SMART GARDEN MONITORING APPLICATION

NUR SHAHIRAH BINTI AZMI

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

SMART GARDEN MONITORING APPLICATION

NUR SHAHIRAH BINTI AZMI

This report is submitted in partial fulfillment of the requirements for the Bachelor of Computer Science (Computer Networking) with Honors.

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2021

DECLARATION

I hereby declare that this project report entitled

SMART GARDEN MONITORING APPLICATION

is written by me and is my own effort and that no part has been plagiarized without citations.

STUDENT : NUR SHAHIRAH BINTI AZMI Date : 10/9/2021

I hereby declare that I have read this project report and found this project report is sufficient in term of the scope and quality for the award of Bachelor of Computer Science (Computer Networking) with Honors.

SUPERVISOR : TS. MUHAMAD SYAHRUL AZHAR BIN SANI Date : 10/9/2021

DEDICATION

In the name of God, the Most Gracious, the Most Merciful.
I dedicate this project to my beloved parents, Azmi Bin A.Ghani and Norizan Binti Salleh who always support me, inspired me, guided me tightly, support me with words of encouragement from the beginning of my PSM journey. To my supervisor, evaluator, and lecturers for modelling me into a knowledgeable person. To my friends and course mates for guidance, sharing information, and giving support throughout my study in UTeM.

ACKNOWLEDGEMENTS

First and foremost, All Praises to Allah SWT the Almighty, for providing me with the strength to complete my project. Thanks to Allah, because I manage to successfully complete my final year project.

A special gratitude I would like to thank to my final year project supervisor, Ts. Muhamad Syahrul Azhar Bin Sani for giving me assistant to complete this project successfully. His kindness and suggestion during the preparation of this research are highly appreciated.

I would also like to thank my family, especially my beloved parents, Azmi bin A.Ghani and Norizan Binti Salleh, who have been giving me a full support and motivation throughout my project during this pandemic era. In addition, I also want to thank my all of my friends with their help that have guided and supported me in order to finish my final year project on time. Furthermore, I would like to thank everyone who indirectly contributed to the successful completion of the report. Thank you very much.

ABSTRACT

The Internet of Things (IoT) is a network of connected devices that interact with one another. Using IoT, it allows user to monitor and control a device remotely with a computer or smartphone to collect and share information. This project presents a cost-effective automated Smart Garden Monitoring Application which can be utilized to monitor the plant in the garden every day. This technology was designed to replace the traditional method of plant monitoring, which required people to be present. The objective of this project are to study on smart garden method with Internet of Things technology, to develop hardware prototype that able to monitor garden's environment and to integrate hardware prototype with mobile application to provide alert. The Node MCU ESP 8266 has an inbuilt WiFi system is used as the central core for IoT applications where the code is implemented in developing an algorithm to differ the data from sensors. The system include a variety of sensors, such as humidity and temperature sensors, moisture sensors in the soil, and movement detection sensors. All sensors are connected to the Node MCU ESP 8266, and the data is saved in the cloud and displayed on the mobile app. Certain parameters is notified to the user if they detect certain condition. This project describes a wireless network and sensor-based Internet of Things architecture for a garden. The expected output result of this project is to provide and alert user with notification about the condition of the plant while remotely monitoring it via mobile application. Hardware testing is carried out to verify that the proposed system is completely functioning.

ABSTRAK

Internet of Things (IoT) adalah rangkaian peranti bersambung yang saling berinteraksi antara satu sama lain. Dengan menggunakan IoT, pengguna dapat memantau dan mengendalikan peranti dari jauh dengan komputer atau telefon pintar untuk mengumpulkan dan berkongsi maklumat. Projek ini membentangkan Sistem Pemantauan Taman Pintar automatik yang dapat digunakan untuk memantau tanaman di taman setiap hari. Teknologi ini dirancang untuk menggantikan kaedah tradisional pemantauan tanaman, yang memerlukan individu untuk hadir. Objektif projek ini adalah Untuk mengkaji kaedah taman pintar dengan teknologi IoT, untuk mengembangkan prototaip perkakasan yang dapat memantau lingkungan taman dan untuk mengintegrasikan prototaip perkakasan dengan aplikasi mudah alih untuk memberi peringatan. Node MCU ESP 8266 mempunyai sistem WiFi yang digunakan sebagai teras utama untuk aplikasi IoT di mana kod tersebut dilaksanakan dalam mengembangkan algoritma untuk membezakan data daripada sensor. Sistem ini merangkumi pelbagai sensor, seperti sensor kelembapan dan suhu, sensor kelembapan di dalam tanah, dan sensor pengesanan pergerakan. Semua sensor disambungkan ke Node MCU ESP 8266, dan data disimpan di awan dan dipaparkan di aplikasi mudah alih. Parameter tertentu akan diberitahu kepada pengguna jika mereka mengesan keadaan tertentu. Projek ini menerangkan rangkaian tanpa wayar dan seni bina Internet of Things berasaskan sensor untuk taman. Hasil keluaran yang diharapkan dari projek ini adalah memberi dan memberi tahu pengguna tentang pemberitahuan mengenai keadaan taman sambil memantau secara jarak jauh melalui aplikasi mudah alih. Ujian perkakasan dilakukan untuk mengesahkan bahawa sistem yang dicadangkan berfungsi sepenuhnya.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

PAGE

LIST OF ABBREVIATIONS

FYP	**-**	**Final Year Project**
IoT	**-**	**Internet of Things**
USB	**-**	**Universal Serial Bus**
Wi-Fi	**-**	**Wireless Fidelity**
Apps	**-**	**Application**
GUI	**-**	**Graphical User Interface**
RAD	**-**	**Rapid Application Development**
Mobile Apps	**-**	**Mobile Application**
Fig.	**-**	**Figure**

LIST OF ATTACHMENTS

CHAPTER 1: INTRODUCTION

1.1 Introduction

Gardening is extensively done by Malaysians, especially those who live in rural areas. They garden for various purposes, including food production, money generating, home decoration, and even self-satisfaction or as a hobby. Although gardening is not as widely embraced in cities as in rural areas, the development of edible gardens, urban farming, and community gardens in cities has inspired residents to start their gardens. Furthermore, it has driven the growth of urban farming or urban gardening among Malaysians for food supply, environmental sustainability, and health. (Navesh, 2014)

In 2006, the Malaysian government launched a campaign called "Bumi Hijau" to encourage Malaysians to grow their food while encouraging people to care for the environment (LAr. Noriah Mat, CA, 2017). According to the Putrajaya Corp Landscape and Park Department, there are currently nine community gardens in Putrajaya due to this program. Furthermore, it was recently announced that Universiti Putra Malaysia (UPM) had established Urban Agriculture to encourage contemporary farming among city people in the limited area available in their homes as a guaranteed supply of food for the nation by 2020. (Noh, 2006)

However, gardening, on the other hand, is not an activity that can be completed in a short amount of time. Gardening needs some skill, expertise, and dedication on the part of the gardener. Some plants, for example, require daily watering, which is a difficulty for individuals who are preoccupied with their jobs, particularly those who reside in metropolitan areas. Noriah Mat, senior deputy director of Putrajaya Corp Landscape and Park Department, claimed that community member involvement in urban gardening projects had fallen 2 to a mere 5% over the years, although many have shown interest. Although some of the projects were successful, some were skeptical about the long-term viability of urban farming, and some were having trouble finding time to accomplish it because they all have day jobs. (Viewed, 2015)

Living in the modern world has shown that technology may be utilized to improve human activities daily. Many inventions have been created to improve human existence, including communication, health, home, and others.

Thus, this project aims to integrate technology in gardening activities to assist people in monitoring their gardens. The project proposed a smart garden monitoring application that allows users to monitor current garden parameters and monitor the system via a mobile.

1.2 Problem Statement (PS)

Gardening is a task that requires human supervision. In today's busy world, the problem arises when people neglect to nourish their plants because they are preoccupied with their jobs and outdoor lifestyle. In addition, the unpredictable nature of the weather also hinders the plant's ability to thrive. As a result, the garden gets destroyed due to environmental conditions and lack of proper care. Because of this problem, a smart garden monitoring application is going to be developed to overcome the problem.

Table 1.1 Problem Statement

PS	Project Statement
PS1	Manual ways of monitoring garden which caused plant in the garden get destroyed due to environmental conditions and lack of proper care.

1.3 Project Question (PQ)

Project questions are fundamental to help the effectiveness of the project. It is used to identify the existing ways of taking proper care of the plant. Based on the research, it can conclude that there are few weaknesses of the garden system.

Table 1.2 Project Question

PS	PQ	Project Question
PS1	PQ1	How to monitor the condition of the garden remotely?
PS1	PQ2	How to make system work automatically?
PS1	PQ3	How to alert user on certain condition of the garden?

1.4 Project Objective (PO)

Project objective defines the improvement that the project aims to achieve in the completion of the project. The improvement must be considered based on the problem statement and the project question of this project. The objectives for this project are:

PO1 : To study on smart garden method with IoT technology.

The monitoring methods used in the smart garden system is discussed to ensure it fulfills the user's needs. Besides, the requirement for suitable microprocessors and sensors are studied before proceeding to the development process of the garden's prototype.

PO2 : To develop hardware prototype that able to monitor the garden's environment.

In this project, the prototype is developed using Node MCU ESP8266 and sensors to monitor the garden's environment. The monitoring sensor of the plant includes soil moisture, humidity, and temperature, and motion sensor.

PO3 : To integrate hardware prototype with mobile application to provide alert.

The prototype is integrated with a mobile application that displays the data of monitoring. Thus, users can monitor the plant remotely.

Table 1.3 Project Objective

PS	PQ	PO	Project Objective
PS1	PQ1	PO1	To study on smart garden method with IoT technology.
PS1	PQ2	PO2	To develop hardware prototype that able to monitor garden's environment.
PS1	PQ3	PO3	To integrate hardware prototype with mobile application to provide alert.

1.5 Project Scope

This project is conducted in a small garden for the research area, with a device functions as the system and a user as the administrator. The scope of this project is to discover the feasibility of using a microcontroller. The sensor is applied to the micro-controller to monitor the parameter of the plant. Besides, this system also consists of mobile applications that provide GUI components to interact with.

i. **Target Area:** The smart garden monitoring application is installed in a garden with Wi-Fi access to facilitate communication inside the garden.

ii. **Device:** The device executes the monitoring system based on the reading of the sensors.

iii. **User :** The user can use a mobile application to control and monitor garden activity through the system.

1.6 Project Contribution (PC)

Project contribution determines the expected result from this project and the significant contribution of this project. This project provides a tangible advantage to people who have a garden and want to monitor them remotely, whether they are present or not. The project's contribution saves users time when monitoring their plants since they might be preoccupied with other tasks. Aside from that, since this system uses a low-cost component, this project can also save money and be easily procured.

Table 1.4 Project Contribution

PS	PQ	PO	PC	Project Contribution
PS1	PQ1	PO1	PC1	Knowledge about smart garden and IoT technology.
	PQ2	PO2	PC2	A prototype that able to monitor garden remotely for tracking garden's parameter.
	PQ3	PO3	PC3	Provide alert to the user by integrating smart garden device with mobile application.

1.7 Report Organization

There are 7 chapters in this thesis, including Introduction, Literature Review, Project Methodology, Analysis & Design, Implementation, Testing and Conclusion.

Chapter 1: Introduction

The first chapter discusses an overview of the project to be developed. This chapter also discusses the problem statement, project question, objective, scope, contribution, and conclusion.

Chapter 2: Literature Review

The second chapter discusses on the literature review, which supported by reading materials, and the publication of a study that was completed before the system was introduced to the public. Hence, this chapter also discusses some relevant published thesis and journal articles on the Internet of Things-enabled Smart Garden Monitoring Application.

Chapter 3: Project Methodology

The third chapter discusses the project's methodology, including a brief description of the project prototype and block diagram. The method used to develop this project is described in this chapter. This chapter also includes project milestones, which make reviewing and organizing the project more manageable.

Chapter 4: Analysis and Design

The fourth chapter defines the results of the preliminary design analysis and the development of the detailed design. This chapter focused on problem analysis, requirement analysis, and project design. The high-level design, user interface design, and detailed design are also briefly covered in this section.

Chapter 5: Implementation

The fifth chapter discussed the activity involved in the implementation phase and expected output after this phase has been completed. This chapter covered the software development environment setup, software configuration management, and the implementation status.

Chapter 6: Testing

The sixth chapter focuses on the project test plan, test design, result, and project analysis. The testing plan includes test organization, test environment, and test schedule. Following that, in the testing results, the test design is descriptive. The test results are analyzed to complete the testing process.

Chapter 7: Conclusion

The final chapter summarizes the overall project based on goals met by combining data from the implementation and testing phases, project contribution, and project limitations. This chapter explains additional work that may be done in the future.

1.8 Conclusion

In conclusion, this chapter helps understand the project's context, the goals and objectives that must be accomplished, and the problems that arose before the start of the project. This study wants to propose a new approach to Smart Garden Monitoring Application based on the related topics. The next chapter focuses on the literature review, which covered related work regarding the technology to be discussed. Aside from that, the current system's issue that necessitates a new system is examined and clarified.

CHAPTER 2: LITERATURE REVIEW AND PROJECT METHODOLOGY

2.1 Introduction

A literature review is a review from other sources that apply to a specific research subject. A literature review is essential in creating a project because it is required to identify information that is relevant to the project. In this chapter, the research and development of devices for smart garden monitoring applications are discussed. Journal articles, websites, reference books, and other information related to this topic were used to construct the literature review.

Thus, this chapter has analyzed the previous work that discussed the research of smart gardens, components used, and table of the problem tabulated based on the previous work. Six previous work has been selected to be analyzed, and from the table in critical review, a proposed solution is proposed to overcome the problem. Finally, for this chapter is the conclusion which concludes the overall of this chapter.

2.2 Internet of Things

Automation dominates the globe in the new technological age, and it holds the secret to fundamentally empowering many sectors of the economy. Technology helps to improve the capabilities, efficiencies, and production efficiency of any human operation, from engineering to agriculture to services and logistics (Yeo et al., 2015). The Internet of Things (IoT) is a technique of monitoring and tracking basic life parameters using computers, cell phones, or other digital technologies (Dorsemaine et al., 2016). New systems based on sensors, applications, and communication protocols can be built using IoT frameworks and information to automate specific tasks. The exchange of data sharing is the key factor of IoT. The practice of using automation for basic tasks would nourish our living standards (Albishi et al., 2017). Technology nowadays are in the midst of the fourth industrial revolution, in which manual processes are being replaced with automated ones (Trotta & Garengo, 2018).

2.3 Smart Garden

People that tend to cultivate gardens and other plants in their homes are always vigilant during the initial stages of care. As a result of this neglect, the garden is ruined. Climate conditions can also shorten the garden's life, as certain plants can die due to a lack of rain, extreme heat, or humidity. Gardeners are hired to manage people's gardens. Plants are often harmed by environmental factors and a lack of adequate treatment. This is where the concept of an integrated garden control device for the internet of things emerges to address the above issues (Kavitha et al., 2020). Internet of Things contributes to eliminating the traditional practices that require human workforce by providing real-time knowledge about plant's parameter (Gaur & Mittal, 2013).

A revolutionary technology known as an Internet of Things-based Smart Garden is beneficial to gardeners. It is mostly used for monitoring and providing vital services that can speed up plant development. Essentially, smart garden systems must have an automated mode and be equipped with multiple sensors for process control. The Agricultural based Internet of Things is increasingly becoming one of the fastest-growing sectors of the IoT (Sambath et al., 2019).

2.4 Microprocessor and Microcontroller

An integrated circuit (IC) is a microprocessor that combines the core functions of a computer's central processing unit (CPU). It's a programmable multifunctional microprocessor device that receives binary data as input and processes it according to instructions stored in memory (Ligo George, 2020)

A microcontroller is a system that includes at the very least a microprocessor, internal storage, data memory, and input-output (I/O). A microcontroller, also known as an embedded controller, is a single-chip computer used to perform control operations on other devices or equipment (Lutkevich, 2019)

2.4.1 Node MCU ESP 8266

This project uses Node Mcu since it is an open-source IoT platform with inbuilt networking features that can connect devices and transfer the data using Wi-Fi protocol. NodeMCU also refers to firmware and development kits. It includes firmware running on the ESP8266 Wi-Fi SoC and hardware supported the ESP-12E module. The firmware uses the scripting language of C (Tonage, Yemul, Jare, & Patki, 2018).

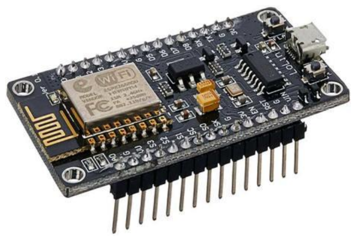


Figure 2.1 Node MCU

2.5 Sensor

2.5.1 Soil Moisture Sensor

Soil moisture sensor senses and measures the volumetric and moisture water content of the soil. A moisture sensor is occupied with a probe inserted together inside the soil to measure the water content of the soil. The device has two conductors that are separated with electrodes. This sensor also is particularly suitable for use as a soil moisture indicator of the percentage of endpoints of humidity, such as wet or dry environments (Iltis & Jolla, 1987).

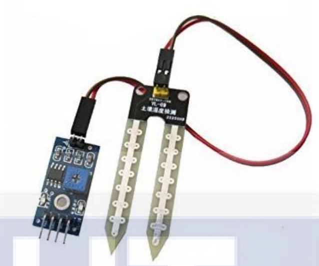


Figure 2.2 Soil Moisture Sensor

2.5.2 DHT11 Temperature and Humidity Sensor

DHT11 sensor is a low-cost temperature and humidity sensor for Arduino and Raspberry Pi popular among electronics enthusiasts due to its numerous benefits. It detects ambient air using a capacitive humidity sensor and a thermistor and outputs a digital signal on the data pin. Then, it is uploaded to the server using Node MCU ESP 8266 (Pleva GmbH, 1995).

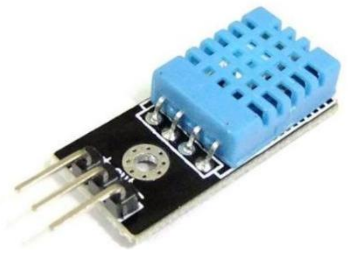


Figure 2.3 DHT11 Temperature & Humidity Sensor

2.5.3 PIR Motion Sensor

A passive infrared sensor (PIR sensor) detects infrared (IR) light generated by objects in their range of vision. When motion is detected, the I/O pin receives a high signal. Typically used as a security device to detect the presence of humans or animals in a monitored area. The Fresnel lens is needed to ensure that the PIR's two slots can view beyond a certain distance. When the sensor is turned off, both places detect the same quantity of infrared. The ambient amount comes from the outside, the walls, and the room (Li, 2019).

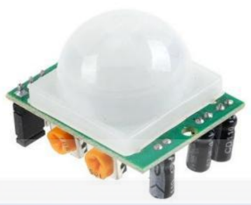


Figure 2.4 PIR Motion Sensor

2.6 Relay

A relay is a switch that is activated or deactivated by electricity. The relay uses an electromagnet (coil) energy to mechanically operate a switching mechanism (contact). When a relay contact is open, it switches power ON a circuit when the coil is activated (Gautam, 2020).

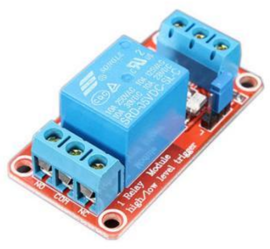


Figure 2.5 Relay

2.7 Communication Technologies in IoT

The internet of things (IoT) is a network of physical devices, tools, automobiles, buildings, and other things embedded with electronics, circuits, software, sensors, and network connections to gather and share data. The internet of things allows items to be sensed and controlled remotely using existing network infrastructure, allowing for more efficient use of resources (Gokhale, Bhat, & Bhat, 2018).

2.7.1 Wi-Fi

Wi-Fi is a wireless communication technology that enables devices to interact without the use of wires. It is a technical word used to refer to a sort of wireless local area network protocol based on the IEEE 802.11 network standard. In a fixed location, Wi-Fi is the most common kind of wireless data connection (Ding et al., 2018).

2.7.2 Bluetooth

Bluetooth is mainly used to communicate across wireless Personal Area Networks (PANs). It is a widely utilized technology that allows data to send from one device to another. It enables users to establish ad hoc networks to transmit data between a variety of devices (Tsira & Nandi, 2014).

2.7.3 Zig Bee

ZigBee relies on the IEEE 802.15.4 personal-area network standard. ZigBee is considered a substitute to Wi-Fi and Bluetooth for specific applications, such as low-powered devices that do not require much bandwidth, such as smart home sensors. The focus of ZigBee is not on point-to-point communication (Tillman, 2021).

Table 2.1 Communication Technologies Specification

Standard	Wi-Fi	Bluetooth	Zigbee
IEEE Spec	IEEE 802.11a/b/g/n	IEEE 802.15.1	IEEE 802.15.4
Topology	Star	Star	Mesh, Star, Tree
Bandwidth	Up to 54 Mbps	1Mbps	250 Kbps
Power Consumption	Low	Very Low	Very Low
Max. Data rate	54	0.72	0.25
Range	4-20 m	<30 m	10-300 m
Spectrum	2.5 - 5GHz	2.4 GHz	2.4 GHz
Channel Bandwidth	22 MHz	1 MHz	0.3/0.6 MHz, 2 MHz

Zigbee communication is commonly for large areas and requires more cost rather than Wi-Fi and Bluetooth. One downside for Bluetooth is it is built for short-range communication, which means that the user can only control the smart device from a short distance. The main benefit of Wi-Fi networks over other wireless technologies is that Wi-Fi is relatively easy to set up, allowing IoT devices with Wi-Fi modules to connect directly to the Internet. Thus, this project chooses to use Wi-Fi as a communication technology for IoT devices.

2.8 Related Work/Previous Work

This segment discusses the findings of previous research related to the proposed method. In this field, various type of implementations has been carried out by the researcher. There are several journals on the garden monitoring project.

2.8.1 Smart Garden Management System

According to Shireen & Devi (2018), stated that Agriculture IoT is gradually becoming one of the most rapidly expanding areas within the IoT. The authors present a proposed system for measuring and monitoring the plant. The system is occupied with an Arduino UNO microcontroller and sensor that can measure the plant's soil moisture, temperature, and humidity. The author also describes that the system uses GPRS as a network overlay for GSM mobile stations. The system transfers the data it receives to GPRS and shows the results on an LCD. The Garden management system use device such as PC to keep the owner update.

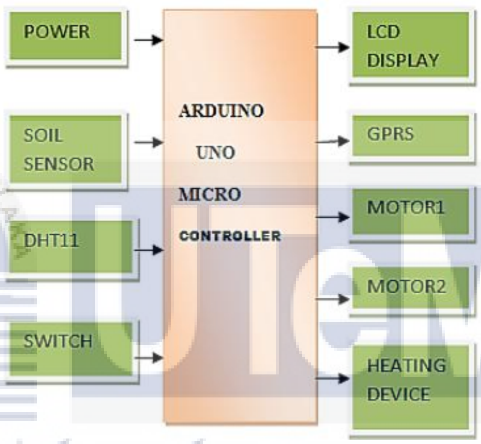


Figure 2.6 Block Diagram Smart Garden Management System

2.8.2 IoT Smart Garden

Ismalyza et al., (2019) presented research on “IoT Smart Garden” to create a system that used Node-Red technologies to develop a distributed wireless network of soil moisture, humidity sensor, and temperature sensors. The system worked through the NodeMCU microcontroller, where the real-time values from the sensors were uploaded to Node-RED. A gateway device also managed sensor data, activated actuators, and transmitted data to the web application for control purpose of the smart garden monitoring system.

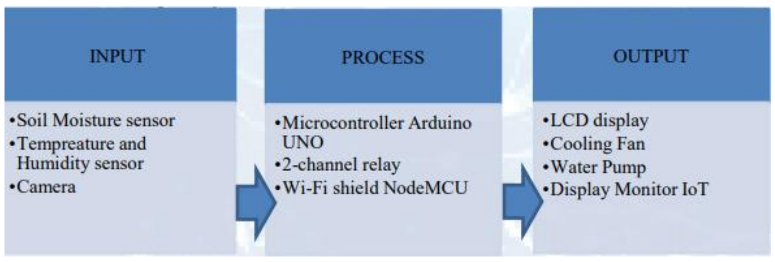


Figure 2.7 Block Diagram of IoT Smart Garden

2.8.3 Tomen: A Plant Monitoring and Smart Gardening system using IoT

Elangovan et al., (2018) have introduced that the primary goal of automation is to make people's lives easier by eliminating physical work and enhancing the efficiency of any machine without requiring user interaction. This paper gives an overview of Tomen's plant monitoring and smart gardening system using IoT for tomato gardens. Tomen is a Raspberry Pi-based plant control and smart gardening system. Tomen manages the sensors and monitors the state of the plants using a cloud-based server and a smartphone program that runs on both Android devices. Tomen can sense variations in temperature and moisture and perform essential activities on the plants, such as irrigation. The system keeps track of all of the garden parameters and saves the data in a database.

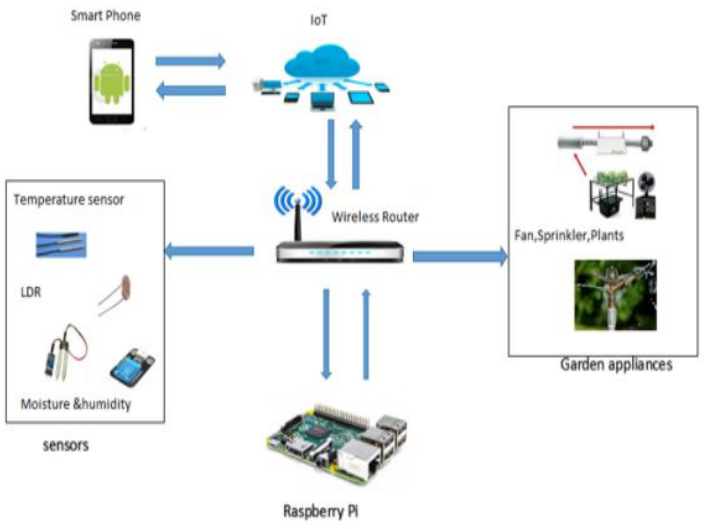


Figure 2.8 Architecture of Tomen System

2.8.4 Smart Garden Monitoring System Using IOT

Thamaraimanalan et al., (2018) have proposed a method that analyses the garden's environmental parameters to simplify plant watering. A mobile application is developed to check the condition of the garden for irrigation. This device uses temperature and moisture sensors to detect temperature and moisture in extensive gardens. Soil and other supporting sensors are incorporated by Arduino, which collects soil values and sends them to Firebase via the built-in WIFI facility. This project uses Firebase as a database, and the sensor's real-time values are changed every second. The researcher created an Android program linked to Firebase and allows the user to control the parameters from anywhere.

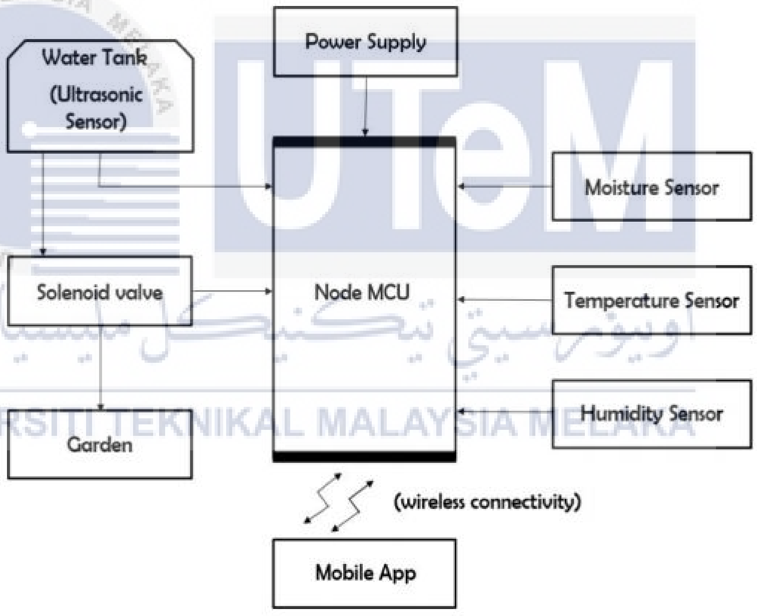


Figure 2.9 Block Diagram of Smart Garden Monitoring System using IoT

2.8.5 Garden Monitoring and Automation Using Atmega 16

In this project, the author Bharti et al., (2020) of the paper indicates proper care for maintaining the plant's growth. Hence, this paper proposed ease of way to maintain and monitor the garden monitoring by using ATMega 16 microcontroller. The article described that the hardware used a soil moisture sensor and a temperature sensor to calculate the moisture in the soil and the temperature of the atmosphere, respectively. It used an IR Module sensor to track the entry and exit of any human inside the yard. This IR Module also calculates the number of people who visit and exit the garden. LDR is also used here to determine whether it is day or night. The limitation of this project is the project does not have a system that can store data about the monitoring activities.

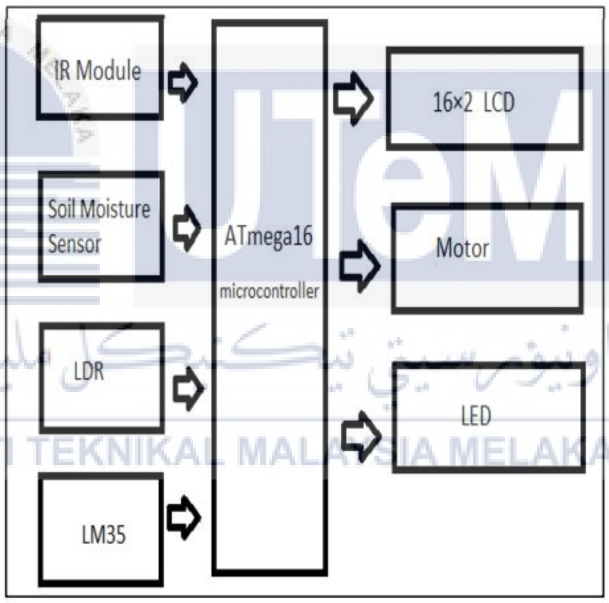


Figure 2.10 Block Diagram of the System

2.9 Critical review of current problem and justification

A critical review is a written task that involves summarizing and evaluating a text. It might be a journal article, a website, a book, or another medium. The researcher must review the selected text in depth to present a fair and accurate evaluation of the chosen content. As a result, several journals were to be used as guides for this project. The summary of the finding is tabulated as shown below.

Based on the current project review and the equivalent project that has been researched, several comparison features have been made. The comparison is made to define improvement of the project to develop. Table 2.2 below shows the comparison of features for smart garden monitoring.

Table 2.2 Comparison of Journal

Journal Title/ Author	Communication Protocol	Micro controller	Sensors	Interface	Motion Detection	Description	Research Gaps
Smart Garden Management System **By :** (Shireen & Devi, 2018)	Wi-Fi	Arduino UNO	Soil sensor, DHT11 sensor	Website	✗	The project is developed for user to monitor plant's parameters, then transfer data to GPRS and displays it on the LCD and website.	There is no notification or alarm system in place to warn users if there is an unusual in the garden.
IoT Smart Garden **By :** (Ismalyza et al., 2019)	Wi-Fi	Node MCU	Soil sensor, DHT11 sensor	Mobile Apps	✗	The project is developed to detect temperature and humidity of plant. Detect soil moisture and pump out water if plant is dry.	There is no discussion storing data from sensor.
Tomen: A Plant Monitoring and Smart Gardening system using IoT **By :** (Elangovan et al., 2018)	Wi-Fi	Raspberry Pi	Soil sensor, DHT11 sensor	Mobile Apps	✗	The project developed for user to monitor the parameter of sensor that connect with Raspberry Pi.	Quite expensive in cost but have a better quality in implementation.
Smart Garden Monitoring System Using IOT **By :** (Thamaraimanalan et al., 2018)	Wi-Fi	Node MCU	Soil sensor, DHT11 sensor, ultrasonic	Mobile Apps	✗	The developed system is used to save manpower and effectively uses the accessible water supply, resulting in increased profits.	There is no discussion on storing data from sensor.
Garden Monitoring and Automation Using At mega 16 **By :** (Bharti et al., 2020)	Not mentioned	Atmega16	Soil sensor, IR module sensor	LCD	✓	The project is developed for detecting moisture in soil and and use IR module for tracking the number of people who visit the garden is also calculated.	The data output display is only available on LCD. There is no display on other GUI interface.

Table 2.2 shows the previous research on a smart garden monitoring application. According to previous research, most garden monitoring devices focus on soil moisture, temperature, and humidity as a parameter in monitoring the garden system. A mobile application appears to be the ideal solution in terms of user interaction because it can be accessed remotely rather than website monitoring or LCD. In contrast to the website and LCD interface that operates on a computer, the user must be close to the system to control it.

Table 2.2 also shows that there are still weaknesses encountered in previous studies, but in this research, the Smart Garden has its strength as it carefully improvised the weakness of earlier studies. Shireen & Devi, 2018 initial idea of sending an alert via the website is ineffective since checking the website is not a fast action. The user must check their website, which may result in a delayed response. Bharti et al., 2020 have developed a prototype that monitors the moisture level in the soil in plants. The product, however, lacks a display since it only displays on LCD only. Thus, the proposed solution is integrating a mobile application platform for the same purpose instead of utilizing a website.

2.10 Proposed Solution/Further Project

Based on a previous study, a proposed solution has been proposed to improvise the previous version of the smart garden system. This project is highly beneficial to gardeners as it can provide a monitoring function system.

The proposed method comprises various sensors for the garden environment. The proposed system uses Node MCU, with multiple sensors such as moisture, humidity, temperature, and motion sensors. The Node MCU is initially being used to connected to Wi-Fi and get data from the sensor. It begins to read the parameters of sensors once the Wi-Fi is enabled. The requisite sensors' threshold levels have been set. The sensor data is sent and stored in a cloud server for monitoring analysis. The user gets a notification if the parameter is abnormal through the smartphone. The data can be dissected at any time and in any place.

Enhanced features such as the motion sensor are being added in this project to sense an intruder and send alerts to users' smartphones. The motion sensor detects any motion caused by animals or humans moving around the garden environment. The buzzer connected with the motion sensor is turned on when it detects any movement around the area. The LED is turn ON when the motion sensor detects movement. Then it alerts the user about the current condition of the detected activity.

2.11 Conclusion

This chapter summarizes that the literature review provides necessary chapter details, and it is a vital part of building the project concept. This is to ensure that the studies had been done based on the smart garden monitoring application as mentioned. A literature review also helps to understand the system's existing features and get a clear picture to develop the system. Based on this chapter, the domain involved in this chapter is the Internet of Things-based smart garden system. From this chapter, a comparison to choose a better project for references has been made to get a trend from the research. In the next chapter, the methodology used for the project is discussed.

CHAPTER 3: PROJECT METHODOLOGY

3.1 Introduction

This chapter consists of two-part which are project methodology and project milestones. The project methodology refers to techniques or methods applied to collect information and data in conducting a project. In the software engineering field, methodology refers to a framework used to design, plan and monitor the development process of information systems (CMS, 2005). To ensure that the project is finished within the allotted period, all tasks involved with this project are specified in the project milestone.

The Rapid Application Development (RAD) methodologies are used in this project. RAD is a software development approach that prioritizes accelerated preparation over minimal planning. This method enables the developer to engage more with end-users to the prototype throughout the project. As a result, the project meets the customer's requirements and is completed in the shortest time possible.

3.2 Methodology

In this project, the Rapid Application Development (RAD) model is used as a project methodology. This method is suitable because it is designed to deliver a high-quality product in the shortest time possible, instead of conventional life cycle software production. Aside from that, the RAD approach provides some product quality in a short period while still receiving input from the end consumer on their requirements. RAD methodology consists of four phases: Requirements Planning, User Design, Construction, and Cut Over. (Bargavi, 2018)

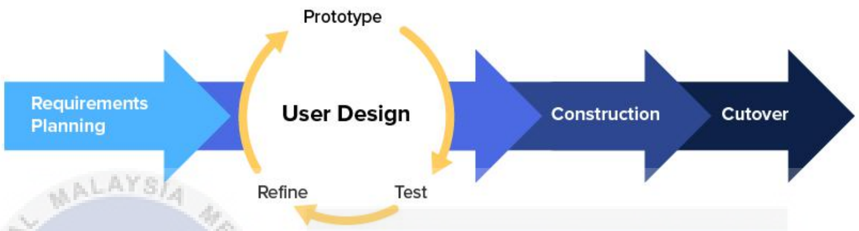


Figure 3.1 Rapid Application Development (RAD)

3.2.1 Phase I: Requirement Planning

This phase analyzes problems that occur and are faced by gardeners or busy workers who manually monitor their garden and then determine solutions that can solve the problems. The hardware and software requirements appropriate for this project are identified for the development to meet the project's objectives.

a. Hardware Requirement

i. **Node MCU ESP 8266:** Node MCU ESP 8266 is being chosen for this project. The Node MCU ESP8266 is used in this project to implement code program to communicate with IoT devices.

ii. **Temperature and Humidity Sensor (DHT11):** DHT11 is a sensor that is used to detect the temperature and humidity of the garden environment. This project uses the DHT11 sensor to monitor the temperature and humidity of the garden environment.

iii. **Soil Moisture Sensor:** A sensor that able to detect the moisture of parameter of the soil.

iv. **IR Motion Sensor:** A motion sensor is applied in this project. It is used to detect the movement if animal crossing the garden field.

v. **Water Pump:** The water pump is used to pump out from the container to water the plant.

vi. **Buzzer:** When the motion detection sensor senses movement, a buzzer trigger a sound, which might be useful for garden safety.

vii. **LED:** LED will turn light on along with buzzer when the motion detection sensor sense any intruder movement.

b. **Software Requirement**

i. **Arduino IDE:** The open-source of Arduino Software IDE (Integrated Development Environment) that runs on the computer, used to write and upload computer code to the physical board.

ii. **Blynk:** This project uses Blynk as a mobile application that displays data from IoT platforms. It will display all data from sensors. The Blynk application also has a manual button for watering purposes. It also has monitoring features and alert notifications if such as it will send alert notifications to the user when soil moisture is dry and if motion is detected.

iii. **ThingSpeak:** Cloud storage used to store data from sensors.

c. Other Materials

i. **Breadboard:** A tool that build and test circuits before finalizing any circuit design.

ii. **Jumper Wire:** Cable is used to connect the prototype between sensor and Node MCU ESP 8266.

3.2.2 Phase II: User Design

User design is a method used by product design teams to create a product that can give users a positive and appropriate experience. This phase discusses the architecture of the project. It involves building the project prototype by using the data and solution determined in the previous stage. The prototype design is to uses the sensor to detect the parameter of the plant condition. The design of the project is created to design the interaction of Node MCU ESP8266 with the sensors.

3.2.3 Phase III: Construction

The phase of construction involves refining the previous phase design. The rapid application development phase's goal is to complete the comprehensive design of the proposed system. There are three components to the hardware process. The first step is to connect the Node MCU ESP 8266 to Wi-Fi. Second, set up the Node MCU ESP 8266 and sensors to work with the made specification. Finally, set up the update section so that users can get alerts. The Arduino IDE is required for writing code or functions and uploading them to the microcontroller. The constructed phase also includes the ThingSpeak cloud storage for storing data. The created ThingSpeak cloud storage channel has four fields that indicate each sensor: the DHT11 temperature and humidity sensor, soil moisture sensor, and motion sensor.

3.2.4 Phase IV: Cut over

Cut over is the phase of the project implementation where the final product is finalized. Based on the previous stage's testing results, the system's functionality is enhanced in this phase. For the test features functionality, the product integrates both software and hardware. The Node MCU ESP 8266 is connected to the Wi-Fi and transmits data to the application during the testing process. Users are then notified about the parameters in their garden.

3.3 Project Milestone

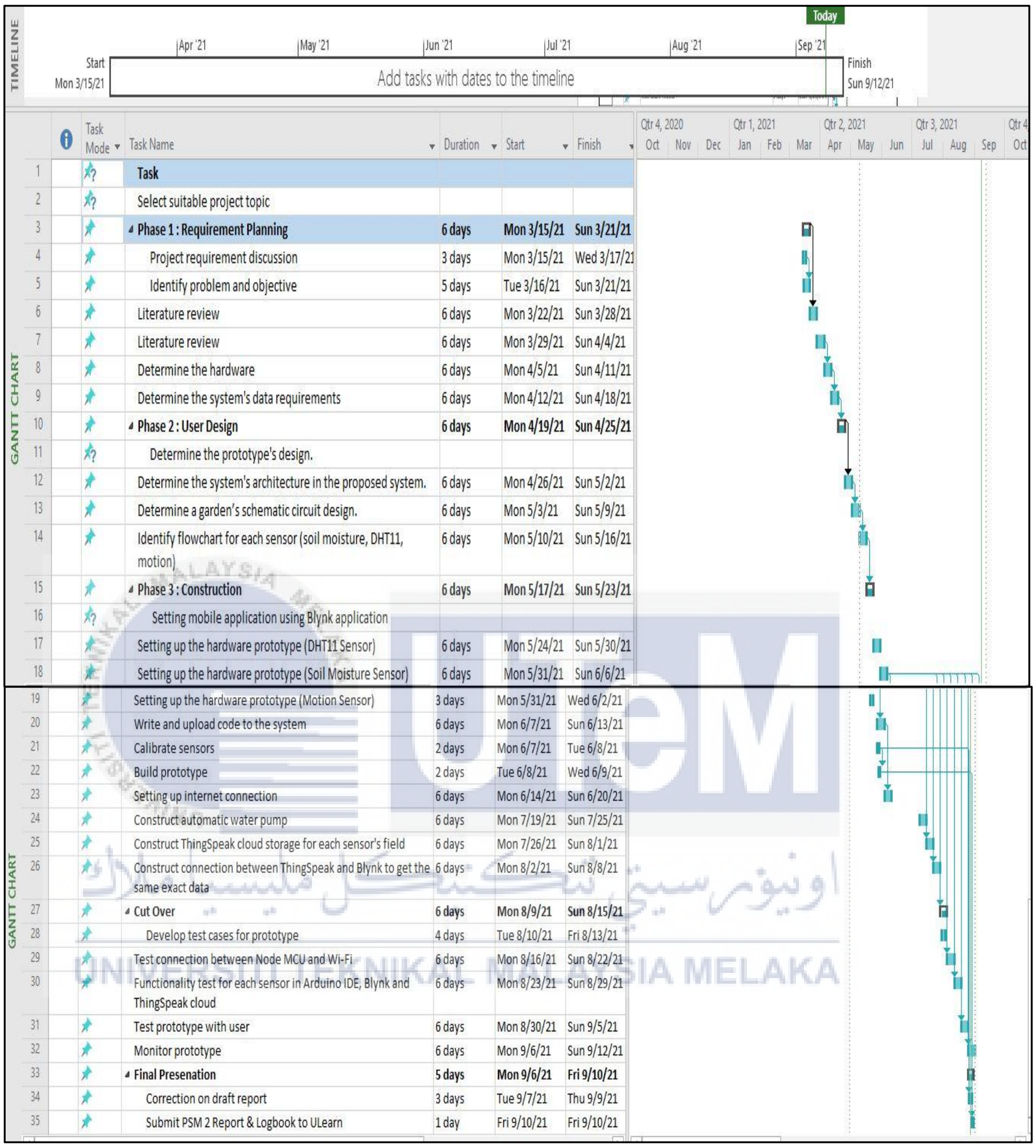
Project milestones are a type of management tool used to monitor the project's progress to identify a particular stage in a project's timeline. A project milestone is developed to ensure that all the project's tasks are accomplished within the specific time to ensure that the project runs smoothly and is finished on schedule. Project milestones organize the task into a series of related events with dates for each to be started and finished.

Table 3.1 Project Milestone

Phase	Activity
Phase 1 : Requirement Planning	✓ Select suitable project topic ✓ Project requirement discussion ✓ Identify problem and objective ✓ Literature review ✓ Determine the hardware (NodeMCU ESP8266, DHT11 sensor, motion sensor, buzzer, LED, soil moisture sensor, relay switch, water pump and Battery) and software requirements (Blynk, ThinkSpeak, Arduino IDE). ✓ Determine the system's data requirements
Phase 2 : User Design	✓ Determine the prototype's design. ✓ Determine the system's architecture in the

	proposed system. ✓ Determine a garden's schematic circuit design. ✓ Identify flowchart for each sensor (soil moisture, DHT11, motion)
Phase 3 : Construction	✓ Setting mobile application using Blynk application ✓ Setting up the hardware prototype (DHT11 sensors) ✓ Setting up the hardware prototype (Soil Moisture sensors) ✓ Setting up the hardware prototype (Motion sensors) ✓ Write and upload code to the system ✓ Calibrate sensors ✓ Build prototype ✓ Setting up internet connection ✓ Construct automatic water pump ✓ Construct ThingSpeak cloud storage for each sensor's field. ✓ Construct connection between ThingSpeak and Blynk to get the same exact data
Phase 4 : Cut Over	✓ Develop test cases for prototype ✓ Test connection between Node MCU and Wi-Fi ✓ Test for functionality of each sensors in real environment. ✓ Test prototype with users ✓ Monitor prototype

3.4 Project Gantt Chart



	Task Mode	Task Name	Duration	Start	Finish
1		**Task**			
2		Select suitable project topic			
3		**Phase 1 : Requirement Planning**	**6 days**	**Mon 3/15/21**	**Sun 3/21/21**
4		Project requirement discussion	3 days	Mon 3/15/21	Wed 3/17/21
5		Identify problem and objective	5 days	Tue 3/16/21	Sun 3/21/21
6		Literature review	6 days	Mon 3/22/21	Sun 3/28/21
7		Literature review	6 days	Mon 3/29/21	Sun 4/4/21
8		Determine the hardware	6 days	Mon 4/5/21	Sun 4/11/21
9		Determine the system's data requirements	6 days	Mon 4/12/21	Sun 4/18/21
10		**Phase 2 : User Design**	**6 days**	**Mon 4/19/21**	**Sun 4/25/21**
11		Determine the prototype's design.			
12		Determine the system's architecture in the proposed system.	6 days	Mon 4/26/21	Sun 5/2/21
13		Determine a garden's schematic circuit design.	6 days	Mon 5/3/21	Sun 5/9/21
14		Identify flowchart for each sensor (soil moisture, DHT11, motion)	6 days	Mon 5/10/21	Sun 5/16/21
15		**Phase 3 : Construction**	**6 days**	**Mon 5/17/21**	**Sun 5/23/21**
16		Setting mobile application using Blynk application			
17		Setting up the hardware prototype (DHT11 Sensor)	6 days	Mon 5/24/21	Sun 5/30/21
18		Setting up the hardware prototype (Soil Moisture Sensor)	6 days	Mon 5/31/21	Sun 6/6/21
19		Setting up the hardware prototype (Motion Sensor)	3 days	Mon 5/31/21	Wed 6/2/21
20		Write and upload code to the system	6 days	Mon 6/7/21	Sun 6/13/21
21		Calibrate sensors	2 days	Mon 6/7/21	Tue 6/8/21
22		Build prototype	2 days	Tue 6/8/21	Wed 6/9/21
23		Setting up internet connection	6 days	Mon 6/14/21	Sun 6/20/21
24		Construct automatic water pump	6 days	Mon 7/19/21	Sun 7/25/21
25		Construct ThingSpeak cloud storage for each sensor's field	6 days	Mon 7/26/21	Sun 8/1/21
26		Construct connection between ThingSpeak and Blynk to get the same exact data	6 days	Mon 8/2/21	Sun 8/8/21
27		**Cut Over**	**6 days**	**Mon 8/9/21**	**Sun 8/15/21**
28		Develop test cases for prototype	4 days	Tue 8/10/21	Fri 8/13/21
29		Test connection between Node MCU and Wi-Fi	6 days	Mon 8/16/21	Sun 8/22/21
30		Functionality test for each sensor in Arduino IDE, Blynk and ThingSpeak cloud	6 days	Mon 8/23/21	Sun 8/29/21
31		Test prototype with user	6 days	Mon 8/30/21	Sun 9/5/21
32		Monitor prototype	6 days	Mon 9/6/21	Sun 9/12/21
33		**Final Presenation**	**5 days**	**Mon 9/6/21**	**Fri 9/10/21**
34		Correction on draft report	3 days	Tue 9/7/21	Thu 9/9/21
35		Submit PSM 2 Report & Logbook to ULearn	1 day	Fri 9/10/21	Fri 9/10/21

3.5 Conclusion

Overall, this chapter concludes by describing the methodology or approach that is used in this project. The Rapid Application Development (RAD) model is applied to develop the system. The Rapid Application Development methodology consists of several phases that assist in developing the system in a more reliable and timely manner. The project milestones determine when the project is completed, ensuring that the project stays on schedule. This is important to ensure the project is completed on time. The next chapter aims to discuss the analysis and design for the project, including the problem analysis, hardware and software, and the project's design.

CHAPTER 4: ANALYSIS AND DESIGN

4.1 Introduction

This chapter discusses the design and the activities involved in the design phase. To ensure the functionality of the project, all the information hardware and software requirements used on the smart garden monitoring application technology project are stated. For the architecture, the process diagram is indicated as an essential step. This process is to identify whether the design can be operated or not.

4.2 Problem Analysis

Gardening is an activity that requires human supervision. The problem occurs when the current system's role of taking care of the garden is done manually. Monitoring the plant is cumbersome when done manually because it isn't systematic and has no monitoring devices. Other than that, people who cultivate gardens and other plants in their homes are always vigilant during the initial stages of care. As a result of this neglect, the garden is ruined. Climate conditions can also shorten the plant's life, as certain plants can die due to a lack of rain, extreme heat, or humidity. It also becomes difficult for individuals to monitor their gardens while they are away. The plant is neglected and stunted if it is left for an extended period. Their plants can be cared for from afar due to the capability of using a real-time monitoring system.

4.3 Requirement Analysis

This section describes in detail the requirement analysis involved in the project. The analysis of requirements is required because it is a task to determine the project's needs and conditions. This chapter explains the data requirement, functional requirements, and other hardware and software requirements.

4.3.1 Data Requirement

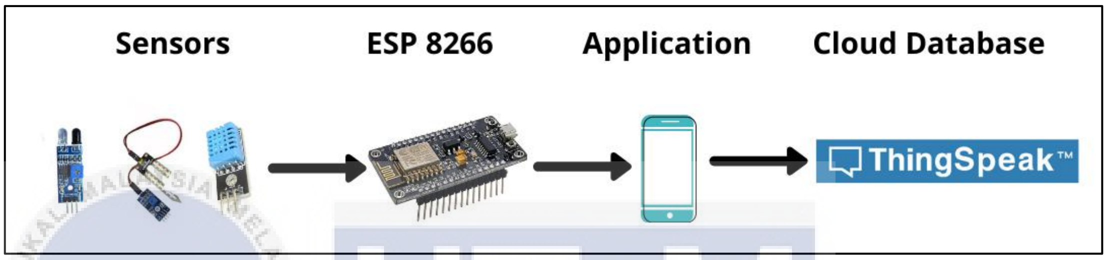


Figure 4.1 Data Flow Requirement

The project has a design that includes input and output results because the system interacts with the hardware. The system's input is the soil moisture sensor, humidity and temperature sensor, and motion sensor. The soil moisture sensor is used to detect a parameter of the level of moisture of the soil. A humidity and temperature sensor is used to detect the humidity and temperature of the garden environment. A motion detector sensor measures changes and detects the movement of animals being in a monitored garden area. The water pump begins to pump out water to the soil moisture sensor after received input below the threshold from the moisture sensor. The detected data from the sensor display on the Blynk application and ThingSpeak cloud storage.

4.3.2 Functional Requirement

Based on the block diagram in Figure 4.2, the proposed smart garden monitoring application in functional block diagram. It consists of various functional blocks, including input, microcontroller, and output blocks.

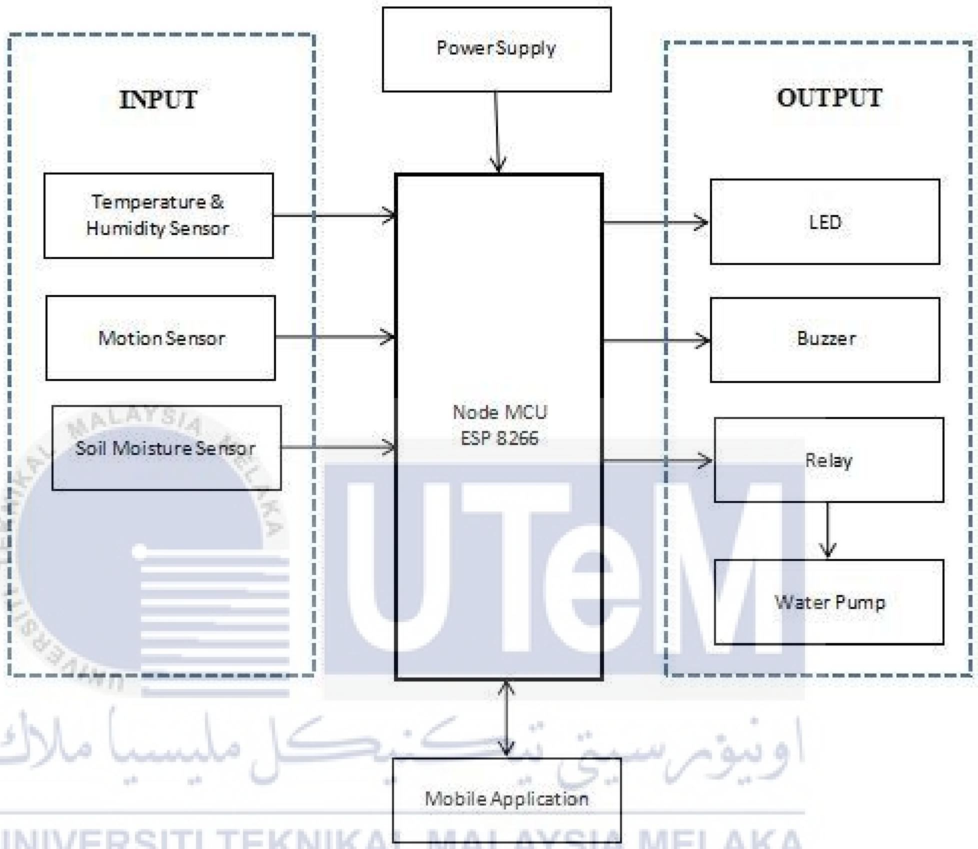


Figure 4.2 Functional Block Diagram

- **Input Block :** This block consist of 3 sensors that collect data from the garden's parameter. The sensor includes a motion sensor, soil moisture sensor, and temperature and humidity sensor.

- **Microcontroller Block :** NodeMCU is the microcontroller in this block, which is the project's main hardware. It collects data from sensors and processes it according to the requirements programmed into the microcontroller.

- **Output Block :** This block includes the system's LED, buzzer, and automatic watering function which have relay and water pump.

4.4 Hardware Requirement

This section identifies the hardware items necessary for the experiment that fulfills a certain specification in the project.

4.4.1 Node MCU ESP 8266

This project use Node MCU which is cost-effective that has an inbuilt ESP 8266 Wi-Fi module. Node MCU acts as a microcontroller programmed to read the sensors data and send the data through ESP 8266 Wi-Fi module to the user's smartphone.

Figure 4.3 Node MCU ESP 8266

4.4.2 Soil Moisture Sensor

This project use a moisture sensor to measure the soil's moisture content in the garden. The sensor is inserted through the soil using the two probes and then detects the resistance to determine the moisture level. In this project, if the sensor detects the soil moisture below 38%, it is considered dry. Otherwise, if the sensor detects the soil moisture above 45%, it is considered wet.

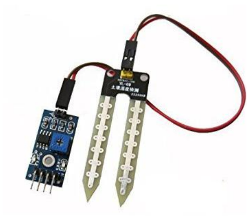


Figure 4.4 Soil Moisture Sensor

4.4.3 DHT11 Humidity and Temperature Sensor

The DHT11 sensor is a basic and low-cost digital temperature and humidity sensor. The DHT11 is a combination of humidity and temperature sensor. This project uses the DHT11 sensor to measure the air surrounding the garden environment. It measures the air around it using a capacitive humidity sensor and a thermistor and outputs a digital signal on the data pin.

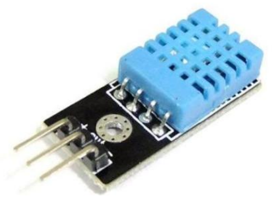


Figure 4.5 DHT11 Humidity and Temperature Sensor

4.4.4 IR Motion Sensor

IR Motion Sensor is a type of sensor to detect motion. It consists of 3 pins attached to the sensor: GND, Output, and DC voltage. The IR Motion Sensor is connected to the Node MCU ESP 8266 by using the Female-to-Male jumper wire. This sensor is used in this project to sense an intruder and alert the user's smartphone. The motion sensor detects any motion caused by animals or humans moving around the garden environment.

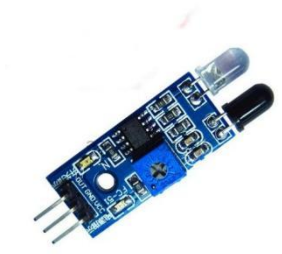


Figure 4.6 IR Motion Sensor

4.4.5 Relay

The channel module relay is used to operate high-voltage equipment (such as the water pump) and acts as a switch to turn the water pump on or off. A 3-way screw terminal brings out the relay terminals (COM, NO, and NC). The onboard relay has a 10A rating. It may be used to manage high voltage, high current loads such as motors, solenoid valves, lights, and AC loads, among other things. It also has two LEDs that show the state of the power supply (VCC) and the relay condition.

Figure 4.7 Relay

4.4.6 Water Pump

This project uses the micro submersible water pump to pump water out of the water container. The water in the water container is pumped out using the water pump. The motors of this pump operate at a DC voltage. A tube drain of 7.5 millimeters, an internal diameter of 4.7 millimeters, and a maximum stroke of 40 to 110 centimeters can pump 80 to 120 liters of water per 51 hours. The power supply operates at a voltage of up to 6 volts, with current usage of up to 220 milliamps when loaded (mA).

Figure 4.8 Water Pump

4.4.7 Buzzer

A buzzer is an audio communication system, either electronic or piezoelectric. Warning systems and timers are uses for buzzers and beepers. This buzzer is used in this project to give an audible warning if a motion sensor detects the motion caused by intruders.

Figure 4.9 Buzzer

4.4.8 Light-Emitting Diode (LED)

A semiconductor device, which is an LED, emits light when the electrical current passes through it. Light is created when the particles holding the current are integrated into the semiconductor material. This project use LED as a warning if the motion sensor detects movement, such as the LED turn ON if motion is detected.

Figure 4.10 LED

4.4.9 Breadboard

The breadboard shown in Figure 4.11 is a solder-less device used to prototype electronics and test circuit designs. Interconnected electrical components in electronic circuits can be done by inserting leads or terminals into the holes and making connections using wires.

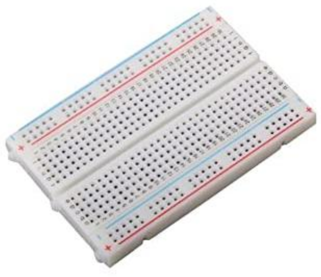


Figure 4.11 Breadboard

4.4.10 Jumper Wires

Jumper wires are wires with connector pins at each end that two points can be connected without soldering. This project uses jumper wires are used with breadboards and other prototyping devices to make it easy to modify a circuit.

Figure 4.12 Jumper Wire

4.4.11 USB Cable

The Arduino USB cable connects the Node MCU ESP 8266 or any other board to a USB female port on a computer. The cable is approximately 178 cm long. Then there's this USB 2.0 cable type A / B standard.

Figure 4.13 USB Cable

4.5 Software Requirement

This section identifies the specific software involved to perform the project activities. The main programming language used was C++. C++ was the primary programming language used. It was for the system's code development. It is integrated with numerous libraries to fulfill the system's functions.

4.5.1 Arduino IDE

This project uses Arduino IDE (Integrated Development Environment) to write and upload code to Node MCU compatible boards and communicates with them. The Arduino IDE program software can be written in the C and C++ language.

Figure 4.14 Arduino IDE

4.5.2 Blynk

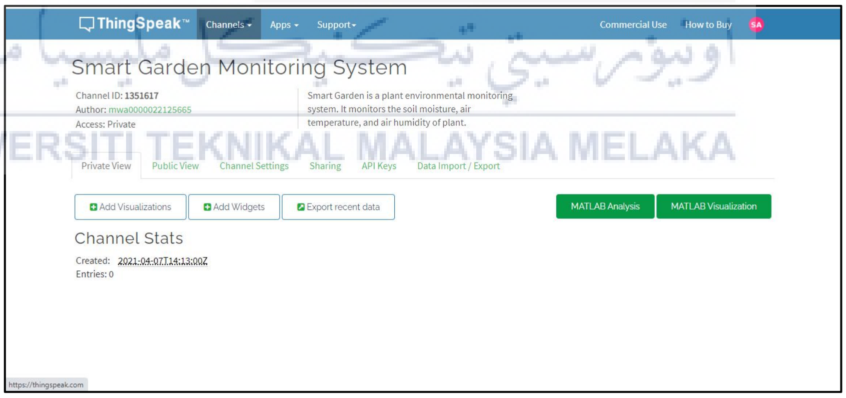
Blynk is an iOS and Android platform to control Node MCU through the internet. It can remotely control hardware, display, and read sensor data. It's a virtual dashboard that provides developers to build a project's graphical interface by dragging and dropping widgets.

Blynk

Figure 4.15 Blynk

4.5.3 ThingSpeak

ThingSpeak is a platform IoT analytic service that enables aggregate, view, and analysis of live data streams in the cloud. ThingSpeak provides an instant display of information posted to ThingSpeak by the devices.



Smart Garden Monitoring System

Channel ID: 1351617
Author: mwa0000022125665
Access: Private

Smart Garden is a plant environmental monitoring system. It monitors the soil moisture, air temperature, and air humidity of plant.

Channel Stats

Created: 2021-04-07T14:13:00Z
Entries: 0

https://thingspeak.com

Figure 4.16 ThingSpeak

4.5.4 Fritzing

Fritzing is open-source software for prototyping electrical components. Fritzing software allows the user to create a schematic and hence a separate diagram attached to highly professional-looking wiring circuit designs.

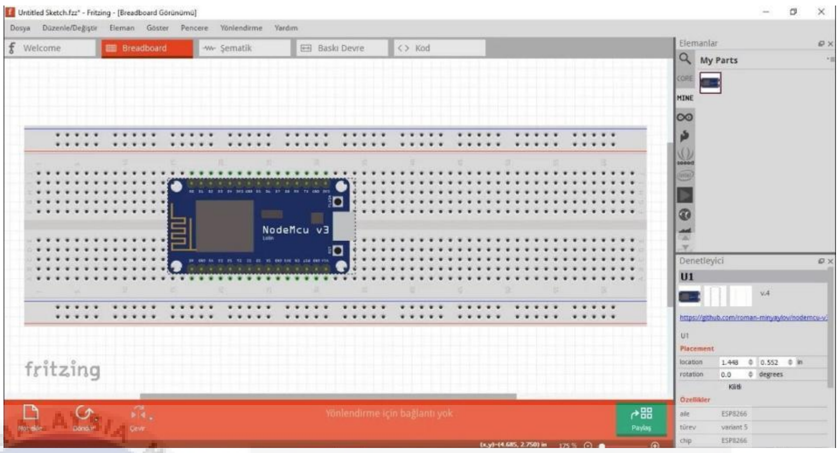


Figure 4.17 Frtizing

4.6 High Level Design

High-level design explains the architecture used to build a product. This section shows the detailed design of the project. It contains the structure view of the system to be developed, including the system architecture design, schematic circuit diagram, interface design, and flow chart of the system.

4.6.1 System Architecture

The following Figure 4.18 illustrates the system architecture for Smart Garden Monitoring Application. The sensors are integrated with Node MCU ESP 8266, and then the data is monitored through a mobile application. This project uses the soil moisture sensor to detect a parameter of the level of moisture of the soil. If the sensor detects the soil is dry, it gives a notification to the user to water the plant. The water pump begins to pump out water to the soil moisture sensor after receiving input below the moisture sensor threshold. A humidity and temperature sensor is used to detect the humidity and temperature of the garden environment. A motion detector sensor measures changes and detects the movement of animals being in a monitored garden area. The sensor's detected data is displayed on the application and monitored by the ThingSpeak cloud database for further analysis.

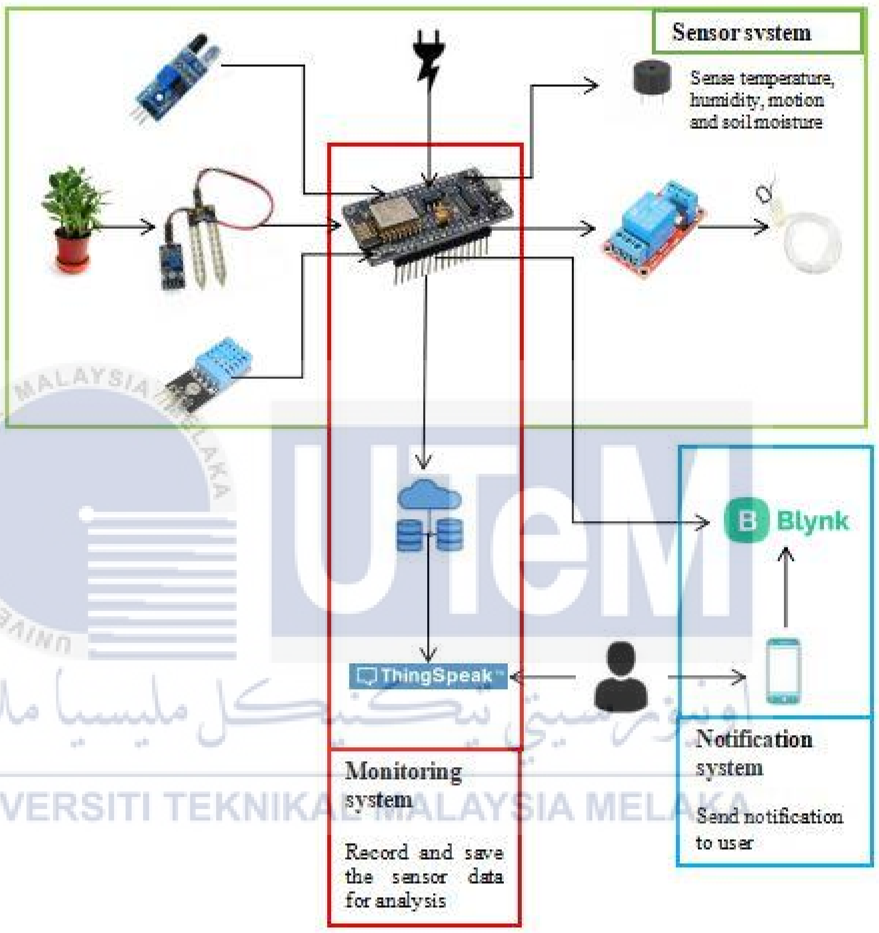

Sensor system
Sense temperature, humidity, motion and soil moisture
Blynk
ThingSpeak
Monitoring system
Record and save the sensor data for analysis
Notification system
Send notification to user

Figure 4.18 System Architecture Diagram

The first function is the sensor system. The Node MCU ESP 8266 is connected to the sensor. The hardware needed includes Node MCU microcontroller, soil moisture sensor, DHT11 temperature and humidity sensor, IR motion sensor, buzzer, LED light, relay, 5v battery, and water pump. The Node MCU is connected to the PC via the USB cable while the rest of the hardware is connected to the Node MCU using jumper wires.

Next, the second function is the monitoring system, which uses Thingspeak. When the data is received, it stores it in the ThingSpeak cloud for monitoring for future references. Users need to log in to access the sensor data. Then, the user can view and monitor sensor data using the graphical form on ThingSpeak cloud.

Table 4.1 Monitoring System Detail

Monitoring System	
Soil sensor	i. Sensor monitor soil moisture (dry or wet). ii. Send, update and store data from Node MCU ESP 8266 to ThingSpeak about motion history.
Motion sensor	i. Sensor monitor motion from animal that enter garden area. ii. Send, update and store data from Node MCU ESP 8266 to ThingSpeak about motion history.
DHT11 sensor	i. Sensor monitor temperature and humidity of current condition of garden. ii. Send, update and store data from Node MCU ESP 8266 to ThingSpeak about temperature and humidity history.

The third system is a notification system. The system sends the notification to the user's smartphone. The user receives a notification that the watering device has been disabled when it has been deactivated. The details are as below:

Table 4.2 Notification System Detail

Notification System	
Soil sensor	i. Watering systems is turning on or off. ii. The Blynk app receives data from the Nodemcu, analyses it, and sends a notification to the user's smartphone. iii. Blynk apps give alert and notification to user.
Motion sensor	i. Motion detection sensor detect motion. ii. Blynk apps give alert and notification to user.
DHT11 sensor	i. Blynk apps display temperature and humidity of current condition of garden.

4.6.2 Schematic Circuit Diagram

Figure 4.19 shows the sketch circuit design for Smart Garden Monitoring Application using Fritzing software. The design includes various sensors connected to the main microprocessor, Node MCU ESP 8266. The sensor consists of a soil moisture sensor, temperature and humidity sensor, and motion sensor. The schematic diagram also has a relay as a switch to control the water pump. The water pump is connected with a 4xAA battery as a power supply to the water pump. There is also a buzzer and LED as an output to the motion sensor development.

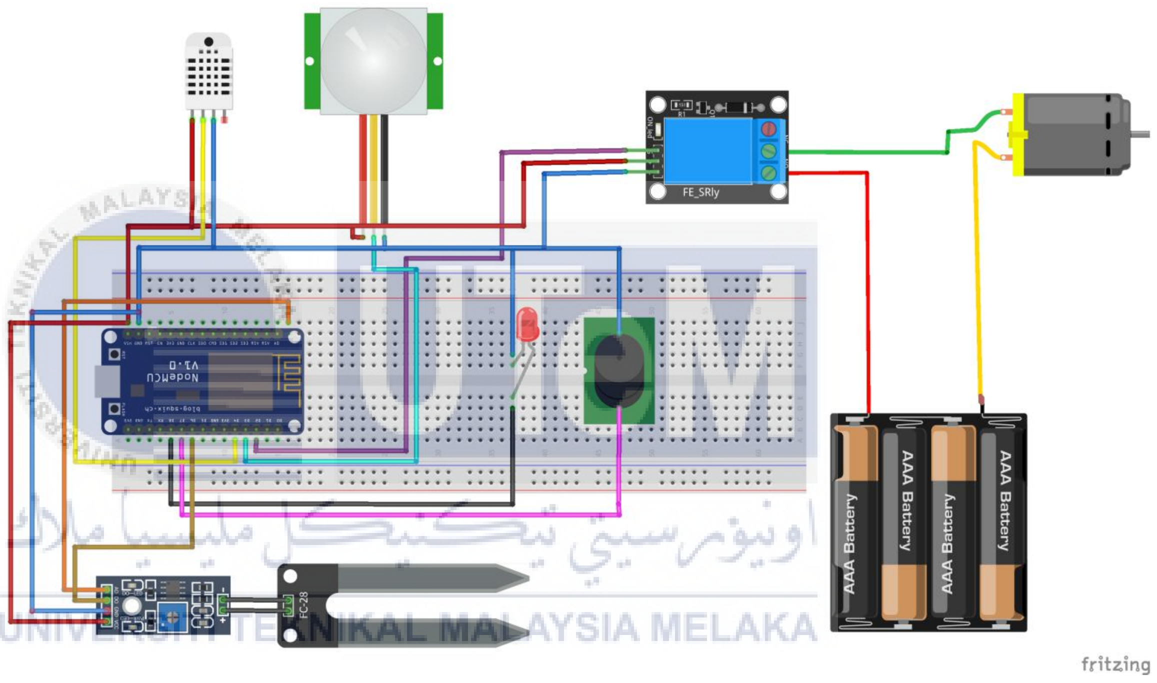


Figure 4.19 Schematic Circuit Diagram

4.6.3 User Interface Design

For this project, a graphical user interface design (UI) is used to become successful and increasingly competent gradually. The UI for this application is user-friendly so that the user can indeed access it. The user interface design for Smart Garden Monitoring Application is based on the Blynk application. Figure 4.20 shows the user design of the Blynk application. The design displays the parameter of the plant in the Garden.

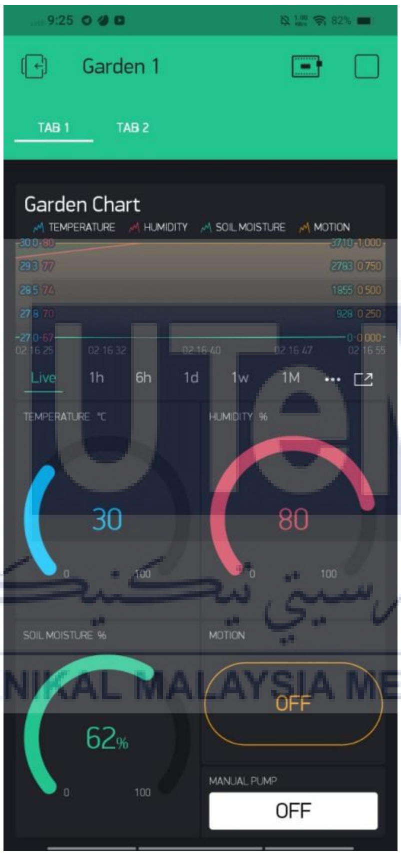


Figure 4.20 Interface Design

4.6.4 Flowchart

Figure 4.21 below shows the flow chart of the system. The system starts with all sensors detect the data parameter of the plant. The data is transfer to Node MCU ESP 8266 and display to the mobile application. If the data of moisture soil is below the threshold, a notification is sent to the user. The user may take suitable action to make sure the parameter of the plant gets enough parameters. If the data reach the level parameter, the process is repeated to analyze the data.

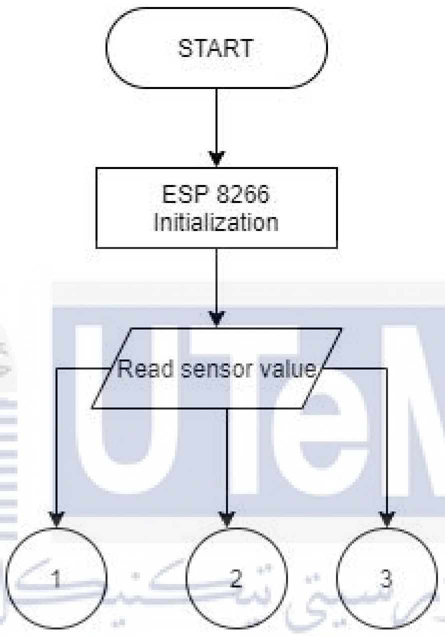


Figure 4.21 Overall Flowchart

4.6.4.1 Flowchart for Soil Moisture Sensor

Figure 4.22 shows the soil moisture sensor flowchart. This shows how this sensor can run through this system by using Node MCU ESP 8266. Soil moisture sensors need to connect with ESP 8266 microcontroller first to detect the parameter of soil moisture. If the sensor detects that the moisture of the soil is below the threshold, the sensor gives notification to water the plant by pump on the water pump using relay module. Then, the sensor can display the data through the mobile application.

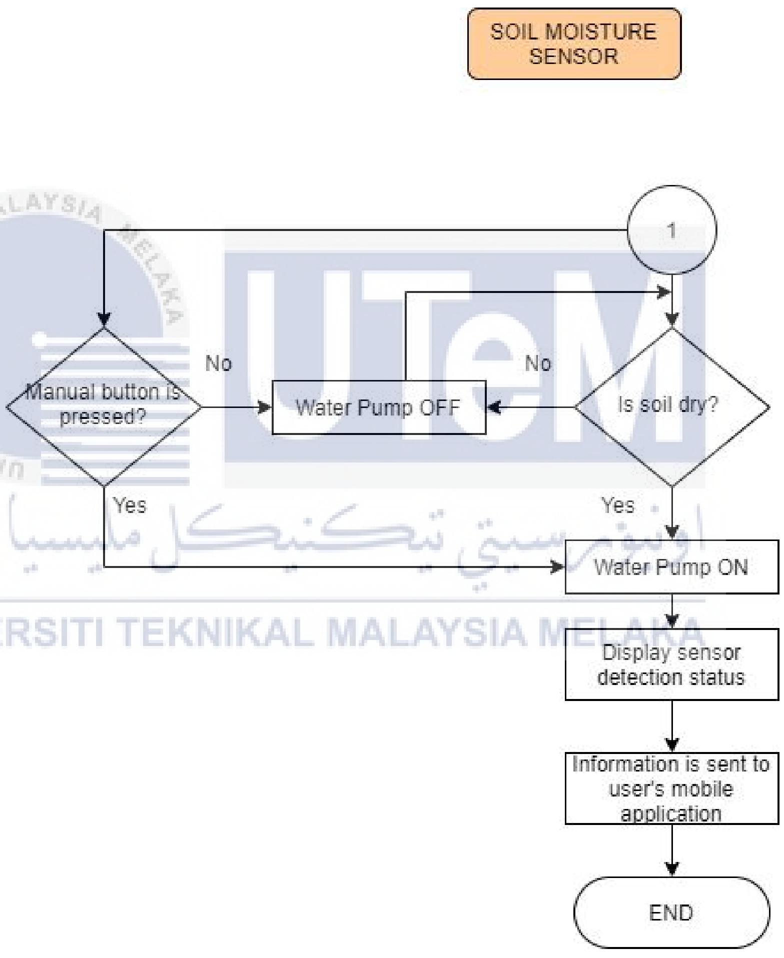

SOIL MOISTURE SENSOR
1
Manual button is pressed?
No
Water Pump OFF
No
Is soil dry?
Yes
Yes
Water Pump ON
Display sensor detection status
Information is sent to user's mobile application
END

Figure 4.22 Flowchart Soil Moisture Sensor

4.6.4.2 Flowchart for DHT11 Temperature & Humidity Sensor

Figure 4.23 shows the DHT11 temperature and humidity flowchart. This shows how this sensor can run through this system. DHT11 sensor needs to connect with ESP 8266 microcontroller first to have access to detect the condition of the environment. Then, the sensor can display the data through the mobile application.

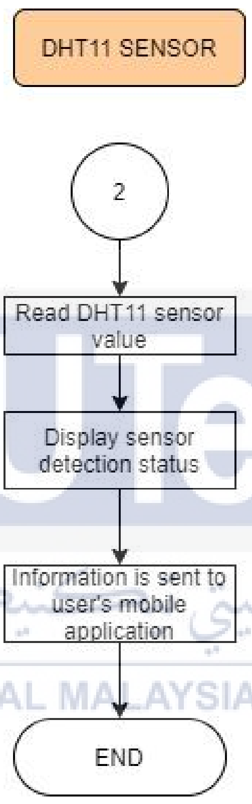

DHT11 SENSOR
2
Read DHT11 sensor value
Display sensor detection status
Information is sent to user's mobile application
END

Figure 4.23 Flowchart DHT11 Sensor

4.6.4.3 Flowchart for Motion Sensor

Figure 4.24 shows the motion sensor flowchart. The flowchart shows how the sensor can run through this system. The sensor needs to connect with ESP 8266 microcontroller first to detect any movement from within the environment. Then, the sensor can display the data through the mobile application. If the sensor detects any movement, the sensor turn on the buzzer and displays output to the user's mobile application.

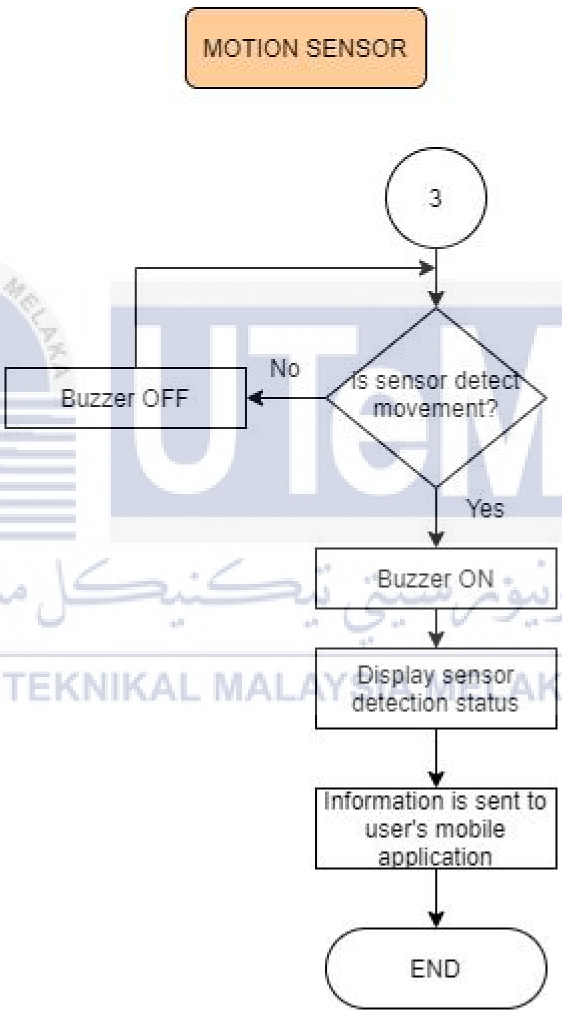

MOTION SENSOR
3
Is sensor detect movement?
No
Buzzer OFF
Yes
Buzzer ON
Display sensor detection status
Information is sent to user's mobile application
END

Figure 4.24 Flowchart IR Motion Sensor

4.7 Conclusion

In conclusion, this chapter is one of the crucial chapters before proceeding to the next chapter, which is implementation. This is because all the requirements, including hardware and software requirements, are collected in this chapter. The Node MCU ESP 8266's features have previously revealed why it was chosen as a microcontroller because it have an inbuilt Wi-Fi module. Other components, such as the sensors, LED, and buzzer, it also has been discussed. The source code must be well-organized to get data from the sensor to the Smart Garden Monitoring Application. A prototype of a smart garden monitoring application during the design phase. The next chapter discussed the implementation phase of the project Smart Garden Monitoring Application.

CHAPTER 5: IMPLEMENTATION

5.1 Introduction

This chapter focuses on implementing the Smart Garden Monitoring Application in both software and hardware implementation development. System implementation is a phase where the design results are transferred to software code and logical procedure. The system is tested in this phase to determine that the development of the system meets the standard criteria. The Implementation method involves developing and testing a functioning system that meets organizational design criteria and implementing the system's interface with existing systems. It must construct the database, application programs, user and system interfaces.

5.2 Development Environment Setup

This section describes the project development environment setup in-depth to help users understand the project. The project's environment setup involves:

5.2.1 Hardware Development Setup

There is some essential hardware that is needed in the development of the Smart Garden Monitoring Application. Table 5.1 shows the detail pins that are connected to the Node MCU.

Table 5.1 Sensor Pin Detail to Node MCU pin

Hardware	Wire	Node MCU Pins
Soil Moisture Sensor	VCC	3V3
	GND	GND
	A0	A0
	D0	D6
DHT11 Sensor	VCC	Vin
	GND	GND
	Data	D4
Motion Sensor	VCC	3V3
	GND	GND
	Out	D3
Buzzer	Signal (+)	D7
	Power (-)	GND
LED	Signal (+)	D8
	Power (-)	GND
Relay	DC+	Vin
	DC-	GND
	IN	D2
Water Pump	Battery Wire	NO
	Pump Wire	COM

5.2.2 Prototype Implementation

A proposed model is developed for the Smart Garden Monitoring Application. A prototype hardware design for the system is shown in Figure 5.1. From Figure 5.1, it shows the Node MCU ESP 8266 is the center core of this system which connects all the required hardware. The sensors involved in this project are the soil moisture sensor that measures the level of moisture from the soil, the motion sensor that detects the presence of motion, and the temperature and humidity sensor that sense the plant's environment.

The value that the Node MCU received from the sensors is displayed on the application display. Concurrently, the obtained data is delivered to the relay module, which determines whether the water pump should be turned on or off. If the condition requires turning on the water pump, it begins to draw water from the water tank and push it to the opposite side of the water pipe, completing the soil watering process.

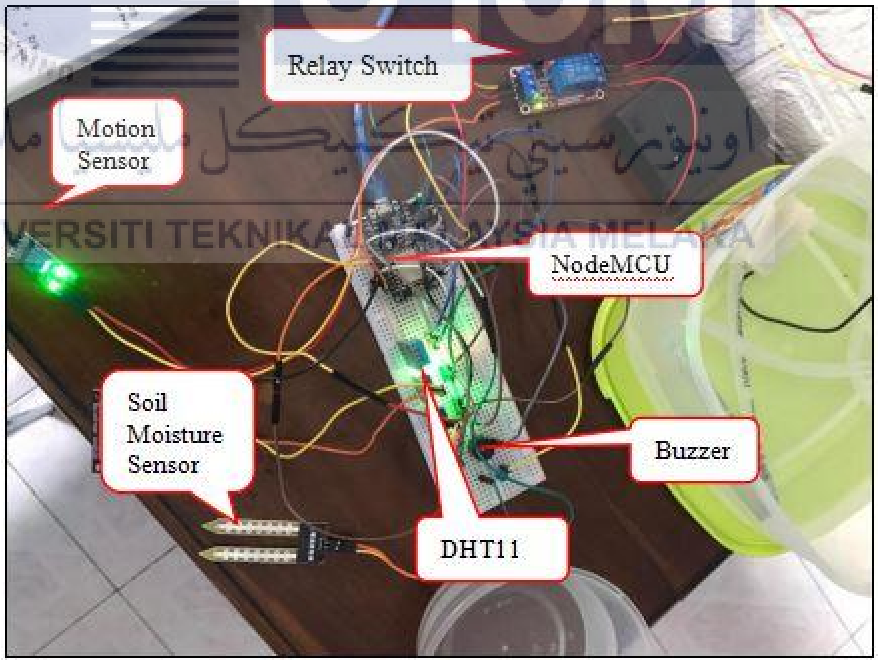


Figure 5.1 Prototype Hardware Design

Figure 5.2 shows the final prototype to demonstrate the real environment of the garden. This prototype is set up with all of the necessary hardware. The hardware connection, such as Node MCU ESP8266 and sensors, is connected to the PC via a USB port. The actual prototype is developed and shown in Figure 5.2.

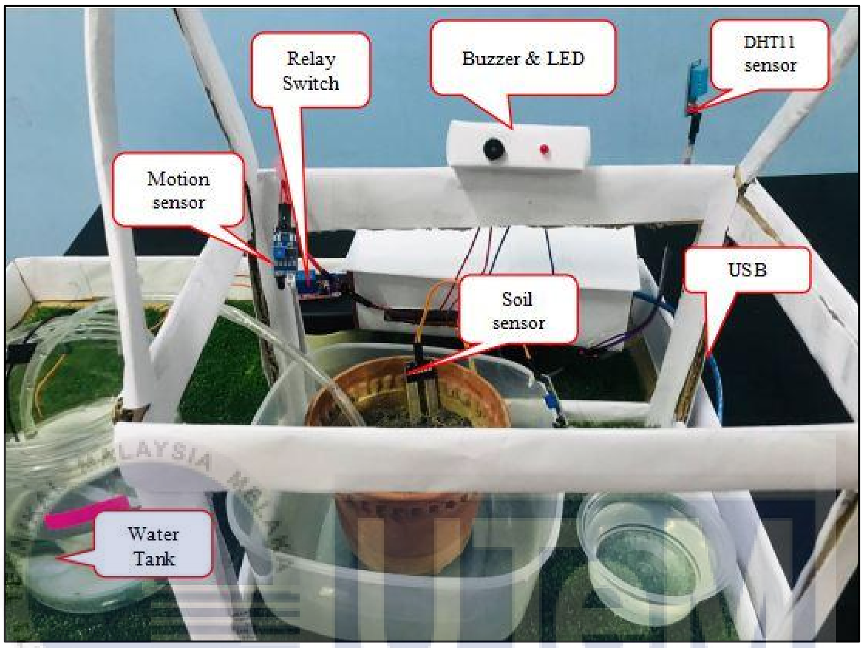


Figure 5.2 Hardware Development Setup (Side View)

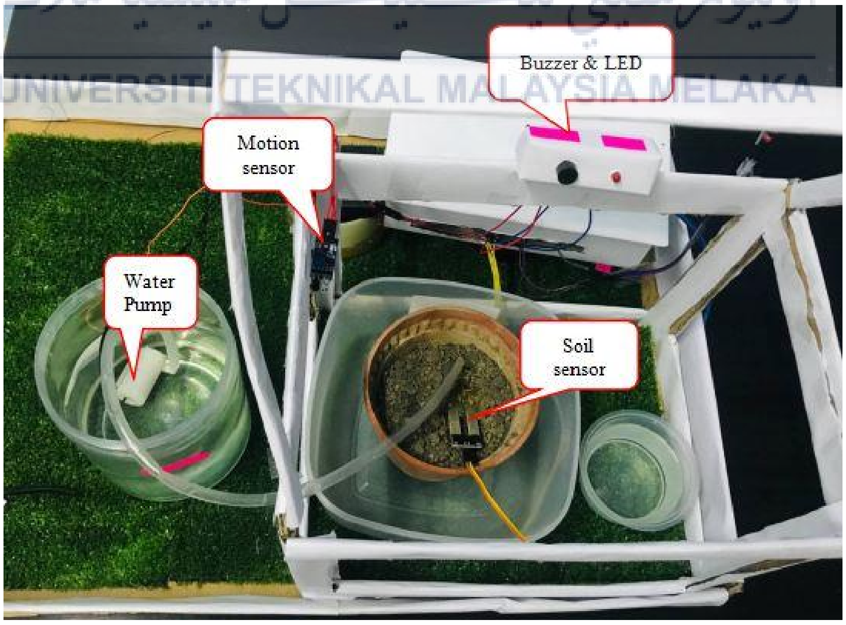


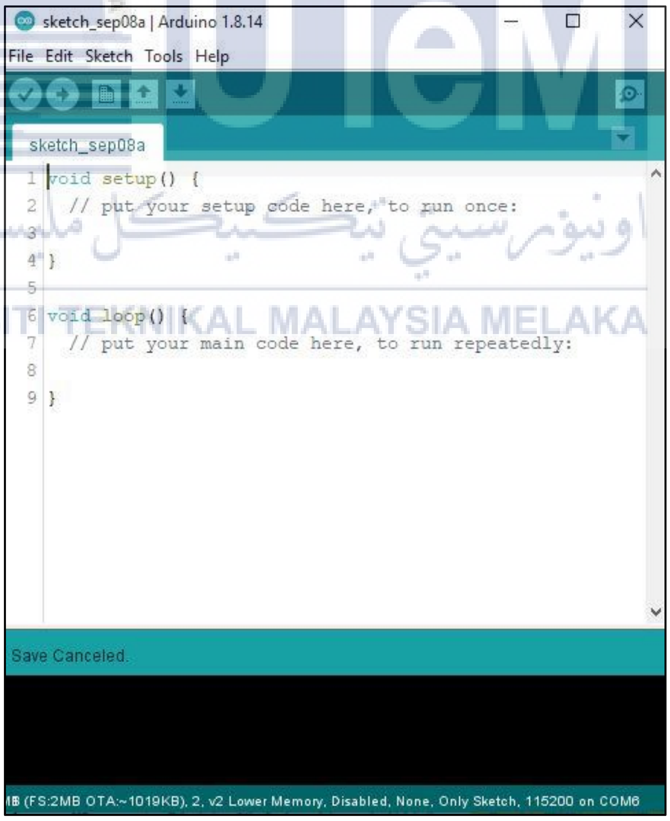
Figure 5.3 Hardware Development Setup (Top View)

5.2.3 Software Development Setup

As defined in Chapter 4, there are several software needed to be installed for this project.

1. Arduino IDE Setup

The Arduino IDE is installed first, whereby the board management and library are downloaded from the internet. Node MCU is used as the board management, and numerous libraries were included in this project. The Arduino IDE is a software program used in this project to program the Node MCU board. When a Node MCU is connected to a sensor, it can display and calculate data that can be viewed on the Arduino IDE serial monitor.



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

}

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

}
```

```
Save Canceled.
```

```
(FS:2MB OTA:~1019KB), 2, v2 Lower Memory, Disabled, None, Only Sketch, 115200 on COM6
```

Figure 5.4 Arduino IDE

2. Blynk Application Setup

Blynk is an Android service that allows users to create a simple interface to control and monitor the hardware project. Blynk is used in this project as a monitoring, reporting, and graphing application for moisture, temperature, humidity, and motion.

a) Firstly, install blynk applications for free on Google Play.

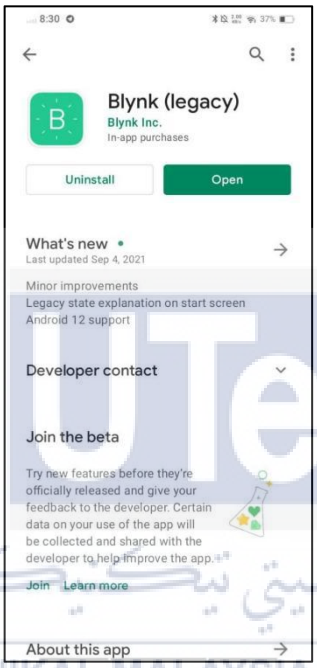

8:30
Blynk (legacy)
Blynk Inc.
In-app purchases
Uninstall
Open
What's new
Last updated Sep 4, 2021
Minor improvements
Legacy state explanation on start screen
Android 12 support
Developer contact
Join the beta
Try new features before they're officially released and give your feedback to the developer. Certain data on your use of the app will be collected and shared with the developer to help improve the app.
Join
Learn more
About this app

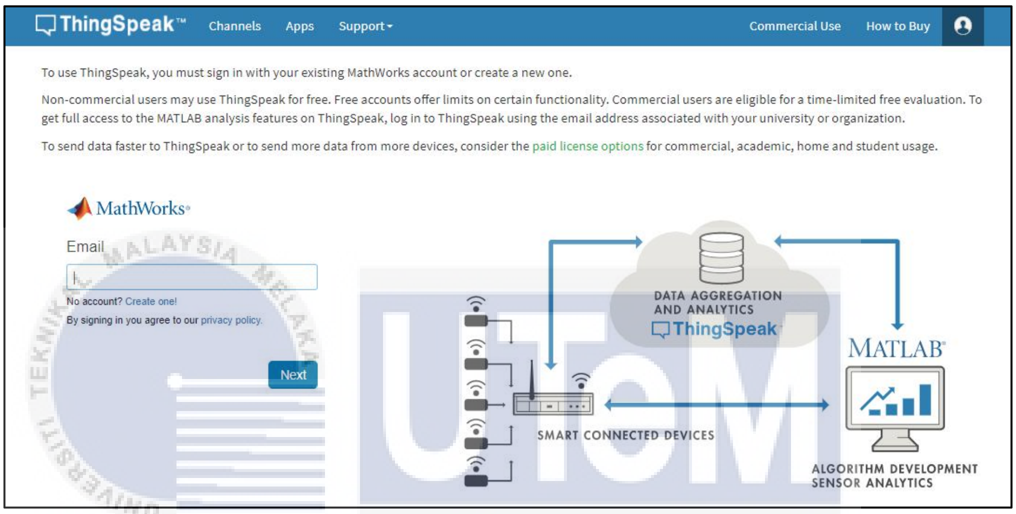
Figure 5.5 Blynk installation

b) To begin designing the interface, open the Blynk apps and log in with the Blynk account. Drag and drop the interface to any location on the blank interface. The verified password sent by Blynk into the email account must be used to connect Node MCU and Blynk.

3. ThingSpeak Cloud Storage Setup

ThingSpeak is a real-time cloud platform that allows application developers to synchronize data between clients and save it in the ThingSpeak cloud via an API. ThingSpeak is free and does not require installation on the computer.

a) Firstly, go to the ThingSpeak cloud website to establish a new real-time channel,



To use ThingSpeak, you must sign in with your existing MathWorks account or create a new one.

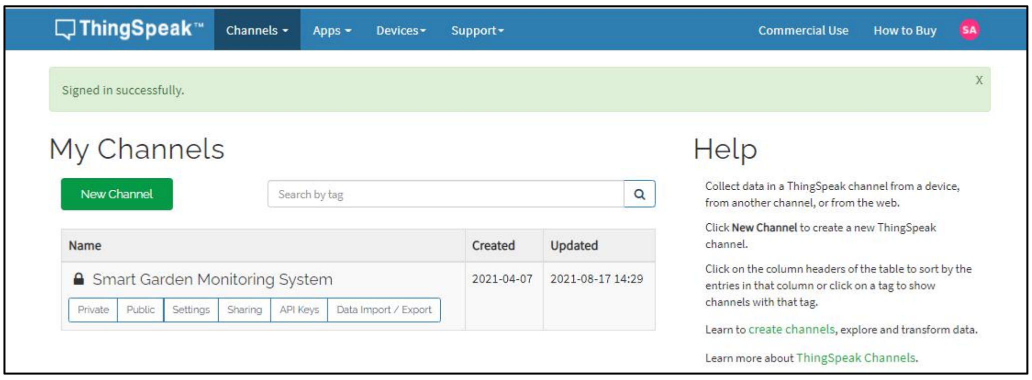
Non-commercial users may use ThingSpeak for free. Free accounts offer limits on certain functionality. Commercial users are eligible for a time-limited free evaluation. To get full access to the MATLAB analysis features on ThingSpeak, log in to ThingSpeak using the email address associated with your university or organization.

To send data faster to ThingSpeak or to send more data from more devices, consider the paid license options for commercial, academic, home and student usage.

MathWorks
Email
No account? Create one!
By signing in you agree to our privacy policy.
Next
DATA AGGREGATION AND ANALYTICS
ThingSpeak
MATLAB
SMART CONNECTED DEVICES
ALGORITHM DEVELOPMENT SENSOR ANALYTICS

Figure 5.6 ThingSpeak interface

b) Sign in and begin developing a new project channel.



Signed in successfully.

My Channels

Name	Created	Updated
Smart Garden Monitoring System	2021-04-07	2021-08-17 14:29

Help

Collect data in a ThingSpeak channel from a device, from another channel, or from the web.

Click **New Channel** to create a new ThingSpeak channel.

Click on the column headers of the table to sort by the entries in that column or click on a tag to show channels with that tag.

Learn to create channels, explore and transform data.

Learn more about ThingSpeak Channels.

Figure 5.7 ThingSpeak Channel

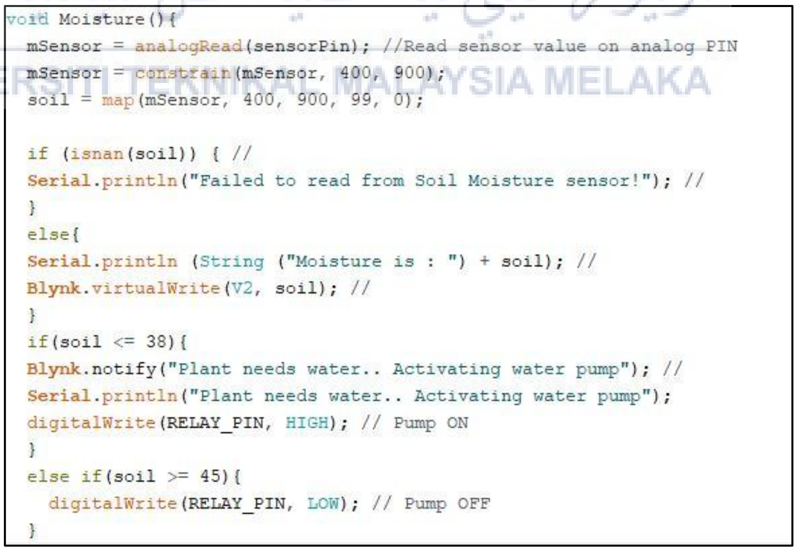
5.3 Software Configuration Management

The system's configuration and code implementation are described in this section. Once the hardware is connected, the Node MCU is configured with code to communicate with the device using the Arduino IDE. The following code is used to configure the Node MCU program.

5.3.1 Arduino IDE Configuration

5.3.1.1 Code Fragment Soil Moisture Sensor

The code fragment in Figure 5.8 is used to print the percentage received from the soil moisture sensor to assess whether the soil is wet or dry. The sensor's value also determines whether the water pump is turned on or off. The code fragment transforms the upper bound 400 to 100 percent and the lower bound 900 to 0 percent. When the soil moisture percentage falls below 38%, automatically water pump turns ON and sends a notification to Blynk apps. This system has set the value for the watering process. The water pump turns off automatically when the soil moisture percentage exceeds 45%, and the Blynk application indicates the water pump state as 'OFF.



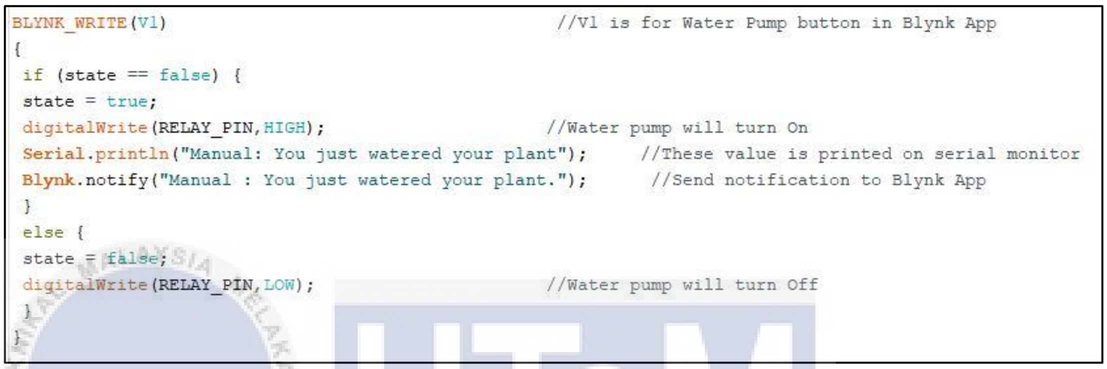
```
void Moisture(){
  mSensor = analogRead(sensorPin); //Read sensor value on analog PIN
  mSensor = constrain(mSensor, 400, 900);
  soil = map(mSensor, 400, 900, 99, 0);

  if (isnan(soil)) { //
  Serial.println("Failed to read from Soil Moisture sensor!"); //
  }
  else{
  Serial.println (String ("Moisture is : ") + soil); //
  Blynk.virtualWrite(V2, soil); //
  }
  if(soil <= 38){
  Blynk.notify("Plant needs water.. Activating water pump"); //
  Serial.println("Plant needs water.. Activating water pump");
  digitalWrite(RELAY_PIN, HIGH); // Pump ON
  }
  else if(soil >= 45){
    digitalWrite(RELAY_PIN, LOW); // Pump OFF
  }
```

Figure 5.8 Code Fragment for Soil Moisture Configuration

5.3.1.2 Code Fragment Water Pump Switch

The code in Figure 5.9 shows the manual water pump when the user clicks the button in the Blynk application. The V1 is indicated as Virtual Pin1 in the Blynk Application, which controls the relay switch to ON or OFF. If the user clicks the manual button for the watering process on the Blynk application, then the water pump is ON and sends a notification about manual watering.

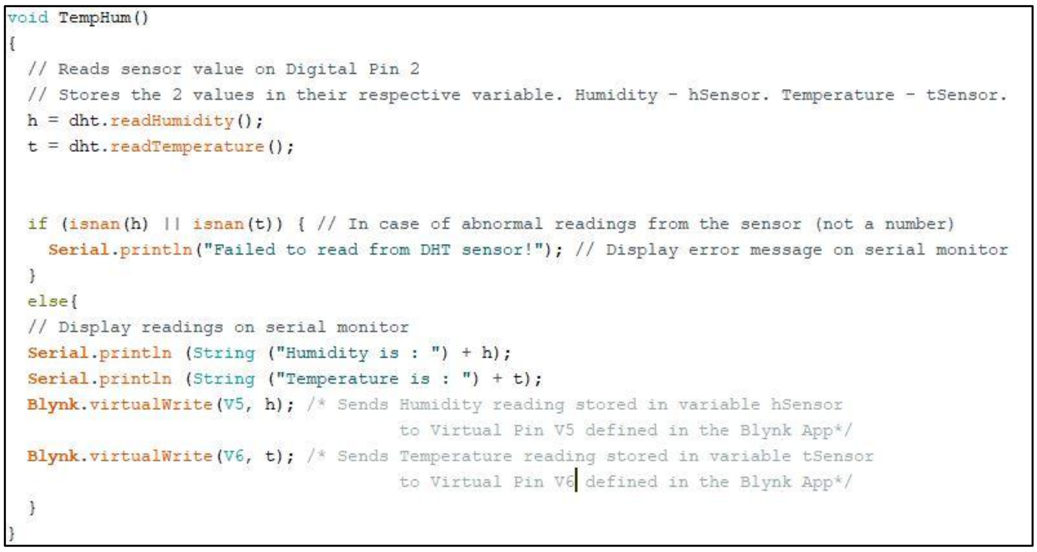


```
BLYNK_WRITE(V1)                                     //V1 is for Water Pump button in Blynk App
{
 if (state == false) {
 state = true;
 digitalWrite(RELAY_PIN,HIGH);                     //Water pump will turn On
 Serial.println("Manual: You just watered your plant");     //These value is printed on serial monitor
 Blynk.notify("Manual : You just watered your plant.");     //Send notification to Blynk App
 }
 else {
 state = false;
 digitalWrite(RELAY_PIN,LOW);                      //Water pump will turn Off
 }
}
```

Figure 5.9 Code Fragment for Water Pump Configuration

5.3.1.3 Code Fragment DHT11 Sensor

The code fragment in Fig. 5.10 is used to print the temperature and humidity of the garden environment from the DHT11 sensor for monitoring purposes. The function void TempHum has two variables to monitor, which are temperature and humidity. The ‘t’ is indicated as temperature while ‘h' is indicated as humidity. The ‘V5’ is indicated as Virtual Pin5 defined in the Blynk application for humidity. While for ‘V6’, it is indicated as a Virtual Pin6 variable in the Blynk Apps. If there are abnormal readings from the DHT11 sensor, it displays an error message on the serial monitor. Otherwise, if the sensor detects the reading temperature and humidity data, it displays the readings on a serial monitor. Then, after the reading data is displayed on the serial monitor, it shows the Blynk application data for the user’s monitoring.



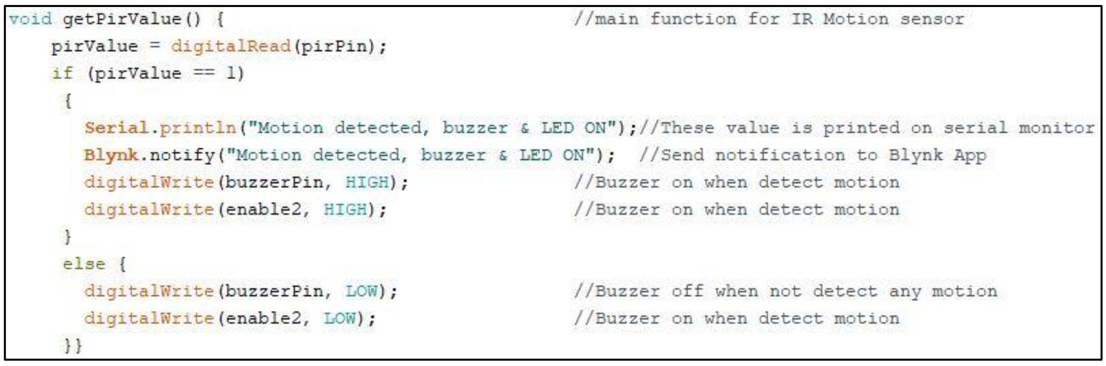
```
void TempHum()
{
  // Reads sensor value on Digital Pin 2
  // Stores the 2 values in their respective variable. Humidity - hSensor. Temperature - tSensor.
  h = dht.readHumidity();
  t = dht.readTemperature();


  if (isnan(h) || isnan(t)) { // In case of abnormal readings from the sensor (not a number)
    Serial.println("Failed to read from DHT sensor!"); // Display error message on serial monitor
  }
  else{
  // Display readings on serial monitor
  Serial.println (String ("Humidity is : ") + h);
  Serial.println (String ("Temperature is : ") + t);
  Blynk.virtualWrite(V5, h); /* Sends Humidity reading stored in variable hSensor
                                to Virtual Pin V5 defined in the Blynk App*/
  Blynk.virtualWrite(V6, t); /* Sends Temperature reading stored in variable tSensor
                                to Virtual Pin V6 defined in the Blynk App*/
  }
}
```

Figure 5.10 Code Fragment for DHT11 Configuration

5.3.1.4 Code Fragment Motion Sensor

The code fragment in Figure 5.11 is used to print the motion detected from the sensor and send alert to user's smartphone. In the getPirValue function, the statement pirValue = digitalRead(pirPin) is checking the status of motion sensor. If sensor detects the pirValue is equal to one which indicate that the motion is detected, then it triggers the buzzer and LED to turn on which is used to produce alarm and sound as alert. If pirValue is not equal to one, then the else statement will execute and buzzer and LED is switch off.

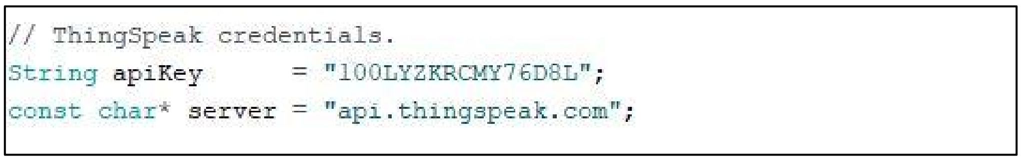


```
void getPirValue() {                                //main function for IR Motion sensor
    pirValue = digitalRead(pirPin);
    if (pirValue == 1)
     {
       Serial.println("Motion detected, buzzer & LED ON");//These value is printed on serial monitor
       Blynk.notify("Motion detected, buzzer & LED ON");  //Send notification to Blynk App
       digitalWrite(buzzerPin, HIGH);               //Buzzer on when detect motion
       digitalWrite(enable2, HIGH);                 //Buzzer on when detect motion
     }
     else {
       digitalWrite(buzzerPin, LOW);                //Buzzer off when not detect any motion
       digitalWrite(enable2, LOW);                  //Buzzer on when detect motion
     }}
```

Figure 5.11 Code fragment for Motion Detection

5.3.1.5 Code Fragment ThingSpeak Connection

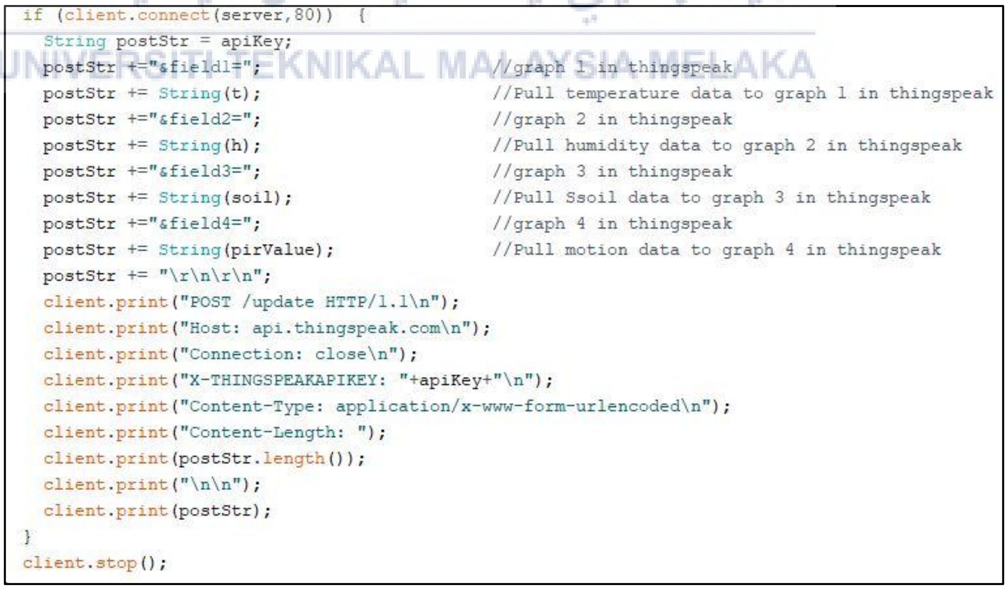
The code fragment in Figure 5.12 is used to connect to ThingSpeak cloud that has been configured in the Arduino IDE. This section is a key that is used to write API key data to ThingSpeak channel and ThingSpeak server.



```
// ThingSpeak credentials.
String apiKey      = "10OLYZKRCMY76D8L";
const char* server = "api.thingspeak.com";
```

Figure 5.12 ThingSpeak credentials

The code fragment in Figure 5.13 is used to for ThingSpeak server fields. In the ThingSpeak, there will have 4 fields which are field1 is for ‘t’ function which indicates for temperature reading, field2 is for ‘h’ function which indicates for humidity reading, field3 is for mSensor function which indicates for soil moisture reading and field4 is for pirValue function for motion detection. Then, the client print the sensors value on serial monitor.



```
 if (client.connect(server,80))  {
   String postStr = apiKey;
   postStr +="&field1=";                      //graph 1 in thingspeak
   postStr += String(t);                      //Pull temperature data to graph 1 in thingspeak
   postStr +="&field2=";                      //graph 2 in thingspeak
   postStr += String(h);                      //Pull humidity data to graph 2 in thingspeak
   postStr +="&field3=";                      //graph 3 in thingspeak
   postStr += String(soil);                   //Pull Ssoil data to graph 3 in thingspeak
   postStr +="&field4=";                      //graph 4 in thingspeak
   postStr += String(pirValue);               //Pull motion data to graph 4 in thingspeak
   postStr += "\r\n\r\n";
   client.print("POST /update HTTP/1.1\n");
   client.print("Host: api.thingspeak.com\n");
   client.print("Connection: close\n");
   client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n");
   client.print("Content-Type: application/x-www-form-urlencoded\n");
   client.print("Content-Length: ");
   client.print(postStr.length());
   client.print("\n\n");
   client.print(postStr);
 }
 client.stop();
```

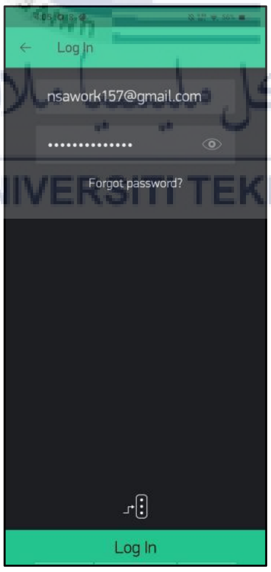
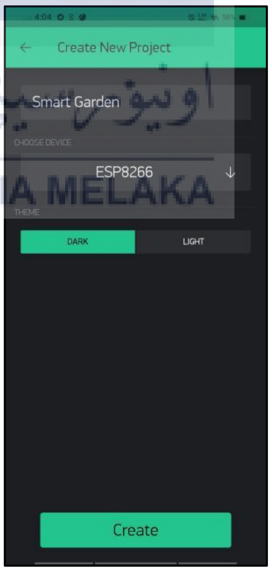
Figure 5.13 Code fragment for ThingSpeak cloud connection

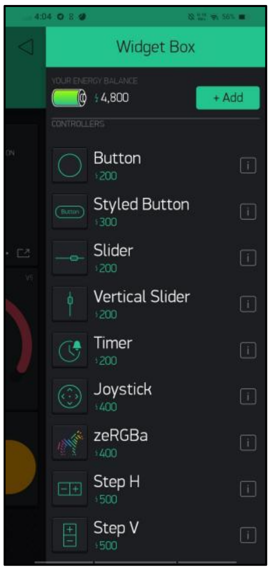
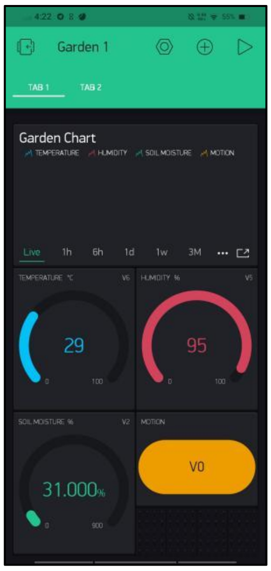
5.3.2 Blynk Application Configuration

The Blynk application has two components for this project. The first is a sensor reading, while the second is a button that controls a specific situation, such as a water pump in manual.

There is only one button in this Blynk application named 'Water.' When the button is pressed, the plant is watered manually, and the current condition of the watering system is shown. In simple terms, if the button is pressed, the system is in manual mode.

If the soil humidity reading falls below the system's specified threshold in AUTOMATE mode, the system automatically switches on the pump. The threshold is estimated to be 38% for this project. As a result, any reading below is considered low, and the Node MCU automatically switches on the water pump.

Step 1 : Create account or Log in to Blynk application.  Log In nsawork157@gmail.com ••••••••••••••• Forgot password? Log In **Figure 5.14 Create Blynk account**	**Step 2 :** Create new project for Smart Garden Monitoring Application.  Create New Project Smart Garden CHOOSE DEVICE ESP8266 THEME DARK LIGHT Create **Figure 5.15 Create new project**

Step 3 : Drag any widget for hardware sensors.	**Step 4 :** Current interface in Blynk application.
 **Figure 5.16 Choose widget for each sensor**	 **Figure 5.17 User interface in Blynk application**

5.3.3 ThingSpeak Cloud Configuration

The user of ThingSpeak has to create a channel and choose how many chart fields to use. Four fields are developed for this system. Blynk transmits data to ThingSpeak when sensors give data to it. After data is collected in a channel, the user can analyze and display it with ThingSpeak applications.

Step 1 : Create an account and sign in to ThingSpeak server. After sign in, click *New Channel* to create channel for this system.

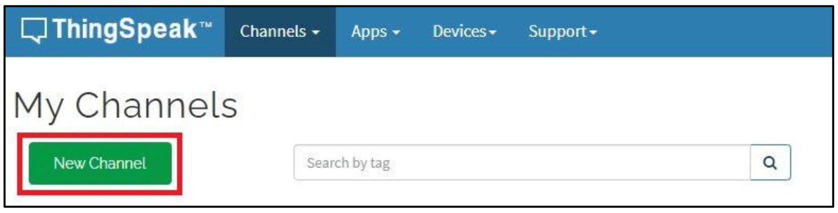
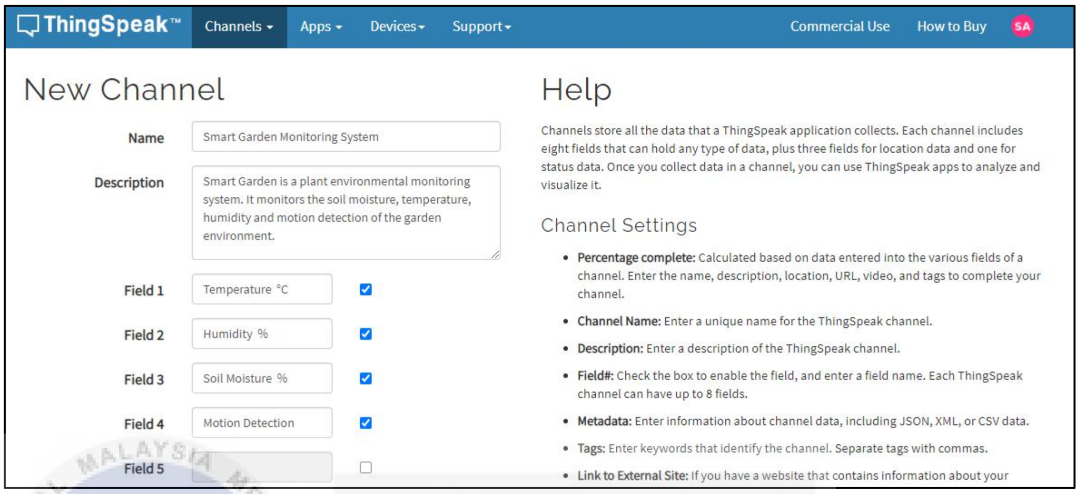


Figure 5.18 Create new channel in ThingSpeak

Step 2 : Then, fill in the channel details for this project. This project requires four fields which is field one stores the temperature data, field two stores the humidity data, field three stores the soil moisture data and field four stores the motion detection data. Then, save the channel.



ThingSpeak™ Channels ▾ Apps ▾ Devices ▾ Support ▾ Commercial Use How to Buy SA

New Channel

Name	Smart Garden Monitoring System	
Description	Smart Garden is a plant environmental monitoring system. It monitors the soil moisture, temperature, humidity and motion detection of the garden environment.	
Field 1	Temperature °C	✓
Field 2	Humidity %	✓
Field 3	Soil Moisture %	✓
Field 4	Motion Detection	✓
Field 5		☐

Help

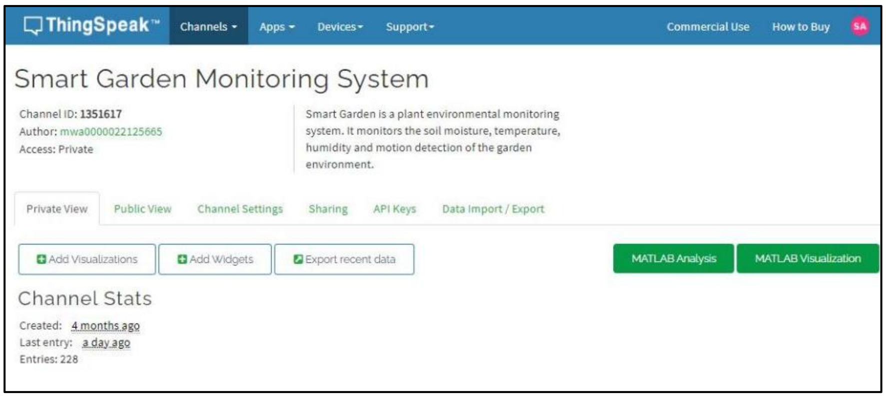
Channels store all the data that a ThingSpeak application collects. Each channel includes eight fields that can hold any type of data, plus three fields for location data and one for status data. Once you collect data in a channel, you can use ThingSpeak apps to analyze and visualize it.

Channel Settings

- **Percentage complete:** Calculated based on data entered into the various fields of a channel. Enter the name, description, location, URL, video, and tags to complete your channel.
- **Channel Name:** Enter a unique name for the ThingSpeak channel.
- **Description:** Enter a description of the ThingSpeak channel.
- **Field#:** Check the box to enable the field, and enter a field name. Each ThingSpeak channel can have up to 8 fields.
- **Metadata:** Enter information about channel data, including JSON, XML, or CSV data.
- **Tags:** Enter keywords that identify the channel. Separate tags with commas.
- **Link to External Site:** If you have a website that contains information about your

Figure 5.19 Fill in channel details for the project

Step 3 : After saving the channel, then it shows the channel has been created as shown in Figure 5.20. It also shows the graph for each sensors created as shown in Figure 5.21 for temperature and humidity, while Figure 5.22 shows graph for soil moisture and motion detection.



ThingSpeak™ Channels ▾ Apps ▾ Devices ▾ Support ▾ Commercial Use How to Buy SA

Smart Garden Monitoring System

Channel ID: 1351617
Author: mwa0000022125665
Access: Private

Smart Garden is a plant environmental monitoring system. It monitors the soil moisture, temperature, humidity and motion detection of the garden environment.

Private View | Public View | Channel Settings | Sharing | API Keys | Data Import / Export

Add Visualizations | Add Widgets | Export recent data | MATLAB Analysis | MATLAB Visualization

Channel Stats

Created: 4 months ago
Last entry: a day ago
Entries: 228

Figure 5.20 ThingSpeak Main Channel

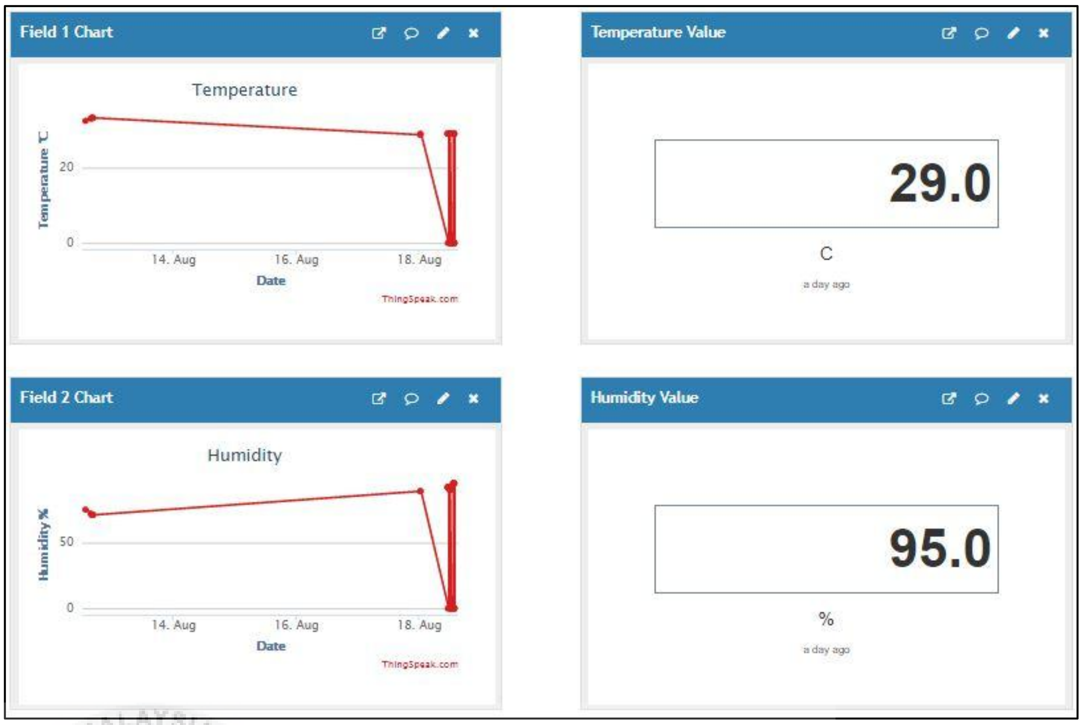


Figure 5.21 Graph Generated for DHT11 Sensor

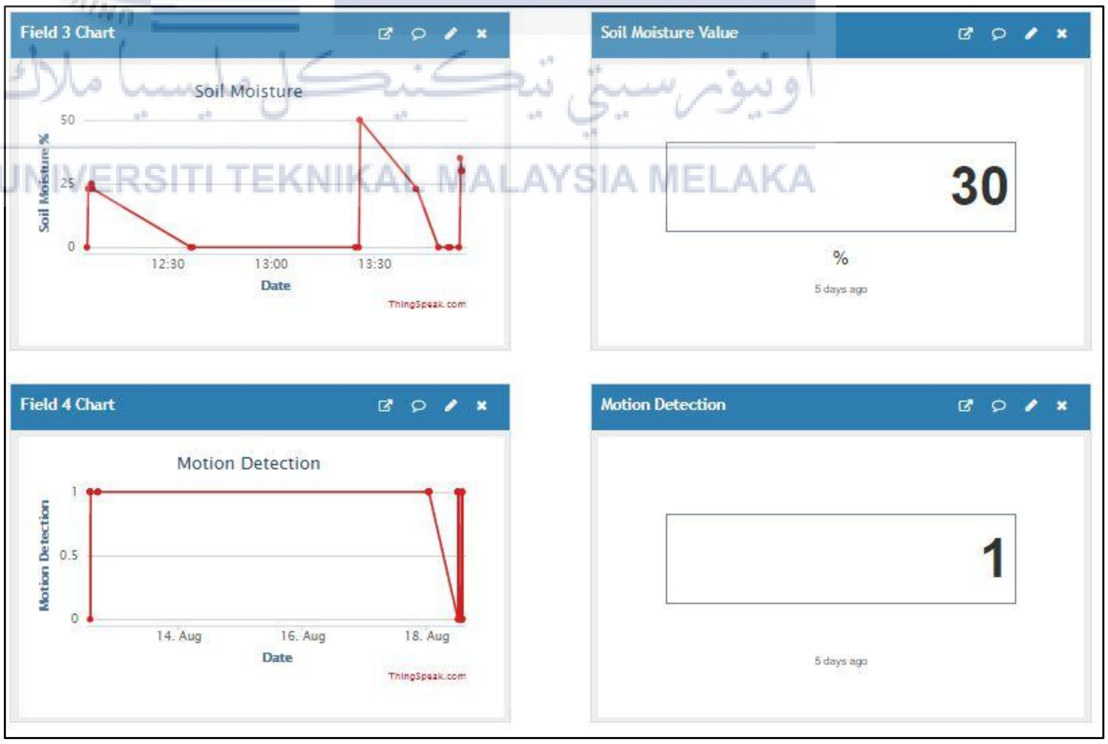


Figure 5.22 Graph Generated for Motion and Soil Sensor

5.3.4 Version Control Procedure

Smart Garden Monitoring Application is new implemented and documented. It is new to introduce to the market and the version for the final product system is 1.0.

5.4 Implementation Status

The implementation status section explains the duration of the development status for each component and configuration.

Table 5.2 Implementation Status

No.	Component	Description	Duration
1.	Assembling the hardware	Process of gathering all hardware required in the project.	7 days
2.	Building the prototype	Process of building the prototype of garden and connecting Node MCU and sensors together.	14 days
3.	Upload code to Node MCU	This process is to ensure the Node MCU microcontroller can control the hardware attached to it.	10 days
4.	Developing system	All of the function of the system is created by using Blynk.	10 days
5.	Database configuration	Configure the ThingSpeak cloud server for data storage and connect to the system	15 days

5.5 Conclusion

This chapter demonstrates the project's implementation process, which involves creating the development environment for the sensors and the project's control. On both sides of the implementation, the sensors and the microcontroller Node MCU are connected. It also demonstrates how a sensor is used to track the parameters of the garden environment. The system's software setup is implemented and configured using the Arduino IDE. Both hardware and software parts are vital to control the garden system using a smoothly functioning application. The next chapter is to focuses on the testing of the system. The prototype is used to test the functionality of the system.

CHAPTER 6: TESTING

6.1 Introduction

Chapter 6 discussed the testing phase of the project. The results of the testing phase are presented in this section. The testing step is crucial for ensuring that the prototype and system satisfy the specifications and operate effectively. Furthermore, this phase can enhance the project's function to fulfill the project's goal. All components and modules are tested to ensure that the result is precise and correct.

The Node MCU is the main component in this project that controls the system processes. Receiving alerts from the sensor to notify the user about the garden's status is one of the procedures involved. The testing is repeated until the desired result is obtained. Several testing planning topics are covered at this phase, including the test plan, test schedule, and test design.

6.2 Test Plan

The testing plan outlines the system's operations, scope, and fundamental testing. It is to guarantee that all objectives are achieved and that the system runs smoothly. Before delivering to the customer, any defects can be detected and fixed.

6.2.1 Test Organization

This subsection describes the individuals that involve during the testing process activities are conducted. For this project, there are 2 categories that are responsible for carrying out the testing process in the system. The first is the system developer who develops the system, and the second is the end-user. Both testers play different roles in the testing process.

1) **System Developer**

- The System Developer is in charge of testing the system as a whole, searching for errors, the system's stability, and smooth operation.

2) **End User**

- The person who tests the device's functionality and provides feedback on the product's usability, functionality, and effectiveness.

Table 6.1 Test Organization

Tester Name	Developer	Responsibilities
Nur Shahirah Binti Azmi (Student)	System Developer	Developed hardware prototype and mobile application that operates efficiently.
Nur Shahirah Binti Azmi (Student)	Architecture System	To satisfy the end-user about the proposed project is suitable or not.

Nur Shahirah Binti Azmi (Student)	Programmer System	The mobile applications and hardware are tested by the programmer to be used by the end-user.
TS. Muhamad Syahrul Azhar Bin Sani (Supervisor)	End User	The end-user gives feedback to the proposed project that can improve the project system.

6.2.2 Test Environment

This section outlines the environment in which the system is tested and how the project is carried out. The condition of the project to be carried out must be identified to understand the system limitations. The test environment includes the prototype of Smart Garden Monitoring Application hardware and the device's graphical user interface for this portion. The environment for testing is in an indoor garden with Wi-Fi access to facilitate communication inside the garden. For this project, the motion sensor is tested to detect any movement detection. The DHT11 sensor detects the current temperature and humidity of the garden environment. While for soil moisture sensor, it detects the moisture of the plant, whether dry or wet, to switch on the water pump. The microcontroller is placed inside the box and connected with a relay switch. The Wi-Fi module for Node MCU connection also needs to be tested to ensure the connectivity is enabled. The data used to monitor the parameter is stored in the ThingSpeak cloud for further review.

6.2.3 Test Schedule

This section explains how the system developer conducts testing during the testing phase. If weakness in the system's operation is discovered during testing, the system developer has to return to the implementation phase to double-check and attempt to resolve the issue. This is a continuous testing cycle until the system's functionality has been effectively developed.

6.3 Test Strategy

The testing strategy used in the project is described in this section. Black box testing is being performed to test the project's functioning for this project. Black box testing is a testing approach that examines the project's functioning without looking at the internal code structure or implementation details of the project's software. Then, for white box testing, white box testing assesses an application's internal structure or operation mode rather than its role as an internal perspective or programming skill.

Table 6.2 Test Strategy

Test Type	Description	Person in Charge
Black box testing	Evaluate the group's actions.	Developer, supervisor and user.
White box testing	Examine the project's code framework.	Developer

6.4 Test Design

This section describes the project's test description and the expected outcomes for the scenario selected, which are created and documented in this project.

6.4.1 Test Description

The test descriptions that were tested in this project are listed in this section. This part covered all the components and modules used to achieve an accurate and effective result. The table below lists all of the test cases.

Table 6.3 Node MCU Wi-Fi Connection Test

Test case ID	TC001
Test functionality	Wireless Communication Test
Test purpose	To be able to connect to home network
Test environment	To test this project, the home's network WPS SSID and password must be identified and it must support wireless connection to devices.
Execution steps	i. Run Arduino IDE. ii. Write code to upload into Node MCU ESP 8266 iii. Click upload
Expected results	DHCP IPv4 address is assigned to the device after it connected to home network.
Error Message	None
Result	Pass

Table 6.4 DHT11 Sensor Test

Test case ID	TC002
Test functionality	Temperature and Humidity Sensor
Test purpose	To detect temperature and humidity of garden environment.
Test environment	Sensor is place inside the garden area and Arduino IDE.
Execution steps	i. Implement code DHT11 in Arduino IDE. ii. Upload code and see the detected value in real time iii. Verify value in Blynk apps same to the ThingSpeak cloud.
Expected results	Detect temperature and humidity in real time and display the value in Blynk application and ThingSpeak.
Error Message	None
Result	Pass

Table 6.5 Motion Sensor Test

Test case ID	TC003
Test functionality	Movement Detection Sensor
Test purpose	To be able to detect motion around plant.
Test environment	A person needs to create a motion/block within the sensor's area.
Execution steps	i. Implement and upload code of motion detection in Arduino IDE see the value if no motion detected. ii. Block the sensor's front area and see for the motion detected in serial monitor. iii. Verify value in Blynk apps same to the ThingSpeak cloud.
Expected results	Motion detected, then, buzzer and LED turn on.
Error Message	None
Result	Pass

Table 6.6 Soil Moisture Sensor Test

Test case ID	TC004
Test functionality	Soil Moisture Sensor
Test purpose	To test the integration between Node MCU ESP 8266 and sensor to detect soil moisture.
Test environment	Bottle that filled with water and Arduino IDE.
Execution steps	i. Implement code of soil moisture in Arduino IDE. ii. Upload code and see the value if sensor is dry condition. iii. Put the soil sensor in the bottle that filled with water and see if there is another changes to the value for wet condition. iv. Verify value in Blynk apps same to the ThingSpeak cloud.
Expected results	Sensor detect soil moisture according to certain condition and turn on or off relay switch for watering process.
Error Message	None
Result	Pass

Table 6.7 Water Pump Test

Test case ID	TC005
Test functionality	Water Pump Test
Test purpose	To test the integration between relay switch and soil moisture sensor to water the plant.
Test environment	Relay switch, battery and Arduino IDE.
Execution steps	i. Implement code of water pump in Arduino IDE. ii. Upload code and see if the soil sensor is dry, water pump is turning ON. iii. Then, put the soil sensor in water and see if water pump is turn OFF.
Expected results	Relay switch is turn ON the water pump when the soil moisture sensor detect that it is dry condition. Then it turn OFF when the sensor detect the condition is wet enough. User can also turn ON/OFF using manual button.
Error Message	None
Result	Pass

6.4.2 Monitoring System Test

Table 6.8 ThingSpeak Monitoring Test

Test case ID	TC006
Test functionality	ThingSpeak Monitoring System
Test purpose	To test the functionality of monitoring system
Test environment	Soil Moisture Sensor, DHT11 sensor, Motion Sensor, Arduino IDE, ThingSpeak Cloud
Execution steps	i. Implement code of ThingSpeak in Arduino IDE. ii. Upload code see the value content. iii. Then, see if the data is uploaded to cloud.
Expected results	All of sensors data is uploaded to ThingSpeak.
Error Message	None
Result	Pass

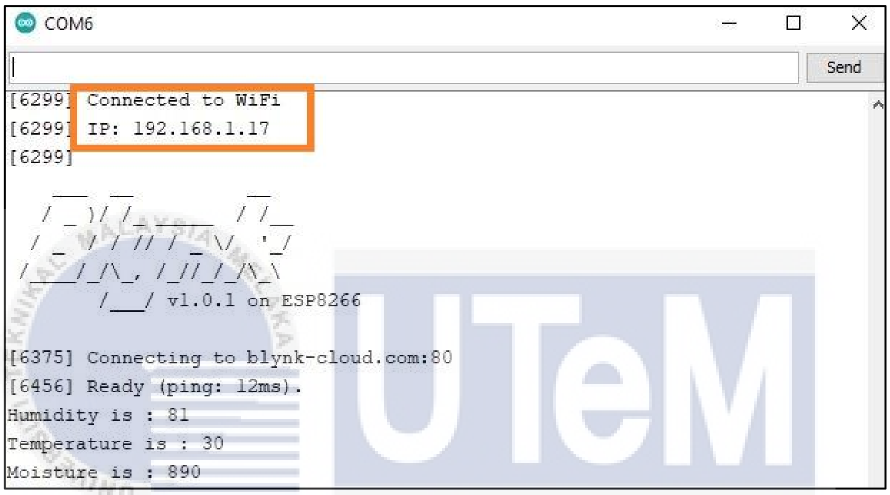
Table 6.9 Blynk Notification System

Test case ID	TC007
Test functionality	Blynk Notification System
Test purpose	To test the functionality of notification and monitoring system
Test environment	Soil Moisture Sensor, DHT11 sensor, Motion Sensor, Arduino IDE, Blynk Application
Execution steps	i. Implement and upload code of Blynk in Arduino IDE. ii. Then, see if the data is displayed in Blynk apps. iii. Make condition or scenario to get notification alert.
Expected results	All of sensors data is uploaded to ThingSpeak.
Error Message	None
Result	Pass

6.5 Test Result and Analysis

6.5.1 Node MCU ESP 8266 Testing

Wi-Fi detection is done by referring to Table 6.3 above. The result of the test is shown below. To test the Node MCU ESP 8266 functionality in the serial monitor, it shows the output of programming code that has been successfully uploaded into the Node MCU board. Figure 6.1 shows the IP address of the Node MCU after it has been connected to the internet through Wi-Fi.



```
[6299] Connected to WiFi
[6299] IP: 192.168.1.17
[6299]
    ___  __          __
   / _ )/ /_ _____  / /__
  / _  / / // / _ \/  '_/
 /____/_/\_, /_//_/_/\_\
        /___/ v1.0.1 on ESP8266

[6375] Connecting to blynk-cloud.com:80
[6456] Ready (ping: 12ms).
Humidity is : 81
Temperature is : 30
Moisture is : 890
```

Figure 6.1 Node MCU Connection Test

6.5.2 DHT11 Sensor Testing

Temperature and humidity detection is done by referring to Table 6.4 below. The result of the test is shown below. In Figure 6.2, the Blynk apps display the temperature and humidity of the surrounding air detected using the DHT11 sensor. It is used to monitor measurement since each plant requires a different type of air to develop. The data of the state is stored in the ThingSpeak cloud for future analysis. The data received and detected by the sensor is stored at ThingSpeak cloud.

Table 6.10 System Condition Test

Condition	Scenario	Result
1	If temperature and humidity is detected, this system update current condition to user smartphone and cloud storage.	Success

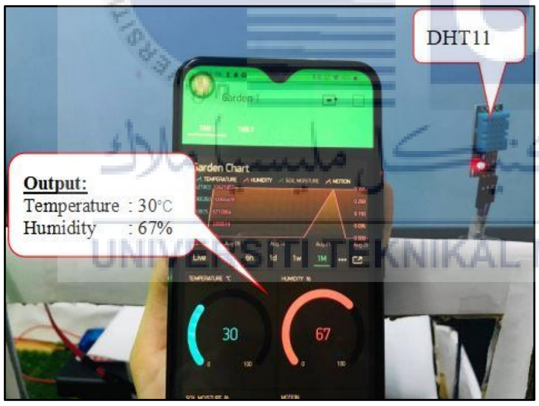


Figure 6.2 Output from Blynk Apps

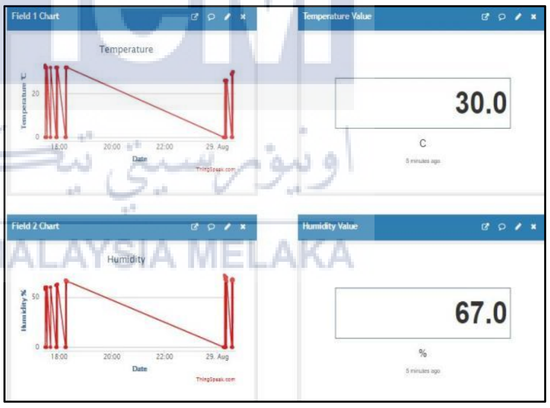


Figure 6.3 Output from ThingSpeak

According to the observations, temperature and humidity are some of the main factors of a plant’s production for better growth. The project aims to monitor the temperature and humidity environment of the garden because every species of plant have different condition temperature and humidity. From Figure 6.2, data display an output from sensor in Blynk apps which user can monitor remotely in real-time. Figure 6.3 shows the data and stored it in the cloud. Thus, users can analyze future references.

6.5.3 Motion Detection Sensor Testing

Motion detection is done by referring to Table 6.5 below. The result of the test is shown below. The sensor used in this project is to detect animal movement that passing through the detector. The sensor activates the buzzer and LED and gives alert notifications of the current state. Serial monitor print motion detection detected by the sensor. The data received is stored in the ThingSpeak cloud for future analysis.

Table 6.11 Motion System Condition Test

Condition	Scenario	Result
1	If no motion is detected by sensor, this system not do anything and continue to monitor.	Success
2	If motion is detected, buzzer trigger sound, LED turn ON, and the system give alert notification to user smartphone and update to cloud storage.	Success

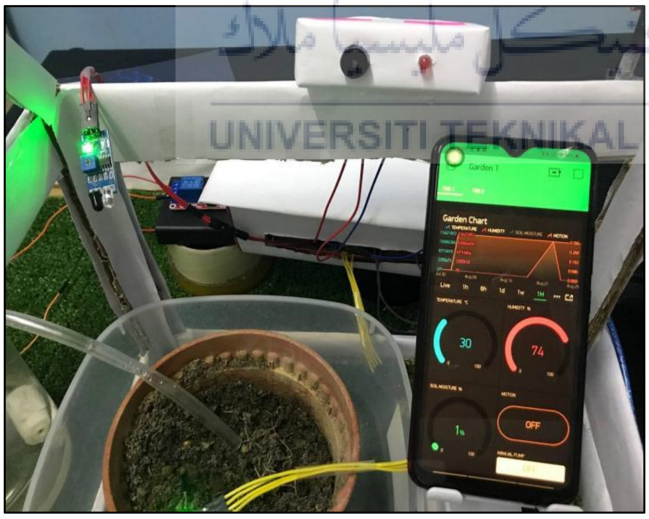


Figure 6.4 Output Condition 1

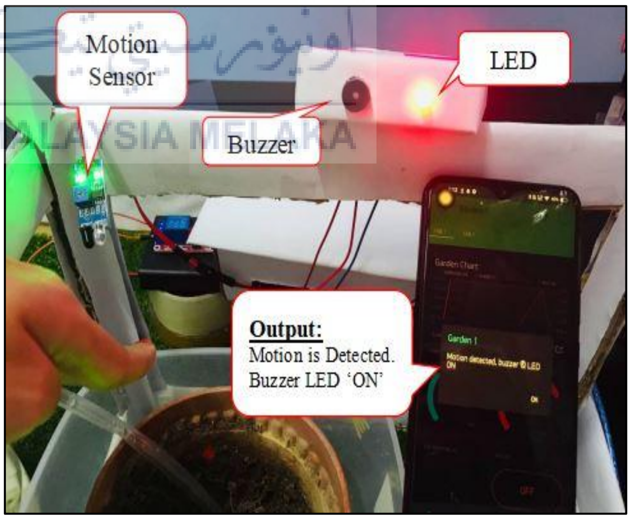


Figure 6.5 Output Condition 2

The motion detection sensor is used in this project to detect movement within the sensor's range. This project use motion sensor to monitor the garden environment from animal that can threaten plant's area.

Figure 6.4 shows when no motion is detected, and then the system keeps on monitoring the area. Figure 6.5 shows an output result notification when the sensor detects motion in front of the sensor. The test has been done by blocking the sensor's area to see the functionality of the sensor. Then, after motion has been detected by the sensor, it triggers the buzzer as an alarm system to make a loud noise as well as LED turn ON as a warning that motion is detected. Then, the system updates the detected data into ThingSpeak cloud storage, as shown in Figure 6.6 below.

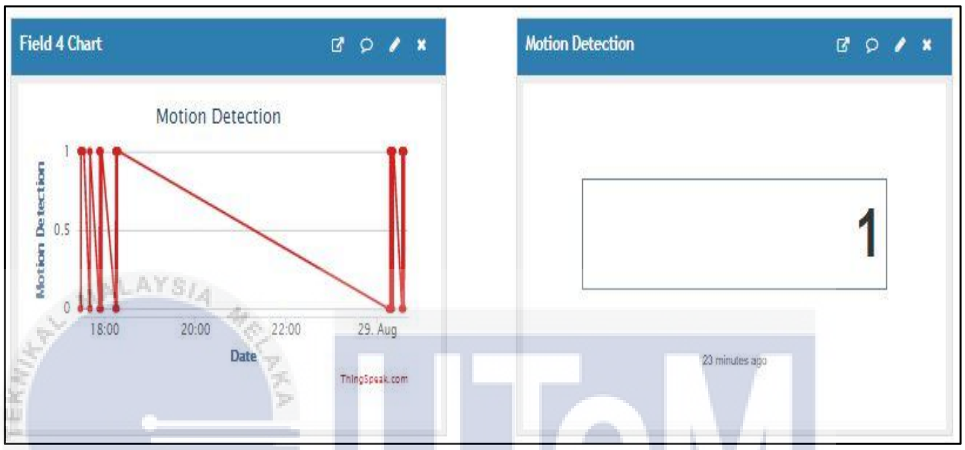


Figure 6.6 Output of motion detected display on ThingSpeak

From Figure 6.6, the result of the graph shows that the value of plotted data is 1, which means the sensor has detected the movement that has passed through the garden area. Then, the data is stored there to enable the user to observe the garden's condition in the future.

The aim of using ThingSpeak cloud storage for motion sensors is because, by collecting the data, the user can analyze when the system detects motion happened and how many times the sensor detects movement. Thus, this data is vital for the user to monitor their garden effectively.

6.5.4 Soil Moisture Sensor Testing

Soil moisture detection is done by referring to Table 6.6. The result of the test is shown below. For soil moisture sensor, it detects the moisture content of the soil. If the soil is dry, the Blynk application sends a notification that it is dry and automatically waters the plant. If the soil is wet enough, the water pump is turned off.

Table 6.12 Functionality System Test

No.	Soil Moisture	Relay	Water Pump	Result
1.	If soil <=38% (Dry)	ON	Open	Success
2.	If soil >=45% (Wet)	OFF	Closed	Success

i. Output based on dry soil condition

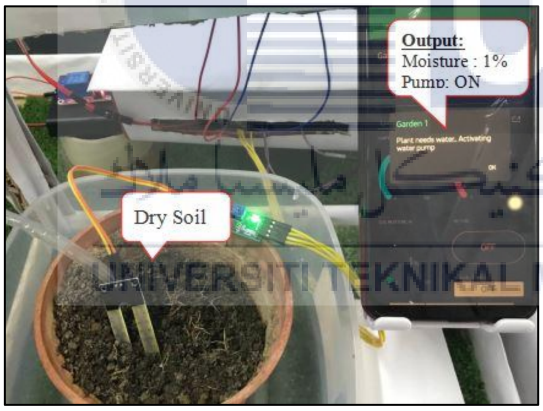


Figure 6.7 Output dry condition in Blynk apps

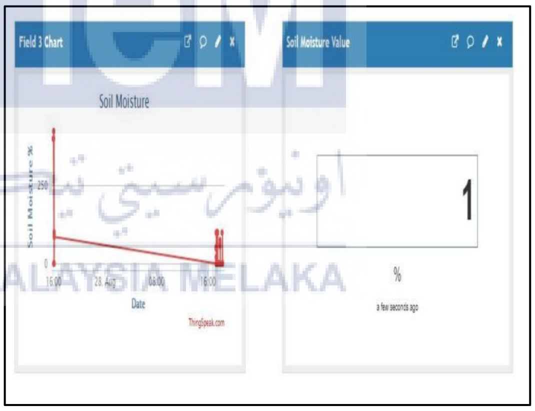


Figure 6.8 Output dry condition in ThingSpeak cloud

According to the observations, the soil moisture sensor is initially placed in dry soil for testing. From Figure 6.7, the soil moisture reading falls below 1%, which is below the threshold that has been set for moisture. Thus, the water pump is turned on, and Blynk sends a notification to the user's smartphone.

From Figure 6.8, the reading of data moisture is shown in ThingSpeak cloud for data storage. Thus, it tells the user that the reading data for soil moisture is below the threshold value set. From the graph input, the user can analyze and examine when the system detects soil is dry. Then from the data, the user can take the initiative to prepare more water for the plant. Thus, this data is important for the user to monitor their garden effectively.

ii. Output based on wet soil condition

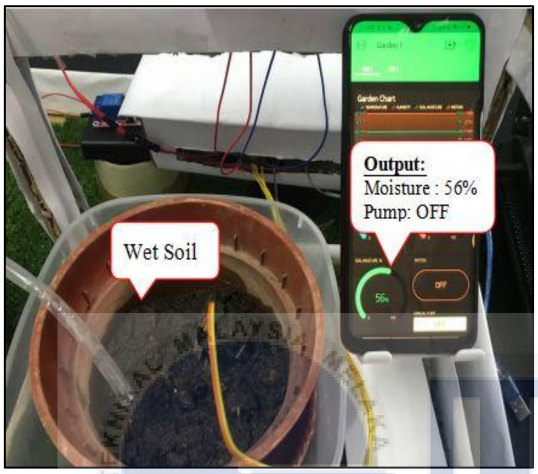


Figure 6.9 Output wet condition in Blynk apps

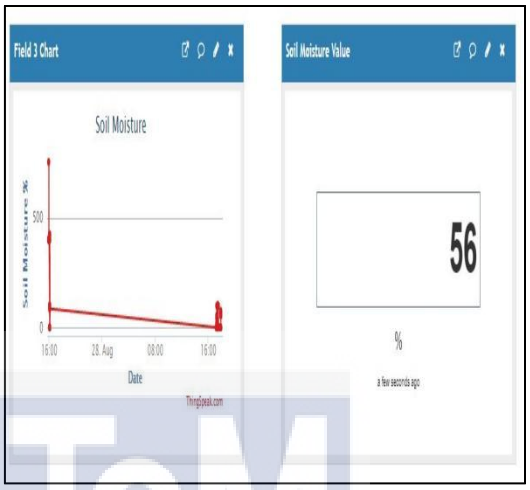


Figure 6.10 Output wet condition in ThingSpeak cloud

For the system to be regulated automatically, a threshold value must be specified during this testing. The optimum moisture level for this prototyping has been set to be 45%. The threshold has been set is because it prevents the system from being over-watering which can threaten the plant's growth and production.

According to Figure 6.9, it has shown that the reading of moisture detected by the soil sensor is 56%, which is higher than the threshold that has been set for moisture value. Thus, the water pump is turned off.

Figure 6.10 shows the output data for the wet condition in ThingSpeak cloud storage, showing the same value as in the Blynk application. The user can get input on how long the soil moisture can be beheld from the graph display.

6.5.5 Manual Button Water Pump Testing

Manual water pump testing is done for testing button functionality. The result of the test is shown below. Besides an automatic water pump, this system also consists of a manual button for the watering process. When the user wants to pump more water, the user can click a button named ‘Manual Pump’ in the Blynk application.

Table 6.13 Water Pump Manual Button Test

No.	Soil Moisture	Relay	Water Pump	Result
1.	If user click button manual for water pump, it turn ON and user received notification.	ON	Open	Success
2.	If user do not click button manual water pump, the system not do anything.	OFF	Closed	Success

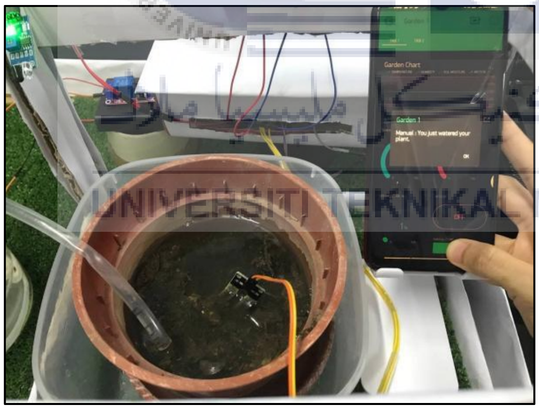
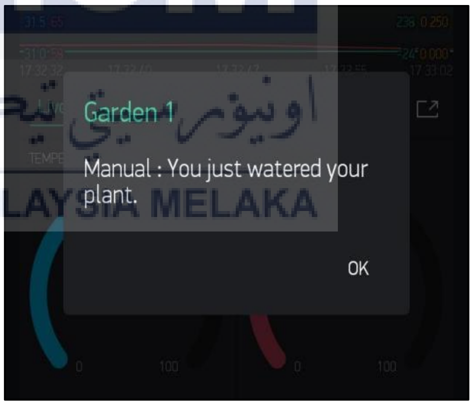


Figure 6.11 Output for manual button pump


Garden 1

Manual : You just watered your plant.

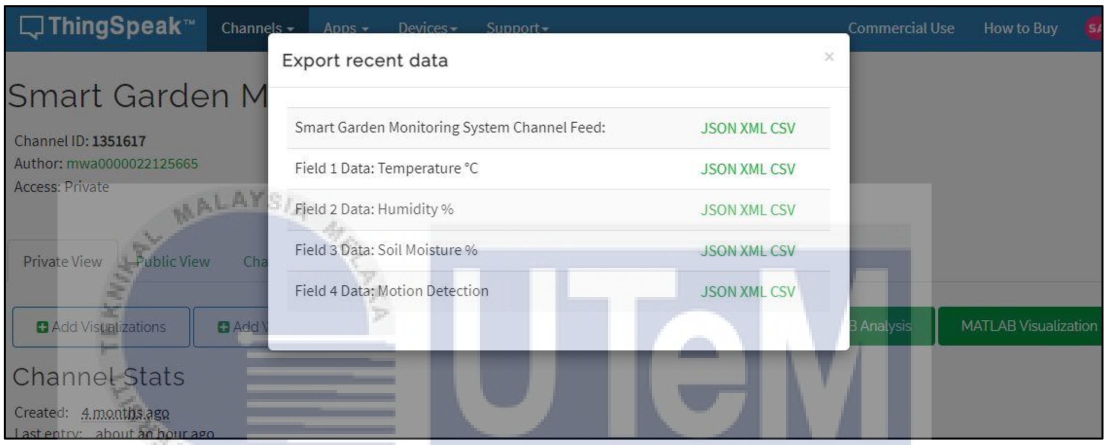
OK

Figure 6.12 Notification display after click Manual button

This project includes a manual button in case the user wants to water their plant more frequently. As shown in Figure 6.11, the system has been tested by clicking the manual button provided for this system. Then, the water pump is turned on, and water flows when the button is pressed. Figure 6.12 shows that, after the button is pressed, the user receives a notification display on their Blynk application showing that their plant has been watered manually.

6.5.6 ThingSpeak Cloud Storage Testing

ThingSpeak is one of the cloud storage for IoT platforms that have been used in this project that can store data. The use of ThingSpeak in this project is to store data that all sensors have detected for future use. All of the data are recorded in the ThingSpeak channel, followed by displaying the data according to the field created for each sensor. As shown in Figure 6.13 below, a button lets users export recent data from each sensor field. Then, users can download the data in .CSV format and do analysis using the sensor readings.



Export recent data

Smart Garden Monitoring System Channel Feed:	JSON XML CSV
Field 1 Data: Temperature °C	JSON XML CSV
Field 2 Data: Humidity %	JSON XML CSV
Field 3 Data: Soil Moisture %	JSON XML CSV
Field 4 Data: Motion Detection	JSON XML CSV

Channel ID: 1351617
Author: mwa0000022125665
Access: Private

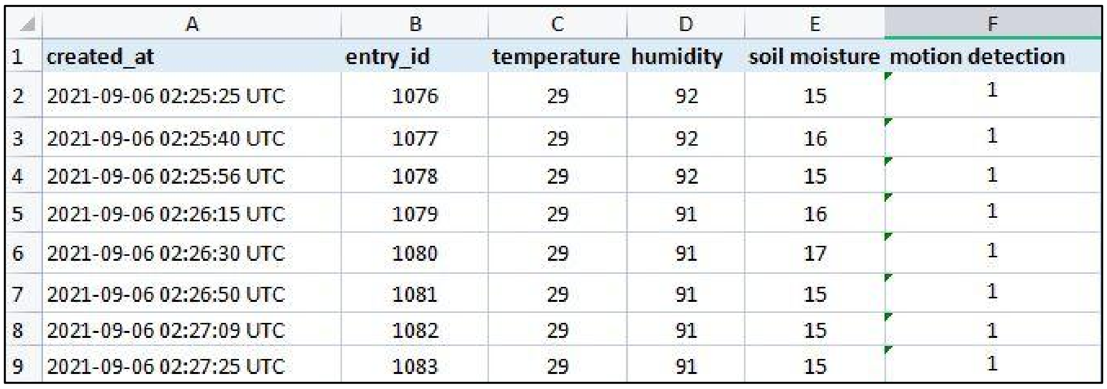
Channel Stats
Created: 4 months ago
Last entry: about an hour ago

Figure 6.13 ThingSpeak log data

Figure 6.14 shows the .csv files that have been downloaded from the ThingSpeak channel. The export data is from the field that has been created for each sensor. Users can get an option to export data from the whole channel feed or export data from each sensor field.

Figure 6.14 Downloaded csv file from ThingSpeak channel

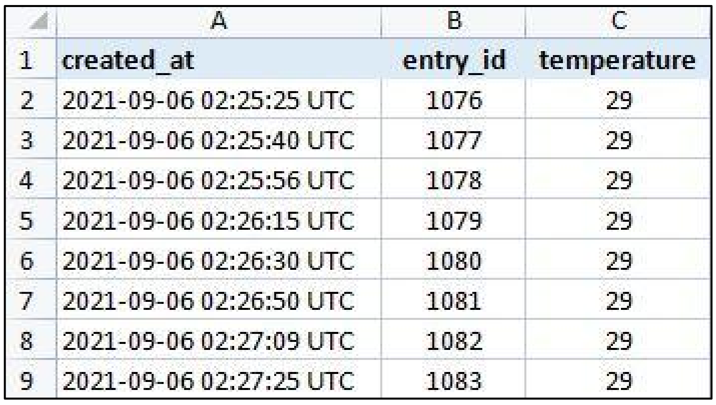
Figure 6.15 shows the file *feed.csv* that includes the reading from each sensor's data from ThingSpeak channel. The reading data consists of temperature sensor, humidity sensor, soil moisture sensor and motion detection sensor. From the exported file, user can get to know the condition and history of detected data.



	A	B	C	D	E	F
1	created_at	entry_id	temperature	humidity	soil moisture	motion detection
2	2021-09-06 02:25:25 UTC	1076	29	92	15	1
3	2021-09-06 02:25:40 UTC	1077	29	92	16	1
4	2021-09-06 02:25:56 UTC	1078	29	92	15	1
5	2021-09-06 02:26:15 UTC	1079	29	91	16	1
6	2021-09-06 02:26:30 UTC	1080	29	91	17	1
7	2021-09-06 02:26:50 UTC	1081	29	91	15	1
8	2021-09-06 02:27:09 UTC	1082	29	91	15	1
9	2021-09-06 02:27:25 UTC	1083	29	91	15	1

Figure 6.15 ThingSpeak export file feed.csv

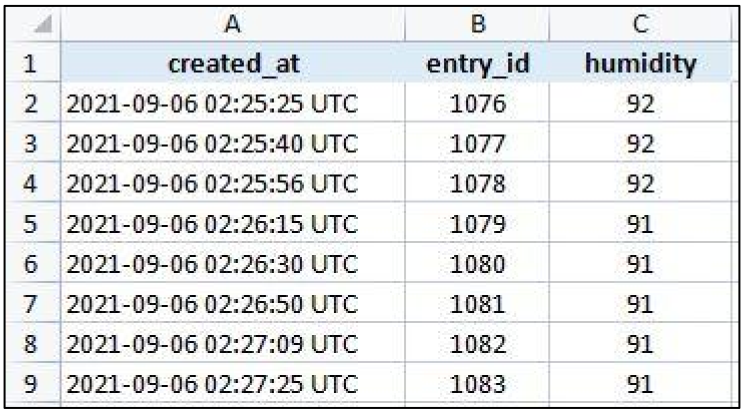
Figure 6.16 shows the file *temperature.csv* that includes the reading from DHT11 sensor's data from ThingSpeak channel. The data is updated to ThingSpeak channel to monitor the temperature reading. From the exported file, user can get to know the condition and history of detected data.



	A	B	C
1	created_at	entry_id	temperature
2	2021-09-06 02:25:25 UTC	1076	29
3	2021-09-06 02:25:40 UTC	1077	29
4	2021-09-06 02:25:56 UTC	1078	29
5	2021-09-06 02:26:15 UTC	1079	29
6	2021-09-06 02:26:30 UTC	1080	29
7	2021-09-06 02:26:50 UTC	1081	29
8	2021-09-06 02:27:09 UTC	1082	29
9	2021-09-06 02:27:25 UTC	1083	29

Figure 6.16 ThingSpeak export file temperature.csv

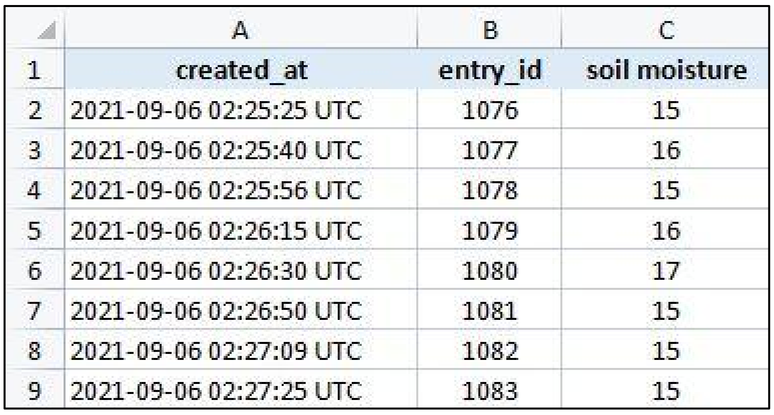
Figure 6.17 shows the file *humidity.csv* that includes the reading from DHT11 sensor's data from ThingSpeak channel. The data is updated to ThingSpeak channel to monitor the humidity reading. From the exported file, user can get to know the condition and history of detected data.



	A	B	C
1	created_at	entry_id	humidity
2	2021-09-06 02:25:25 UTC	1076	92
3	2021-09-06 02:25:40 UTC	1077	92
4	2021-09-06 02:25:56 UTC	1078	92
5	2021-09-06 02:26:15 UTC	1079	91
6	2021-09-06 02:26:30 UTC	1080	91
7	2021-09-06 02:26:50 UTC	1081	91
8	2021-09-06 02:27:09 UTC	1082	91
9	2021-09-06 02:27:25 UTC	1083	91

Figure 6.17 ThingSpeak export file humidity.csv

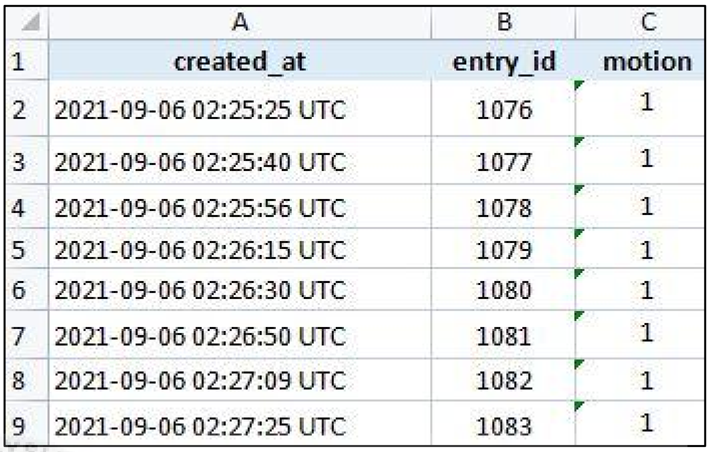
Figure 6.18 shows the file *soil moisture.csv* that includes the reading from soil moisture sensor's data from ThingSpeak channel. The data is updated to ThingSpeak channel to monitor the soil moisture reading. From the exported file, user can get to know the condition and history of detected data.



	A	B	C
1	created_at	entry_id	soil moisture
2	2021-09-06 02:25:25 UTC	1076	15
3	2021-09-06 02:25:40 UTC	1077	16
4	2021-09-06 02:25:56 UTC	1078	15
5	2021-09-06 02:26:15 UTC	1079	16
6	2021-09-06 02:26:30 UTC	1080	17
7	2021-09-06 02:26:50 UTC	1081	15
8	2021-09-06 02:27:09 UTC	1082	15
9	2021-09-06 02:27:25 UTC	1083	15

Figure 6.18 ThingSpeak export file soil moisture.csv

Figure 6.19 shows the file *motion detection.csv* that includes the reading from motion sensor's data from ThingSpeak channel. The data is updated to ThingSpeak channel to monitor the motion detected reading. From the exported file, user can get to know the condition and history of detected data.



	A	B	C
1	created_at	entry_id	motion
2	2021-09-06 02:25:25 UTC	1076	1
3	2021-09-06 02:25:40 UTC	1077	1
4	2021-09-06 02:25:56 UTC	1078	1
5	2021-09-06 02:26:15 UTC	1079	1
6	2021-09-06 02:26:30 UTC	1080	1
7	2021-09-06 02:26:50 UTC	1081	1
8	2021-09-06 02:27:09 UTC	1082	1
9	2021-09-06 02:27:25 UTC	1083	1

Figure 6.19 ThingSpeak export file motion detection.csv

6.5.7 Notification Blynk Application Testing

The notification functionality has also been implemented in this project. If the motion is detected in the garden area, or the soil moisture level falls below the average level, then a notification appears on the user's phone, alerting them to the plant's critical condition.

6.6 Conclusion

In conclusion, the testing phase is conducted to examine the functionality of the smart garden prototype which include Node MCU ESP 8266, sensors, Blynk application and ThingSpeak cloud. Testing process helps the developer in improving the system's performance by undergone multiple tests and analyzed the result based on different type of sensors. It is also enhancing user experiences as a result of end-user input. Next chapter discuss about the conclusion of the project system.

CHAPTER 7: PROJECT CONCLUSION

7.1 Introduction

This chapter summarizes the project development and conclusion. It discusses the overall project development and achievement of this project based on the objective and analysis result of the project that has been implemented in the previous chapter. This chapter discusses the project's contribution and limitations. This chapter has suggested how the system can be improved better in the future.

7.2 Project Summary

Smart Garden Monitoring Application is a project for monitoring the garden environment and sending notifications to smartphones by using the Blynk application to perform all system functions. This project is developed for users who have an indoor garden but neglect to nourish their plants and do not have enough time to care for them because they are preoccupied with their jobs and outdoor lifestyle. Thus, this project objective is developed to overcome the problem that arises.

The first objective is to study on smart garden method with IoT technology. Through the research studies, as stated in Chapter 2, the project has discovered several methods to monitor the garden with various sensors without human intervention. The research also includes watering the plant, components to use, and ways to monitor the garden. .

The second objective is to develop a hardware prototype that can monitor the garden’s environment. In this project, the prototype has been developed using Node MCU ESP8266 and sensors to monitor the garden’s environment. The monitoring sensor of the plant includes soil moisture, humidity and temperature, and motion sensor. The developed prototype does help users to monitor the garden environment remotely. Thus, this project has achieved its target to develop a hardware prototype that can monitor the garden environment without user intervention.

The third objective is to integrate hardware prototypes with a mobile application to provide an alert. The prototype has been integrated with a mobile application that displays the data of monitoring. Thus, users can monitor the plant remotely. Real-time notification is sent to the user's smartphone to inform the user about the current condition and take immediate action for future references. ThingSpeak cloud storage has also been included in this project to store the detected data sensor in cloud storage. From that, the user can analyze the behavior of the detected data as a future reference. Thus, the integration between hardware prototypes with mobile apps to provide alerts has been achieved in this project.

Overall, Smart Garden Monitoring Application has been fully designed and tested, and it fulfills all of the project's objectives. The hardware and software components are effectively integrated, resulting in a comprehensive smart garden that can overcome the drawbacks of previous traditional gardening on the market. This Smart Garden Monitoring Application is not as advanced as other smart irrigation applications, but it has the potential to improve over time.

7.2.1 Project Strength and Weakness

This section emphasis on the strength and weaknesses of the developed project. This project typically has a strength and weakness during and after implemented. The strength and weaknesses are stated as below:-

1) Project Strength

a. Provide a real-time monitoring

This system provides real-time monitoring on both platforms, which is Blynk mobile application and ThingSpeak cloud storage. Thus, users can monitor and get updated information about the environment from anywhere in real-time.

b. Lower operational costs

The developed project system was found to be feasible and cost-effective. The system is simple to set up and maintain because it only requires a few components. Aside from that, the cost of developing the system is low, making it accessible to those who love gardening.

2) Project Weakness

a. Need 24/7 internet connection

This system uses a mobile app to notify users and cloud storage to store data. As a result, this system must be connected to the network 24/7 to maintain the server and monitor the garden environment via sensor readings. If the connection is lost, the server and the reading data from the sensor may not function properly.

b. USB connection lost between hardware

This system only supports the USB connection between the hardware. The command is sent through a USB connection from the software program to the hardware. If the connection is lost, the hardware may not get the control.

7.3 Project Contribution

This project was developed to help users know about IoT technology that can monitor their garden by replacing traditional methods. The implementation of Smart Garden Monitoring Application was practical and cost-effective with the functionality installed for plant growth, such as optimizing water resources for plant growth, help users to keep track of sensors that detects animals try to enter into the plant area.

Besides, this project also contributes to help people to monitor their plants in the garden remotely via a monitoring function where users can track the garden’s condition based on the reading displayed on the Blynk mobile application and ThingSpeak cloud storage.

This project also contributes to providing alerts to the user by integrating hardware devices with a mobile application. Plant damage by wild animals, temperature and humidity, and watering has become serious societal issues in recent years. It demands immediate attention because there is currently no viable answer to this problem. Thus, users are notified about the garden’s condition which they can take action in the future from the alert notification that has been developed in this project.

7.4 Project Limitation

There is some limitations that has been identified in this project. The limitations are stated as below.

7.4.1 Security

The system uses a motion detection sensor to sense movement in the garden area. However, this sensor's limitation is that the sensor cannot detect whether it is human or what kind of animal has entered the garden area since the sensor only detects motion.

7.4.2 Water Supply

In this project, the water supply is stored in the water container without any sensor. As a result, the user must check the water supply manually to ensure that it is sufficient.

7.5 Future Works

This project has its potential and future works for improvement to make it more functional and comprehensive. Therefore, the future works that can be added are the Smart Garden Monitoring Application can be enhanced in the future is by adding camera CCTV. For now, this project only uses a motion sensor to detect movement. For future work, it can be improved by adding a camera that can record video and images. Thus, users can monitor via CCTV of their garden more accurately in the future. Besides, the system may require an alert to warn the user that their water storage is running low, which needs another sensor to measure the water level in the water container. Aside from that, future work can include developing a mobile application and making it more appealing for users.

7.6 Conclusion

In conclusion, the Smart Garden Monitoring Application is successfully completed, and its functionality fulfills all the objectives stated for this project. The system has been successfully developed and demonstrated using an actual prototype with sensors that can detect data from sensors and seamlessly monitor reading data. By connecting different parameters of the sensors to the cloud and effectively controlling them remotely using a mobile application, the development of a Smart Garden Monitoring Application has been proven to work satisfactorily.

REFERENCES

Albishi, S., Soh, B., Ullah, A., & Algarni, F. (2017). Challenges and Solutions for Applications and Technologies in the Internet of Things. *Procedia Computer Science*, *124*, 608–614. https://doi.org/10.1016/j.procs.2017.12.196

Bargavi. (2018). *Rapid Application Development - Application development*. https://blog.aleph-technologies.com/rapid-application-development/

Bharti, L., Chandrapuri, M., Nagle, V., Dhurve, P., & Patil, J. (2020). Garden Monitoring and Automation Using Atmega 16. *Asian Journal of Applied Science and Technology*, *04*(01), 127–130. https://doi.org/10.38177/ajast.2020.4113

Ding, J., Li, T. R., & Chen, X. L. (2018). The Application of Wifi Technology in Smart Home. *Journal of Physics: Conference Series*, *1061*(1). https://doi.org/10.1088/1742-6596/1061/1/012010

Dorsemaine, B., Gaulier, J. P., Wary, J. P., Kheir, N., & Urien, P. (2016). Internet of Things: A Definition and Taxonomy. *Proceedings - NGMAST 2015: The 9th International Conference on Next Generation Mobile Applications, Services and Technologies*, 72–77. https://doi.org/10.1109/NGMAST.2015.71

Elangovan, R., Santhanakrishnan, D. N., Rozario, R., & Banu, A. (2018). Tomen:A Plant monitoring and smart gardening system using IoT. *International Journal of Pure and Applied Mathematics*, *Volume 119*(March), 703–710. https://www.researchgate.net/publication/323811801_TomenA_Plant_monitoring_and_smart_gardening_system_using_IoT

Gaur, J., & Mittal, P. (2013). *Enhancing The Security Of Wsn In Agriculture*. *4*(4), 22–25.

Gautam, A. (2019). *Rain drop Sensor Module Pinout, Datasheet & How to Use it in a Circuit*. https://components101.com/sensors/rain-drop-sensor-module

Gautam, A. (2020). *5V Single-Channel Relay Module - Pin Diagram, Specifications, Applications, Working*. https://components101.com/switches/5v-single-channel-relay-module-pinout-features-applications-working-datasheet

Ismalyza, M. A., Rohaida, M. Y., & Khatijah, M. S. (2019). PolyCCRISe 2019. *4th National Research of Intellectual Seminar POLYCC Proceeding (RISE)*, 42–49.

Kavitha, B. C., Vallikannu, R., & Sankaran, K. S. (2020). Delay-aware concurrent data management method for IoT collaborative mobile edge computing environment. *Microprocessors and Microsystems*, *74*, 103021. https://doi.org/10.1016/j.micpro.2020.103021

LAr. Noriah Mat, CA, C. (2017). *Putrajaya urban farming*. *September*.

Li, I. (2019). *PIR Sensor: Overview, Applications and Projects - Latest open tech from seeed studio*. https://www.seeedstudio.com/blog/2019/08/03/pir-sensor-introduction-and-how-pir-motion-sensor-works-with-arduino-and-raspberry-pi/

Ligo George. (2020). *What is a Microprocessor ? How does it work ?* EletroSome. https://electrosome.com/microprocessor/

Lutkevich, B. (2019). *What is a Microcontroller and How Does it Work?* https://internetofthingsagenda.techtarget.com/definition/microcontroller

Navesh, N. (2014). *Social entrepreneurs out to promote edible gardens in Malaysia - SME | The Star Online*. The Star Newspaper Online. https://www.thestar.com.my/Business/SME/2014/08/05/Planting-the-seeds-for-success-Social-entrepreneurs-out-to-promote-edible-gardens-in-Malaysia/?style=biz

Noh. (2006). *BERITA | Universiti Putra Malaysia*. http://www.upm.edu.my/berita/details/jaminanmakanannegarabi?LANG=en

Pleva GmbH. (1995). Temperature sensor. *Melliand Textilberichte*, *76*(12), 1112. https://doi.org/10.1117/3.1002910.ch11

Randofo. (2016). *Intro to Arduino : 15 Steps (with Pictures) - Instructables*. https://www.instructables.com/Intro-to-Arduino/

Sambath, M., Prasant, M., Bhargav Raghava, N., & Jagadeesh, S. (2019). Iot based garden monitoring system. *Journal of Physics: Conference Series*, *1362*(1), 380–383. https://doi.org/10.1088/1742-6596/1362/1/012069

Shireen, N., & Devi, B. V. (2018). Smart Garden Management System. *International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)*, *5*(3), 6–10.

Thamaraimanalan, T., Vivekk, S. P., Satheeshkumar, G., & Saravanan, P. (2018). Smart Garden Monitoring System Using IOT. *Asian Journal of Applied Science and Technology (AJAST) (Open Access Quarterly International Journal*, *2*(2), 186–192. www.ajast.net

Tillman, M. (2021). *What is Zigbee and why is it important for your smart home?* https://www.pocket-lint.com/smart-home/news/129857-what-is-zigbee-and-why-is-it-important-for-your-smart-home

Trotta, D., & Garengo, P. (2018). Industry 4.0 key research topics: A bibliometric review. *2018 7th International Conference on Industrial Technology and Management, ICITM 2018*, *2018-January*, 113–117. https://doi.org/10.1109/ICITM.2018.8333930

Tsira, V., & Nandi, G. (2014). Bluetooth Technology : Security Issues and Its Prevention. *International Journal of Computer Applications in Technology*, *5*(5), 1833–1837.

Viewed, M. (2015). *Urban farms grow out of community gardens*. https://www.thestar.com.my/News/Community/2014/05/06/Urban-farms-grow-out-of-community-gardens-Putrajaya-Corp-embarks-on-pilot-project-and-provides-techn/

Yeo, K. S., Chian, M. C., Wee Ng, T. C., & Tuan, D. A. (2015). Internet of things:

Trends, challenges and applications. *Proceedings of the 14th International Symposium on Integrated Circuits, ISIC 2014*, 568–571. https://doi.org/10.1109/ISICIR.2014.7029523

APPENDICES I (SOURCE CODE)

```
#define  BLYNK_PRINT Serial
#define  ONE_WIRE_BUS 16
#define  BLYNK_PRINT Serial
#include <SPI.h>
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <DHT.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#include "ThingSpeak.h"
OneWire  oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);
#define DHTPIN 2
#define DHTTYPE DHT11
#define pirPin 5
// Get Auth Token in the Blynk App.
char auth[] = "swyMIhGGDd1YtVIM6gTWc4zgU5MZQrHr";
// WiFi credentials.
char ssid[] = "azmigh@unifi";
char pass[] = "azmi2020";
// ThingSpeak credentials.
String apiKey      = "100LYZKRCMY76D8L";
const char* server = "api.thingspeak.com";
// Declare variables and constants
int sensorPin=A0;                //read sensor value from analog pin
int t;                           //temperature
int h;                           //humidity
int enable2 = 15;                //GPIO15 or D8 is for LED
int pirValue;                    //Motion sensor
int mSensor ;                    //soil sensor
int soil;
const int RELAY_PIN = 4;     //water pump
const int buzzerPin = 13;    //GPIO13 or D7 is for buzzer connected
boolean state = false;          //For water pump to turn on or off
BlynkTimer timer;               //Initialize BlynkTimer Library
DHT dht(DHTPIN, DHTTYPE);    //Initialize DHT Library
WiFiClient client;

void setup()
{
  // Debug console
  Serial.begin(9600);
  Blynk.begin(auth, ssid, pass);
  dht.begin();
  pinMode(sensorPin,  INPUT);
  pinMode(pirPin,     INPUT);
  pinMode(enable2,    OUTPUT);
  pinMode(RELAY_PIN,  OUTPUT);
  pinMode(buzzerPin,  OUTPUT);
  digitalWrite(RELAY_PIN, HIGH);
  digitalWrite(buzzerPin, LOW);
  digitalWrite(enable2, LOW);

  // Setup a function to be called
  timer.setInterval(10000L, Moisture);  //Moisture Sensor : 10 second
  timer.setInterval(600000L, TempHum);     //DHT 11 :10 min
  timer.setInterval(500L, getPirValue);   //main function for motion  500 ms
}

 //=============== Humidity and Temperature ===============

void TempHum()
{
  // Reads sensor value on Digital Pin 2
  // Stores the 2 values in their respective variable. Humidity - hSensor. Temperature
```

```
- tSensor.
  h = dht.readHumidity();
  t = dht.readTemperature();
  if (isnan(h) || isnan(t)) { // In case of abnormal readings from the sensor (not a
number)
Serial.println("Failed to read from DHT sensor!"); // Display error message on serial
monitor
  }
  else{
  // Display readings on serial monitor
  Serial.println (String ("Humidity is : ") + h);
  Serial.println (String ("Temperature is : ") + t);
  Blynk.virtualWrite(V5, h); /* Sends Humidity reading stored in variable hSensor
                        to Virtual Pin V4 defined in the Blynk App*/
  Blynk.virtualWrite(V6, t); /* Sends Temperature reading stored in variable tSensor
                        to Virtual Pin V5 defined in the Blynk App*/
  }
}

 //=============== Soil Moisture ===============
void Moisture(){
  mSensor = analogRead(sensorPin); //Read sensor value on analog PIN
  mSensor = constrain(mSensor, 400, 900);
  soil = map(mSensor, 400, 900, 99, 0);
  if (isnan(soil)) { //
  Serial.println("Failed to read from Soil Moisture sensor!"); //
  }
  else{
  Serial.println (String ("Moisture is : ") + soil); //
  Blynk.virtualWrite(V2, soil); //
  }
  if(soil <= 38){
  Blynk.notify("Plant needs water.. Activating water pump"); //
  Serial.println("Plant needs water.. Activating water pump");
  digitalWrite(RELAY_PIN, HIGH); // Pump ON
  }
  else if(soil >= 45){
    digitalWrite(RELAY_PIN, LOW); // Pump OFF
  }
}

 //=============== Motion Detection ===============

void getPirValue() {
    pirValue = digitalRead(pirPin);
    if (pirValue == 1)
     {
       Serial.println("Motion detected, buzzer & LED ON");
       Blynk.notify("Motion detected, buzzer & LED ON");
       digitalWrite(buzzerPin, HIGH);
       digitalWrite(enable2, HIGH);
     }
     else {
       digitalWrite(buzzerPin, LOW);
       digitalWrite(enable2, LOW);
     }}

BLYNK_WRITE(V1)
{
 if (state == false) {
 state = true;
 digitalWrite(RELAY_PIN,HIGH);
 Serial.println("Manual: You just watered your plant");
 Blynk.notify("Manual : You just watered your plant.");
 }
 else {
```

```
 state = false;
 digitalWrite(RELAY_PIN,LOW);
 }
}

void loop()
{
 Blynk.run();
 timer.run(); // initiates BlynkTimer

  if (client.connect(server,80))  {
    String postStr = apiKey;
    postStr +="&field1=";                  //graph 1 in thingspeak
    postStr += String(t);                  //Pull temperature data to graph 1 in thingspeak
    postStr +="&field2=";                  //graph 2 in thingspeak
    postStr += String(h);                  //Pull humidity data to graph 2 in thingspeak
    postStr +="&field3=";                  //graph 1 in thingspeak
    postStr += String(soil);               //Pull temperature data to graph 1 in thingspeak
    postStr +="&field4=";                  //graph 1 in thingspeak
    postStr += String(pirValue);           //Pull temperature data to graph 1 in thingspeak
    postStr += "\r\n\r\n";
    client.print("POST /update HTTP/1.1\n");
    client.print("Host: api.thingspeak.com\n");
    client.print("Connection: close\n");
    client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n");
    client.print("Content-Type: application/x-www-form-urlencoded\n");
    client.print("Content-Length: ");
    client.print(postStr.length());
    client.print("\n\n");
    client.print(postStr);
  }
  client.stop();
}
```