation system is provided by a user interface module. It is a web-based dashboard. Farmers can monitor the system and change threshold values for irrigation.

# System Development Methodology

## Introduction

This chapter talks about the functional and non-functional requirements of the Smart IoT-Based Irrigation System for Farm Management. It also talks about how users will engage with the developed system, the various components of the system and how the system’s various components interact with one another. This is demonstrated though a use case diagram, database schema, Entity Relationship Diagram (ERD) and a sequence diagram. As stated in chapter 3, the system will use the Object-Oriented Analysis and Design (OOAD) approach.

## System Requirements

Some of the functional and non-functional requirements of the system include:

### Functional Requirements

1. The system will automatically irrigate the plants once the soil moisture threshold decreases past 50%.
2. The system will automatically stop irrigating the plants once the soil moisture threshold increases past 70%.
3. The system will automatically refill the water tank once the water level decreases past 20%.
4. The system will automatically stop refilling the water tank once the water level increases past 60%.
5. The system will send data to firebase for storage since it is the database.
6. The system will display data from the sensors on a web-based dashboard.
7. The farmer will be able to change the soil moisture and water level thresholds on the web-based dashboard.
8. The system will use Wi-Fi to send data to the database and for the data to be displayed on the web-based dashboard by use of a NodeMCU ESP8266.

### Non-Functional Requirements

1. Environmental Impact - The system aims to save water by ensuring minimal water is used for irrigation and this has been done through enabling the farmer to change the threshold values on the dashboard. This feature can be very handy during the dry season when water is scarce.
2. Usability - The system should have an intuitive user interface and be user-friendly for farmers and this has been done through providing a web-based dashboard which displays the data from the sensors and allows the farmer to change the threshold values.
3. Interoperability - The system should be able to integrate with other farm management and automation systems and technologies and this can be achieved by using standardized communication protocols (e.g., MQTT) and ensuring compatibility with common data formats.
4. Maintainability - The components of the system should be easily replaceable, and this has been done through using a breadboard to connect the components which allows them to be disconnected easily and replaced if necessary. They do not need to be soldered to the board permanently.

## System Analysis Diagrams

Some of the system analysis diagrams that have been considered are as follows:

### Use Case Diagram

A use case diagram is used to show the various ways a user might interact with a system. The sensors take the measurements of temperature, humidity, soil moisture content and water level and then relay it to the NodeMCU ESP8266 which then sends the data to Google firebase. The data is retrieved from firebase and displayed on the web application dashboard. Table 4.1, Table 4.2, Table 4.3, Table 4.4, Table 4.5, Table 4.6, Table 4.7, Table 4.8, and Table 4.9 give more details on the main use cases in the system.

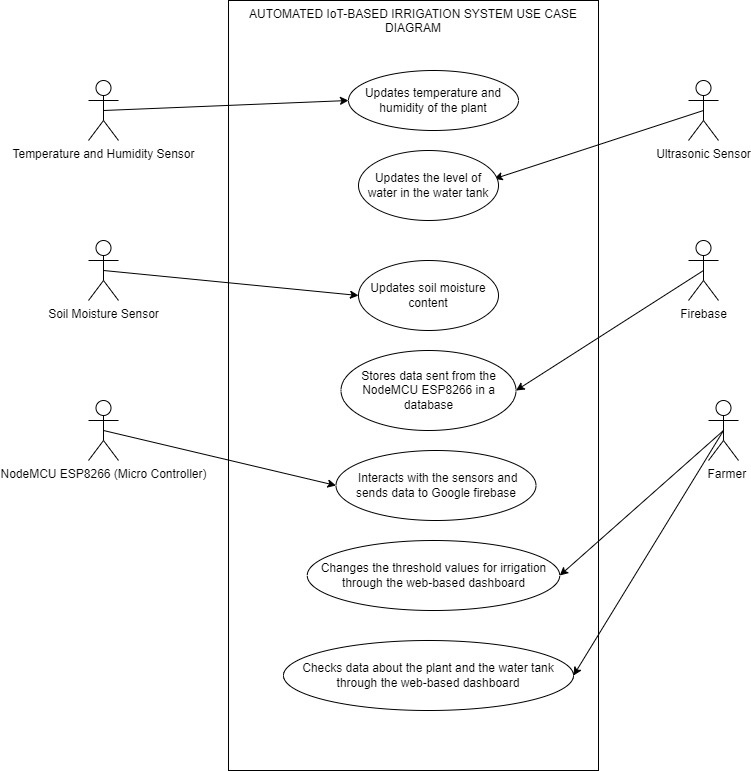


Figure 4. 1: System Use Case Diagram

Table 4. 1: Use case description of temperature and humidity sensor

|  |  |
| --- | --- |
| **Use Case Name:** | Update temperature and humidity of the plant |
| **Actor:** | Temperature and humidity sensor |
| **Precondition:** | Connection of sensor to NodeMCU ESP8266 and sensor being near plant |
| **Main Sequence:** | 1. Measure temperature and humidity of plant. 2. Send data to NodeMCU ESP8266. 3. Data is checked for a certain threshold value. 4. If it has reached the value, plant is irrigated. 5. Data is displayed on web application dashboard. |
| **Alternate Sequence:** | 1. If threshold value has not been met, plant is not irrigated. |

Table 4. 2: Use case description of soil moisture sensor

|  |  |
| --- | --- |
| **Use Case Name:** | Update soil moisture of the plant |
| **Actor:** | Soil Moisture sensor |
| **Precondition:** | Connection of sensor to NodeMCU ESP8266 and insertion of sensor into soil |
| **Main Sequence:** | 1. Measure soil moisture of plant. 2. Send data to NodeMCU ESP8266. 3. Data is checked for a certain threshold value. 4. If it has reached the value, plant is irrigated. 5. Data is displayed on web application dashboard. |
| **Alternate Sequence:** | 1. If threshold value has not been met, plant is not irrigated. |

Table 4. 3: Use case description of ultrasonic sensor

|  |  |
| --- | --- |
| **Use Case Name:** | Update water level of the tank |
| **Actor:** | Ultrasonic sensor |
| **Precondition:** | Connection of sensor to NodeMCU ESP8266 and insertion of sensor into water tank |
| **Main Sequence:** | 1. Measure water level in tank. 2. Send data to NodeMCU ESP8266. 3. Data is checked for a certain threshold value. 4. If it is under a certain value, the water tank is refilled. |

Table 4. 4: Use case description of Firebase module

|  |  |
| --- | --- |
| **Use Case Name:** | Stores data from the NodeMCU ESP8266 to a database |
| **Actor:** | Firebase module |
| **Precondition:** | Having a firebase project with a real-time database and incorporating firebase into the automated irrigation code |
| **Main Sequence:** | 1. Sensors send data to NodeMCU ESP8266. 2. NodeMCU ESP8266 sends data to Firebase module. 3. Data is stored into the firebase database. |

Table 4. 5: Use case description of NodeMCU ESP8266

|  |  |
| --- | --- |
| **Use Case Name:** | Send data from sensors to Firebase module |
| **Actor:** | NodeMCU ESP8266 |
| **Precondition:** | Connection of NodeMCU ESP8266 to laptop |
| **Main Sequence:** | 1. Receives data from sensors. 2. Sends data to Firebase module. |

Table 4. 6: Use case description of changing threshold values for farmer

|  |  |
| --- | --- |
| **Use Case Name:** | Changing threshold values for irrigation through web application dashboard |
| **Actor:** | Farmer |
| **Precondition:** | Login into the web application dashboard |
| **Main Sequence:** | 1. Farmer accesses web application dashboard by login. 2. Farmer changes threshold values if it is a dry season and they want to use threshold values that will lead to less water being used by the system. |

Table 4. 7: Use case description of checking plant data and water tank data for farmer

|  |  |
| --- | --- |
| **Use Case Name:** | Checking plant data and water tank data through the web-based dashboard |
| **Actor:** | Farmer |
| **Precondition:** | Login into the web application dashboard |
| **Main Sequence:** | 1. Farmer accesses web application dashboard by login. 2. Farmer checks temperature and humidity values as well as soil moisture values of their plant. 3. Farmer checks water tank data about water level in the tank. |

### Sequence Diagram

A sequence diagram essentially shows how things interact with one another sequentially, or the order in which these interactions occur. A sequence diagram can also be referred to as event diagrams or event scenarios. Sequence diagrams show the actions taken by the components of a system in chronological order.

A diagram with text and arrows

Description automatically generated with medium confidence

Figure 4. 2: System Sequence Diagram

### Entity Relationship Diagram (ERD)

ERD diagrams provide a visual representation of relationships among people, objects, places and concepts or events within the system. This ERD diagram shows how the entities relate to each other in the system. It starts with the sensors which send data to the NodeMCU ESP8266 which in turn sends data to the web application dashboard through the Firebase module which stores data in a database. Users can access the data through the dashboard.

A diagram of a diagram

Description automatically generated

Figure 4. 3: System Entity Relationship Diagram

### Class Diagram

It represents the application's static view. Class diagrams are used to create executable code for software applications as well as for visualizing, explaining, and documenting various elements of systems. The characteristics and functions of a class are described in a class diagram, along with the restrictions placed on the system.

A close-up of a computer screen

Description automatically generated

Figure 4. 4: System Class Diagram

### Database Schema

A database schema represents how data is organized within a relational database. It works as a guide when creating the database of the system. It is a diagrammatic representation of the database. The main entities in the database are represented in the below.

A diagram of a computer flowchart

Description automatically generated

Figure 4. 5: System Database Schema

# System Implementation and Testing

## Introduction

This chapter focuses on the description of the implementation environment, the hardware and software specifications used to make the automated irrigation system. It also covers additional software that contributed to making the project successful. The setup of the project and testing of the components has been covered here as well as testing of the overall project.

## Description of the Implementation Environment

The Hardware used for this project includes a NodeMCU ESP8266 which serves as the microcontroller responsible for sending data from the sensors to the database. The sensors used are a DHT11 Temperature and Humidity sensor which measures the temperature and humidity, a soil moisture sensor which measures the moisture of the soil and HC-SR04 Ultrasonic sensor which measures the distance to an object by use of sonar. Two relays were used which act as a switch to control the pumps and two pumps were used to supply water to the tank and to the farm. Each pump gets its power supply from three batteries housed in a battery holder. The hardware is connected on a breadboard which is used to build temporary circuits.

### Hardware Specifications

|  |  |
| --- | --- |
| A black circuit board with a small chip  Description automatically generated with medium confidence | The **NodeMCU ESP8266** is a microcontroller that can be used alone or to allow other microcontrollers to connect to a Wi-Fi network.  It has features such as:   1. Operating voltage: 3.3V 2. Input voltage: 7-12V 3. Digital I/O Pins (DIO): 16 4. Analog Input Pins (ADC): 1 |
|  | The **DHT11 Temperature and Humidity Sensor** is used to measure temperature and relative humidity.  It has features such as:   1. Temperature Range: 0 to 50 ºC +/-2 ºC 2. Humidity Range: 20 to 90% +/-5% 3. Operating Voltage: 3-5.5V DC 4. Current Supply: 0.5-2.5 mA |
| A black device with wires  Description automatically generated with medium confidence | The **Soil Moisture Sensor** is used to detect moisture in the soil and has a voltage comparator that evaluates the amount of moisture.  It has features such as:   1. Voltage: DC 3.3-12V 2. Current: 20mA-30Ma 3. Interface: + - Digital Value (DO), Analog Value (AO) 4. Operating Temperature: 25-85 Celsius |
| A close-up of a blue circuit board  Description automatically generated | The **HC-SR04 Ultrasonic Sensor** is used to measure distance to an object through sonar and reads from 2cm to 400cm with an accuracy of 0.3cm.  It has features such as:   1. Power Supply: +5V DC 2. Working Current: 15mA 3. Measuring Angle: 30 ° 4. Effectual Angle: <15 ° |
| A blue electronic device with red and green lights  Description automatically generated | The **5V/12V 1 Channel Relay Module** is used to interface low-voltage digital circuits (such as those from microcontrollers like Arduino with high-voltage devices (such as pumps) therefore acting as an isolated switch between the two.  It has features such as:   1. Has isolation grove in control and load area. 2. Can control both DC and AC signal as well as 220V AC load. 3. Has a permanent open and close vents contact 4. Public and common side will breakover when there is a signal in signal input |
| A white plastic pump with black wire  Description automatically generated | The **Micro-Submersible pump** is used to supply water to the plant.  It has features such as:   1. Small and light, compact and easy to install. 2. Voltage: DC 3V 3. Safe and explosion proof 4. High efficiency and energy saving |
| A clear plastic case with wires  Description automatically generated | The **Battery Case with a Power Switch** is used to hold the batteries that supply power to the pumps and has a switch that enables turning on the pump to be simple as compared to disconnecting wires.  It has features such as:   1. Series Circuit Configuration 2. Includes integrated on/off switch. 3. Holds three AA batteries. 4. Clear Plastic |
| A close-up of a circuit board  Description automatically generated | The **Breadboard** is used to build temporary circuits and allows components to be removed and replaced easily.  It has features such as:   1. Wire Size: Suitable for 29-20 AWG wires 2. Dimension: 165mm \* 55mm \* 10mm 3. Tie Points: 830 tie point consists of 630 tie point terminal strip tie point distribution strips 4. Matrix: 126 separate 5 point terminals, plus 4 horizontal bus lines of 50 test points each |

### Software Specifications

|  |  |  |  |
| --- | --- | --- | --- |
| Software | Version/Release | Platform | Additional Information |
| Arduino IDE | 2.2.1 | Windows, macOS, Linux | Open-source software for writing, compiling, and uploading code to almost all Arduino Modules. |
| Firebase | 12.9.0 | Cross-Platform | Backend-as-a-service platform for web and mobile applications and is a NOSQL database program. |
| Windows 10 Pro | 22H2 | Windows | Operating system by Windows. |
| node-red | 3.1.0 | Cross-Platform | Flow-based development tool for visual programming of IoT |
| Node.js | 20.9.0 | Cross-Platform | Server-side JavaScript runtime for building scalable network applications and node-red is built on top of it. |
| Google Chrome | 119.0.6045.160 (Official Build) (64-bit) | Windows, macOS, Linux | Web browser developed by google |
| C++ | C++11 | Cross-Platform | C++ variant used in Arduino Programming. |
| MQTT Protocol | Latest Version | Cross-Platform | Lightweight messaging protocol for small sensors and mobile devices. |

## Description of Testing

Unit testing involves testing the smallest parts of the system which are called units to check whether they are working properly. In this case, each sensor, the microcontroller, the pumps, and relays are tested individually to ensure they are functioning by writing code in Arduino for each sensor and the microcontroller and it is uploaded to see if it will compile successfully.

Integration testing involves testing the individual units as a group to check that there are no bugs when the units are interacting. It confirms that all the units are communicating properly. In this case, code is written in the Arduino that combines all the units together and it is uploaded to see if it will compile successfully.

### IoT

The setup was done by connecting a NodeMCU ESP8266 to the breadboard. The NodeMCU has a pinout that consists of digital and analog pins. The digital pins are denoted by a D and the analog ones are denoted by an A.

The DHT11 temperature and humidity sensor has a DATA pin, a VCC, a GND and NC. The NC pin is not in use. The D2 pin was connected to the NodeMCU which corresponds to GPIO4 and the GND (common ground) and VCC (powers the sensor and can be 5V or 3.3V) were connected to the positive and negative terminal of the breadboard.

The soil moisture sensor has an analog pin, a digital pin, and the GND and VCC. The analog pin is mostly used thus it was connected to the A0 pin of the NodeMCU and the ground and VCC were connected to the positive and negative terminal of the breadboard.

The HC-SR04 Ultrasonic Sensor has an echo output pin, trigger input pin, GND and VCC. The echo pin was connected to D6 pin which corresponds to GPIO12, trigger pin was connected to D5 which corresponds to GPIO14, and the ground and VCC were connected to the positive and negative terminal of the breadboard.

The 5V/12V 1 Channel Relay Module has a front and back piece. At the terminal block there are three terminals: normally open, normally closed, and common. The pump has a ground wire and a VCC wire. The ground of the pump is connected to the ground of the battery holder and the VCC is connected to the normally closed terminal. At the input jumper section, there is VCC, GND and IN (Input) pins. The system has two relays: One for controlling the irrigation pump and one for controlling the pump that supplies water to refill the tank.

The IN pin for the Irrigation relay is connected to D7 pin of the NodeMCU which corresponds to GPIO13 and ground and VCC were connected to the positive and negative terminal of the breadboard while the IN pin for the Tank relay is connected to D8 pin of the NodeMCU which corresponds to GPIO15 and ground and VCC were connected to the positive and negative terminal of the breadboard.

When the setup was complete, the code to send the data from the sensors to firebase was written in the Arduino IDE. The data was saved in a Realtime database on firebase and sent to node-red to be displayed through a UI dashboard.

Each component was tested by connecting it to its respective pin of the NodeMCU and writing the appropriate code for it. The code was then uploaded in the Arduino IDE and successful uploading was confirmed by having no errors in the output bar of the Arduino IDE and having the appropriate units displayed in the serial monitor.

The Wi-Fi module which is the NodeMCU ESP8266 was tested by connecting it to the breadboard and writing some code to check if the LED on it would blink. To test whether it would send data through Wi-Fi, the DHT11 was connected to it and some code was written and upon successful uploading, it sent data to firebase.

### Testing Paradigm

Unit testing is a form of white box testing thus the reason why it was used. It also helps in ensuring that each component is working correctly. In this case, a few of the functional requirements were tested to check that they were accurate.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Functional Requirement | Test Data | Expected Result | Actual Result | Pass/Fail | Evidence |
| The system will automatically irrigate the plants once the soil moisture threshold decreases past 50%. | Dry Soil | The system will automatically irrigate the plants | The system irrigated the plants | Pass | Appendix 5  Appx 5.3 |
| The system will automatically stop irrigating the plants once the soil moisture threshold increases past 70%. | Wet Soil | The system will automatically stop irrigating the plants | The system stopped irrigating the plants | Pass | Appendix 5  Appx 5.5 |
| The system will automatically refill the water tank once the water level decreases past 20%. | Water level being below 20% | The system will automatically fill the water tank | The system filled the water tank | Pass | Appendix 5  Appx 5.7 |
| The system will automatically stop refilling the water tank once the water level increases past 60%. | Water level being above 60% | The system will automatically stop filling the water tank | The system stopped filling the water tank | Pass | Appendix 5  Appx 5.8 |

# Conclusion, Recommendations and Future Works

## Conclusion

The system was developed to automatically irrigate plants once the soil moisture decreases past 50% and to stop irrigating the plants once the soil moisture increases past 70%. The water tank is automatically refilled once the water level decreases past 20% and when the water level increases past 60%, the pump is switched off. The system also displays the temperature and humidity of the plant and allows the farmer to change the threshold values of irrigation and refilling the water tank in case of a dry season. This was done by using sensors, a microcontroller, relays, and pumps. The data was displayed on a web-based dashboard.

The results of the system show that water can be conserved efficiently during irrigation while saving the farmer the labour of manual irrigation since the plants are only irrigated when their soil moisture is low and once, they are saturated, the system switches the pump off which conserves water and power. Additionally, the system ensures that the farmer is never worried about not having enough water in their tank for irrigation since the system automatically detects when the water level is low and refills the tank and stops filling it once it is moderately full. The farmer can see the data about their plants on a web-based dashboard which gives them more insights on the health of their plants. During droughts, the farmer can conserve even more water by changing the threshold values of irrigation and refilling the water tank which ultimately contributes to conserving the environment and using precious resources sparingly.

## Recommendations

The challenges that were encountered during the project include the high cost of purchasing the components and the inconvenience of some of the components being out of stock with no guarantee of restocking. Another challenge was transporting the final system safely in a way that the components will not break or disconnect. Since IoT is not well covered the syllabus, learning how to do various things had to be done on YouTube or through online articles which was tricky due to some missing information.

The recommendations for this project include:

1. Using a GSM module to send data from the sensors to the database instead of using the NodeMCU ESP8266 since the farmer can buy bundles for it in case, they live on a farm that has no access to Wi-Fi.
2. Including a login functionality for the farmer to ensure that no unauthorized person has access to their system.
3. Using a more accurate sensor than the HC-SR04 Ultrasonic Sensor to measure the water level since it gives faulty readings when the distance is at below 3cm.
4. A mobile application can be created which would make it easier for the farmer to see their data on the go.

## Future Works

More research could be done on expanding the system to incorporate spraying pesticide on weeds remotely and sending an SMS to the farmer to notify them about the pump being on or off. Additionally, further research could be done on including analytics about the crop so that the farmer can be able to make useful predictions about the crop.

**References**

AGRIVI. (2022, May 16). Drip Irrigation: Full Guide │ Function │ Parts. *AGRIVI*. https://www.agrivi.com/blog/drip-irrigation-as-the-most-efficient-irrigation-system-type/

Burton, D. M. (2016). *Water management issues at the farm level*. https://www.wbadmip.org/demo/intweb/images/uploaded/brochure\_files/d045c59a90d7587d8d671b5f5aec4e7c.pdf

Chien, C. (2020, February 4). *What is Rapid Application Development (RAD)?* Codebots. https://codebots.com/app-development/what-is-rapid-application-development-rad

Daccache, A., Knox, J. W., Weatherhead, E. K., Daneshkhah, A., & Hess, T. M. (2015). Implementing precision irrigation in a humid climate – Recent experiences and on-going challenges. *Agricultural Water Management*, *147*, 135–143. https://doi.org/10.1016/j.agwat.2014.05.018

Eshete, D. G., Sinshaw, B. G., & Legese, K. G. (2020). Critical review on improving irrigation water use efficiency: Advances, challenges, and opportunities in the Ethiopia context. *Water-Energy Nexus*, *3*, 143–154. https://doi.org/10.1016/j.wen.2020.09.001

crop

*IBM Documentation*. (2021, March 5). https://www.ibm.com/docs/en/rsm/7.5.0?topic=uml-sequence-diagrams

Jayant, B., Dahiya, K., Rukhiyar, A., Ahirwar, R., & Meena, R. (2022). *A REVIEW OF THE DRIP IRRIGATION SYSTEM*. *01*. https://doi.org/10.55953/JERA.2022.1103

Karpagam, J., Merlin, I. I., Bavithra, P., & Kousalya, J. (2020). Smart Irrigation System Using IoT. *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, 1292–1295. https://doi.org/10.1109/ICACCS48705.2020.9074201

Mafuta, M., Zennaro, M., Bagula, A., Ault, G., Gombachika, H., & Chadza, T. (2013). Successful Deployment of a Wireless Sensor Network for Precision Agriculture in Malawi. *International Journal of Distributed Sensor Networks*, *9*(5), 150703. https://doi.org/10.1155/2013/150703

Object Oriented Analysis and Design. (2020, April 7). *GeeksforGeeks*. https://www.geeksforgeeks.org/object-oriented-analysis-and-design/

Ojha, T., Misra, S., & Raghuwanshi, N. S. (2015). Wireless sensor networks for agriculture: The state-of-the-art in practice and future challenges. *Computers and Electronics in Agriculture*, *118*, 66–84. https://doi.org/10.1016/j.compag.2015.08.011

*Rapid Application Development (RAD) Guide | Definition, Steps*. (2023, June 2). https://quixy.com/blog/rapid-application-development/

Roy, S. K., Misra, S., Raghuwanshi, N. S., & Das, S. K. (2021). AgriSens: IoT-Based Dynamic Irrigation Scheduling System for Water Management of Irrigated Crops. *IEEE Internet of Things Journal*, *8*(6), 5023–5030. https://doi.org/10.1109/JIOT.2020.3036126

Shock, C., Flock, R., Eldredge, E., Pereira, A., & Jensen, L. (n.d.). *Drip Irrigation Guide for Potatoes in the Treasure Valley*.

Unit Testing | Software Testing. (2019, April 18). *GeeksforGeeks*. https://www.geeksforgeeks.org/unit-testing-software-testing/

w3computing.com. (2014, July 17). *Object-Oriented Systems Analysis and Design*. https://www.w3computing.com/systemsanalysis/object-oriented-systems-analysis-design/

Zotarelli, L., Dukes, M. D., Simonne, E. H., & Barrett, C. E. (n.d.). *2020–2021 Vegetable Production Handbook: Chapter 3. Principles and Practices of Irrigation Management for Vegetables*.

Brush, K. (2022, March 23). *TechTarget*. Retrieved from TechTarget: https://www.techtarget.com/searchsoftwarequality/definition/use-case

C. Shock, R. F. (2006). Drip Irrigation Guide for Potatoes in the Treasure Valley.

C.Brouwer, K. P. (2023, July 6). *vikaspedia.* Retrieved from vikaspedia: https://vikaspedia.in/agriculture/agri-inputs/farm-machinary/sprinkler-irrigation-system

*Crop X Agronomic Farm Management System*. (2013). Retrieved from Crop X Agronomic Farm Management System: https://cropx.com/

Gillis, A. S. (2022). *TechTarget*. Retrieved from TechTarget: https://www.techtarget.com/iotagenda/definition/Internet-of-Things-IoT

Hamilton, T. (2023, October 7). Retrieved from GURU99: https://www.guru99.com/white-box-testing.html

*IBM*. (2023, February). Retrieved from IBM: https://www.ibm.com/topics/database-schema

*java point*. (2020, August 8). Retrieved from java point: https://www.javatpoint.com/integration-testing

*kissflow.* (2023, June 19). Retrieved from kissflow: https://kissflow.com/application-development/rad/rapid-application-development/

*kissflow*. (2023, May 30). Retrieved from kissflow: https://kissflow.com/application-development/rad/benefits-of-rapid-application-development/

Lincoln Zotarelli, M. D. (2019). 2020–2021 Vegetable Production Handbook: Chapter 3. Principles and Practices of Irrigation Management for Vegetables. *Florida Online Journals*.

*LinkedIn*. (2013). Retrieved from CropX Technolgies: https://www.linkedin.com/company/cropx-ltd/?originalSubdomain=nl

*LinkedIn*. (2021). Retrieved from Semios: https://www.linkedin.com/feed/update/urn:li:activity:6757417371289432064/

*Merriam-Webster*. (2023, June 25). Retrieved from Merriam-Webster: https://www.merriam-webster.com/dictionary/irrigation

Nehlin, M. (2016, June 13). *Diva Portal.* Retrieved from https://www.diva-portal.org/smash/get/diva2:945941/FULLTEXT01.pdf

*Oracle*. (2020, March). Retrieved from Oracle: https://www.oracle.com/ke/internet-of-things/what-is-iot/

*ScienceDirect.* (2022, October 28). Retrieved from ScienceDirect: https://www.sciencedirect.com/topics/computer-science/oriented-analysis

*Semios*. (2010). Retrieved from Semios: https://semios.com/

*slideshare.* (2016, November). Retrieved from slideshare: https://www.slideshare.net/KashifKhan76/system-development-methodologies-64648700

*Soil Scout*. (2013, June). Retrieved from Soil Scout: https://soilscout.com/

Stockle, C. O. (2010, July 9). Retrieved from https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=8321f0dbb29814bd74e6bc858210809ca847a227

Taghvaeian, S. (2017, February 14). Retrieved from https://extension.okstate.edu/fact-sheets/print-publications/bae/surface-irrigation-systems-bae-1527.pdf

*TechTarget*. (2022, April 1). Retrieved from TechTarget: https://www.techtarget.com/searchsoftwarequality/definition/unit-testing

*tutorialspoint.* (2013, June 19). Retrieved from tutorialspoint: https://www.tutorialspoint.com/sdlc/sdlc\_rad\_model.htm

*tutorialspoint*. (2014, January 17). Retrieved from tutorialspoint: https://www.tutorialspoint.com/dbms/dbms\_data\_schemas.htm

*Visual Paradigm*. (2019, July 11). Retrieved from Visual Paradigm: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/

*Visual Paradigm*. (2019, January 22). Retrieved from Visual Paradigm: https://www.visual-paradigm.com/guide/data-modeling/what-is-entity-relationship-diagram/

# 

# Appendix 1: Gantt Chart

# A screenshot of a computer Description automatically generated

# Appendix 2: Marking Guide

**Strathmore University**

**School of Computing and Engineering Sciences**

**Information Systems Project Documentation Assessment Guide**

|  |  |
| --- | --- |
| **Student Number(s)** |  |
| **Working Title:** |  |
|  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Evaluation Points** | **Weight** | **Score** | **Notes** |
| Title | 1 |  |  |
| Abstract  *Updated to include chapter 1-6* | 3 |  |  |
| **Chapter 1-3**  \**checking previous proposal chapters for the correctness of title and problem statement, project scope as implemented and change of tenses* |  |  |  |
| Problem Statement | 1 |  |  |
| Justification | 1 |  |  |
| Scope | 1 |  |  |
| Limitation | 1 |  |  |
| Literature Review | 2 |  |  |
| Methodology | 2 |  |  |
| **Chapter 4**  Correct functional requirements  Correct non-functional requirements  System Architecture and accompanying literature  4 Design diagrams and accompanying literature | 3  3  2  4 |  |  |
| **Chapter 5**  Setup Description: Hardware, software, support libraries, frameworks, versions and compatibility  Description of how the solution works to meet problem and business needs  Description of the test environment, data, test case   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Functional Requirement | Test Data | Expected Result | Actual Result | Pass/Fail | Evidence |   *\*Check 3 core functional requirements and evidence of test available as appendix* | 6  3  6 |  |  |
| **Chapter 6**  Valid Conclusion  Sound Recommendation | 2  2 |  |  |
| **Presentation**  Document Structure as per template provided and grammar  Citation and References  Document Numbering and Table of Contents/figures  Existence of required appendices | 2  2  2  1 |  |  |
| **Total Marks** | **50** |  |  |

Comments

Examiner Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Appendix 3: Plagiarism Report**

**A screenshot of a computer

Description automatically generated**

**Appendix 4: GitHub Analytics and Link**

The link to the Github is: <https://github.com/Paula-Wang>

**A screenshot of a calendar

Description automatically generated**

Appx 4. 1: Github Contribution

These are the contributions that I have made to Github throughout the year 2023.

A screenshot of a computer

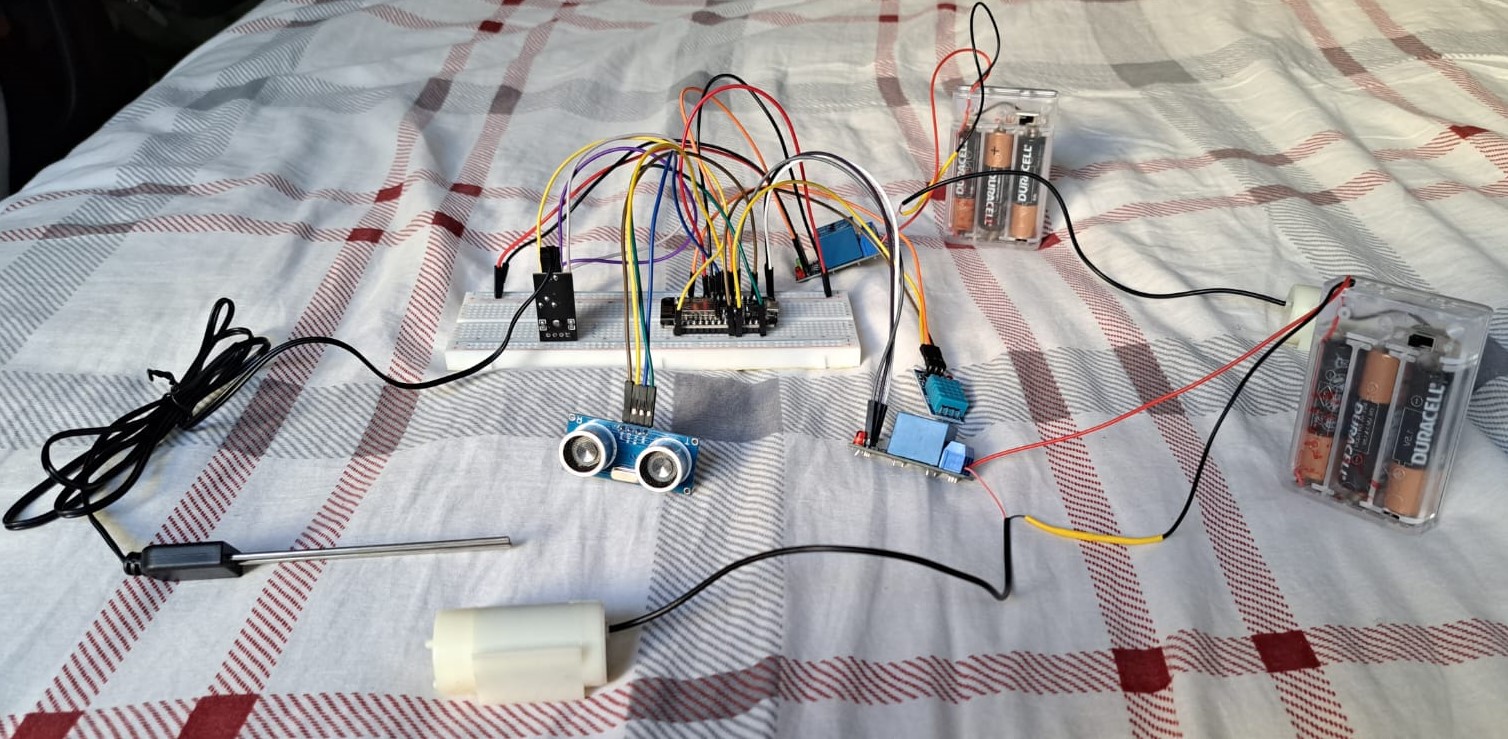
Description automatically generated

Appx 4. 2: Github Activity Overview

This is the activity overview of my Github throughout the year 2023.

**Appendix 5: User Manual**

The system can be setup as shown below:



Appx 5. 1: Setup of the system

To test that the system automatically irrigates the plants once the soil moisture threshold decreases past 50%, the setup can be done as shown below:

A computer with wires and a bottle on a bed

Description automatically generated with medium confidence

Appx 5. 2: Setup of the system testing dry soil

The results of the setup above are shown below:

A screenshot of a computer

Description automatically generated

Appx 5. 3: Dashboard displaying irrigation pump is on and soil moisture is low

To test that the system automatically stops irrigating the plants once the soil moisture threshold increases past 70%, the setup can be done as shown below:

A computer with wires and a container on a table

Description automatically generated

Appx 5. 4: Setup of the system testing wet soil

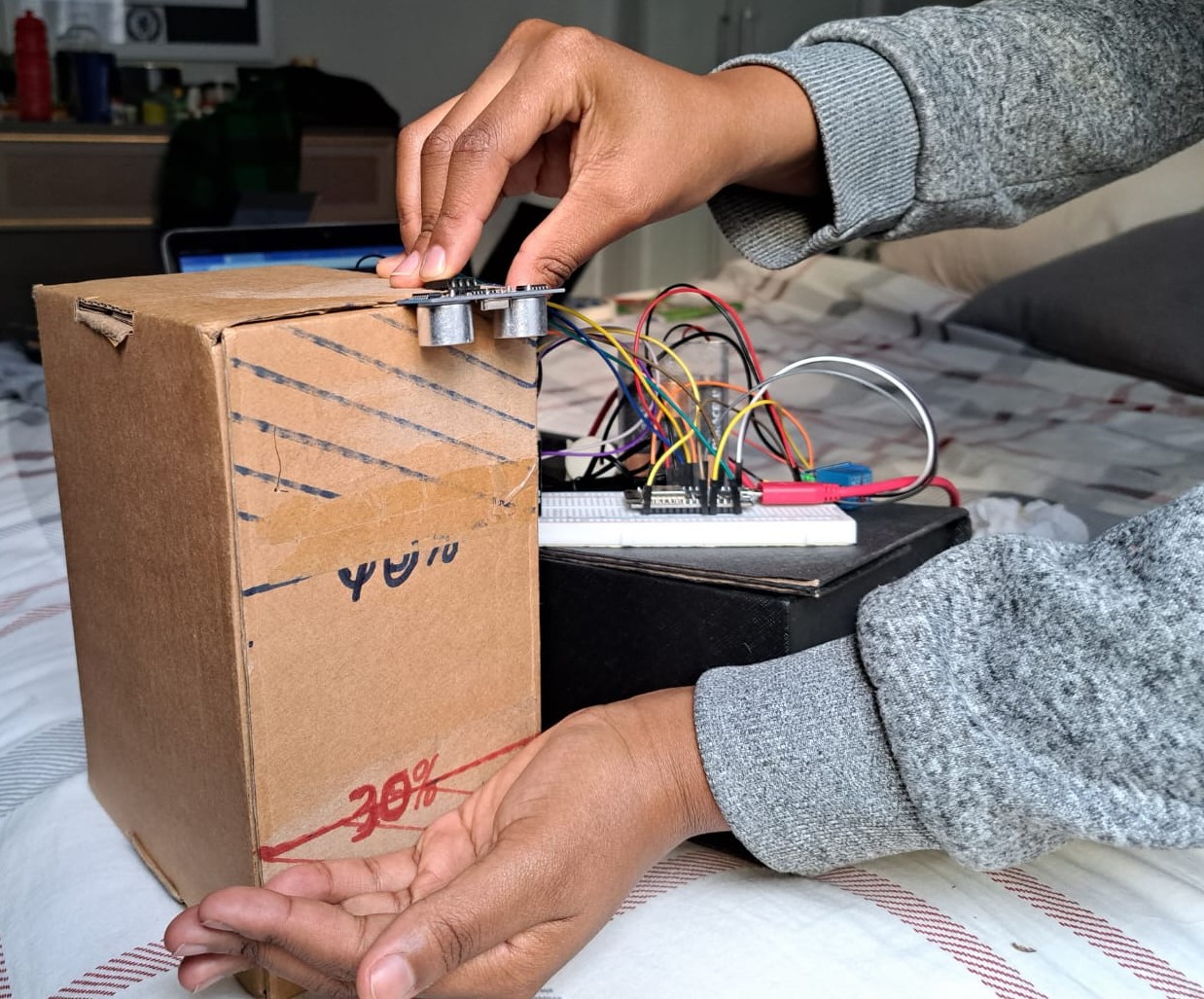
The results of the setup above are shown below:

A screenshot of a computer

Description automatically generated

Appx 5. 5: Dashboard displaying irrigation pump is off and soil moisture is high

To test that the system automatically refills the water tank once the water level decreases past 20%, the setup can be done as shown below:



Appx 5. 6: Setup of the system testing that the water level is below 20%

The results of the setup above are shown below:

A screenshot of a computer

Description automatically generated

Appx 5. 7: Dashboard displaying tank pump is on and tank water level is low

To test that the system automatically stops refilling the water tank once the water level increases past 60%, the setup can be done as shown below:



Appx 5. 8: Setup of the system testing that the water level is above 60%

The results of the setup above are shown below:

A screenshot of a computer

Description automatically generated

Appx 5. 9: Dashboard displaying tank pump is off and tank water level is full