

ABSTRACT

The purpose of this study was to develop a system that can help farmers and garden enthusiasts to manage their greenhouses by the use of a network of sensors connected together and controlled by a central hub. This complies with the world vision of connecting as much devices together as possible that has been dubbed the Internet of Things phenomenon. The system makes use of sensors that a controlled by a central controller that sends the data to a server that streams the data to an android application that allows the user to view minute by minute sensory values of the greenhouse sensors. In a country like Zimbabwe were the rain patterns are changing due to global warming it could be a great asset, since it has the ability to measure the amount of water used by plants can help optimize the use of the resource and it will allow farmers to venture into producing crops that are not native to the country since a ventilation system will be used to control the climate of the greenhouse (climate control).

DECLARATION

I **ASHLEY NASH C. CHAKANYUKA (R156517P)**, hereby declare that I am the sole author of this dissertation. I authorize the Midlands State University to lend this to other institutions or individuals for the purpose of scholarly research.

Signature..... Date.....

APPROVAL

This dissertation entitled “**Internet of Things based Greenhouse** by **Ashley Nash C. Chakanyuka**” meets the regulations governing the award of the degree of **Bachelor of Computer Science Honors Degree** of the **Midlands State University**, and is approved for its contribution to knowledge and literacy presentation.

Supervisor : Ms. B Mugoniwa

Signature :

Date :

ACKNOWLEDGEMENTS

My appreciation goes to those whose support, advice and encouragements made this project a success. I am heartily thankful to my supervisor, Ms. B Mugoniwa whose encouragement, guidance and support from the initial to the final phase enabled me to develop this system. Special thanks to my family who have given me the strength to continue when the road looked bleak. I also thank my classmates who have been with me these past four years who have been with me at every turn. May the Lord continue to bless you.

DEDICATIONS

To my family, my pillar of strength, cause of your sacrifice I have been able to reach this far.

Contents

ABSTRACT	i
DECLARATION	ii
APPROVAL.....	iii
ACKNOWLEDGEMENTS	iv
DEDICATIONS.....	v
List of Appendices	x
List of Figures	xi
List of Tables	xiii
Chapter 1: Introduction	1
1.1 Introduction	1
1.2 Background of Study.....	1
1.3 Research Problem.....	2
1.3.1 Aim of Research Study.....	2
1.3.2 Objectives of Research	3
1.4 Methods and Instruments	3
1.4.1 Instruments	3
1.4.2 Methods	4
1.5 Expected Results/ Contribution and future scope of the project.....	5
1.5.1 Expected Results.....	5
1.5.2 Future Scope	6
1.6 Research Limitations and Delimitations	6
1.6.1 Limitations.....	6
1.6.2 Delimitations	6

1.7 Significance of the Research	7
1.8 Proposed Budget	7
1.9 Time Scheduling	8
1.10 Conclusion.....	9
Chapter 2: Literature Review	10
2.1 Introduction	10
2.2 Background of Greenhouses	10
2.3 Evaluation of previous implementations.....	11
2.4 Proposed Implementation.....	13
2.4.1 Limitations of proposed system.....	13
2.4.2 Benefits of the proposed system.....	14
2.5 Foreknowledge	14
2.6 Conclusion.....	15
Chapter 3: Methodology	16
3.1 Introduction	16
3.2 Component Description.....	16
3.2.1 Sensors.....	16
3.2.2 Actuators.....	19
3.2.3 Arduino Mega 2560.....	20
3.2.4 ESP8266 Wi-Fi Module	22
3.2.5 Four Channel Relay	23
3.2.6 ATX Power Supply	23
3.3 Software Components of the Project.....	24
3.3.1 IDEs	24
3.3.2 Programming Language	24

3.4 Conclusion.....	25
Chapter 4: System Design.....	26
4.1 Introduction.....	26
4.2 Data flow.....	26
4.3 System architectural design.....	27
4.4 Circuit and Schematic design.....	28
4.5 Physical Design.....	29
4.6 Interface design.....	30
4.7 Pseudocode.....	32
4.8 Client-Server Architecture.....	32
4.9 Security Design,.....	33
4.9.1 Physical Security.....	33
4.9.2 Network Security.....	33
4.9.3 Operational Security.....	34
4.10 Conclusion.....	34
Chapter 5: Implementation.....	35
5.1 Introduction.....	35
5.2 Coding.....	35
5.2.1 Mobile application source code.....	35
5.2.2 Microcontroller source code.....	37
5.3 Software testing.....	40
5.3.1 Android Application testing.....	40
5.4 Results.....	43
5.4.1 Objective one.....	43
5.4.2 Objective two.....	44

5.4.3 Objective three.....	45
5.5 Objectives vs System	46
5.6 Installation.....	46
5.6.1 Implementation Strategies	47
5.6.2 User training	47
5.7 System Maintenance	47
5.7.1 Corrective Maintenance.....	48
5.7.2 Perfective Maintenance	48
5.7.3 Adaptive Maintenance.....	48
5.7.4 Preventive Maintenance	48
5.8 Recommendations for future development	48
5.9 Conclusion.....	49
References.....	50
Appendices.....	54

List of Appendices

Appendix A: User Manual.....	54
Appendix B: Hardware Code.....	58

List of Figures

Figure 1. 1 Smart greenhouse Block Diagram.....	5
Figure 1. 2 Gantt chart	9
Figure 2. 1 Wireless sensors using ZigBee technologies	12
Figure 3. 1 Soil Moisture Sensor	17
Figure 3. 2 Inside view of a DHT22 sensor	18
Figure 3. 3 Water Flow Sensor	18
Figure 3. 4 Water Pump	19
Figure 3. 5 Cooling Fan	20
Figure 3. 6 Arduino Mega 2560 board.....	21
Figure 3. 7 ESP8266 Wi-Fi module.....	22
Figure 3. 8 Four Channel relay	23
Figure 3. 9 ATX Power Supply Module.....	24
Figure 4. 1 Flow Chart of Greenhouse Project	27
Figure 4. 2 Architectural design of the components	28
Figure 4. 3 Circuit design for greenhouse components	29
Figure 4. 4 Physical Design of the Greenhouse	30
Figure 4. 5 UI for Greenhouse control panel	31
Figure 4. 6 UI for historical data view	31
Figure 5. 1 First Android code snippet	36
Figure 5. 2 Second Android code snippet.....	36
Figure 5. 3 Third Android code snippet.....	37
Figure 5. 4 First Arduino code snippet	38
Figure 5. 5 Second Arduino code snippet.....	38
Figure 5. 6 Third Arduino code snippet.....	39
Figure 5. 7 Fourth Arduino code snippet	39
Figure 5. 8 Application interface when server is connected.....	40
Figure 5. 9 Application interface when server is disconnected	41
Figure 5. 10 Sensor values from serial monitor	42
Figure 5. 11 Esp8266 blue light indicating activity	42
Figure 5. 12 Connection status using serial monitor	43

Figure 5. 13 Humidity level alert.....	44
Figure 5. 14 Values from sensors	45
Figure 5. 15 Historical values sensor data	46
Figure A1. 1 Start screen	54
Figure A1. 2 Select period to view historical results	55
Figure A1. 3 Legend of graphical Data	56
Figure A1. 4 Graphical data	56
Figure A1. 5 Circuit setup of components	57

List of Tables

Table 1. 1 Hardware Components	7
Table 1. 2 Time Plan	8
Table 3. 1 Specifications of DHT22 Sensor	17
Table 3. 2 Arduino Mega 2560 specifications	20

Chapter 1: Introduction

1.1 Introduction

This chapter will seek to introduce the proposed system then it will go further on to describing the background of the given study that lead the developer to propose the given system. An articulate look with be done on the problem definition and the objectives that will be set that will signify the successful completion of the systems development lifecycle. After this the developer will look at the components needed for the system and the limitations and the delimitations of the system will be discussed. A proposed budget and timeline will be set on how to complete the project.

1.2 Background of Study

The beginning of the 21st century saw an increase in technological advancements at an alarming rate. This made way for new schools of thought that would help shape the internet as we know it. The concept of the Internet of the things is a budding new idea that focuses on connecting all devices used by humans together over the internet. Over the years the number of devices connected to the internet has increased exponentially every year giving way to a much easier and efficient ways of life. A good example of this movement is the introduction of smart home appliances such as fridges, microwaves, home lights and so many more other devices, these devices are able to link to your phone and this allows you the ability to manipulate their functionality from one local hub. It has been projected that by the year 2020 they will be as much as 7 trillion “Things” or devices that are connected to the internet that will be used by individuals to make their lives better. This movement hasn’t been limited to just improving the general life for people in their day to day living, but has been commercialized and is now being used in industries such as commerce, the military revolution, medicine and also agriculture.

The application of the Internet of things in agriculture is by far a step in the right direction. This will allow a more efficient and smart way of dealing with the growth of crops. The use of sensor will greatly increase this. The introduction of greenhouses has allowed farmers to easily manage the growth of their crops and by introducing the concept of internet of things this will greatly automate and make this farming more efficient robust. The use of sensors will produce large volumes of data which can be manipulated using the concept of Big Data. Where smart

algorithms and AI systems are used to foretell future events by drawing conclusions from patterns exhibited by the data that is collected. These predictions will help the smart farmer to efficiently gauge the amount of water, light the plants are being exposed to, temperature and humidity conditions experienced within the greenhouse and allow a better understanding on what the crops really need produce 100% yield.

In the past couple of years Zimbabwe has seen a decline in their agricultural market either due to mismanagement of resources, draught and lack of knowledge of their farmers. This situation has taken the once thriving agricultural sector down a dark path that has left the nation having food shortages and in turn the government has had to rely on foreign aid and buying the needed crops from countries such as Brazil who are the world's leaders in wheat farming. In order for Zimbabwe to regain its place in the world in the agricultural sector a more conscious effort must be made to trying to integrate the local farming industry with the new upcoming technologies that will make farming more efficient.

1.3 Research Problem

After a much-needed reflection into the farming sector in this country the developer has found areas that the concepts of the Internet of Things could help address and thereby solving problems that the average farmer would find hard to solve. These problems being faced are:

- Due to draughts in the country a decline in the availability of water has had a huge impact on the production of crops that are vital to the people of this country.
- Due to the scarcity of water, the need of more dams is of paramount importance to the farmers of this country and due to lack of the necessary funds to build them ways have to be formulated to save this precious resource.
- Due to climate change factors farming is becoming more difficult due to the ever-changing seasons and rain patterns, the need for technology to step in is greater than before in order to find lasting solutions.

1.3.1 Aim of Research Study

The aim of this study is to create a more efficient and smarter greenhouse that will allow the farmer to capitalize on in house climate control to allow the greenhouse to adapt to different plant growths that require different temperature standards. In doing this this will allow farmers

in Zimbabwe to broaden the variety of crops their farm, in so doing this will allow a more efficient way of usage of resources.

1.3.2 Objectives of Research

- To monitor soil moisture using soil moisture sensors that will allow the farmer to view these statistics on an android device. This will allow a more automated water delivery method thereby curbing wastage of the resource.
- Use temperature humidity sensors to monitor the temperature and humidity fluctuations that occur inside the greenhouse, this will enable the greenhouse to be climate controlled to suit the needs of different plants.
- To alert the farmer when soil is below or above a critical level of humidity on his/her android device.

1.4 Methods and Instruments

The methods and instruments are defined as the tools and the steps that are required to develop the project from stretch. Step by step procedures are planned and formulated in order to produce a quality product.

1.4.1 Instruments

In order to demonstrate this concept a prototype must be constructed that will demonstrate at a small scale how this technology will help greenhouse farmers compete in this global village. To achieve this goal following components and software developing tools will be needed for the construction of this prototype.

1.4.1.1 Software Tools

- Android Studio – The aim is to produce a portable interface that the farmer can interact with so as to view the data from the sensors. To achieve this the Android Studio interface will be used to develop an application that can be installed and used on any android device. The main languages that will be used to achieve this fit are Java and Kotlin which are well supported by Android studio.

- Arduino IDE – This is an Integrated Development Environment (IDE) that was developed to suit the needs of individuals that would want to use Arduino micro-controllers in the development of their projects.
- JetBrains WebStorm – This IDE was developed by JetBrains. It allows a developer to use the power of IntelliJ which is a tool that gives code suggestions to the developer. It supports languages like HTML5, JavaScript, PHP, ruby, groovy and a host of other web development languages. In this project it shall be used to develop the JavaScript scripts that will make use of the Node.js environment which will work as an API that will enable the communication between the android application and the micro-controller.
- MongoDB – This is a NoSQL database that stores data in JSON form which is an unstructured form when compared to the traditional databases like MySQL.

1.4.1.2 Components

1. Arduino Uno micro-controller
2. Wi-Fi Module ESP8266
3. Android Device
4. Soil moisture sensor
5. DHT22
6. Resistors
7. Water pump
8. Water flow sensor
9. 4 channel Relay
10. ATX Power supply
11. Tubing to be used as irrigation pipes
12. Water reservoir
13. Fans

1.4.2 Methods

As it is important to have a set of tools that will help develop a well formulated system it is all or not if there is no well defined method in which to execute each step. Fig1.1 shows a block diagram that identifies the inputs and outputs that will be seen within the project.

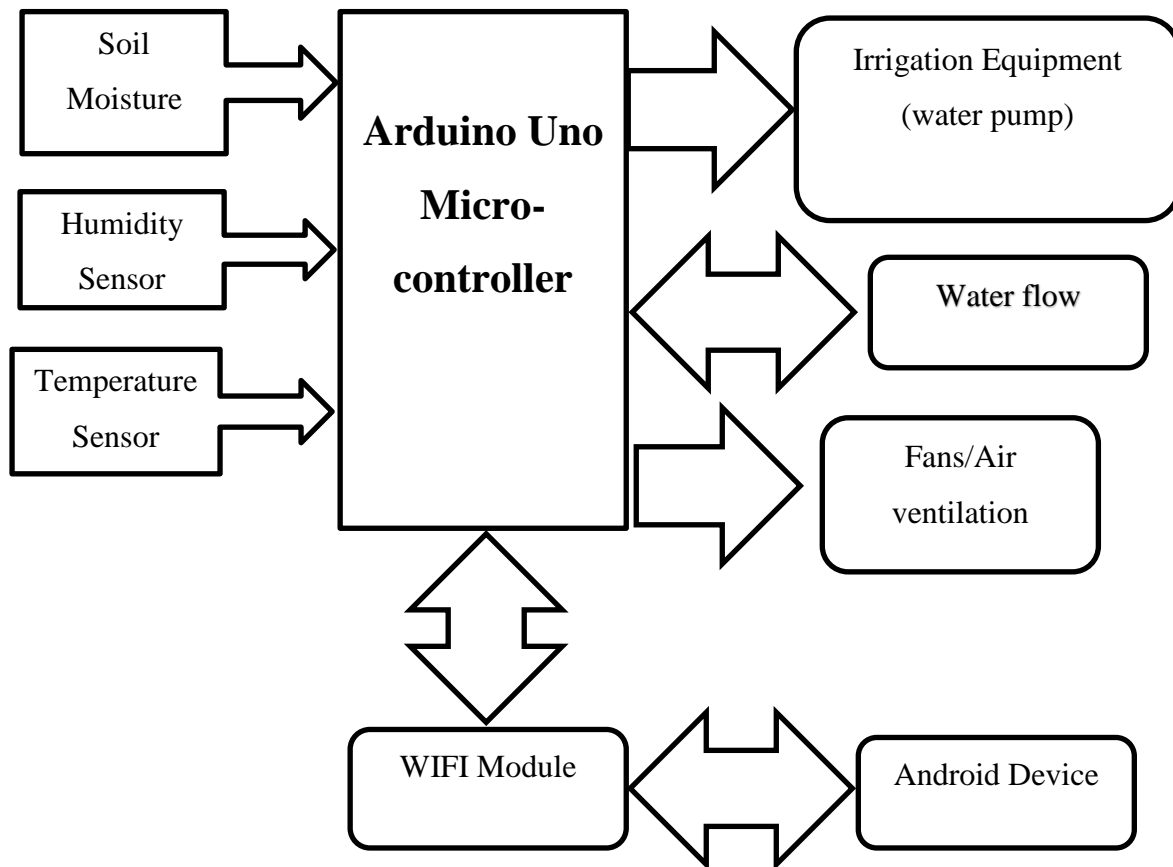


Figure 1. 1 Smart greenhouse Block Diagram

(Source: Author)

1.5 Expected Results/ Contribution and future scope of the project

In any undertaking or project certain results are expected to be the outcome of the project on its completion. Rouse (2015) defines project scope as part of the planning procedure that deals with identifying and documenting project goals, costs, deadlines, deliverables. From this definition it can be derived that the future scope of the project deals with identifying any future goals that can be achieved after the final implementation of the project. This entails that project team is given the task of further researching on any technologies that can be added in any future development as to further enhance the functionality of the system. This section seeks to outline the results that will be expected and the future scope of the project.

1.5.1 Expected Results

- To be able to regulate the temperature and humidity in the greenhouse (climate control) which will allow the farming of different offseason crops.
- To be able to maintain an optimum moisture level of soil to allow optimum plant growth and saving of water.
- Receive real-time data about what is happening in the greenhouse.

1.5.2 Future Scope

- Instead of using electricity to power the greenhouse to make it more ecofriendly use of solar panels could be used to power the greenhouse.
- The use of glass panels that can reflect light and control the amount of light that enters the greenhouse can be implemented in future development.

1.6 Research Limitations and Delimitations

Simon (2011) defines limitations as the potential weaknesses that can be found within a study and are beyond the researchers' control. Limitations can be found in every aspect of a person's life and should not be ignored. To produce a quality project, one must be able to identify these limitations and devise a way of mitigating these problems. Baron (2008) defines delimitators as factors that can affected the study but which the researcher has a degree of control over them. Delimitators help control the boundaries/scope of the study. In this section the limitations and delimitators of this project shall be stated.

1.6.1 Limitations

- Network failures could hinder the communication between the android device and the smart greenhouse system.
- Components used to develop the system might be hard to come by due to economic hardships.

1.6.2 Delimitations

- Farmer confidence might be difficult to gain as they are used to their old tried and tested methods of doing things.
- User training might be difficult due to user resistance.

1.7 Significance of the Research

As a developing country Zimbabwe needs a way for feeding its nation by the use of more technologically efficient methods of farming that allow the affect use of scarce resources like water. Implementation of these concepts will see a growth in the farming sector in Zimbabwe by allowing the country to broaden the types of crops it can produce and increase its market share.

1.8 Proposed Budget

The development of any system needs to have an adequate budget that clearly outlines the software and hardware needs of a project. This budget is formulated so that the developer knows clearly how much is needed to complete the project. Table 1.1 states the hardware and costs that will be incurred in the development of the project.

Table 1. 1 Hardware Components

<i>Component</i>	<i>Unit Price (\$USD)</i>	<i>Quantity</i>	<i>Price (\$USD)</i>
<i>Arduino Mega 2560</i>	23	1	23
<i>DHT22 Temperature and humidity sensor</i>	8	1	8
<i>Soil Moisture Sensor</i>	4	1	4
<i>Wi-Fi Module ESP8266</i>	9	1	9
<i>Water Pump</i>	14	1	14
<i>Water Flow Sensor</i>	10	1	10
<i>4 channel Relay</i>	8	1	8
<i>Fan</i>	5	1	5
<i>Bread board</i>	10	1	10
<i>Resistors</i>	0.10	6	0.60
		Total	91.6

(Source: Author)

1.9 Time Scheduling

In order to complete this project successfully adequate time management must be employed so as to finish the project in the best amount of time. Table 1.2 shows a Time plan that outlines the stages and time frame that will be needed to complete each stage.

Table 1. 2 Time Plan

Activity	Start of Activity	End Activity	Duration (Weeks)
Project Proposal	21/09/2018	11/10/2018	3
Planning	12/10/2018	03/11/2018	3
Analysis	04/11/2018	04/12/2018	4
Design	05/12/2018	05/02/2019	8
Coding	06/02/2019	20/04/2019	10
Testing	21/04/2019	12/05/2019	3
Maintenance	13/05/2019	Ongoing	Ongoing

(Source: Author)

Activity	Week 1-3	Week 4-6	Week 7-9	Week 10-12	Week 13-15	Week 16-18	Week 19-21	Week 22-24	Week 25-27	Week 28-31
Project Proposal										
Planning										
Analysis										
Design										
Coding										
Testing										
Maintenance										ONGOING

Figure 1. 2 Gantt chart (Source Author)

1.10 Conclusion

After the implementation of the concept in this chapter farmers will now able to easily manage their greenhouses by allowing climate control that will allow plants to get adequate resources such as water, light to improve efficient growth of the crops. The next chapter will deal with a comparison of the proposed system and other existing systems in the market.

Chapter 2: Literature Review

2.1 Introduction

In this age where advancements in technology are happening at a more rapid rate, many industries have begun to digitalize in order to stay competitive. Most visibly is the introduction of automation in the agriculture sector and more specifically in the management of greenhouses. This new technology now allows farmers to drastically reduce cost in the production of their crops in the long run. This section seeks to deal with the previous works that have spearheaded this technology to where it is today and also look at current alternatives that are applicable to the Zimbabwe environment.

2.2 Background of Greenhouses

According to the Oxford Dictionary (2005, 3rd edition) a greenhouse can be defined as a structure that is made out of glass or plastic used to protect plants from weather conditions that could threaten their growth and reproduction. Greenhouse-like structures first emerged during the periods of Tiberius Caesar who was then the emperor of Rome. These structures were built as a way to provide the Roman populous with an all-year-round supply of cucumbers. With the coming of the ages, many other civilisations began to adopt this concept of using methods of artificially creating a climate that would allow the growth of plants not indigenous to that region or would have been out of season as compared to the traditional farming methods which did not allow this and were limited in scope. Countries like the Netherlands and England began to also use these structures which in that time were known as orangeries or pineries (for pineapple farming) depending on the plant being farmed in it.

Greenhouses have a wide variety of uses. The most popular is their adaptation in the science of botany, where herbs and flowers are farmed in large quantities for either medical purposes, commercial use or even the preservation of extinct plant species. A perfect example of one of these greenhouse conservations is one found in South Africa named the Pearson Conservatory after Mr Henry Pearson who was then the mayor of Port Elizabeth at the time of its commission.

Greenhouses can be plant-specific or house many species of plants as indicated above. They also come in many different sizes and levels of complexity from the most basic greenhouse

that uses manual labour to maintain it to more high-tech greenhouses that are controlled by expert systems in order to maintain an ideal climate for the desired production output.

This study will aim at looking at the more modern types of greenhouse and how to improve on the works that have already been implemented in this sector of agriculture in order to better improve their efficiency.

2.3 Evaluation of previous implementations

Greenhouses have survived the testament of time and have become the primary method of farming that is used by developed countries that lack the vast land resources possessed by Zimbabwe. This lack of land as a resource has forced these countries to research and focus on how to make their greenhouses more effective and efficient as compared to the very first greenhouses that were used in ancient Rome.

Stipanicev and Marasovic (2010) proposed a system that is based on a technology developed by Dallas Semiconductor Corp. that uses an embedded web server on a TINI (Tiny Internet Interface) board. It utilises the use of a MicroLan system that allows a master device to communicate to small devices like thermometers. This form of network provides power, signalling and low-speed data transmission with the use of a single conductor. This board uses a network of distributed sensors and actuators to collect data. Once the data is connected it is transmitted to the internet by the use of a network cable. The TINI board is a great example of the use of embedded systems. These systems are much cheaper than the PC counter parts that need a large amount of resources to process data that is given by the network or sensors as compared to a microcontroller that are more task specific. For greenhouse enthusiasts this enables them to access the full functionality of an automated system at a lower cost.

As new technology is invented the rise of better and more efficient ways of transmitting data from sensors have been formulated. ZigBee is one of these that have become popular and joined the race with other technologies such as WI-FI and Bluetooth that allow a microprocessor communicate with sensors that are non-as a wireless sensors network. Qianet et al (2010) goes on to compare how this ZigBee technology would fair when compare with both WI-FI and Bluetooth technologies when implemented on Chinese greenhouses. Qianet et al (2010) goes on further to identify five factor that can be used to compare these 3 technologies. The conclusion that ZigBee is a more superior technology was reached due to its high flexibility

and unlimited installations for both large scale and small-scale greenhouses. This solution is clear demonstrated in Figure 2.1 that illustrates the proposed implementation design of this system.

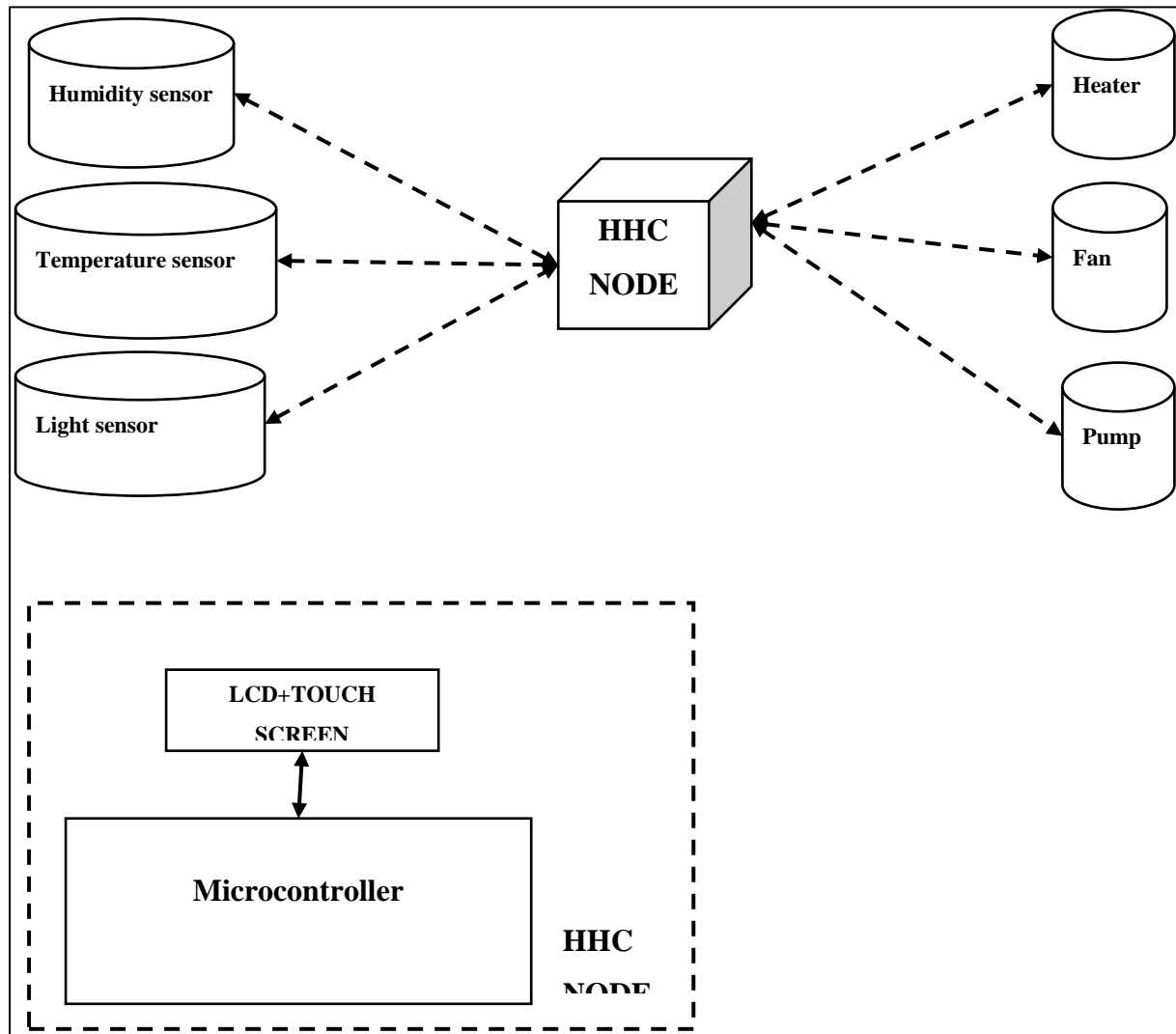


Figure 2. 1 Wireless sensors using ZigBee technologies (Source: Qianet et al (2010))

One drawback of this design is that a central computer needs to be used in order to manage this system. This greatly reduces the flexibility of the farmer.

Abdul Aziz et al (2009) have suggested a system capable of measuring and detecting various temperature ranges using temperature sensors connected to a wireless network of sensors and then transmits the data to the farmer via SMS (Short Message Service). This system will enable the farmer to receive alerts the farmer beforehand if they are any abnormal changes in the greenhouses climate. This form of technique can also further be integrated to use other sensors that can alert the farmer on matters such as the moisture levels of the soil, luminous intensity

being experienced within the greenhouse. A trigger text can be used to activate certain functionalities within the greenhouse such as activating the fans in order to reduce a temperature spike, when moisture level is low the sprinklers can also be activated through this. The drawback of this system is its inability to learn from past experiences and user input in order to create a knowledge base that will enable the system to tackle more trivial issues and make the system more intelligent.

2.4 Proposed Implementation

After evaluating past implementations, a trend is then clearly shown that the need for a more mobile way of monitoring these greenhouses is needed a farmer to be able to manage the events that will be transpiring within the greenhouse and allow more control as the farmer will have access to real time greenhouse data from anywhere in the world. The proposed system borrows many of its ideas from the previous implementation and seeks to further improve and rectify this element of mobility that would greatly improve the efficiency of the greenhouse.

This new system will include an android application that will enable the farmer to view Real-time data of the events that are happening in the greenhouse. The farmer will be able to view the temperature fluctuations and be able to switch on the fans that will ventilate the greenhouse. The android device will also allow the farmer to view historical data for all the events that will be taking place in the greenhouse.

2.4.1 Limitations of proposed system

During the development lifecycle a number of factors are identified that can affect the project negatively and can act as limitations to the develop of the project as a whole. In the development of this system the following limitations have been recognized.

- Due to the nature of Arduino as an open source platform it is vulnerable to exploitation by any individual with intimate knowledge of the system.
- According to Manske (2018) the operational cost of buying or evening building a greenhouse from scratch is an expensive task before mentioning the addition of sensors and other equipment to help automate the greenhouse.

- The application might need a longer learning curve for the more senior citizens as they have a higher resistance to adoption of new technologies which is also argued by Vaportzis et al (2017).
- Need for a constant and reliable internet connection might be a hindrance to the farmer receiving Real-time updates from the greenhouse due to the internet connectivity within the country.

2.4.2 Benefits of the proposed system

The greenhouse system in Zimbabwe is still in its infancy, in order for it to reach the level seen in the developed countries so to allow even competition with the fast-evolving technological powerhouses of the world reforms need to be made to empower Zimbabwean farmers to embrace it. The proposed system seeks to close this gap. The following are the benefits of the proposed system.

- Manske (2018) speaks of an inability to regulate the temperature in the greenhouse. The proposed system seeks to make that task trivial even for the novice of greenhouse farmers by giving them the ability to monitor the greenhouse even from the comfort of their homes.
- Labor force is greatly reduced as most if not all the processes are automated.
- Climate control is essential when a farmer needs to farm crops that are off season or are not indigenous to the region. The sensor can allow the farmer to change the threshold for the various factors/variables such as temperature, humidity, etc. that are needed for these plants.
- The irregular rain patterns that have recently affected that country have made resources such as water a highly important commodity that should be used wisely and sparingly in the day to day survival of animals, humans and plants alike. Implementation of this system will see this resource being used wisely as the farmer is able to accurately allocate the amount of water a plant would need to grow.

2.5 Foreknowledge

In order to acquire a deep appreciation of the implementation of this project one must at least possess a little bit of background information or foreknowledge about the construction and

maintenance of your average day greenhouse. Further knowledge should be attained in the greenhouse automation and this will allow good decision making for which parts can best be automated in order to produce the most efficient produce. This basic understanding will allow the proper choice of electronic equipment and components that will be conducive for the scale of the project. In this project the knowledge of components like the Arduino board and the host of sensors it supports is a must will not just help cut operational costs but also help with maintenance costs as well.

A basic understanding of the interconnection of these components will coming in handy as it will allow the implementation of the project to be smoother. Depending with the size of the green house one may decide to use a ZigBee network that makes use of radio frequencies to transfer data from nodes in the network which are connected to sensors (also known as a Wireless Sensor Network) or just use a central micro-controller that will be able to control all its complimentary components at once. This knowledge can be obtained from various areas on the internet such as Arduino forums that have a host of projects that have been done by other engineers and the way they overcame some difficult areas in the implementation of their projects.

2.6 Conclusion

In this chapter the reader has been exposed to history of greenhouses and how they have come to be the structures they are today. This includes how they first came about and different types of traditional greenhouses that have model and resulted in the ones we have today. This information went on to solidify the stance of why greenhouse automation will help push this field forward in this technologically rich era. The proposed system was then compared toe-to-toe with previous implantations of the system where their pros and cons were clearly outlined.

Chapter 3: Methodology

3.1 Introduction

In this chapter the components and the methods of gathering data will be discussed. The components to be discussed in this chapter are as follows: Arduino Mega 2560 (microcontroller), DHT22 Temperature and Humidity sensor, Soil moisture sensor, WIFI-module ESP8266, Water pump, Water flow sensor, 4 channel relay, Fan, and an ATX power supply.

3.2 Component Description

This section seeks to give a more in depth look at all the components mentioned one at a time in order to give the reader a greater appreciation of their specifications and how these components will collectively work together in order to produce a fully functioning system.

3.2.1 Sensors

According to Ravi (2017) a device with the ability to measure and detect some type of physical quantity be it light, humidity, temperature, etc. as a form of input and then produces an output signal that carries values that can be used for various process is what we define as a sensor. These devices are divided into many categories according to their intended use. For this project the following sensors are going to be utilized to collect the readings for the numerical data:

i) Soil Humidity Sensor

A soil humidity sensor is a two-part component that consists of conductive probes (main sensor) and the control board (LM393 IC) as seen in figure 3.1. The main use of the probes is to measure the volumetric water content in the soil and the control board which functions as a voltage comparator (Ravi 2018).

The sensor uses electric current to measure the amount of water in the soil. The amount of water in the soil will cause the conductivity of the probe to vary. If the water content in the soil is low an LED will glow and this will indicate that the output of the control board(comparator) is high. The reverse is true for when the water content of the soil is high.

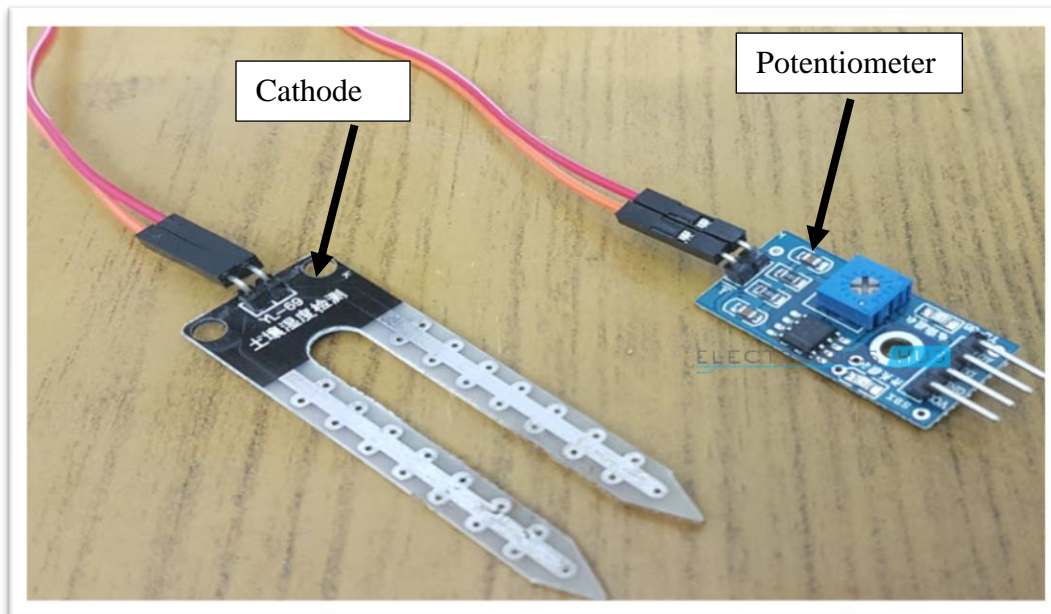


Figure 3. 1 Soil Moisture Sensor

(Source: Ravi (2018))

ii) **DHT22 Temperature Humidity Sensor**

As stated by Last minute engineers this is a special designed low-cost sensor that has the ability of measuring the temperature and humidity of the surrounding environment. Under the hood as seen in figure 3.2 the sensor comprises of a Humidity Sensing Component and NTC Temperature Sensor (Thermistor). Table 3.1 Identifies the specifications of the sensor.

Table 3. 1 Specifications of DHT22 Sensor

<i>Property</i>	<i>Value</i>
<i>Operating Voltage</i>	3 To 5V
<i>Max Operating Current</i>	2.5mA max
<i>Humidity Range</i>	0 - 100% / 2.5%
<i>Temperature Range</i>	-40 to 80°C / ±0.5°C
<i>Body Size</i>	15.5mm x 25mm x 7.7mm
<i>Sensor Period</i>	2s

(Source: Dejan (2016))

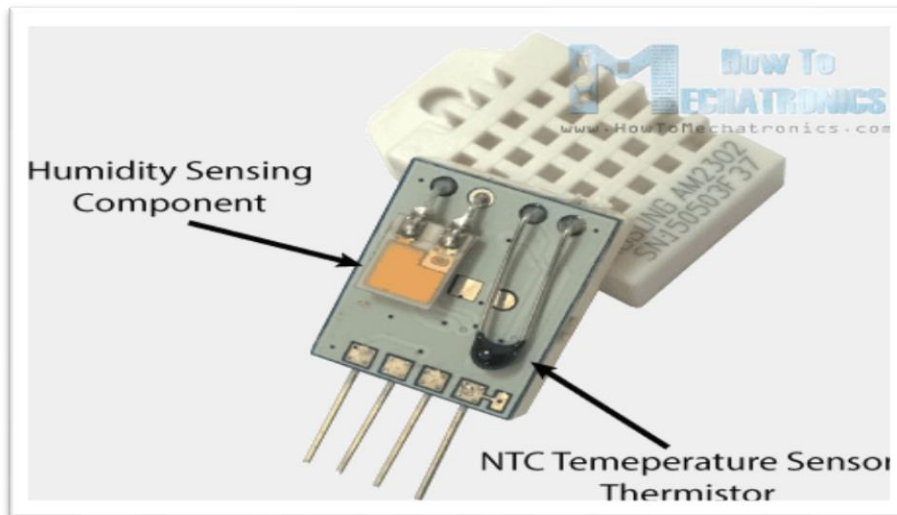


Figure 3. 2 Inside view of a DHT22 sensor

(Source: Dejan (2016))

iii) Water flow Sensor

A water flow sensor consists of a water rotor, half-effect sensor and a plastic body valve. It is used to measure the rate of flow of water in a pipe. It achieves this by using the amount of times the rotor is turned by the water to pulse signal (done by the hall-effector sensor) that determines how much water has flown through the sensor. In this project this sensor will be used to give an accurate representation of the daily water usage of the plants in order to avoid wastage. Figure 3.3 shows the representation of the sensor.



Figure 3. 3 Water Flow Sensor

(Source: Ravi (2018))

3.2.2 Actuators

Taymanov and Sapozhnikova (2018) define an actuator as anything with the ability to change energy so as to produce a form of motion. This energy can be created by many elements such as electricity, liquid, air which in turn causes an action/motion to occur. The actuators in this project are driven by the Arduino microcontroller. The following is a description of the actuators used within the project.

i) **Water Pump**

A water pump can be defined as any device that has the ability to move water. Water pumps have been used since the beginning of time so as to move water through mediums from one region to another. In this project the pump will be controlled by a microcontroller that will give it an electric signal telling it to start moving the water from the water reservoir to the direction of the plants. Figure 3.5 shows the type of pump to be used in the project.



Figure 3. 4 Water Pump

(Source: Ravi (2018))

ii) **Fan**

The main purpose of a fan is cool the environment due to an increase in temperature. In this project a fan will be used to cool down the greenhouse when the temperature

and humidity reach a certain level that is not conducive for the plant. This means that the fans will play a major role in the climate control of the greenhouse.



Figure 3. 5 Cooling Fan

(Source: Ravi (2018))

3.2.3 Arduino Mega 2560

The Arduino Mega 2560 is what we call a microcontroller. Its major role is to accept input from the sensors and perform calculations and then provide output in the form of electric signals that will be interpreted by the actuators that will in turn carry out the relevant task needed of them. The board is based on an ATmega2560 chip. Table 3.2 shows the specifications of the board.

Table 3. 2 Arduino Mega 2560 specifications

<i>Property</i>	<i>Value</i>
<i>Microcontroller</i>	ATmega2560
<i>Operating Voltage</i>	5V
<i>Input Voltage (recommended)</i>	7-12V
<i>Input Voltage (limit)</i>	6-20V
<i>Digital I/O Pins</i>	54 (15 provides PWM output)

<i>Analog Input Pins</i>	16
<i>DC Current per I/O Pin</i>	20 mA
<i>DC Current for 3.3V Pin</i>	50 mA
<i>Flash Memory</i>	256 KB (bootloader uses 8 KB)
<i>SRAM</i>	8 KB
<i>EEPROM</i>	4 KB
<i>Clock Speed</i>	16 MHz
<i>LED_BUILTIN</i>	13
<i>Length</i>	101.52 mm
<i>Width</i>	53.3 mm
<i>Weight</i>	37 g

(Source: SM (2017))

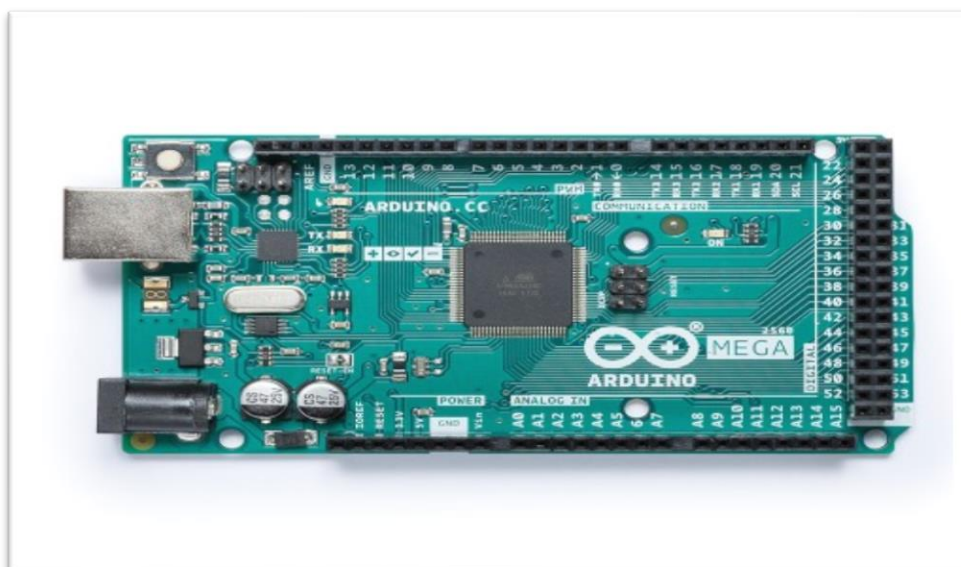


Figure 3. 6 Arduino Mega 2560 board

(Source: SM (2017))

This board is able to interface with other devices like Wi-Fi modules, GSM modules, Ethernet modules and ZigBee that help further boost its capabilities in achieve a true network of sensors that help produce a perfect standalone network of **Things** that will be working together as one unit.

3.2.4 ESP8266 Wi-Fi Module

ESP8266 Wi-Fi module is defined by Aqeel (2018) to being an open source firmware that has been developed to give support to developers who are willing to venture into the realm of I.o.T design and development. It has similar features with the microcontroller mentioned above but with one added advantage. This Wi-Fi module is able to interface with other device over a Wi-Fi connection due to the fact that the Wi-Fi module is based on an ESP8266 Wi-Fi chip. As much as it can work as a standalone device, when a large number of sensors are need to interface with it an Arduino board will come in handy due to the vast number of digital and analogy pins on the board that allow it to interface with much more components than the Wi-Fi module can on its own. Another great advantage of this component is the ability for it to act as an access point. This can allow the Arduino device to be controlled without a serial connection to any server and use a wireless connection which can be accessed by a web server on a pc or even an android device. Figure 3.8 shows what the Wi-Fi module looks like.

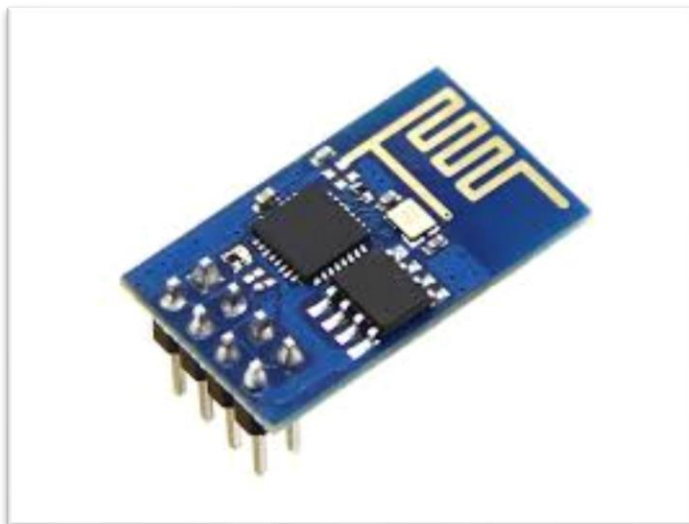


Figure 3. 7 ESP8266 Wi-Fi module

(Source: Espressif (2019))

3.2.5 Four Channel Relay

A relay is described as an electrical switch used in a circuit that has the ability to use a small electric current to control or switch on or off a much larger current than the one that initiated it (Woodford 2018). The small current passes through a coil that in turn will create a magnetic field that will be used to control the switch. A relay has dual roles, as a switch/lever that switches a circuit on or off and also as an amplifier that has the ability to convert small currents to large ones. In the project the relay will be used to control when the cooling fan is switched on due to an increase in temperature within the greenhouse and also when the pump will be started given the level of moisture need by the plant has decreased below a certain threshold level.

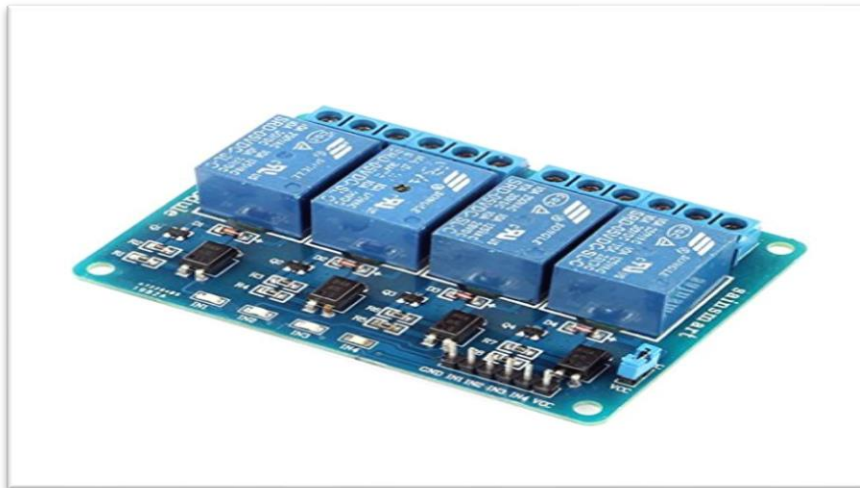


Figure 3. 8 Four Channel relay (Source: Ravi (2018))

3.2.6 ATX Power Supply

The developer will repurpose an ATX power supply that is used by computers to power their motherboards in order to produce a 12v voltage that will be able to power the water pump and the use a L7805 voltage regulator to step down the voltage to 5v in order not to fry the other components.



Figure 3. 9 ATX Power Supply Module

(Source: Electronics Tutorials (2018))

3.3 Software Components of the Project

The software components are the intangible parts that will be used to develop the project. These components include the Integrated Development Environments (IDEs), programming languages that will be used to develop the project.

3.3.1 IDEs

Rouse et al (2018) defines an IDE as an environment that is used to consolidate all the tools that might be needed to develop in a specific programming language. It helps bring all the tools like compilers, text editors, etc. into one environment making it easier for developers to manage these programming aspects. The software needed for this project will be developed using three different IDEs which are namely Arduino IDE, Android Studio and WebStorm.

3.3.2 Programming Language

According Hemmendinger a computer programming language is a high-level language that when translated to its corresponding machine language equivalent acts a set of instructions that allow a computer to perform a specific task. For this project the languages that will be used are Java, C/C++ and JavaScript.

3.4 Conclusion

In conclusion with the vast understanding of these components the development of the project will be made easier and faster. In the next chapter the reader will be introduced to the system and the expected results that will be produced by the system.

Chapter 4: System Design

4.1 Introduction

When an architect is drawing a plan for a house, he/she has to define each and every part of the house in details so as to be able to clearly visualise the idea that is being thought up by the owner of the house. In part system design tries to play that role by defining the elements that make up a system from the user interfaces, flow of data within the system, system architecture. System design implies an approach that is systematic, which enables the developer to cover all the user requirements so as to deliver the best system possible. In the following section a closer look will be given to the components that make up the process of system design.

4.2 Data flow

System design deals with the flow of information that occurs between two different points be it in an organisation between two people or the communication of two servers in different geographical locations. In this system the developer will illustrate the movement of data between the sensors, microcontroller and the android phone which acts as the client. In order to illustrate this flow of data a flow chart should be used, Kimber et al (2018) describes flow charts as a significant tool that can be used to convey information between two parties be it in the classroom or in a business setting. Figure 4.1 illustrates the way the data will flow within the greenhouse, starting from the point that the sensors collect the data that then sets off a chain reaction of events that lead to the end user receiving the information on his/hers tablet or smartphone. In the event of a fault it is easy to follow the flow as the steps of how the data is moving is clearly shown.

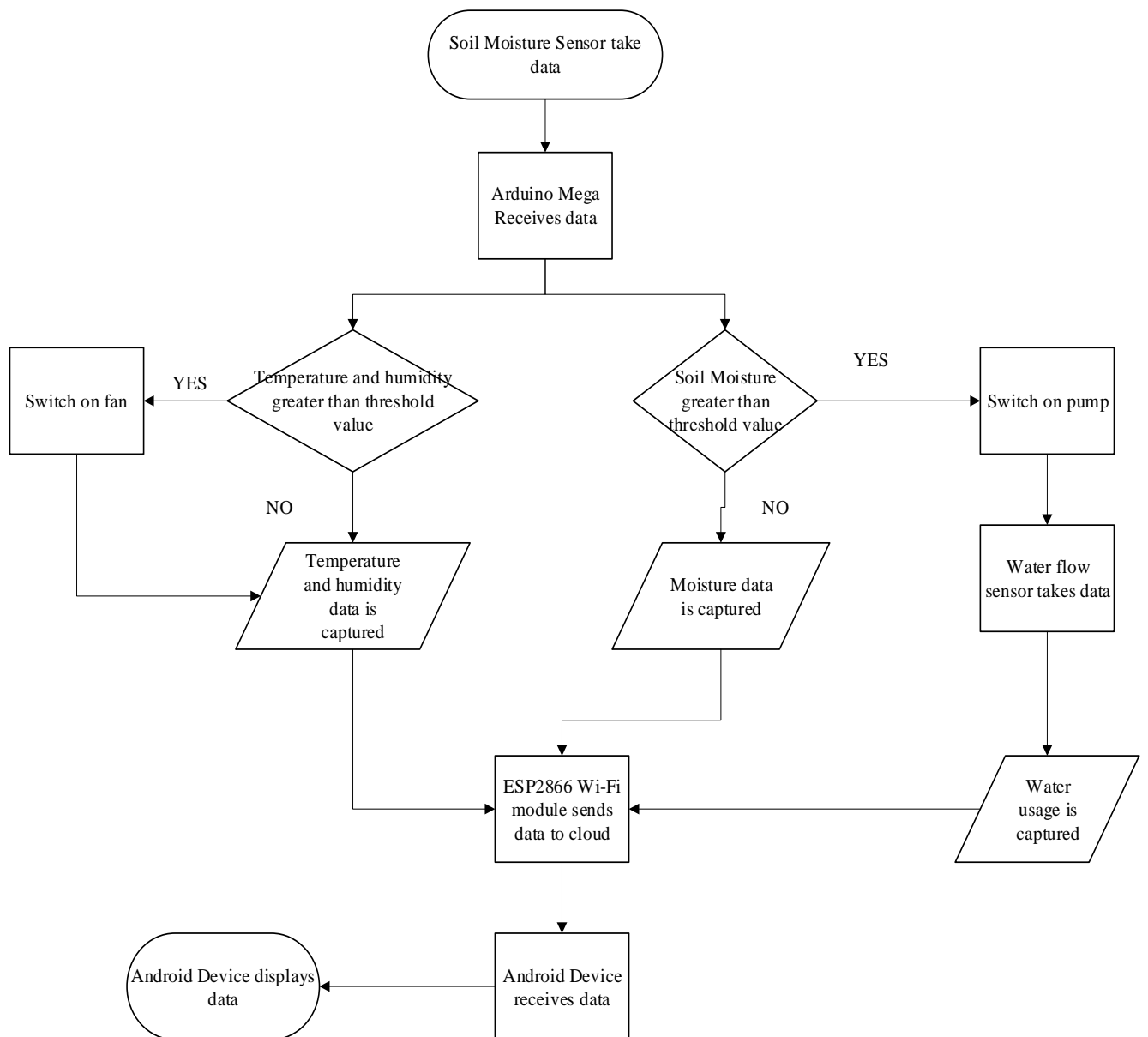


Figure 4. 1 Flow Chart of Greenhouse Project

(Source: Author)

4.3 System architectural design

In order to describe what the system architecture is the reader should have a basic appreciation of what an architecture is. Gough (2013) defines the term architecture as a relative idea used to

design and implement the construction of a building or other physical entity. From this definition the reader can appreciate that a systems architectural design is a blueprint of the interrelationship between all the components be it software or hardware. This relationship is shown

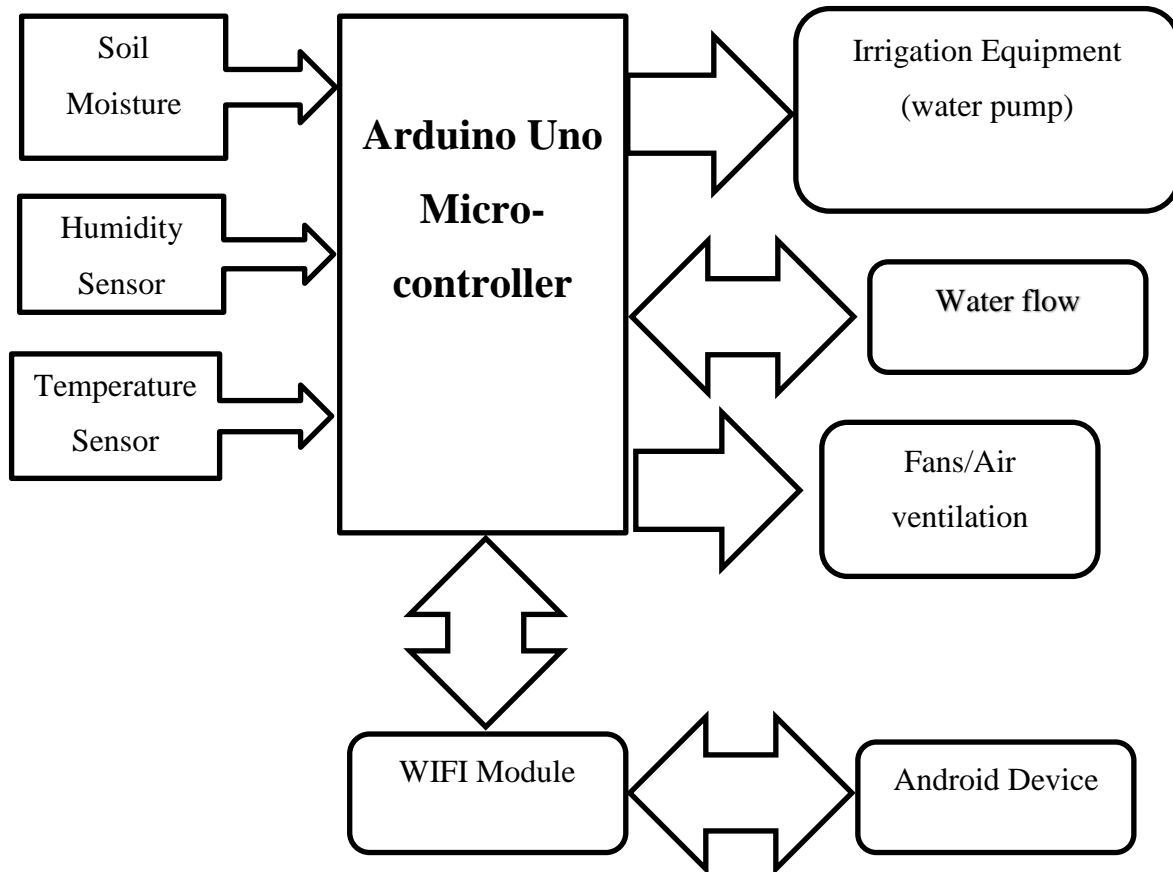


Figure 4. 2 Architectural design of the components (Source: Author)

4.4 Circuit and Schematic design

According to Science daily circuit design is process used to determine the logical pathways that are needed to join electrical components to produce a functional and complete circuit. This results in the development of instruction that can be used to develop the electric circuit. This will further help in the development of a schematic that can be used to develop a Printed Circuit Board (PCB) when one wants to develop a more integrated system. The developer will dwell more on the circuit design done on a bread board for the development of this project. This circuit design is seen in Figure 4.3 which illustrates how each component will be interconnect with each other on a logical level.

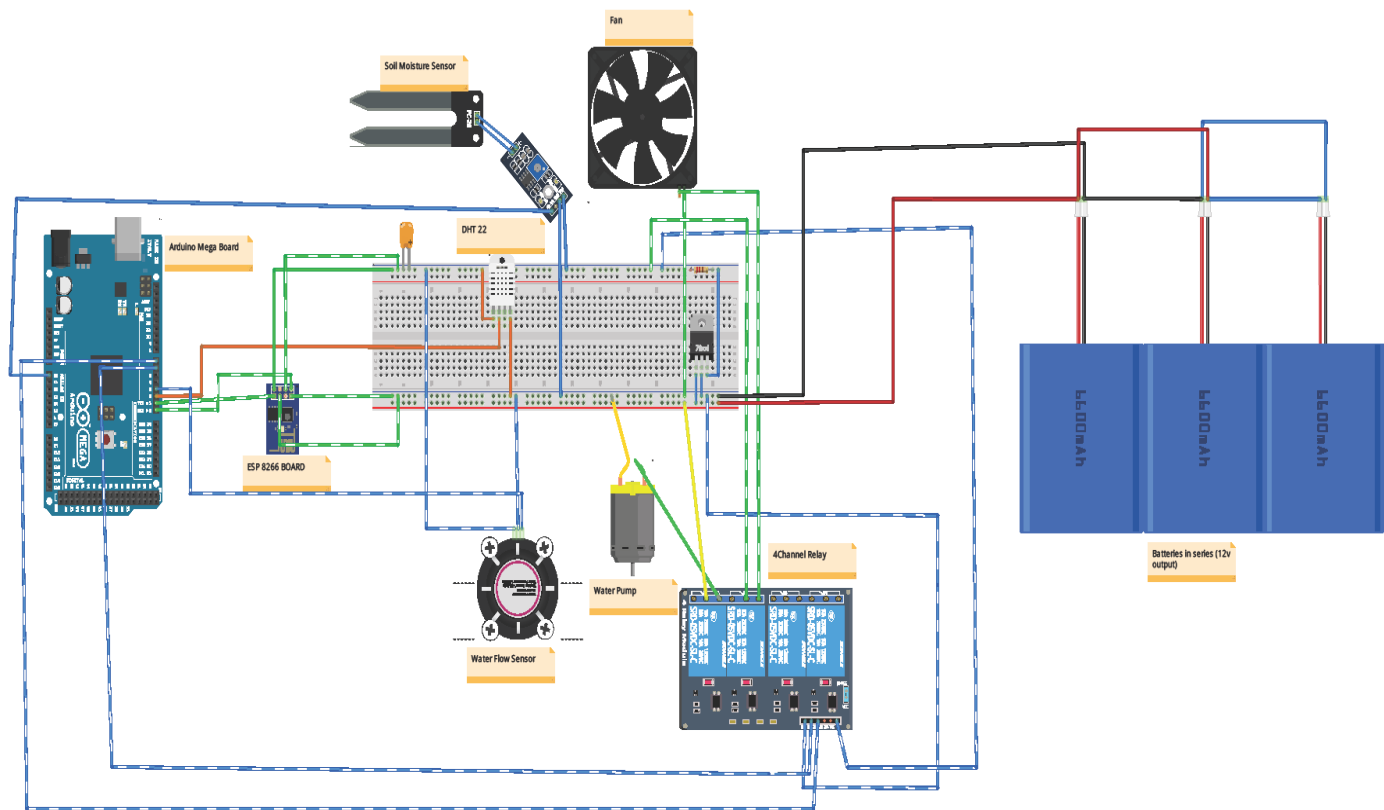


Figure 4. 3 Circuit design for greenhouse components

(Source: Author)

4.5 Physical Design

This section deals with the physical outlook of the greenhouse upon completion and how the I.o.T aspect of the project is achieved. Figure 4.4 illustrates how the proposed system will look after its completion. The ESP8266 Wi-Fi module is represented by the sensor nodes that will communicate with a router and send information to the android device via a server that will store the daily values taken by the sensor and can later be mined to produce relevant data by the use of data mining techniques. These stored values will also be displayed via the android device that will be used by the end user to view the relevant information pertaining the greenhouse i.e. soil moisture values, temperature and humidity values and the amount of water used each time the pump is initiated.

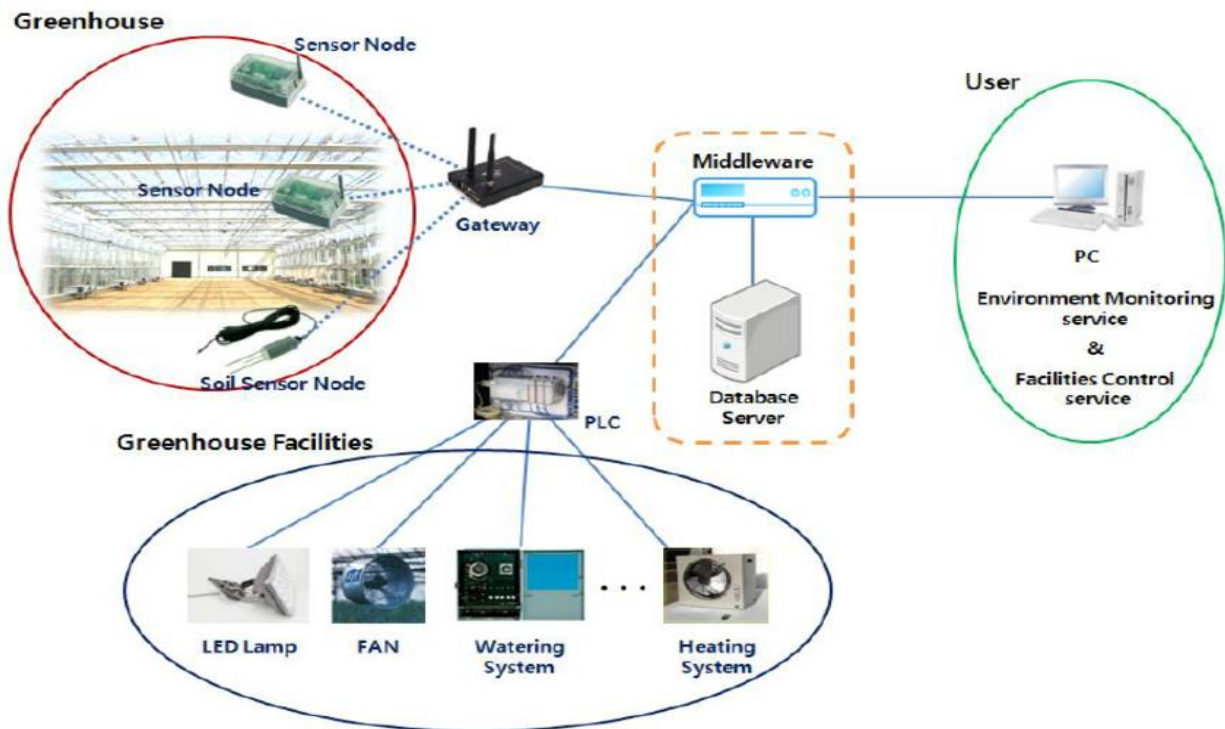


Figure 4. 4 Physical Design of the Greenhouse (Source: Shamshiri (2018))

4.6 Interface design

Interface design deals with the design of the user interface which is the first port of contact between the user and the system. The interface being designed is for the android system where the user will be able to view the values recorded by the sensors and also to view historic data that will be stored in the server such as the average temperature per day, average humidity a week, etc. Figure 4.5 illustrates what the end user will be seeing in terms of the data that he/she will be receiving from the greenhouse. On click each display area the user is sent to another UI shown on Figure 4.6 that will allow the user to choose which region of historical data would like to be viewed.

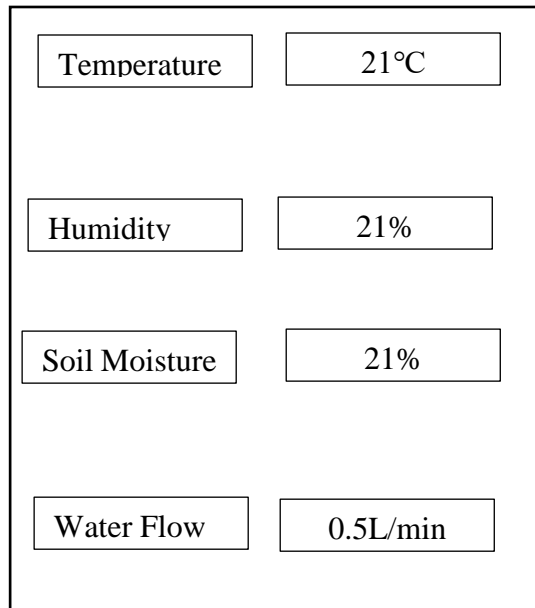


Figure 4. 5 UI for Greenhouse control panel

(Source: Author)

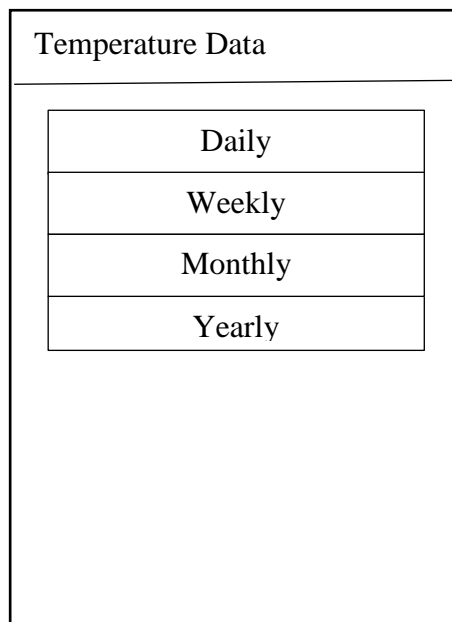


Figure 4. 6 UI for historical data view

(Source: Author)

4.7 Pseudocode

Rouse (2014) defines it as a naturally readable language that is used by designer to describe the implementation they would like to see in a program. After formulation of the pseudocode it is given to programmers who use it as a template to develop the desired system. Following is the pseudocode for the switching on of the fan in the event that the temperature has reached a certain threshold level.

Initialize pins

Declare tempVariable

Initialize maxTemp = 23

Setup Method

Initialize Serial Baud

Start dht sensor

Loop Method

Initialize tempVariable = temperature value from dht sensor

IF maxTemp > tempVariable THEN

 Switch on fan

 Print fan is on

ELSE

 Do not switch on fan

 Print fan is off

Print tempVariable

4.8 Client-Server Architecture

Beal (2019) describes this as a network architecture that depends on the availability of a client and a server. This network architecture is the backbone of the communication between our

client which is the android device and the server which will be implemented in order to handle http requests from the client, storage of data from the sensors connected to the Arduino microcontroller. The server will implement a RESTful API architecture that simplifies the transmission of data between the client and the server by the use of CRUD (Create, Read, Update, Delete) operations.

4.9 Security Design,

Security design is a process by which the security measure is integrated into the software from the ground by including measures such as testing and other safe guards such as authentication. By developing these safeguards at every design stage, the software will have a more effective security policy and help it adhere to the best programming practices. This testing is not only just limited to the development of the software but is a continuous process that goes through the implementation stage and as well as the maintenance stage. In order to gain a more in-depth appreciation the reader will be introduced to three subsections that make up security design. These sections are Physical security, Network security and Operational security.

4.9.1 Physical Security

Physical security deals with the security of the physical aspects and ways of preventing tampering of these elements. The system being developed is comprises of both hardware and software elements. With this realisation a higher effort must be made in maintaining the physical security of the hardware components that are more accessible to compromise either from threats like natural disasters, animals and humans trying to compromise the integrity of the equipment. In order to maintain security of the hardware element of the project, components that are highly susceptible to damage from the elements should be enclosed in a secure case that has both the ability to protect them from water, fire or any disaster and also that can be locked in order to protect it from being tempered with physically.

4.9.2 Network Security

Network Security deals with the security protocols and authentication methods used to authenticate the data that is transferred with in a network. This is done so that data integrity is maintained since data packets can be compromised as they are sent from the client to the server

and vice versa. In this system the data will be transferred from a Wi-Fi module that takes advantage of modern-day Wi-Fi standards such as 802.11n which according to Mitchell (2019) has backward compatibility with previous standard versions and a wide range of devices and at the same time provides a more bandwidth. Due to this data is transferred fast from the Arduino to the server and a Restful API architecture ensures security by the use of handshake tokens used to verify the authenticity of data packets. After data is transferred to the server it is then repackaged and transferred to the android device that then displays the information.

4.9.3 Operational Security

Techopedia (2012) defines operation security as the process of protecting data that if fallen in the hands of the competitor could give the competitor an advantage. This information can be categorized as data or process that happen in an organization. In the greenhouse system these processes can be equated to the processes of transferring the data from the Arduino board to the server. The information might be using an open source API but still could give any competitors.

4.10 Conclusion

In this chapter the reader has been introduced to the term system design and a detailed description has been given on each component that it comprises and how they relate to the task at hand. In the next and final chapter, the reader will be introduced to the results that the system would have generated and a brief description of recommendations that should be take hide of for future development.

Chapter 5: Implementation

5.1 Introduction

This chapter will seek to introduce the implementation phase. According to the SDLC it is the final stage of software development which in other texts is superseded by the maintenance phase which by all stands is a continues phase that is ongoing and doesn't reach completion till the software has been retired (Usane Rani, 2017). The main discussion will be based on how the Greenhouse system will be implemented in terms of the following aspects covered under software implementation which are: coding, system testing, final installation of the system, system maintenance, the results the system will produce, and finally any recommendations that can be implemented in the future development of the system.

5.2 Coding

This is the most important aspect of software development, as if done right project deadlines are easily met and hence the software is developed in time (Ransom, 2018). In this system this section is divided into two, the first being the mobile application and the second being that of the microcontroller.

5.2.1 Mobile application source code

In order to fully develop an I.o.T enabled system a user should be able to access the device from a remote area anywhere in the world. This notion dictates that this system should be controlled from either a laptop or a mobile device. The developer chose to use a mobile device since it is more portable and has easier access. The android development environment was used to develop the control centre which makes use of java programming language and xml for scripting and interface design purposes. Below are the code snippets illustrating the code used to develop the android application.

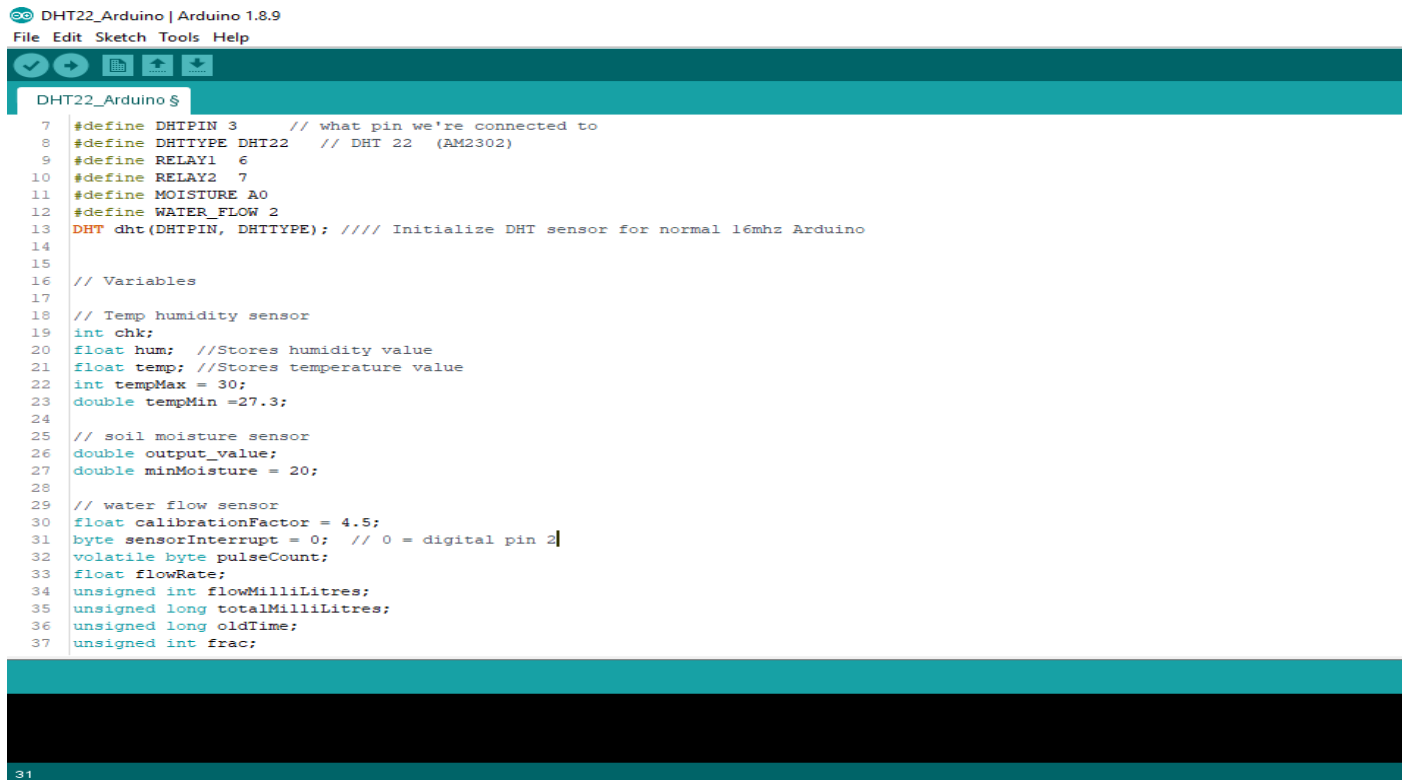

```
128 }
129
130 private void retrieveMenuVales() {
131     final StringRequest stringRequest = new StringRequest(Request.Method.GET, AppConfig.URL_GET_MAIN_MENU, response -> {
132         try {
133             JSONArray jsonArray = new JSONArray(response);
134
135             for (int i = 0; i < jsonArray.length(); i++) {
136                 JSONObject index = jsonArray.getJSONObject(i);
137                 String temperature = index.getString("name: temperature");
138                 String humidity = index.getString("name: humidity");
139                 String water_flow = index.getString("name: water_flow");
140                 String moisture = index.getString("name: moisture");
141
142                 String temp = "\u2103"; //tempSymbol;
143                 String humiditySymbol = "%";
144                 String moistureSymbol = "%";
145                 String waterFlow = "L/min";
146
147                 tempValue.setText(String.format("%s %s", temperature, temp));
148                 humidityValue.setText(String.format("%s %s", humidity, humiditySymbol));
149                 moistureValue.setText(String.format("%s %s", moisture, moistureSymbol));
150                 waterFlowValue.setText(String.format("%s %s", water_flow, waterFlow));
151             }
152         }
153     });
154
155     } catch (JSONException e) {
156         e.printStackTrace();
157     }
158
159     }, error -> {
160         if (error instanceof TimeoutError) {
161             // Handle timeout error
162         }
163     });
164
165     }
166
167     }
168
169     }
170
171     }
172
173     }
174
175     }
176
177     }
178
179     }
180
181     }
182
183     }
184
185     }
186
187     }
188
189     }
190
191     }
192
193     }
194
195     }
196
197     }
198
199     }
200
201     }
202
203     }
204
205     }
206
207     }
208
209     }
210
211     }
212
213     }
214
215     }
216
217     }
218
219     }
220
221     }
222
223     }
224
225     }
226
227     }
228
229     }
230
231     }
232
233     }
234
235     }
236
237     }
238
239     }
240
241     }
242
243     }
244
245     }
246
247     }
248
249     }
250
251     }
252
253     }
254
255     }
256
257     }
258
259     }
260
261     }
262
263     }
264
265     }
266
267     }
268
269     }
270
271     }
272
273     }
274
275     }
276
277     }
278
279     }
280
281     }
282
283     }
284
285     }
286
287     }
288
289     }
290
291     }
292
293     }
294
295     }
296
297     }
298
299     }
300
301     }
302
303     }
304
305     }
306
307     }
308
309     }
310
311     }
312
313     }
314
315     }
316
317     }
318
319     }
320
321     }
322
323     }
324
325     }
326
327     }
328
329     }
330
331     }
332
333     }
334
335     }
336
337     }
338
339     }
340
341     }
342
343     }
344
345     }
346
347     }
348
349     }
350
351     }
352
353     }
354
355     }
356
357     }
358
359     }
360
361     }
362
363     }
364
365     }
366
367     }
368
369     }
370
371     }
372
373     }
374
375     }
376
377     }
378
379     }
380
381     }
382
383     }
384
385     }
386
387     }
388
389     }
390
391     }
392
393     }
394
395     }
396
397     }
398
399     }
400
401     }
402
403     }
404
405     }
406
407     }
408
409     }
410
411     }
412
413     }
414
415     }
416
417     }
418
419     }
420
421     }
422
423     }
424
425     }
426
427     }
428
429     }
430
431     }
432
433     }
434
435     }
436
437     }
438
439     }
440
441     }
442
443     }
444
445     }
446
447     }
448
449     }
450
451     }
452
453     }
454
455     }
456
457     }
458
459     }
460
461     }
462
463     }
464
465     }
466
467     }
468
469     }
470
471     }
472
473     }
474
475     }
476
477     }
478
479     }
480
481     }
482
483     }
484
485     }
486
487     }
488
489     }
490
491     }
492
493     }
494
495     }
496
497     }
498
499     }
500
501     }
502
503     }
504
505     }
506
507     }
508
509     }
510
511     }
512
513     }
514
515     }
516
517     }
518
519     }
520
521     }
522
523     }
524
525     }
526
527     }
528
529     }
530
531     }
532
533     }
534
535     }
536
537     }
538
539     }
540
541     }
542
543     }
544
545     }
546
547     }
548
549     }
550
551     }
552
553     }
554
555     }
556
557     }
558
559     }
560
561     }
562
563     }
564
565     }
566
567     }
568
569     }
569
570     }
571
572     }
573
574     }
575
576     }
577
578     }
579
580     }
581
582     }
583
584     }
585
586     }
587
588     }
589
590     }
591
592     }
593
594     }
595
596     }
597
598     }
599
600     }
601
602     }
603
604     }
605
606     }
607
608     }
609
610     }
611
612     }
613
614     }
615
616     }
617
618     }
619
620     }
621
622     }
623
624     }
625
626     }
627
628     }
629
630     }
631
632     }
633
634     }
635
636     }
637
638     }
639
640     }
641
642     }
643
644     }
645
646     }
647
648     }
649
650     }
651
652     }
653
654     }
655
656     }
657
658     }
659
660     }
661
662     }
663
664     }
665
666     }
667
668     }
669
670     }
671
672     }
673
674     }
675
676     }
677
678     }
679
680     }
681
682     }
683
684     }
685
686     }
687
688     }
689
690     }
691
692     }
693
694     }
695
696     }
697
698     }
699
700     }
701
702     }
703
704     }
705
706     }
707
708     }
709
710     }
711
712     }
713
714     }
715
716     }
717
718     }
719
720     }
721
722     }
723
724     }
725
726     }
727
728     }
729
730     }
731
732     }
733
734     }
735
736     }
737
738     }
739
740     }
741
742     }
743
744     }
745
746     }
747
748     }
749
750     }
751
752     }
753
754     }
755
756     }
757
758     }
759
760     }
761
762     }
763
764     }
765
766     }
767
768     }
769
770     }
771
772     }
773
774     }
775
776     }
777
778     }
779
780     }
781
782     }
783
784     }
785
786     }
787
788     }
789
790     }
791
792     }
793
794     }
795
796     }
797
798     }
799
800     }
801
802     }
803
804     }
805
806     }
807
808     }
809
810     }
811
812     }
813
814     }
815
816     }
817
818     }
819
820     }
821
822     }
823
824     }
825
826     }
827
828     }
829
830     }
831
832     }
833
834     }
835
836     }
837
838     }
839
840     }
841
842     }
843
844     }
845
846     }
847
848     }
849
850     }
851
852     }
853
854     }
855
856     }
857
858     }
859
860     }
861
862     }
863
864     }
865
866     }
867
868     }
869
870     }
871
872     }
873
874     }
875
876     }
877
878     }
879
880     }
881
882     }
883
884     }
885
886     }
887
888     }
889
890     }
891
892     }
893
894     }
895
896     }
897
898     }
899
900     }
901
902     }
903
904     }
905
906     }
907
908     }
909
910     }
911
912     }
913
914     }
915
916     }
917
918     }
919
920     }
921
922     }
923
924     }
925
926     }
927
928     }
929
930     }
931
932     }
933
934     }
935
936     }
937
938     }
939
940     }
941
942     }
943
944     }
945
946     }
947
948     }
949
950     }
951
952     }
953
954     }
955
956     }
957
958     }
959
960     }
961
962     }
963
964     }
965
966     }
967
968     }
969
970     }
971
972     }
973
974     }
975
976     }
977
978     }
979
980     }
981
982     }
983
984     }
985
986     }
987
988     }
989
990     }
991
992     }
993
994     }
995
996     }
997
998     }
999
1000    }
```

Figure 5. 3 Third Android code snippet

(Source: Author)

5.2.2 Microcontroller source code

The ability for the greenhouse system to control the sensors, receive and send data to the android device is through the Arduino Mega board. This is can be termed the brain of the system. This board is a programmable board which makes use of the Arduino IDE to produce and upload the code into the board. The code is developed in the C programming language. Below are some of the code snippets of the source code that will be used to control the sensors.

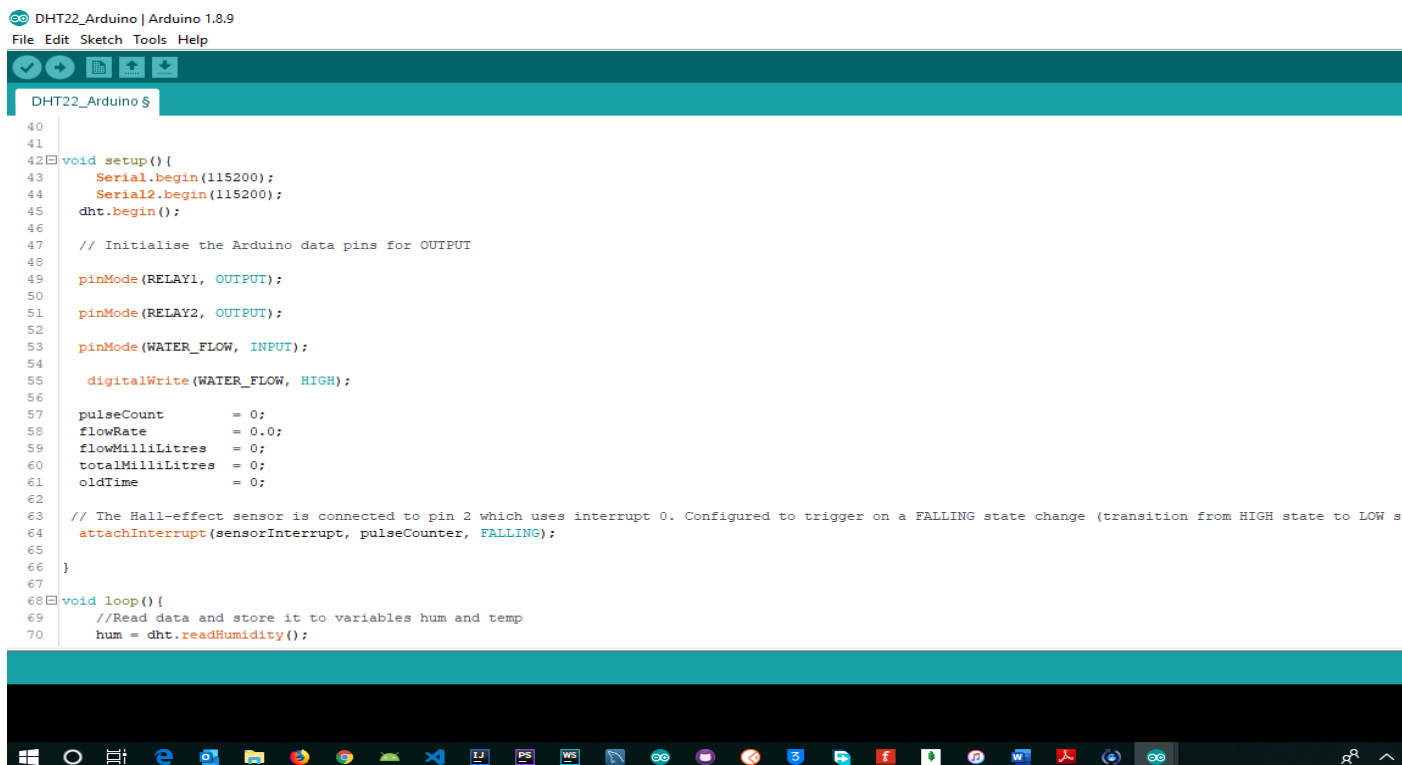


```
DHT22_Arduino | Arduino 1.8.9
File Edit Sketch Tools Help

DHT22_Arduino $
7 #define DHTPIN 3 // what pin we're connected to
8 #define DHITYPE DHT22 // DHT 22 (AM2302)
9 #define RELAY1 6
10 #define RELAY2 7
11 #define MOISTURE A0
12 #define WATER_FLOW 2
13 DHT dht(DHTPIN, DHITYPE); //// Initialize DHT sensor for normal 16mhz Arduino
14
15
16 // Variables
17
18 // Temp humidity sensor
19 int chk;
20 float hum; //Stores humidity value
21 float temp; //Stores temperature value
22 int tempMax = 30;
23 double tempMin =27.3;
24
25 // soil moisture sensor
26 double output_value;
27 double minMoisture = 20;
28
29 // water flow sensor
30 float calibrationFactor = 4.5;
31 byte sensorInterrupt = 0; // 0 = digital pin 2
32 volatile byte pulseCount;
33 float flowRate;
34 unsigned int flowMilliLitres;
35 unsigned long totalMilliLitres;
36 unsigned long oldTime;
37 unsigned int frac;
```

Figure 5. 4 First Arduino code snippet

(Source: Author)



```
DHT22_Arduino | Arduino 1.8.9
File Edit Sketch Tools Help

DHT22_Arduino $
40
41
42 void setup(){
43     Serial.begin(115200);
44     Serial2.begin(115200);
45     dht.begin();
46
47     // Initialise the Arduino data pins for OUTPUT
48
49     pinMode(RELAY1, OUTPUT);
50
51     pinMode(RELAY2, OUTPUT);
52
53     pinMode(WATER_FLOW, INPUT);
54
55     digitalWrite(WATER_FLOW, HIGH);
56
57     pulseCount = 0;
58     flowRate = 0.0;
59     flowMilliLitres = 0;
60     totalMilliLitres = 0;
61     oldTime = 0;
62
63     // The Hall-effect sensor is connected to pin 2 which uses interrupt 0. Configured to trigger on a FALLING state change (transition from HIGH state to LOW state)
64     attachInterrupt(sensorInterrupt, pulseCounter, FALLING);
65
66 }
67
68 void loop(){
69     //Read data and store it to variables hum and temp
70     hum = dht.readHumidity();
```

Figure 5. 5 Second Arduino code snippet

(Source: Author)

```

DHT22_Arduino | Arduino 1.8.9
File Edit Sketch Tools Help
DHT22_Arduino $
97     digitalWrite(RELAY2, HIGH); // Turns Relay Off
98     }
99
100    // //Print temp and humidity values to serial monitor
101    // Serial.print("Humidity: ");
102    // Serial.print(hum);
103    // Serial.print("%,");
104    Serial.print(" Temp: ");
105    Serial.print(temp);
106    Serial.println(" Celsius");
107    // // Print Soil Moisture
108    // Serial.print("Moisture : ");
109    // Serial.print(output_value);
110    // Serial.println("%");
111    delay(2000); //Delay 2 sec.
112
113    if (millis() - oldTime > 1000){ // Only process counters once per second
114        // Disable the interrupt while calculating flow rate and sending the value to the host
115        detachInterrupt(sensorInterrupt);
116
117        // Because this loop may not complete in exactly 1 second intervals we calculate the number of milliseconds that have passed since the last execution
118        // that to scale the output. We also apply the calibrationFactor to scale the output based on the number of pulses per second per units of measure (1
119        // coming from the sensor.
120        flowRate = ((1000.0 / (millis() - oldTime)) * pulseCount) / calibrationFactor;
121
122        // Note the time this processing pass was executed. Note that because we've disabled interrupts the millis() function won't actually be incrementing
123        // at this point, but it will still return the value it was set to just before interrupts went away.
124        oldTime = millis();
125
126        // Divide the flow rate in litres/minute by 60 to determine how many litres have passed through the sensor in this 1 second interval, then multiply b
127        // to millilitres.

```

Figure 5. 6 Third Arduino code snippet

(Source: Author)

```

DHT22_Arduino | Arduino 1.8.9
File Edit Sketch Tools Help
DHT22_Arduino $
132
133    // Print the flow rate for this second in litres / minute
134    Serial.print("Flow rate: ");
135    Serial.print(int(flowRate)); // Print the integer part of the variable
136    Serial.print("."); // Print the decimal point
137    // Determine the fractional part. The 10 multiplier gives us 1 decimal place.
138    frac = (flowRate - int(flowRate)) * 10;
139    Serial.print(frac, DEC); // Print the fractional part of the variable
140    Serial.print("L/min");
141    // Print the number of litres flowed in this second
142    Serial.print(" Current Liquid Flowing: "); // Output separator
143    Serial.print(flowMilliLitres);
144    Serial.print("mL/Sec");
145
146    // Print the cumulative total of litres flowed since starting
147    Serial.print(" Output Liquid Quantity: "); // Output separator
148    Serial.print(totalMilliLitres);
149    Serial.println("mL");
150
151    // Reset the pulse counter so we can start incrementing again
152    pulseCount = 0;
153
154    // Enable the interrupt again now that we've finished sending output
155    attachInterrupt(sensorInterrupt, pulseCounter, FALLING);
156 }
157 }
158
159 void pulseCounter(){
160     // Increment the pulse counter
161     pulseCount++;
162 }

```

Figure 5. 7 Fourth Arduino code snippet

(Source: Author)

5.3 Software testing

According to Rajkumar (2017) it is the process of evaluating a software against the stated user requirements and furthermore ensuring the software is not defected and free of bug or anything that can hinder its peak performance. The definition giving clearly illustrates that this is very important process in the development life cycle as the developer compares the requirements that were set and the actual functionality of the system. In order to ascertain this a quick look on the varies aspects of testing shall be done for both the android application and the Arduino microcontroller.

5.3.1 Android Application testing

In the development of the greenhouse system a local server was used in order to speed up the testing phase. During this testing the application was tested whether it could receive data using a Wi-Fi network since the server was localized different mobile networks could not be used. When a connection is a stablished the system will show values from the server as shown in figure 5.8

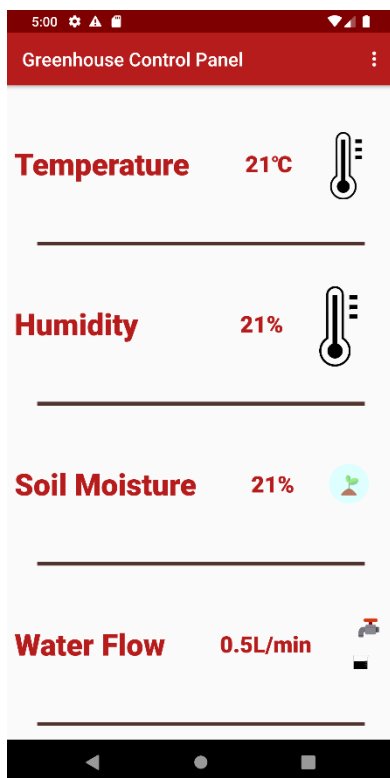


Figure 5. 8 Application interface when server is connected

(Source: Author)

In the event there is no connection between the server and the application the following will be seen as seen in Figure 5.9

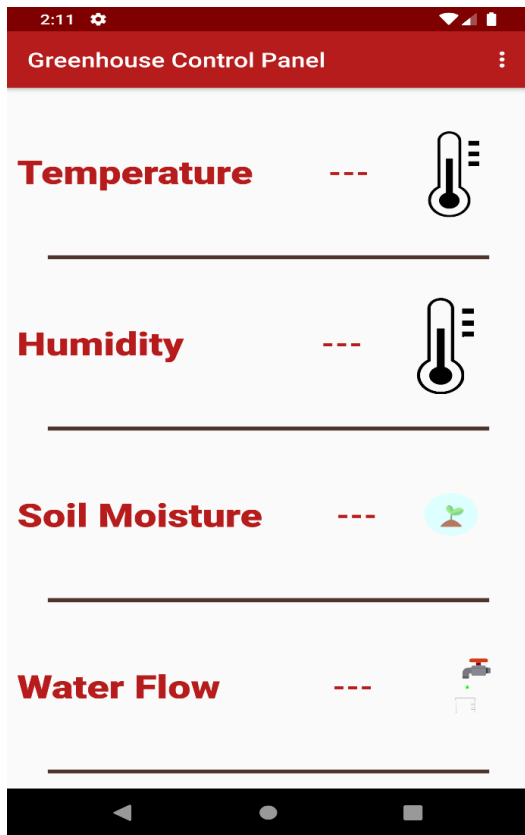


Figure 5. 9 Application interface when server is disconnected (Source: Author)

5.3.2 Microcontroller testing

Testing the microcontroller will deal with testing the ESP8266 module for connectivity to the network that will allow it to send data to the server and also the values produced by the sensors. Figure 5.10 will illustrate the values coming from the sensors as viewed through the Arduino serial monitor.

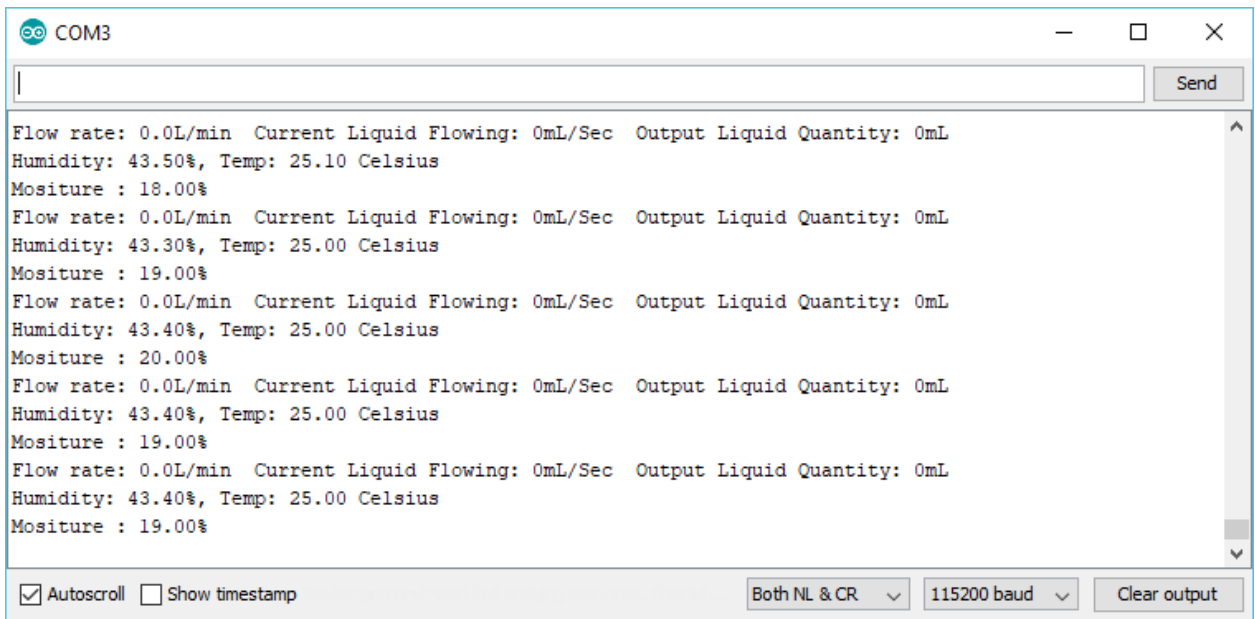


Figure 5. 10 Sensor values from serial monitor

(Source: Author)

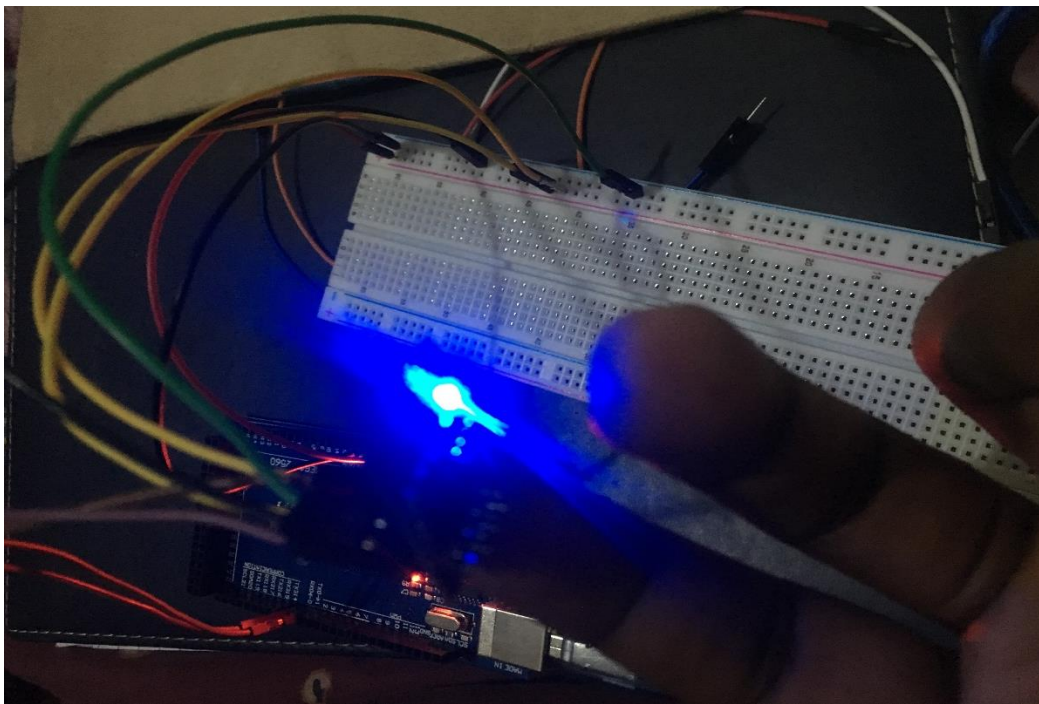


Figure 5. 11 Esp8266 blue light indicating activity

(Source: Author)

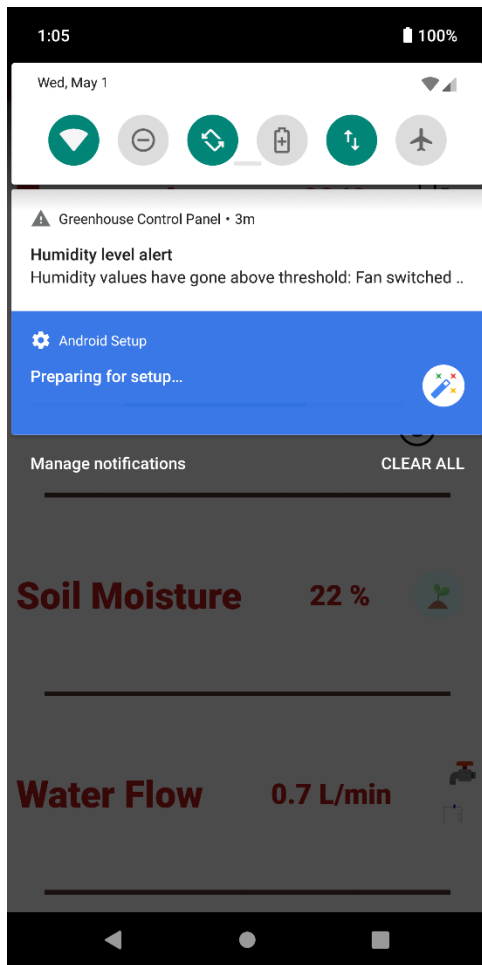


Figure 5. 13 Humidity level alert

(Source: Author)

5.4.2 Objective two

The other objective to be viewed is that of the viewing the fluctuation of sensor values using the greenhouse control interface. Figure 5.14 illustrates this.

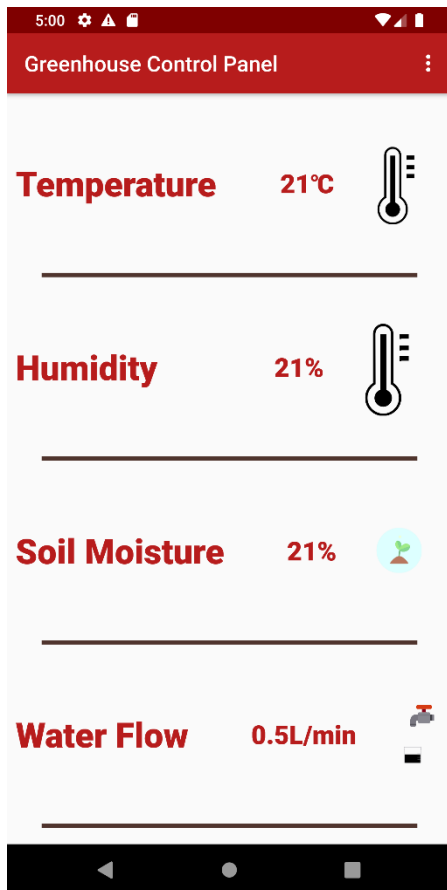


Figure 5. 14 Values from sensors

(Source: Author)

5.4.3 Objective three

The system should also have the ability to show the user historical data that has been collected throughout a certain time frame depending on the chosen time frames that are group has daily, weekly, monthly and yearly. Figure 5.15 shows how these historical values will be viewed.

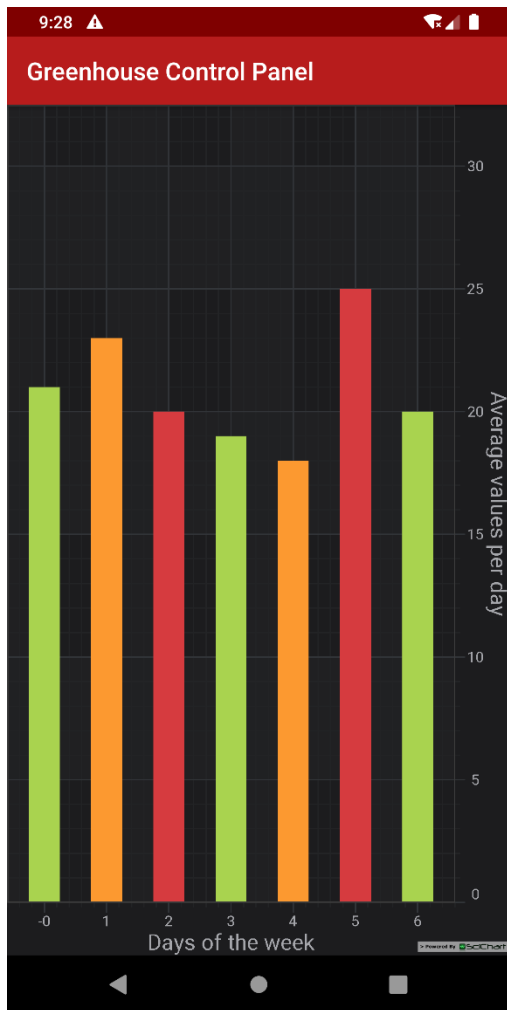


Figure 5. 15 Historical values sensor data

(Source: Author)

5.5 Objectives vs System

Objectives vs design is the comparison between the requirements set by the user and the actual system itself after it has been implemented. The users are supposed to see if the objectives set at the beginning have been met in order to declare that the project was a success. When view the results being produced by the system in the results section on views that all the objectives have been met since the hardware component is only acting the role of collecting data which will then be displayed on the android application.

5.6 Installation

After testing the software needs to be installed in the environment of use. This process deals with preparing both the software and hardware aspects of the system for use. It also deals with

training the users on how the system shall be used and any changeover strategies that could be implemented from the old system to the new one.

5.6.1 Implementation Strategies

Implementation strategies also known as changeover strategies are methods used to migrate from an existing to the new system whilst trying to mitigate any risks factors that can disturb the normal flow of business (Hardeep, 2013). They are four main strategies used in software development and these strategies are direct changeover, parallel changeover, phased change over and pilot changeover. Each of these strategies has its pros and cons over the other, but the develop chose direct changeover since the existing system is more manual, there is no risk to the slowing down of the operation of any specific greenhouse. Direct changeover also allows a seamless introduction of the new system. Implementation of this changeover strategy is effortless since there is no system to replace.

5.6.2 User training

User training is a process of making users familiar with the system being implemented. This is done through workshops were users are given literature such as training manuals and video tutorials that will help them familiarize with the system. For the greenhouse system the training is not extensive since as soon as the greenhouse system has gone online the user just needs to open the android application and view the variables of the sensors in the greenhouse.

5.7 System Maintenance

Software maintenance is a periodic process that deals with correcting, modifying, and updating of a software to suit the needs of the customer or the change in environment or technology used to develop that software. Software Maintenance is rooted from four main philosophies which are: corrective maintenance, adaptive maintenance, perfective maintenance and lastly preventive maintenance.

5.7.1 Corrective Maintenance

Corrective maintenance as the name suggests deals with correcting any flaws or bugs that could arise in the daily use of the system. Measure should be taken in order to correct these flaws. The flaws can range from hardware malfunction, performance issues i.e. a delay in readings due to the Wi-Fi connection or even a glitch in the android application.

5.7.2 Perfective Maintenance

This aspect deals with ever changing user requirements. As the system continues being used the user might find that he or she might want to add extra functionality such as to use a heating system in the greenhouse. Applying this new functionality falls under perfective maintenance.

5.7.3 Adaptive Maintenance

This aspect deals with the way the system will adapt to changes in technology and the ability to use the system on multiple platforms. A practical example is transferring the system from using an Arduino microcontroller to using a raspberry pie microcontroller which uses a different system architecture all together that that offered by Arduino.

5.7.4 Preventive Maintenance

This type of maintenance deals with apply changes and modification that might not have a huge significance in the present but will have a huge one in the future. An example can be given where by instead of using the bread board one can use a more integrated circuit that is able to have all the components on one spot.

5.8 Recommendations for future development

The developmental life cycle of a system is not a static process but one that is dynamic and is continuous due to improvements in technology and also availability of resources. Below are the recommendations given by the developer that he thinks should be implemented to improve the system in the future.

1. Since this project deals with large amounts of data in future a raspberry pie board should be used since it makes use of a python operating system. This will greatly increase the robustness of the system as python has a vast number of tools that can handle data analysis, and predictive analytics of the data from the sensors.
2. During development the developer had problems interfacing the Arduino board with the esp8266. In future a better Wi-Fi module can be used or an integrated board with Wi-Fi capabilities can be used.
3. Use of a PCB (printed circuit board) should be used. This reduces the amount of wires seen and avoids any issues with loose connections between components.

5.9 Conclusion

To bring this chapter to a close it is worth mentioning that the implementation stage is the most important aspect of the development of a software product and selection of the right procedures and strategies will determine the success and failure of even a well-developed software product.

References

Abdul A. I., Hasan M. H., Ismail M. J., Mehat M., and Haroon N. S., “Remote Monitoring in Agricultural Greenhouse Using Wireless Sensor and Short Message Service (SMS)”, In Proceedings of International Journal of Engineering & Technology IJET-IJENS, Volume 9, No 9, PP 1-9, October 2009.

Arduino Software (IDE), Viewed October 5, 2018, <https://www.arduino.cc/en/Guide/Environment>.

Aqeel, A. (2018), “Introduction to NodeMCU V3: A brief tutorial on the Introduction to NodeMCU V3”, The Engineering Project, viewed 19 March 2019, <https://www.theengineeringprojects.com/2018/10/introduction-to-nodemcu-v3.html>

Barn, M. A. (2008). Guidelines of Writing Research Proposals and Dissertations. University of South Dakota. Vermillion.

Beal, V., (2019), “client-server architecture”, webopedia, viewed 20 April 2019, https://www.webopedia.com/TERM/C/client_server_architecture.html

Bostock, J. and Riley, H. T. (1856). Natural History: Vegetables of a cartilaginous nature – cucumbers. Pepones. London, England: Henry G. Bohn, vol. 4, book 19, chapter 23.

Cooper, M. (2015), ‘Project Scope’, viewed October 8, 2018, <https://searchcio.techtarget.com/definition/project-scope>.

Dejan, (2016), “DHT11 & DHT22 Sensors Temperature and Humidity Tutorial using Arduino”, How to Mechatronics, viewed 20 March 2019, <https://howtomechatronics.com/tutorials/arduino/dht11-dht22-sensors-temperature-and-humidity-tutorial-using-arduino/>

Dishant, J. (2017). Internet of things IoT in Agriculture. Viewed May 13, 2018, <https://www.slideshare.net/DishantJames/internet-of-things-iot-in-agriculture>.

Electronics Tutorials, (2018), “Convert ATX PSU to Bench Supply”, Electronics Tutorials, 01 April 2019, <https://www.electronics-tutorials.ws/blog/convert-atx-psu-to-bench-supply.html>

Espressif, (2019), “ESP8266”, Espressif, viewed 20 March 2019, <https://www.espressif.com/en/products/hardware/esp8266ex/overview>

Gough, T., (2013), "Reception Theory of Architecture: Its Pre-History and Afterlife, Architectural Theory Review", Architectural Theory Review, vol 18, issue 3, p279-292, DOI: 10.1080/13264826.2013.889645

Hardeep, S., (2013), "Designing, Engineering, and Analysing Reliable and Efficient Software", IGI Global, Pennsylvania.

Hemmeldiger, D., "Computer programming language", Encyclopedia Britannica, viewed 21 March 2019, <https://www.britannica.com/technology/computer-programming-language#accordion-article-history>

Kimber, O., Cromley, J. G., and Molnar-Kimber, K. L., (2018), "Let Your Ideas Flow: Using Flowcharts to Convey Methods and Implications of the Results in Laboratory Exercises, Articles, Posters, and Slide Presentations", J. Microbiol. Biol. Educ., vol. 19 no. 1.

Last Minute engineers, "How DHT11 DHT22 Sensors Work & Interface with Arduino", Last Minute engineers, viewed 19 March 2019, <https://lastminuteengineers.com/electronics/arduino-projects/>

Manske, S. E. (2018). Greenhouses in Zimbabwe: What you need to know before you buy, viewed November 18, 2018, <http://www.emergingfarmer.com/2018/05/greenhouses-in-zimbabwe-what-you-need.html>

Mitchell, B., (2019), "802.11 Standards Explained: 802.11ac, 802.11b/g/n, 802.11a", Lifewire, view 27 April 2019, <https://www.lifewire.com/wireless-standards-802-11a-802-11b-g-n-and-802-11ac-816553>

"greenhouse". Oxford English Dictionary (3rd ed.). Oxford University Press. September 2005.

Rajkumar, (2017), "What Is Software Testing – Definition, Types, Methods, Approaches", Software Testing Material, viewed 1 May 2019, <https://www.softwaretestingmaterial.com/software-testing/>

Ransom, N., (2018), Implementation & Coding Phase in SDLC, 01/24/2018, online video, viewed 1 May 2019, <https://study.com/academy/lesson/implementation-coding-phase-in-sdlc.html#transcriptHeader>.

Ravi (2017), Different Types of Sensors, Electronics Hub, viewed March 15, 2019, <https://www.electronicshub.org/different-types-sensors/>

Ravi (2018), Interfacing Soil Moisture Sensor with Arduino, Electronics Hub, viewed March 15, 2019, <https://www.electronicshub.org/interfacing-soil-moisture-sensor-with-arduino/>

Ravi, (2018), “Arduino Water Flow Sensor Interface – Hookup Guide & Tutorial”, Electronics Hub, viewed 20 March 2019, <https://www.electronicshub.org/arduino-water-flow-sensor-interface/>

Rouse, M., (2014), “Definition pseudocode”, TechTarget, viewed 20 April 2019, <https://whatis.techtarget.com/definition/pseudocode>

Rouse, M., Gillis, A., Silverthorne, V. (2018), “Integrated development environment (IDE)”, TechTarget, viewed 20 March 2019, <https://searchsoftwarequality.techtarget.com/definition/integrated-development-environment>

Shamshiri, R.R., Kalantari, F., Ting, K. C., Thorp K., Hameed, I. A., Weltzien, C., Ahmad, D., Shad, Z., (2018), “Advances in greenhouse automation and controlled environment agriculture: A transition to plant factories and urban agriculture”, ResearchGate, viewed 5 May 2019, https://www.researchgate.net/publication/322834975_Advances_in_greenhouse_automation_and_controlled_environment_agriculture_A_transition_to_plant_factories_and_urban_agriculture

Simon, M. K. (2011). Dissertation and scholarly research: Recipes for success (2011 Ed.). Seattle, WA, Dissertation Success, LLC.

SM, (2017), “Getting Started with Arduino and Genuino MEGA2560”, Arduino, viewed 20 March 2019, <https://www.arduino.cc/en/Guide/ArduinoMega2560>

Stipanicev, D. and Marasovic, J. (2010), “Networked Embedded Greenhouse Monitoring and Control”, In Proceedings of IEEE Conference on Control Applications (CCA) 23-25 June 2003, Vol. 2, PP.1350 – 1355.

Taymanov, R., Sapozhnikova, K. (2018), “What makes sensor devices and microsystems ‘intelligent’ or ‘smart’?”, Smart Sensors and MEMs, 2nd edn, Woodhead Publishing, pages 1-22.

Techopedia, (2012), “Operations Security (OPSEC)”, Techopedia, viewed 5 May 2019, <https://www.techopedia.com/definition/24144/operations-security-opsec>

Usha Rani, S. B. (2017). A detailed study of Software Development Life Cycle (SDLC) Models. *International Journal of Engineering and Computer Science*, 6(7). Viewed 1 May 2019, <http://www.ijecs.in/index.php/ijecs/article/view/2830>

Vaportzis, E., Clausen, M. G., and Gow, A. J. (2017). Older Adults Perceptions of Technology and Barriers to Interacting with Tablet Computers: A Focus Group Study, The Dunhill Medical Trust, viewed November 18, 2018, <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5649151/>.

Woodford, C. (2018), “Relays”, Explain that stuff, viewed 20 March 2019, <https://www.explainthatstuff.com/howrelayswork.html>

Xi-shan, G., Xiang-long, Y., Li-ren, W., Qian Z., and Yiming, Z., (2010) “A Wireless Solution for Greenhouse Monitoring and Control System based on ZigBee Technology” In Proceedings of Journal of Zhejiang University SCIENCE A, 8(10).

Appendices

Appendix A: User Manual

Android application User Manual

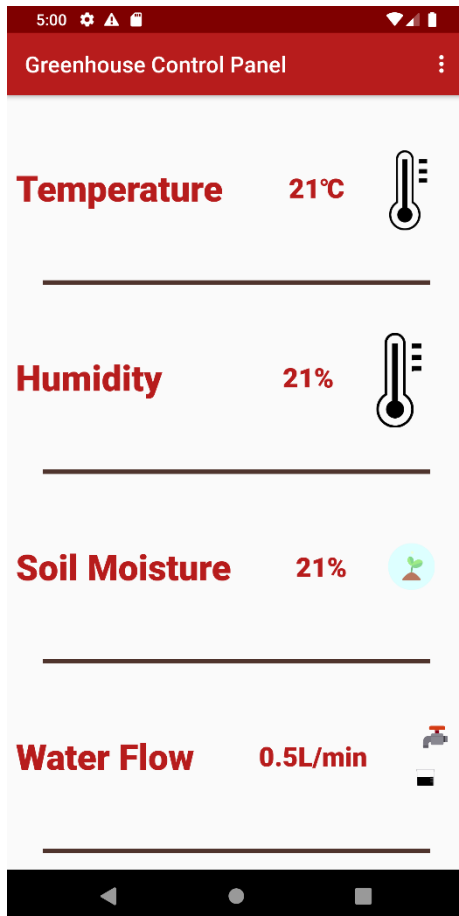


Figure A1. 1 Start screen

(Source: Author)

User encounters the first activity seen in Figure A1.1 which shows the values of the greenhouse sensors. Each value field is clickable and sends user to the next menu.

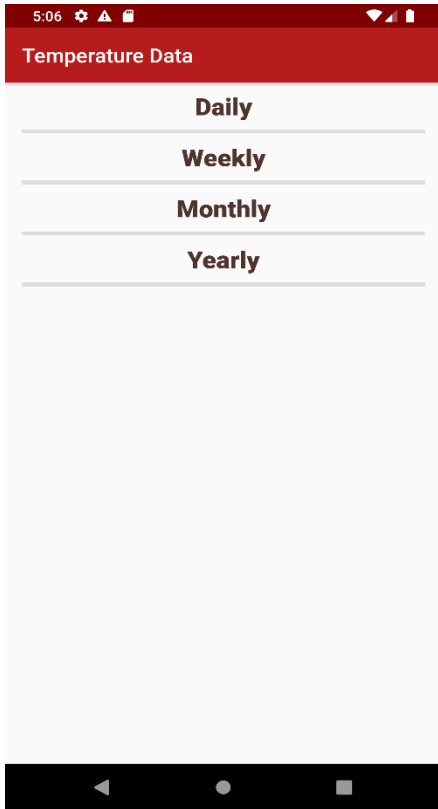


Figure A1. 2 Select period to view historical results

(Source: Author)

User has the ability to choose which period to view historical data by clicking it. View will be sent to screen in Figure A1.3 where a legend is shown which has the path to the graphical representation of the data. If user decides to click the view graphical representation field, they are sent to Figure A1.4 where the data is represented using a bar graph.

Day	Value
Sunday	21
Monday	23
Tuesday	20
Wednesday	19
Thursday	18
Friday	25
Saturday	20

Click here to view in graphical form

Figure A1. 3 Legend of graphical Data

(Source: Author)

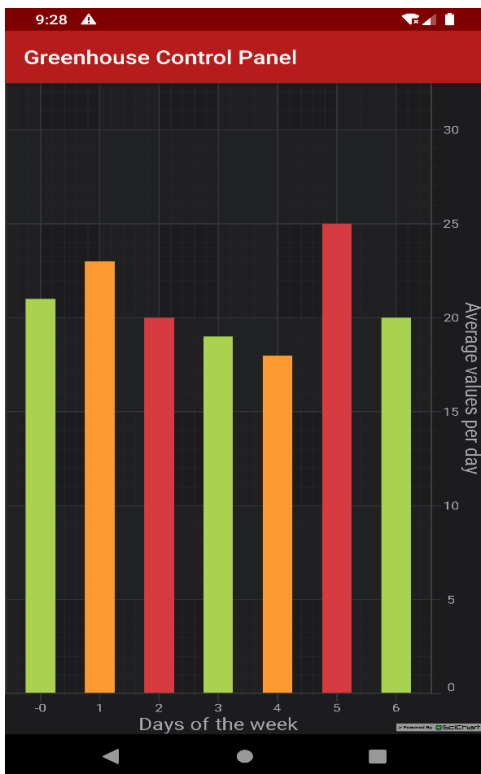


Figure A1. 4 Graphical data

(Source: Author)

Hardware User Manual

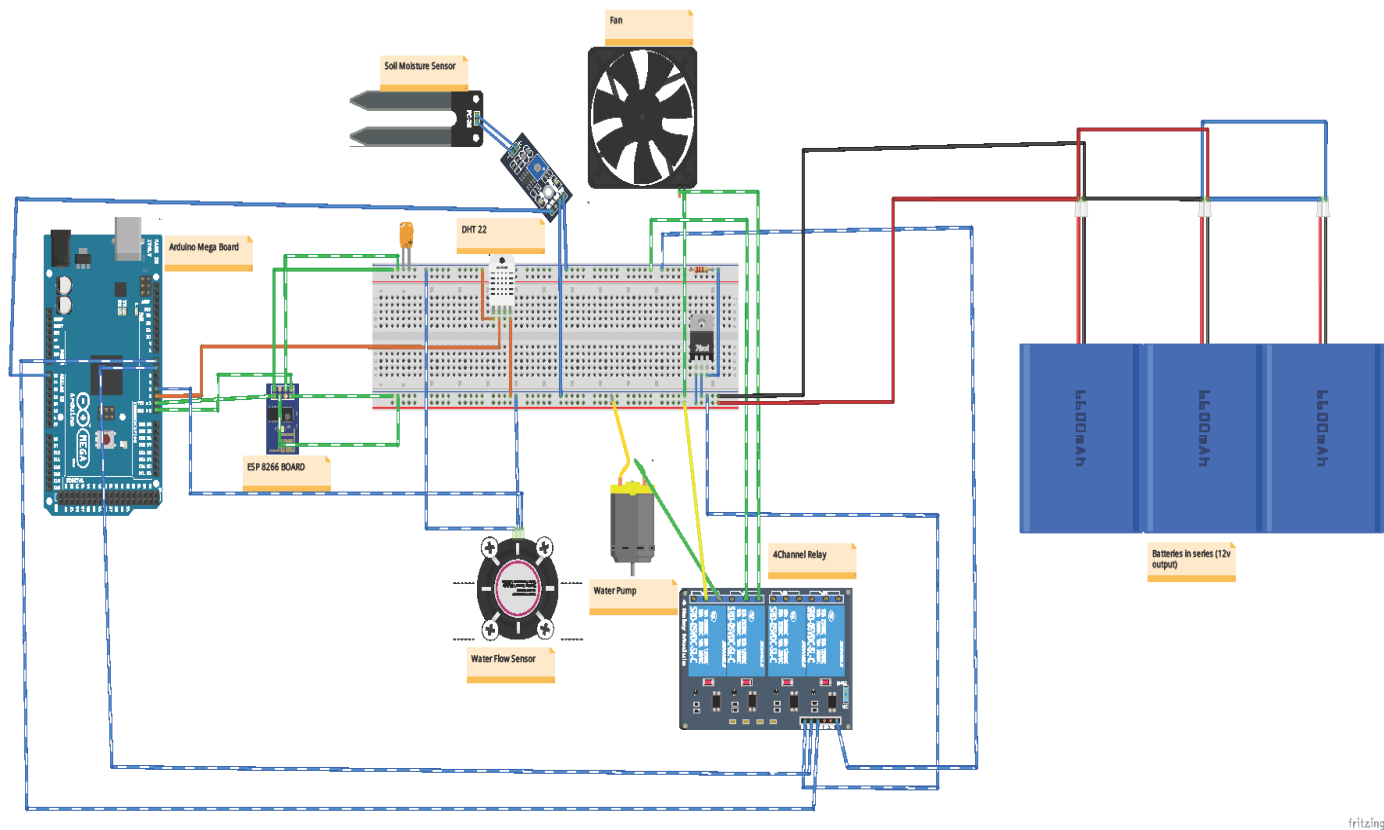


Figure A1. 5 Circuit setup of components

(Source: Author)

Figure A1.5 shows the setup of the component connections that need to be followed by the user to setup the greenhouse system. The Arduino and esp8266 module will be pre-programmed with the software needed to interface with the android application.

Appendix B: Hardware code

WiFi Module source code

```
// Import required libraries

#include <ESP8266WiFi.h>

#include <ArduinoJson.h>

#include <TextFinder.h>

TextFinder finder (Serial);

// WiFi parameters

const char* ssid = "nash";

const char* password = "01234567890";

const int NUMBER_OF_FIELDS = 3; // how many comma-separated fields we expect

int fieldIndex = 0; // the current field being received

double values [NUMBER_OF_FIELDS]; // array holding values for all the fields

const int capacity = JSON_OBJECT_SIZE (4);

StaticJsonDocument<capacity> doc;

// The port to listen for incoming TCP connections

#define LISTEN_PORT      80

// Create an instance of the server

WiFiServer server (LISTEN_PORT);

void setup(void) {

    // Start Serial

    Serial.Begin(115200);

    // Connect to WiFi
```

```

// WiFi.mode(WIFI_STA);

WiFi.begin(ssid, password);

while (WiFi.status() != WL_CONNECTED) {

    delay (500);

    Serial.print(".");

} Serial.println("");

Serial.println("WiFi connected");

// Start the server

server.begin();

Serial.println("Server started");

// Print the IP address

Serial.println(WiFi.localIP());

}

void loop () {

    // Handle REST calls

    WiFiClient client = server.available();

    for (fieldIndex = 0; fieldIndex < 3; fieldIndex ++) {

values[fieldIndex] = finder.getValue(); // get a numeric value

doc["temperature"] = values[0];

doc["humidity"] = values[1];

doc["water_flow"] = values[2];

doc["moisture"] = values[3];

client.connect("192.168.137.1",80);

```



```

client.println("POST /post HTTP/1.1");

client.println("HOST: 192.168.137.1");

client.println("Content-Type: application/json");

client.print("Content-Length: ");

client.println(measureJson(doc));

client.println("Connection: close");

client.println();

serializeJson(doc, client);

fieldIndex = 0; // ready to start over

}

delay (10000);

}

```

Arduino Code

```

#include <DHT.h>

//Constants

#define DHTPIN 3 // what pin we're connected to

#define DHTTYPE DHT22 // DHT 22 (AM2302)

#define RELAY1 6

#define RELAY2 7

#define MOISTURE A0

#define WATER_FLOW 2

DHT dht (DHTPIN, DHTTYPE); /// Initialize DHT sensor for normal 16mhz Arduino

// Variables

```

```

// Temp humidity sensor

int chk;

float hum; //Stores humidity value

float temp; //Stores temperature value

int tempMax = 25;

double tempMin =21;

// soil moisture sensor

double output_value;

double minMoisture = 20;

// water flow sensor

float calibrationFactor = 4.5;

byte sensorInterrupt = 0; // 0 = digital pin 2

volatile byte pulseCount;

float flowRate;

unsigned int flowMilliLitres;

unsigned long totalMilliLitres;

unsigned long oldTime;

unsigned int frac;

void setup () {

    Serial.begin(115200);

    Serial2.begin(115200);

    dht.begin();

    // Initialise the Arduino data pins for OUTPUT

```

```

pinMode (RELAY1, OUTPUT);

pinMode (RELAY2, OUTPUT);

pinMode (WATER_FLOW, INPUT);

digitalWrite (WATER_FLOW, HIGH);

pulseCount    = 0;

flowRate      = 0.0;

flowMilliLitres = 0;

totalMilliLitres = 0;

oldTime       = 0;

// The Hall-effect sensor is connected to pin 2 which uses interrupt 0. Configured to trigger on
a FALLING state change (transition from HIGH state to LOW state)

attachInterrupt (sensorInterrupt, pulseCounter, FALLING);

}

void loop(){

    //Read data and store it to variables hum and temp

    hum = dht.readHumidity();

    temp= dht.readTemperature();

    // temp = 25.7;

    output_value = analogRead(MOISTURE);

    output_value = map(output_value, 550, 0, 0, 100);

    if( temp < tempMin ){

        // Serial.println("Fan is off");

        digitalWrite(RELAY1,HIGH);    // Turns Relay Off

```

```

    }

    if((temp >= tempMin) && (temp <= tempMax)) { // if temperature is higher than minimum
temp

        //Serial.println("Fan is on");

        digitalWrite(RELAY1,LOW);        // Turns ON Relays 1

    }

    if (minMoisture >= output_value){

// Serial.println("Pump is on");

        digitalWrite(RELAY2,LOW);        // Turns ON Relays 2

    }

    if (minMoisture < output_value){

        digitalWrite(RELAY2, HIGH); // Turns Relay Off

    }

//Print temp and humidity values to serial monitor

    Serial.print("Humidity: ");

    Serial.print(hum);

    Serial.print("%,");

    Serial.print(" Temp: ");

    Serial.print(temp);

    Serial.println(" Celsius");

// Print Soil Moisture

    Serial.print("Mositure : ");

    Serial.print(output_value);

```

```

Serial.println("% ");

Serial2.println(temp);

Serial2.println(hum);

Serial2.println(output_value);

delay(2000); //Delay 2 sec.

if((millis() - oldTime) > 1000){ // Only process counters once per second

// Disable the interrupt while calculating flow rate and sending the value to the host

detachInterrupt(sensorInterrupt);

    flowRate = ((1000.0 / (millis() - oldTime)) * pulseCount) / calibrationFactor;

oldTime = millis();

flowMilliLitres = (flowRate / 60) * 1000;

// Add the millilitres passed in this second to the cumulative total

totalMilliLitres += flowMilliLitres;

// Print the flow rate for this second in litres / minute

Serial.print("Flow rate: ");

Serial.print(int(flowRate)); // Print the integer part of the variable

Serial.print("."); // Print the decimal point

// Determine the fractional part. The 10 multiplier gives us 1 decimal place.

frac = (flowRate - int(flowRate)) * 10;

Serial.print(frac, DEC) ; // Print the fractional part of the variable

Serial.print("L/min");

// Print the number of litres flowed in this second

Serial.print(" Current Liquid Flowing: "); // Output separator

```

```
Serial.print(flowMilliLitres);

Serial.print("mL/Sec");

// Print the cumulative total of litres flowed since starting

Serial.print(" Output Liquid Quantity: ");      // Output separator

Serial.print(totalMilliLitres);

Serial.println("mL");

Serial2.println(flowRate);

// Reset the pulse counter so we can start incrementing again

pulseCount = 0;

// Enable the interrupt again now that we've finished sending output

attachInterrupt(sensorInterrupt, pulseCounter, FALLING);

}

}

void pulseCounter(){

// Increment the pulse counter

pulseCount++;

}
```

