

**AUTOMATED FERTIGATION SYSTEM  
(WEB-BASED)**

**CHEW YANG**

**BACHELOR OF COMPUTER SCIENCE  
(SOFTWARE ENGINEERING)**

**UNIVERSITI MALAYSIA PAHANG**

## UNIVERSITI MALAYSIA PAHANG

### DECLARATION OF THESIS AND COPYRIGHT

Author's Full Name : Chew Yang

Date of Birth

Title : Automated Fertigation System (Web)

Academic Session : Semester 2 2022/2023

I declare that this thesis is classified as:

- CONFIDENTIAL (Contains confidential information under the Official Secret Act 1997)\*
- RESTRICTED (Contains restricted information as specified by the organization where research was done)\*
- OPEN ACCESS I agree that my thesis to be published as online open access (Full Text)

I acknowledge that Universiti Malaysia Pahang reserves the following rights:

1. The Thesis is the Property of Universiti Malaysia Pahang
2. The Library of Universiti Malaysia Pahang has the right to make copies of the thesis for the purpose of research only.
3. The Library has the right to make copies of the thesis for academic exchange.

Certified by:

\_\_\_\_\_  
(Student's Signature)

\_\_\_\_\_  
(Supervisor's Signature)

\_\_\_\_\_  
Date: 09/06/2023

\_\_\_\_\_  
Muhammad Zulfahmi Toh Bin  
Abdullah @ Toh Chin Lai  
Date: 09/06/2023

NOTE : \* If the thesis is CONFIDENTIAL or RESTRICTED, please attach a thesis declaration letter.



## **SUPERVISOR'S DECLARATION**

I hereby declare that I have checked this thesis and in my opinion, this thesis is adequate in terms of scope and quality for the award of the degree of Bachelor of Computer Science (Software Engineering).

---

Full Name :Muhammad Zufahmi Toh Bin Abdullah @ Toh Chin Lai  
Position :Lecturer  
Date :09/06/2023



## **STUDENT'S DECLARATION**

I hereby declare that the work in this thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at Universiti Malaysia Pahang or any other institutions.

---

Full Name : CHEW YANG  
ID Number : 001223-02-0055  
Date : 09 June 2023

AUTOMATED FERTIGATION SYSTEM (WEB)

CHEW YANG

Thesis submitted in fulfillment of the requirements  
for the award of the degree of  
Bachelor of Computer Science (Software Engineering)

Faculty of Computing  
UNIVERSITI MALAYSIA PAHANG

JUNE 2023

## **ACKNOWLEDGEMENTS**

The successful development of this system is all because I got a superb supervisor who is Muhammad Zulfahmi Toh bin Abdullah @ Toh Chin Lai. He always support and guide me on my degree final year project. Along the way, I have met a lot of problems but Zulafahmi is giving his best on helping me to solve the problem and give me support.

Furthermore, I would like to thank to all the lecturers and friends of the Faculty of Computing who have taught me in the past. I am grateful that I have them along to help me and I believe I cannot success to manage the project on today without the supporting from them.

Moreover, I would like to thank those who have questioned me and they do provide me with some new idea that can be used to upgrade my project.

Finally, I would like to thank my family members in supporting me to study become as an software engineering and without them, there would be no me today. Thank you.

## **ABSTRAK**

**Automated Fertigation System (Web):** Membina satu laman web yang boleh menyimpan data-data dari IoT produk yang digunakan dengan sistem ini.

**Murid :** Chew Yang

**Sarjana Muda :** Computer Science (Software Engineering)

**Tarikh :** June 2023

Muka Surat : 152

Automated Fertigation System (Web) adalah sistem laman web yang bergabung dengan produk IoT untuk menyediakan transformasi kebun dari kebun tradisional ke kebun pintar. Sistem ini mampu memantau suhu tanah di kebun dan kelembapan tanah di kebun. Sistem dapat mengambil data tanah dengan sensor yang terpasang pada tanah untuk setiap tumbuhan. Kita dapat memperoleh data tanah dan memantau status tumbuhan melalui data dalam sistem kita. Sistem ini dikembangkan untuk mengubah kebun tradisional menjadi kebun pintar dan meningkatkan efisiensi hasil panen tumbuhan. Sebagai kesimpulan, sistem kami akan membantu petani untuk memantau kebun mereka dengan cara yang pintar dan lebih efektif.

## **ABSTRACT**

**Automated Fertigation System (Web):** Creating a smart farm system that works with IoT products.

**Student :** Chew Yang

**Degree :** Computer Science (Software Engineering)

**Date :** June 2023

**Pages :** 152

Automated Fertigation System (Web) is a website system that integrate with the IoT product to provide the transformation of farm from the traditional farm to smart farm. This system is able to monitor the temperature of the soil in farm and the moisture of the soil in the farm. The system can retrieve the data of soil by the sensor that is attached to the soil for every plant. We can obtain the soil data and monitoring the plant status through the data in our system. This system is developed to transform the traditional farm into smart farm and increase the efficiency of the plant's harvest. As a conclusion, our system will help the farmer to monitoring their farm in a smart and more effective way.



## TABLE OF CONTENT

<b>DECLARATION</b>	
<b>TITLE PAGE</b>	
<b>ACKNOWLEDGEMENTS</b>	<b>ii</b>
<b>ABSTRAK</b>	<b>iii</b>
<b>ABSTRACT</b>	<b>iv</b>
<b>TABLE OF CONTENT</b>	<b>v</b>
<b>LIST OF TABLES</b>	<b>viii</b>
<b>LIST OF FIGURES</b>	<b>ix</b>
<b>LIST OF ABBREVIATIONS</b>	<b>xiii</b>
<b>CHAPTER 1 INTRODUCTION</b>	<b>14</b>
1.1 Introduction	14
1.2 Problem Statement	15
1.3 Objective	15
1.4 Scope	16
1.5 Thesis Organization	16
<b>CHAPTER 2 LITERATURE REVIEW</b>	<b>17</b>
2.1 Introduction	17
2.2 Existing Systems/Works	17
2.2.1 Agrivi	17
2.2.2 FarmLogs	18
2.2.3 Conservis	19

2.3	Analysis/ Comparison of Existing System	19
2.4	Overview of IoT	20
	2.4.1 Comparison between IoT products	21
2.5	Overview of Fertilizer	22
2.6	Summary	23
<b>CHAPTER 3 METHODOLOGY</b>		<b>24</b>
3.1	Introduction	24
3.2	Project Management Framework / Methodology	24
3.3	Project Requirement	27
3.4	Proposed Design	28
	3.4.1 Flowchart	28
	3.4.2 Context Diagram	29
	3.4.3 Use Case Diagram	30
	3.4.4 Use Case Description	31
	3.4.5 Activity Diagram	56
	3.4.6 Storyboard	61
3.5	Data Design	88
	3.5.1 ERD	88
	3.5.2 Data Dictionary	89
3.6	Proof of Initial Concept	95
3.7	Testing / Validation Plan	96
3.8	Potential Use of Proposed Solution	99
3.9	Gantt Chart	99
<b>CHAPTER 4</b>		<b>100</b>

4.1	Introduction	100
4.2	Implementation process	100
4.3	User Manual	110
4.4	Database Implementation	138
4.5	Testing and Result Discussion	145
<b>CHAPTER 5</b>		<b>150</b>
5.1	Introduction	150
5.2	Project Constraint	150
5.3	Future Work	151
<b>REFERENCES</b>		<b>152</b>

## LIST OF TABLES

Table 2.1	Table of Comparison of Existing System	19
Table 2.2	Table of Comparison of IoT products	21
Table 3.1	Use Case Description of Manage Login	31
Table 3.2	Use Case Description of Manage Farm	33
Table 3.3	Use Case Description of Manage Farmer	36
Table 3.4	Use Case Description of Manage Project	39
Table 3.5	Use Case Description of Manage Purchase	42
Table 3.6	Use Case Description of Manage Inventory	45
Table 3.7	Use Case Description of Manage Sales	48
Table 3.8	Use Case Description of Manage Schedule	51
Table 3.9	Use Case Description of Manage Report	54
Table 3.10	Data Dictionary of calendars	89
Table 3.11	Data Dictionary of controltest	89
Table 3.12	Data Dictionary of farms	90
Table 3.13	Data Dictionary of farmers	90
Table 3.14	Data Dictionary of inventories	91
Table 3.15	Data Dictionary of project	92
Table 3.16	Data Dictionary of purchase	93
Table 3.17	Data Dictionary of sales	93
Table 3.18	Data Dictionary of users	94

## LIST OF FIGURES

Figure 2.1	Agrivi System	18
Figure 2.2	FarmLogs System	18
Figure 2.3	Conservis System	19
Figure 2.4	ESP32 WiFi module	21
Figure 3.1	Scrum Methodology	24
Figure 3.2	Flowchart	28
Figure 3.3	Context Diagram	29
Figure 3.4	Use Case Diagram	30
Figure 3.5	Manage Login Use Case Diagram	31
Figure 3.6	Manage Farm Use Case Diagram	33
Figure 3.7	Manage Farmer Use Case Diagram	36
Figure 3.8	Manage Project Use Case Diagram	39
Figure 3.9	Manage Purchase Use Case Diagram	42
Figure 3.10	Manage Inventory Use Case Diagram	45
Figure 3.11	Manage Sales Use Case Diagram	48
Figure 3.12	Manage Schedule Use Case Diagram	51
Figure 3.13	Manage Report Use Case Diagram	54
Figure 3.14	Manage Login Activity Diagram	56
Figure 3.15	Manage Farm Activity Diagram	57
Figure 3.16	Manage Farmer Activity Diagram	57
Figure 3.17	Manage Project Activity Diagram	58
Figure 3.18	Manage Purchase Activity Diagram	58
Figure 3.19	Manage Inventory Activity Diagram	59
Figure 3.20	Manage Sales Activity Diagram	59
Figure 3.21	Manage Schedule Activity Diagram	60
Figure 3.22	Manage Report Activity Diagram	60
Figure 3.23	Login	61
Figure 3.24	Home	61
Figure 3.25	Farmer	62
Figure 3.26	Create Farmer	63
Figure 3.27	View Farmer	64
Figure 3.28	Edit Farmer	65
Figure 3.29	Farm	66

Figure 3.30	Register Farm	67
Figure 3.31	View Farm	68
Figure 3.32	Edit Farm	69
Figure 3.33	Project	70
Figure 3.34	Add New Project	71
Figure 3.35	View Project	72
Figure 3.36	Edit Project	73
Figure 3.37	Purchase	74
Figure 3.38	Create Purchase	75
Figure 3.39	View Purchase	76
Figure 3.40	Edit Purchase	77
Figure 3.41	Inventory	78
Figure 3.42	Create Inventory	79
Figure 3.43	View Inventory	80
Figure 3.44	Edit Inventory	81
Figure 3.45	Sales	82
Figure 3.46	Create Sales	83
Figure 3.47	View Sales	84
Figure 3.48	Edit Sales	85
Figure 3.49	Schedule	86
Figure 3.50	Report	87
Figure 3.51	ERD	88
Figure 3.52	Sample IoT concept for Automated Fertigation System	95
Figure 3.53	Sample IoT concept for Automated Fertigation System	96
Figure 3.54	Sample User Acceptance Test Form	98
Figure 3.55	Gantt Chart for PSM1	99
Figure 3.56	Gantt Chart for PSM2	99
Figure 4.1	Register Farmer	101
Figure 4.2	Register Farmer Code	101
Figure 4.3	Edit Farmer	102
Figure 4.4	Edit Farmer Code	102
Figure 4.5	View & Delete Farmer	103
Figure 4.6	View Farmer Code	103
Figure 4.7	Delete Farmer Code	103
Figure 4.8	Download Farmer List	104

Figure 4.9	Download Farmer List Code	104
Figure 4.10	Schedule	105
Figure 4.11	Schedule Code	107
Figure 4.12	Report Code	108
Figure 4.13	Report Chart Code	108
Figure 4.14	Login	110
Figure 4.15	Home	111
Figure 4.16	Farmer	112
Figure 4.17	Create Farmer	113
Figure 4.18	View Farmer	114
Figure 4.19	Edit Farmer	115
Figure 4.20	Farm	116
Figure 4.21	Register Farm	117
Figure 4.22	View Farm	118
Figure 4.23	Edit Farm	119
Figure 4.24	Project	120
Figure 4.25	Add New Project	121
Figure 4.26	View Project	122
Figure 4.27	Edit Project	123
Figure 4.28	Purchase	124
Figure 4.29	Create Purchase	125
Figure 4.30	View Purchase	126
Figure 4.31	Edit Purchase	127
Figure 4.32	Inventory	128
Figure 4.33	Create Inventory	129
Figure 4.34	View Inventory	130
Figure 4.35	Edit Inventory	131
Figure 4.36	Sales	132
Figure 4.37	Create Sales	133
Figure 4.38	View Sales	134
Figure 4.39	Edit Sales	135
Figure 4.40	Schedule	136
Figure 4.41	Report	137
Figure 4.42	Database table	138
Figure 4.43	calendars table	138

Figure 4.44	controptest table	139
Figure 4.45	farm table	139
Figure 4.46	farmers table	140
Figure 4.47	inventorys table	140
Figure 4.48	migrations table	141
Figure 4.49	password_resets table	141
Figure 4.50	password_reset_token table	142
Figure 4.51	project table	142
Figure 4.52	purchase table	143
Figure 4.53	sales table	144
Figure 4.54	users table	145
Figure 4.55	UAT Form 1 Page 1	146
Figure 4.56	UAT Form 1 Page 2	147
Figure 4.57	UAT Form 2 Page 1	148
Figure 4.58	UAT Form 2 Page 2	149



## **LIST OF ABBREVIATIONS**

UAT	User Acceptance Test
UMP	Universiti Malaysia Pahang
PIC	Person In Charge

## CHAPTER 1

### INTRODUCTION

#### 1.1 Introduction

Agriculture is one of the most important fields in the world now as the population is increasing day by day. According to the Worldometer website, our current population number in the world has more than 8 billion people (World Population Clock, 2022). According to the United Nations Food and Agriculture Organization (FAO), they have said that the agricultural system will not be able to continue supply the sufficient of food for everyone in the world once the population of human in Earth have surpassed 9.1 billion. At the same time, they estimated the human population will surpassed 9.1 billion by 2050 and surpassed 8 billion by 2030. Furthermore, the latest research show that the world's agricultural system will not be enough for suppling the food for the people in the world by 2023. (Sohnngen, 2017). However, as we can see that the current human population is just exceeding 8 billion people in 2022 and I do believe that the lack of food issue will be happen faster than expected.

Next, why does agriculture is important to us? Agriculture is supplying the food to the human being for example the paddy, the vegetables, and others. Food is very important to us because food is supplying the energy for us. Without food, we will only able to alive for few weeks. Hence, we must ensure that the agriculture sector is supplying the enough of food for us. In order to ensure that the food is sufficient to be supplied, we need to improve the harvest of the plant with the help of IoT technology. With the help of Arduino programming, we may apply the IoT product at the farm to transform the traditional farm into the smart farm. We can collect the data of the farm through all the IoT products and then we may use the system to generate the analysis of data for the farm.

From the collected data, we can know where the weakness of the farm is that we are facing so that we can encounter the weaknesses and increase the harvest of the farm.

In conclusion, with the help of the current latest technology of IoT, we can achieve and maintain the great harvest result. Hence, the Automated Fertigation Management System is invented to help the farmers enhance the harvest result of the plant in the farm.

## **1.2 Problem Statement**

The aim of developing this Automated Fertigation Management System is to overcome the problems that faced by the admin in managing their farmers and farm. Firstly, the admin cannot access to know about the latest information of the farms. In order to know the status of the farm, the farmers would need to feedback to the admin so that only the admin would know the status of the farm and ensure the farm is in good condition. (Joseph, 2017) Secondly, the admin is difficult to handle many projects onto a farm with the paperwork and hence they need the management system to overcome the issue. Thirdly, the admin cannot monitor the sales of the farm and cannot track on it. Next, the cannot remember the inventory for the project in addition they cannot know the status of the inventory too. Lastly, the admin cannot monitor their farm report at anytime and anywhere for example they can visualize the farm data through the graph. In the meanwhile, the admin need a website to monitor the farm and also plant status with the help of IoT sensors (Ibrahim, 2015).

## **1.3 Objective**

What is objective? Objectives are the goal that we set to be achieved. The aim of this project is to develop an online automated fertigation management system that can ease the admin in managing the farms and farmers through the web at anywhere and anytime.

The objectives of this project are:

- To design an automated fertigation management system which is user friendly that ease the farmers to use the system.

- To develop an automated fertigation management system for easier managing the fertigation in farm.
- To study an automated fertigation management system which work with IoT products to manage the farm effectively.

#### **1.4 Scope**

This system is aimed to serve for the farmers to manage their farm easily who using the IoT Automated Fertigation Management System in Pahang. This system is aimed to manage the farmers, farms, projects, purchase, inventory, sales, schedule and report. This lessens the burden of management and provide the system which is online 24/7 for the admin to use the system whenever and wherever they are.

#### **1.5 Thesis Organization**

This thesis consists of three chapters. Chapter 1 shall discuss on the introduction to the project. Chapter 2 generally explain and discuss about the feature of the system and comparison of the current existing system towards my project. Chapter 3 basically discuss about the methodology of project which discuss the framework, design, timeline, and the structure of the project. Chapter 4 will describe about the implementation of the project, user manual, database implementation and testing result. Lastly, Chapter 5 will describe about the conclusion for this project, the challenges that face and the future work need to be done to improve and make the system become more advanced.

## CHAPTER 2

### LITERATURE REVIEW

#### 2.1 Introduction

In this rapid development technology era, we are transforming every field from the traditional way to the modern way with the help of technology. As we can see that the current industry revolution 4.0, IoT has been massively applied in different kind of fields such as agriculture field which is also taking part on transforming into IoT smart farm (Yaser Gamil, 2020). A lot of IoT product is used in the agriculture field for example, soil moisture sensor, humidity sensor, temperature sensor and many others. These sensors which can collect the data from the farm and analyse the data and make the prediction. With the help of the IoT, the transformation of industry revolution 4.0 towards the agriculture field is no longer a dream and it will achieve soon.

#### 2.2 Existing Systems/Works

##### 2.2.1 Agrivi

Agrivi is a company who provide the farm management software which works together with their IoT products. Agrivi do provide the IoT product such as IoT Meteo, IoT Soil and IoT Fleet. IoT Meteo is the sensor that predict the weather by collecting the environment data such as humidity. For example, Agrivi is able to collect the weather data and predict the upcoming weather so that the farmer will know what they should do to overcome the bad weather issue. In the meanwhile, the IoT Soil is the sensor that collecting the data of soil for example, soil moisture. For example, the Agrivi able to collect the data of the soil moisture and give out the analysis whether the soil need to increase moisture or not. The next IoT product is IoT Fleet which can track the real time movement of machinery in their farm. For example, the farmer can track the movement of machinery such as the harvest lorry in the farm to ensure all the plants are harvested.



Figure 2.1 Agrivi System

### 2.2.2 FarmLogs

FarmLogs is a company that providing the field mapping, recent rainfall, rain and heat history, scouting, soil maps, future prices and input. The farmer can view their farm through the website by using their satellite technology. Next, the farmer can see the recent rainfall and know their field condition without checking the rain gauges. Furthermore, the farmer can know the historical effects of heat and rain by referring to the data that is collected from the past. Moreover, the farmer can check the type of soil of their farm through their system. For example, they can know which area of the farm is sand land, soil land and rock land. Lastly, they do provide the inputs which allow the farmer to record their expenses of their farm.

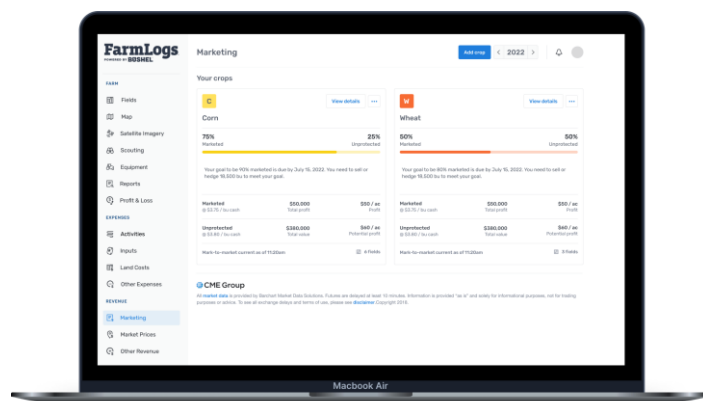


Figure 2.2 FarmLogs System

### 2.2.3 Conservis

Conservis is providing the leading farm management technology service that have a lot of features that can fulfil the requirement of the farmers. The example of services they do provide are data management, harvest management, production management, input management, financial management and analytics.



Figure 2.3 Conservis System

### 2.3 Analysis/ Comparison of Existing System

The comparison of the existing system between Agrivi, FarmLogs and Conservis are compared as below.

Table 2.1 Table of Comparison of Existing System

	Agrivi	FarmLog s	Conservis	Automated Fertigation Management System
Work with IoT products	Yes	Yes	Yes	Yes
Deployment type	Web & mobile	Web	Web	Web & mobile

Control of IoT products	No	No	No	Yes
Report	Yes	Yes	Yes	Yes
Analysis	Yes	Yes	Yes	Yes

The existing system stated above which is the Agrivi, FarmLogs and Conservis have a lot of great technology which can be applied into my project. For example, the three of the existing system got the satellite view function which allow the farmer to see the farm. Next, the existing systems have the weather prediction feature and can collect the actual data of weather. This feature is able to add into my project but using the Google weather prediction API to let the farmer know the latest weather condition. While the existing system does not have the feature to control the IoT product. For example, my project can control the valve of water tank and the fertilizer tank through the IoT product. Furthermore, my project is able to set the timer to control valve for a certain duration.

## 2.4 Overview of IoT

The Internet of Things (IoT) is a product which is the combination of sensors, processor, wire, software and other technologies that make the connection between each other and exchange data with other devices over the Internet.

First of all, the main component that control the product is call Arduino Uno. For example, the module that integrated WiFi module is call Arduino Uno WiFi while the example of the module is ESP32 SIM7600, ESP8266Wifi and others. The Arduino Uno WiFi can connect to the WiFi and transfer the data to the cloud.

How is the ESP32 WiFi module works in the farm? First of all, the ESP32 WiFi module is the main component as central processing unit. Next, all the sensors will be link to the ESP32 module by wires so that the sensor can be integrated and allow for data transferring between each other. In addition, all the data will be transferring to the cloud through WiFi and then store into the cloud database. Lastly, the most important part, all the integration is needed to be programmed so that all the components are working well.



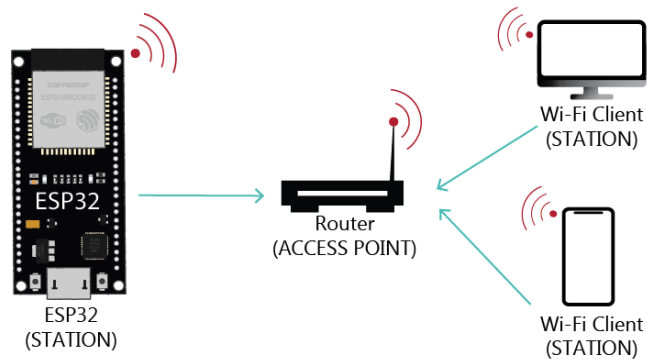


Figure 2.4 ESP32 WiFi module

### 2.4.1 Comparison between IoT products

Table 2.2 Table of Comparison of IoT products

Feature	ESP32 Series	ESP32-S2 Series	ESP32-C3 Series
<b>Launched Year</b>	2016	2020	2020
<b>Number of Core</b>	Dual-core	Single-core	Single-core
<b>WiFi</b>	Supported	Supported	Supported
<b>Bluetooth</b>	Bluetooth 4.2	Not Supported	Bluetooth 5.0
<b>SRAM</b>	520KB	320KB	400KB
<b>ROM</b>	448KB	128KB	384KB
<b>Cache</b>	Two-way	Four-way	Eight-way
<b>Price</b>	~RM20	~RM40	~RM45

## 2.5 Overview of Fertilizer

In agriculture field, we know that the fertilizer is the booster for plant to produce more fruits and grow faster. According to the web, there is 17 essential elements for a plant to grow to their full genetic potential. 14 out of 17 of elements are absorbed by the plant through the soil while the remaining three is come from air and water. (INSTITUTE, 2014). First of all, there are three main element in the fertilizer which are the nitrogen, phosphorus and potassium. The nitrogen is the most important nutrient that make the formation of protein and make up the tissues. Next, is the phosphorus element which is one of the important element that help plants to store energy. It can linked to plant's ability to store energy such as store the energy from the process of photosynthesis. Moreover, potassium is the third main nutrient that can keep the plant healthy and strong. It helps strengthen plants' abilities to resist disease and increasing the crop yields quality.

Next, there are two type of fertilizers that we commonly saw it is granular fertilizers and foliar fertilizers. Granular fertilizers are usually in the pellet form and it is a slow-release form. It need water to dissolve it and slowly absorb by the soil and then only absorb by the plant. The second type of fertilizer is foliar fertilizers. It is in the form of concentrated water soluble powders of synthetic chemical. Usually, the farmers are mixing the water with the foliar fertilizers and then spray it to the plants. It is because the foliar fertilizer is fast acting so it is more controlled and can use it in many different capacities throughout the planting season. (Differences Between Granular Fertiliser and Foliar Fertiliser, 2022)

In conclusion, the foliar fertilizer is more suitable to use with the IoT Automated Fertigation Management System which it is water soluble powders that can perfectly mix with water by our Automated Fertigation Management System's IoT product.

## **2.6 Summary**

As a conclusion for chapter 2, we can take the existing system as a reference so that we can improve our system to become better by adding on the advantages of the system while avoid to implement the disadvantages of the existing system. Hence, we can produce a better system with more great features. Furthermore, we should choose on the most suitable IoT products and the suitable fertilizer in order to work well with the Automated Fertigation Management System and achieve the great result on the harvest of plant.

## CHAPTER 3

### METHODOLOGY

#### 3.1 Introduction

A good project is confirm that there is a good design of paperwork behind it. A good design structure of project is able to produce a good project. It is because the requirement of the project and specification of the project is clearly stated and the timeline of the project is well arranged. With the good timeline designed, we can make sure that the project is going on as per planned.

#### 3.2 Project Management Framework / Methodology

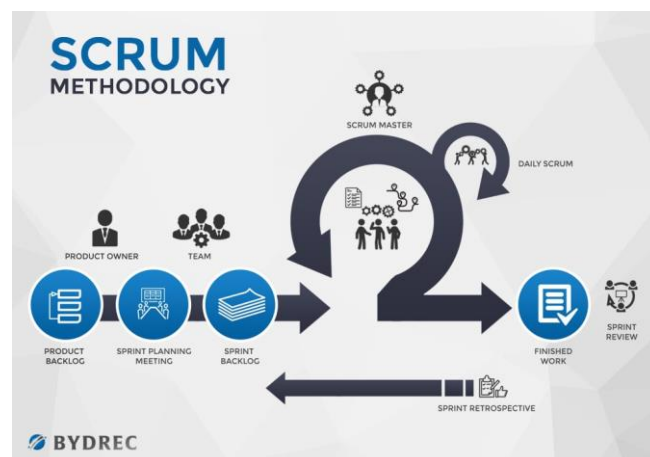


Figure 3.1 Scrum Methodology

The methodology that I have chosen for this project is Scrum methodology. Scrum is categorized under the agile methodology. It will make sure that to deliver an efficient and effective project well on time through the iterative development cycles. Scrum methodology is the one most popular methodology that applied by the software team.

The advantage of Scrum methodology is it is a transparent system that pushes developers to complete their assigned project part and able to deliver it on time. The next advantage of Scrum methodology is it defines the due date of the project and keep the

developers empowered and motivated at every step. Lastly, there is a feedback session at every level of the project to ensure that the quality project is delivered in the end.

Besides, the disadvantage of the Scrum methodology is it is hard to do the planning, structure and organize a project with unspecified mission and vision. Furthermore, there is changes allowed during the software development so that might lead to a delay in delivery time of the project. Last but not least, the Scrum methodology will use much more resources compare to other and there is involvement of stakeholders even there is a small detail change, meeting and discussion. (The SDLC Models & Methodology, n.d.)

### **First Phase : Analysis**

In the analysis phase, it is the first phase within service design and delivery. I will carry out the research and understanding what is the problem statement and the background of Automated Fertigation Management System. To understand the system to be developed, I must understand the information for the project, as well as required function, behavior, performance and interface. I will collect all the system requirements from the client make a meeting or discussion to fully understand the requirement so that the system will be developed and meet the requirements. The important of purpose in this phase is to find out the need and to define the problems that needs to be solved.

### **Second Phase : Design**

In the design phase, I will start gathering the design idea from the client and start to design the interface according to the requirement of the client. First of all, I will make a scratch interface that include important function only. Then, the scratch prototype will be submitted to the client and seek for the feedback from the client. Then the mock up process will be carry out by fulfill the requirement of the client and produce the quality design for the client.

### **Third Phase : Development**

In the development phase, I will now use the collected data in phase one which is analysis phase to start the development process. All the required functions and feature is developed and there are some extra functions add on for the system to make the system become more logical and smooth in operating. In addition, the unwanted or useless function will be taking off if it is unnecessary. Moreover, I will applied the suitable framework for the Automated Fertigation Management System so that the system is well structured and organized. The code in the system will be well arrange and documentation is provided for easier maintenance process and evolution process in the future.

### **Fourth Phase : Testing**

In this phase, I will carry out the testing process to test out the system to know if there is any error or bug on this system. The User Acceptance Test will be carried out to collect the data from the user to know the satisfaction of the system. Furthermore, the system will be soft launching and give access to the client to test out the system and provide some feedback and opinion on the system before the public launching. If there is bug or problem occurs, I will immediately look into the problem and take it seriously to fix the bug in the shortest time and then retest again until the system is satisfied by the client.

### **Fifth Phase : Deployment**

After the client satisfied with the system, the system can be public launching and give access to every users of our system. The system will be deploy to the cloud server and it can be access 24 hours a day by the users.

### **Sixth Phase : Maintenance**

Lastly in the maintenance phase, I will monitor the system from time to time to make sure there are no bugs or errors or problems occurs such as server down, server crash and

many others. I will also collect the feedback from the client so that I can know what is the weakness for this system and will do improvement on it.

### **3.3 Project Requirement**

#### **Functional Requirement**

- The system is able to record and make changes towards farm, farmer, project, purchase, inventory, sales, schedule details and view report.
- The system is able to see the condition of the farm.
- The system is able to see the graph on the collected data.
- The system is able to login into their unique account.

#### **Non-Functional Requirement**

- The system is online 24 hours per day.
- The system is able to support for high users volume at a time.
- The system is running under a https website.
- The system will backup its database twice a day.

#### **Constraints**

- Must have Internet connection to use the system.
- Must have a smart device to access the system.
- Must register an account by admin to use the system.

#### **Limitations**

- Limited support for remote team.

- Limited support for the development process.

### 3.4 Proposed Design

#### 3.4.1 Flowchart

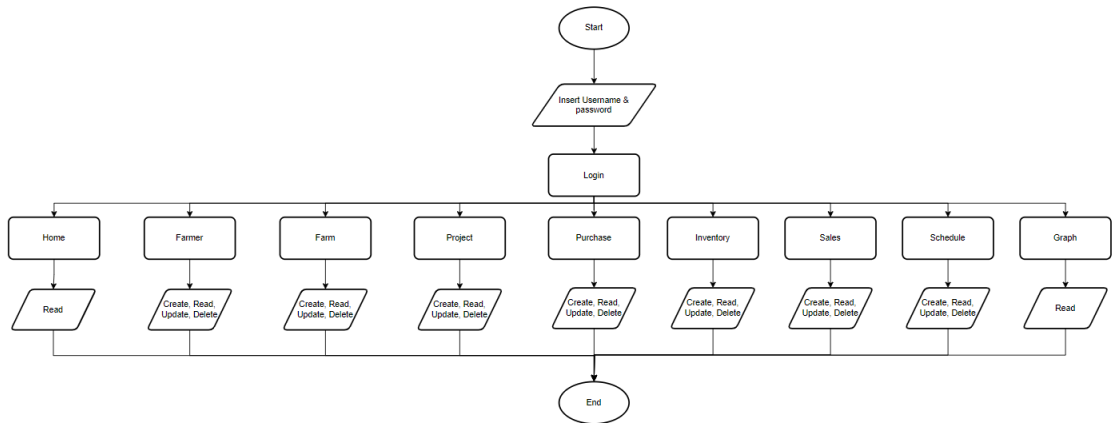


Figure 3.2 Flowchart



### 3.4.2 Context Diagram

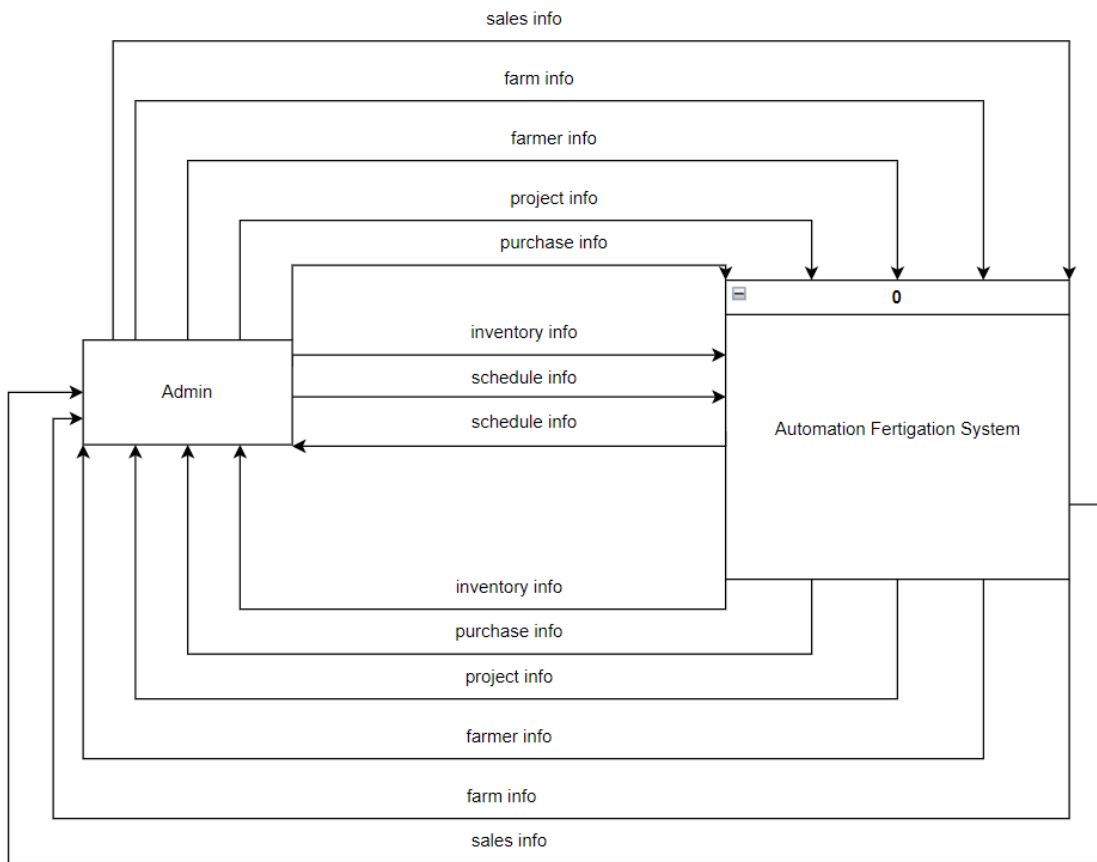


Figure 3.3 Context Diagram

### 3.4.3 Use Case Diagram

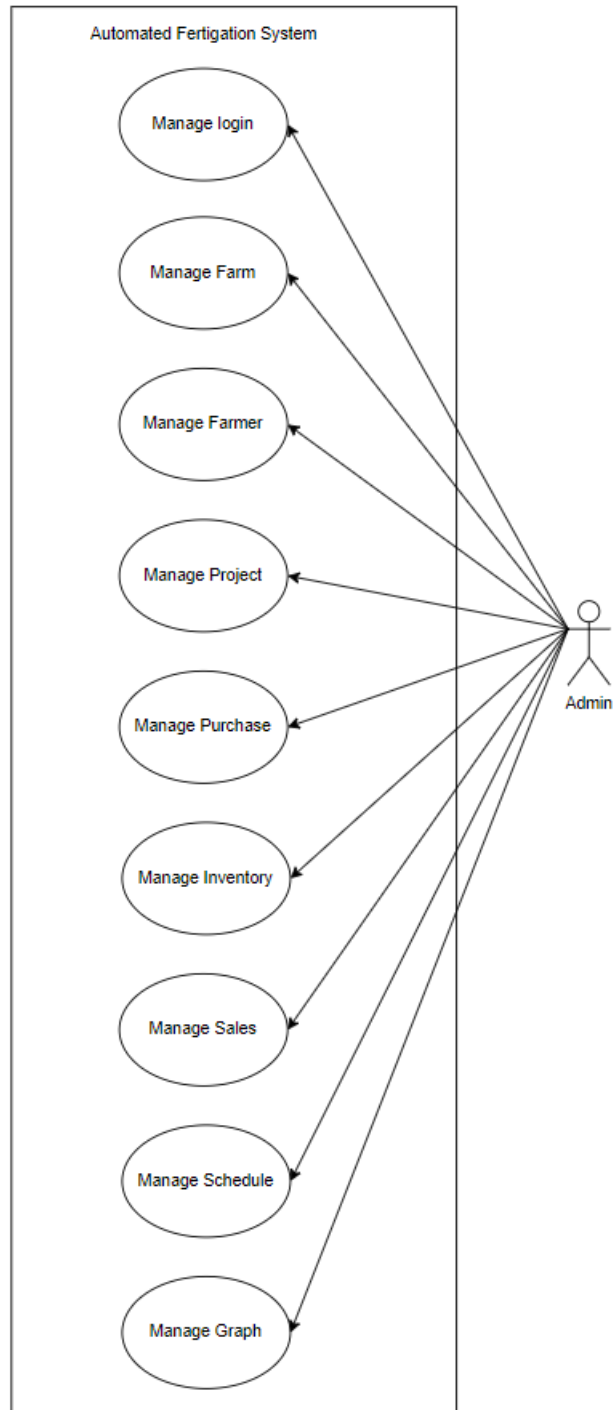


Figure 3.4 Use Case Diagram

### 3.4.4 Use Case Description

Use Case : Manage Login

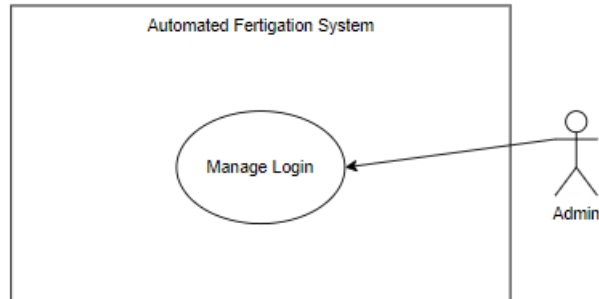


Figure 3.5 Manage Login Use Case Diagram

Table 3.1 Use Case Description of Manage Login

Use Case ID	AFS-UC01
Brief Description	This use case is used by Admin  Admin can login to the system.
Actor	Admin
Pre-Condition	Admin is register to the system.
Basic Flow	Admin  <ol style="list-style-type: none"> <li>1. The use case begins when the Admin land on the landing page.</li> <li>2. The system shows the Automated Fertigation System landing page.</li> <li>3. The user inserts the username and password.</li> <li>4. The user clicks &lt;&lt;Login&gt;&gt; button.</li> <li>5. The system validates the required field data.</li> </ol> <p><b>1. [E1] Invalid Input Data</b></p>

	<p><b>2. [A1] Forgot Password</b></p> <p>6. The system retrieves user data and login successfully.</p> <p>7. The use case ends.</p>
Alternative Flow	<p><b>[A1] Forgot Password [AFS-UC01-A01]</b></p> <p>1. The user selects &lt;&lt; Forgot Password &gt;&gt; button.</p> <p>2. The user inserts the email and click &lt;&lt;Submit&gt;&gt; button.</p> <p>3. The system received the data and sent reset password email.</p> <p>4. The use case continues step 3 in basic flow.</p>
Exception Flow	<p><b>[E1] Invalid Input Data [AFS-UC01-E01]</b></p> <p>1. The system detects invalid input data.</p> <p>2. The system displays invalid input data error message.</p> <p>3. The use case continues step 3 in basic flow.</p>
Post-Condition	<ul style="list-style-type: none"> <li>• User able to login into the system.</li> </ul>
Rules	<p><b>R1: Valid data [AFS-UC01-R01]</b></p> <p>The password include should length at least 8 characters.</p>
Constraints	No applicable.

Use Case : Manage Farm

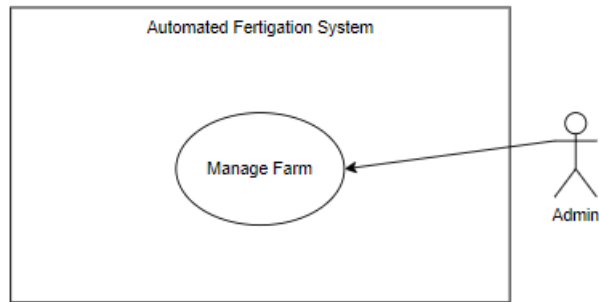


Figure 3.6 Manage Farm Use Case Diagram

Table 3.2 Use Case Description of Manage Farm

Use Case ID	AFS-UC02
Brief Description	This use case is used by Admin.  Admin can manage their profile in the system.
Actor	Admin
Pre-Condition	Admin is login into the system.
Basic Flow	<ol style="list-style-type: none"> <li>1. The use case begins when the Admin login into the system.</li> <li>2. Admin navigate to the farm module page.</li> <li>3. Admin can perform the action as below: <ul style="list-style-type: none"> <li>• Create farm [A1] <b>Create farm</b></li> <li>• View farm [A2] <b>View farm</b></li> <li>• Edit farm [A3] <b>Edit farm</b></li> <li>• Delete farm [A4] <b>Delete farm</b></li> <li>• Download file [A5] <b>Download file</b></li> </ul> </li> <li>4. Action performs successfully.</li> <li>5. The use case ends.</li> </ol>

<p>Alternative Flow</p>	<p><b>[A1] Create farm[AFS-UC02-A01]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Create farm&gt;&gt; button and navigate to Create Farm page.</li> <li>2. Admin input the required data.</li> <li>3. System validates input data. <b>[AFS-UC02-R01]</b></li> <li>4. Admin click &lt;&lt;Add New&gt;&gt; button to create the farm.</li> <li>5. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A2] View farm[AFS-UC02-A02]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;View farm&gt;&gt; button and navigate to View Farm page.</li> <li>2. System display data from database.</li> <li>3. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A3] Edit farm[AFS-UC02-A03]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Edit farm&gt;&gt; button and navigate to Edit Farm page.</li> <li>2. Admin modify the old data.</li> <li>3. System validates input data. <b>[AFS-UC02-R01]</b></li> <li>4. Admin click &lt;&lt;Edit&gt;&gt; button to edit the farm.</li> <li>5. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A4] Delete farm[AFS-UC02-A04]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Delete farm&gt;&gt; button.</li> <li>2. System display confirmation output.</li> <li>3. Admin click &lt;&lt;Confirm&gt;&gt; button to delete the farm.</li> <li>4. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A5] Download farm[AFS-UC02-A05]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Download&gt;&gt; button.</li> <li>2. System retrieve data from database.</li> </ol>
-----------------------------	---

	<ol style="list-style-type: none"> <li>3. The file is downloaded.</li> <li>4. The use case continues step 4 in basic flow.</li> </ol>
Exception Flow	<p><b>[E1] Invalid Input Data [AFS-UC02-E01]</b></p> <ol style="list-style-type: none"> <li>1. The system detects invalid input data.</li> <li>2. The system displays invalid input data error message.</li> <li>3. The use case continues step 3 in basic flow.</li> </ol>
Post-Condition	<ul style="list-style-type: none"> <li>• User able to perform create, view, edit and delete the data from the system.</li> </ul>
Rules	<p><b>R1: Valid data [AFS-UC02-R01]</b></p> <p>The input should not be empty.</p>
Constraints	No applicable.

Use Case : Manage Farmer

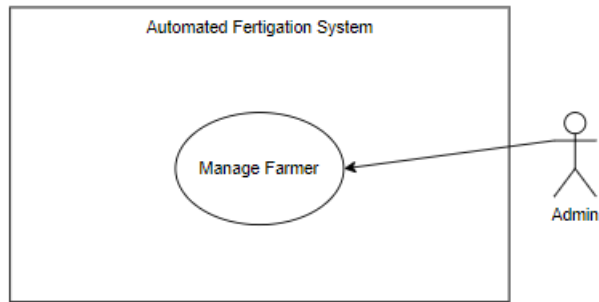


Figure 3.7 Manage Farmer Use Case Diagram

Table 3.3 Use Case Description of Manage Farmer

Use Case ID	AFS-UC03
Brief Description	This use case is used by Admin  Only Admin can register new farmer.
Actor	Admin
Pre-Condition	Admin is login into the system.
Basic Flow	<ol style="list-style-type: none"> <li>1. The use case begins when the Admin login into the system.</li> <li>2. Admin navigate to the farmer module.</li> <li>3. Admin can perform the action as below:                             <ul style="list-style-type: none"> <li>• Create farmer [A1] <b>Create farmer</b></li> <li>• View farmer [A2] <b>View farmer</b></li> <li>• Edit farmer [A3] <b>Edit farmer</b></li> <li>• Delete farmer [A4] <b>Delete farmer</b></li> <li>• Download file [A5] <b>Download file</b></li> </ul> </li> <li>4. Action performs successfully.</li> <li>5. The use case ends.</li> </ol>



<p>Alternative Flow</p>	<p><b>[A1] Create farmer [AFS-UC03-A01]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Create farmer&gt;&gt; button and navigate to Create farmer page.</li> <li>2. Admin input the required data.</li> <li>3. System validates input data. <b>[R1] [E1]</b></li> <li>4. Admin click &lt;&lt;Add New&gt;&gt; button to create the farmer.</li> <li>5. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A2] View farmer [AFS-UC03-A02]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;View farmer &gt;&gt; button and navigate to View farmer page.</li> <li>2. System display data from database.</li> <li>3. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A3] Edit farmer [AFS-UC03-A03]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Edit farmer &gt;&gt; button and navigate to Edit farmer page.</li> <li>2. Admin modify the old data.</li> <li>3. System validates input data. <b>[R1] [E1]</b></li> <li>4. Admin click &lt;&lt;Edit&gt;&gt; button to edit the farmer.</li> <li>5. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A4] Delete farmer [AFS-UC03-A04]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Delete farmer &gt;&gt; button.</li> <li>2. System display confirmation output.</li> <li>3. Admin click &lt;&lt;Confirm&gt;&gt; button to delete the farmer.</li> <li>4. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A5] Download farmer [AFS-UC03-A05]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Download&gt;&gt; button.</li> <li>2. System retrieve data from database.</li> </ol>
-----------------------------	---

	<ol style="list-style-type: none"> <li>3. The file is downloaded.</li> <li>4. The use case continues step 4 in basic flow.</li> </ol>
Exception Flow	<p><b>[E1] Invalid Input Data [AFS-UC03-E01]</b></p> <ol style="list-style-type: none"> <li>4. The system detects invalid input data.</li> <li>5. The system displays invalid input data error message.</li> <li>6. The use case continues step 3 in basic flow.</li> </ol>
Post-Condition	User able to perform create, view, edit, delete and download the data from the system.
Rules	<p><b>R1: Valid data [AFS-UC03-R01]</b></p> <p>The input should not be empty.</p>
Constraints	No applicable.

Use Case : Manage Project

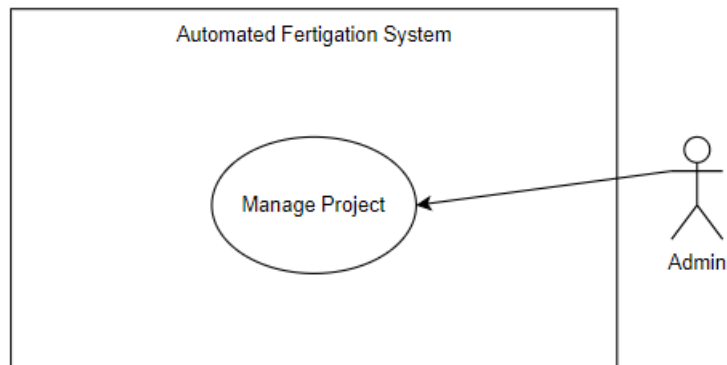


Figure 3.8 Manage Project Use Case Diagram

Table 3.4 Use Case Description of Manage Project

Use Case ID	AFS-UC04
Brief Description	This use case is used by Admin.  Only Admin can manage the project.
Actor	Admin
Pre-Condition	Admin is login into the system and have farm and farmer registered.
Basic Flow	<ol style="list-style-type: none"> <li>6. The use case begins when the Admin login into the system.</li> <li>7. Admin navigate to the project module.</li> <li>8. Admin can perform the action as below:                             <ul style="list-style-type: none"> <li>• Create project [A1] <b>Create project</b></li> <li>• View project [A2] <b>View project</b></li> <li>• Edit project [A3] <b>Edit project</b></li> <li>• Delete project [A4] <b>Delete project</b></li> <li>• Download file [A5] <b>Download file</b></li> </ul> </li> <li>9. Action performs successfully.</li> </ol>

	10. The use case ends.
Alternative Flow	<p><b>[A1] Create project[AFS-UC04-A01]-</b></p> <ol style="list-style-type: none"> <li>6. Admin click &lt;&lt;Create project&gt;&gt; button and navigate to Create project page.</li> <li>7. Admin input the required data.</li> <li>8. System validates input data. <b>[R1] [E1]</b></li> <li>9. Admin click &lt;&lt;Add New&gt;&gt; button to create the project.</li> <li>10. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A2] View project[AFS-UC04-A02]-</b></p> <ol style="list-style-type: none"> <li>4. Admin click &lt;&lt;View project&gt;&gt; button and navigate to View project page.</li> <li>5. System display data from database.</li> <li>6. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A3] Edit project [AFS-UC04-A03]-</b></p> <ol style="list-style-type: none"> <li>6. Admin click &lt;&lt;Edit project&gt;&gt; button and navigate to Edit project page.</li> <li>7. Admin modify the old data.</li> <li>8. System validates input data. <b>[R1] [E1]</b></li> <li>9. Admin click &lt;&lt;Edit&gt;&gt; button to edit the project.</li> <li>10. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A4] Delete project [AFS-UC04-A04]-</b></p> <ol style="list-style-type: none"> <li>5. Admin click &lt;&lt;Delete project&gt;&gt; button.</li> <li>6. System display confirmation output.</li> <li>7. Admin click &lt;&lt;Confirm&gt;&gt; button to delete the project.</li> <li>8. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A5] Download project [AFS-UC04-A05]-</b></p>

	<ol style="list-style-type: none"> <li>5. Admin click &lt;&lt;Download&gt;&gt; button.</li> <li>6. System retrieve data from database.</li> <li>7. The file is downloaded.</li> <li>8. The use case continues step 4 in basic flow.</li> </ol>
Exception Flow	<p><b>[E1] Invalid Input Data [AFS-UC04-E01]</b></p> <ol style="list-style-type: none"> <li>1. The system detects invalid input data.</li> <li>2. The system displays invalid input data error message.</li> </ol> <p>The use case continues step 3 in basic flow.</p>
Post-Condition	User able to perform create, view, edit, delete and download the data from the system.
Rules	<p><b>R1: Valid data [AFS-UC04-R01]</b></p> <p>The input should not be empty.</p>
Constraints	No applicable.

Use Case : Manage Purchase

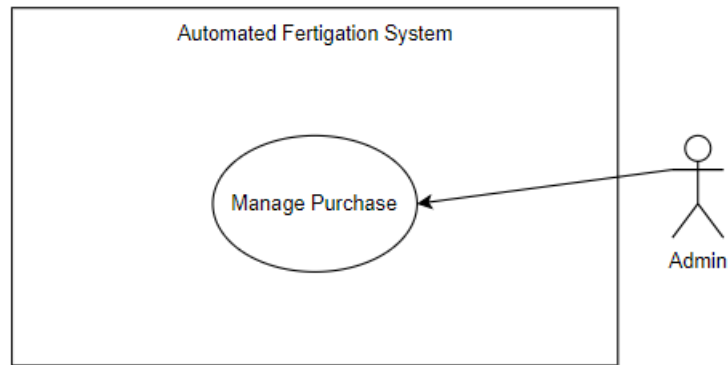


Figure 3.9 Manage Purchase Use Case Diagram

Table 3.5 Use Case Description of Manage Purchase

Use Case ID	AFS-UC05
Brief Description	This use case is used by Admin.  Only Admin can manage the purchase.
Actor	Admin
Pre-Condition	Admin is login into the system and have project registered.
Basic Flow	<ol style="list-style-type: none"> <li>1. The use case begins when the Admin login into the system.</li> <li>2. Admin navigate to the purchase module.</li> <li>3. Admin can perform the action as below:                             <ul style="list-style-type: none"> <li>• Create purchase [A1] <b>Create purchase</b></li> <li>• View purchase [A2] <b>View purchase</b></li> <li>• Edit purchase [A3] <b>Edit purchase</b></li> <li>• Delete purchase [A4] <b>Delete purchase</b></li> <li>• Download file [A5] <b>Download file</b></li> </ul> </li> <li>4. Action performs successfully.</li> <li>5. The use case ends.</li> </ol>

<p>Alternative Flow</p>	<p><b>[A1] Create purchase [AFS-UC05-A01]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Create purchase &gt;&gt; button and navigate to Create purchase page.</li> <li>2. Admin input the required data.</li> <li>3. System validates input data. <b>[R1] [E1]</b></li> <li>4. Admin click &lt;&lt;Add New&gt;&gt; button to create the purchase.</li> <li>5. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A2] View purchase [AFS-UC05-A02]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;View purchase &gt;&gt; button and navigate to View purchase page.</li> <li>2. System display data from database.</li> <li>3. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A3] Edit purchase [AFS-UC05-A03]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Edit purchase &gt;&gt; button and navigate to Edit purchase page.</li> <li>2. Admin modify the old data.</li> <li>3. System validates input data. <b>[R1] [E1]</b></li> <li>4. Admin click &lt;&lt;Edit&gt;&gt; button to edit the purchase.</li> <li>5. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A4] Delete purchase [AFS-UC05-A04]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Delete purchase &gt;&gt; button.</li> <li>2. System display confirmation output.</li> <li>3. Admin click &lt;&lt;Confirm&gt;&gt; button to delete the purchase.</li> <li>4. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A5] Download purchase [AFS-UC05-A05]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Download&gt;&gt; button.</li> <li>2. System retrieve data from database.</li> </ol>
-------------------------	--

	<ol style="list-style-type: none"> <li>3. The file is downloaded.</li> <li>4. The use case continues step 4 in basic flow.</li> </ol>
Exception Flow	<p><b>[E1] Invalid Input Data [AFS-UC05-E01]</b></p> <ol style="list-style-type: none"> <li>1. The system detects invalid input data.</li> <li>2. The system displays invalid input data error message.</li> <li>3. The use case continues step 3 in basic flow.</li> </ol>
Post-Condition	User able to perform create, view, edit, delete and download the data from the system.
Rules	<p><b>R1: Valid data [AFS-UC05-R01]</b></p> <p>The input should not be empty.</p>
Constraints	No applicable.



Use Case : Manage Inventory

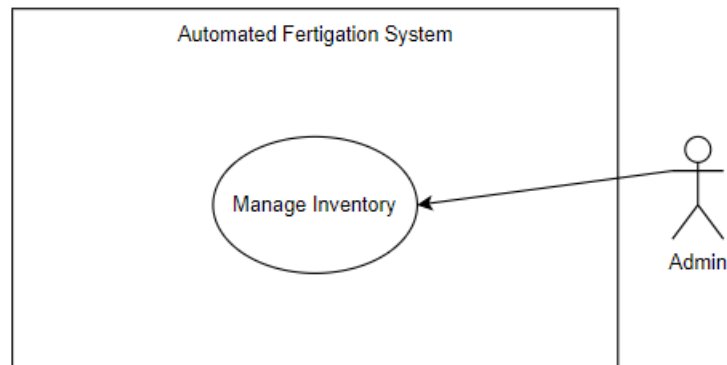


Figure 3.10 Manage Inventory Use Case Diagram

Table 3.6 Use Case Description of Manage Inventory

Use Case ID	AFS-UC06
Brief Description	This use case is used by Admin.  Admin can manage the inventory.
Actor	Admin
Pre-Condition	Admin is login into the system and have purchase.
Basic Flow	<ol style="list-style-type: none"> <li>1. The use case begins when the Admin login into the system.</li> <li>2. Admin navigate to the inventory module.</li> <li>3. Admin can perform the action as below:                             <ul style="list-style-type: none"> <li>• Create inventory [A1] <b>Create inventory</b></li> <li>• View inventory [A2] <b>View inventory</b></li> <li>• Edit inventory [A3] <b>Edit inventory</b></li> <li>• Delete inventory [A4] <b>Delete inventory</b></li> <li>• Download file [A5] <b>Download file</b></li> </ul> </li> <li>4. Action performs successfully.</li> <li>5. The use case ends.</li> </ol>

<p>Alternative Flow</p>	<p><b>[A1] Create inventory [AFS-UC06-A01]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Create inventory &gt;&gt; button and navigate to Create inventory page.</li> <li>2. Admin input the required data.</li> <li>3. System validates input data. <b>[R1] [E1]</b></li> <li>4. Admin click &lt;&lt;Add New&gt;&gt; button to create the inventory.</li> <li>5. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A2] View inventory [AFS-UC06-A02]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;View inventory &gt;&gt; button and navigate to View inventory page.</li> <li>2. System display data from database.</li> <li>3. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A3] Edit inventory [AFS-UC06-A03]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Edit inventory &gt;&gt; button and navigate to Edit inventory page.</li> <li>2. Admin modify the old data.</li> <li>3. System validates input data. <b>[R1] [E1]</b></li> <li>4. Admin click &lt;&lt;Edit&gt;&gt; button to edit the inventory.</li> <li>5. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A4] Delete inventory [AFS-UC06-A04]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Delete inventory &gt;&gt; button.</li> <li>2. System display confirmation output.</li> <li>3. Admin click &lt;&lt;Confirm&gt;&gt; button to delete the inventory.</li> <li>4. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A5] Download inventory [AFS-UC06-A05]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Download&gt;&gt; button.</li> <li>2. System retrieve data from database.</li> </ol>
-----------------------------	---

	<ol style="list-style-type: none"> <li>3. The file is downloaded.</li> <li>4. The use case continues step 4 in basic flow.</li> </ol>
Exception Flow	<p><b>[E1] Invalid Input Data [AFS-UC06-E01]</b></p> <ol style="list-style-type: none"> <li>1. The system detects invalid input data.</li> <li>2. The system displays invalid input data error message.</li> <li>3. The use case continues step 3 in basic flow.</li> </ol>
Post-Condition	User able to perform create, view, edit, delete and download the data from the system.
Rules	<p><b>R1: Valid data [AFS-UC06-R01]</b></p> <p>The input should not be empty.</p>
Constraints	No applicable.

Use Case : Manage Sales

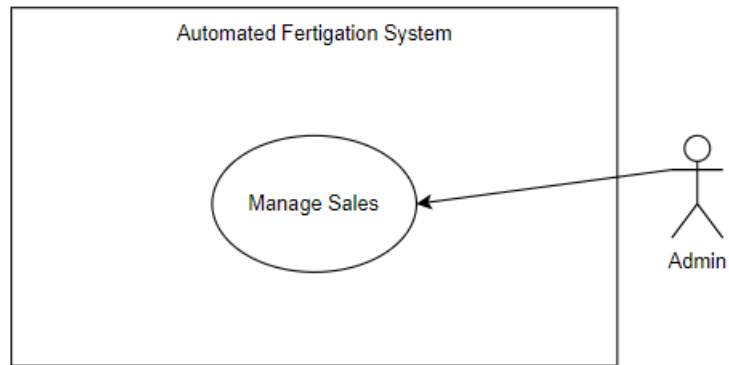


Figure 3.11 Manage Sales Use Case Diagram

Table 3.7 Use Case Description of Manage Sales

Use Case ID	AFS-UC07
Brief Description	This use case is used by Admin.  Admin can manage the sales.
Actor	Admin
Pre-Condition	Admin is login into the system.
Basic Flow	<ol style="list-style-type: none"> <li>1. The use case begins when the Admin login into the system.</li> <li>2. Admin navigate to the sales module.</li> <li>3. Admin can perform the action as below:                             <ul style="list-style-type: none"> <li>• Create sales [A1] <b>Create sales</b></li> <li>• View sales [A2] <b>View sales</b></li> <li>• Edit sales [A3] <b>Edit sales</b></li> <li>• Delete sales [A4] <b>Delete sales</b></li> <li>• Download file [A5] <b>Download file</b></li> </ul> </li> <li>4. Action performs successfully.</li> </ol>

	<p>5. The use case ends.</p>
<p>Alternative Flow</p>	<p><b>[A1] Create sales [AFS-UC07-A01]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Create sales &gt;&gt; button and navigate to Create sales page.</li> <li>2. Admin input the required data.</li> <li>3. System validates input data. <b>[R1] [E1]</b></li> <li>4. Admin click &lt;&lt;Add New&gt;&gt; button to create the sales.</li> <li>5. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A2] View sales [AFS-UC07-A02]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;View inventory &gt;&gt; button and navigate to View sales page.</li> <li>2. System display data from database.</li> <li>3. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A3] Edit sales [AFS-UC07-A03]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Edit sales &gt;&gt; button and navigate to Edit sales page.</li> <li>2. Admin modify the old data.</li> <li>3. System validates input data. <b>[R1] [E1]</b></li> <li>4. Admin click &lt;&lt;Edit&gt;&gt; button to edit the sales.</li> <li>5. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A4] Delete sales [AFS-UC07-A04]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Delete sales &gt;&gt; button.</li> <li>2. System display confirmation output.</li> <li>3. Admin click &lt;&lt;Confirm&gt;&gt; button to delete the sales.</li> <li>4. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A5] Download sales [AFS-UC07-A05]-</b></p>

	<ol style="list-style-type: none"> <li>1. Admin click &lt;&lt;Download&gt;&gt; button.</li> <li>2. System retrieve data from database.</li> <li>3. The file is downloaded.</li> <li>4. The use case continues step 4 in basic flow.</li> </ol>
Exception Flow	<p><b>[E1] Invalid Input Data [AFS-UC07-E01]</b></p> <ol style="list-style-type: none"> <li>4. The system detects invalid input data.</li> <li>5. The system displays invalid input data error message.</li> </ol> <p>The use case continues step 3 in basic flow.</p>
Post-Condition	User able to perform create, view, edit, delete and download the data from the system.
Rules	<p><b>R1: Valid data [AFS-UC07-R01]</b></p> <p>The input should not be empty.</p>
Constraints	No applicable.

Use Case : Manage Schedule

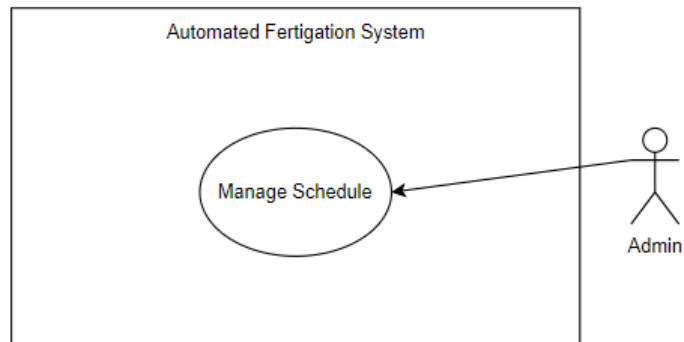


Figure 3.12 Manage Schedule Use Case Diagram

Table 3.8 Use Case Description of Manage Schedule

Use Case ID	AFS-UC08
Brief Description	This use case is used by Admin.  Admin can manage the schedule.
Actor	Admin
Pre-Condition	Admin is login into the system.
Basic Flow	<ol style="list-style-type: none"> <li>1. The use case begins when the Admin login into the system.</li> <li>2. Admin navigate to the sales module.</li> <li>3. Admin can perform the action as below:             <ul style="list-style-type: none"> <li>• Create schedule [A1] <b>Create schedule</b></li> <li>• View schedule [A2] <b>View schedule</b></li> <li>• Edit schedule [A3] <b>Edit schedule</b></li> <li>• Delete schedule [A4] <b>Delete schedule</b></li> </ul> </li> <li>4. Action performs successfully.</li> <li>5. The use case ends.</li> </ol>

<p>Alternative Flow</p>	<p><b>[A1] Create schedule [AFS-UC08-A01]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click on the date on calendar.</li> <li>2. Admin input the required data.</li> <li>3. Admin click &lt;&lt;Add New&gt;&gt; button to create the schedule.</li> <li>4. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A2] View schedule [AFS-UC08-A02]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click navigate to schedule module.</li> <li>2. System display data from database.</li> <li>3. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A3] Edit schedule [AFS-UC08-A03]-</b></p> <ol style="list-style-type: none"> <li>1. Admin drag and drop schedule on new date.</li> <li>2. The use case continues step 4 in basic flow.</li> </ol> <p><b>[A4] Delete schedule [AFS-UC08-A04]-</b></p> <ol style="list-style-type: none"> <li>1. Admin click on the schedule on date.</li> <li>2. System display confirmation output.</li> <li>3. Admin click &lt;&lt;Confirm&gt;&gt; button to delete the schedule.</li> <li>4. The use case continues step 4 in basic flow.</li> </ol>
<p>Exception Flow</p>	<p>-</p>
<p>Post-Condition</p>	<p>User can create, view, edit and delete the schedule from the system.</p>
<p>Rules</p>	<p>-</p>



Constraints	No applicable.
-------------	----------------

Use Case : Manage Report

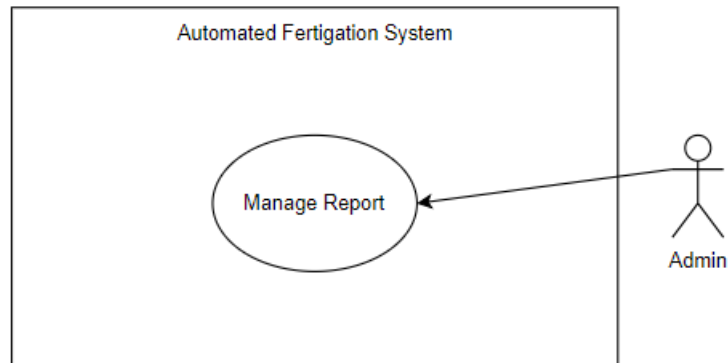


Figure 3.13 Manage Report Use Case Diagram

Table 3.9 Use Case Description of Manage Report

Use Case ID	AFS-UC09
Brief Description	This use case is used by Admin.  Admin can view the report.
Actor	Admin
Pre-Condition	Admin is login into the system.
Basic Flow	<ol style="list-style-type: none"> <li>1. The use case begins when the Admin login into the system.</li> <li>2. Admin navigate to the sales module.</li> <li>3. Admin select the farm.</li> <li>4. System display graph according to the farm.</li> <li>5. Graph display successfully.</li> <li>6. The use case ends.</li> </ol>
Alternative Flow	-

Exception Flow	-
Post- Condition	User able to perform view the graph.
Rules	
Constraints	No applicable.

### 3.4.5 Activity Diagram

Manage Login

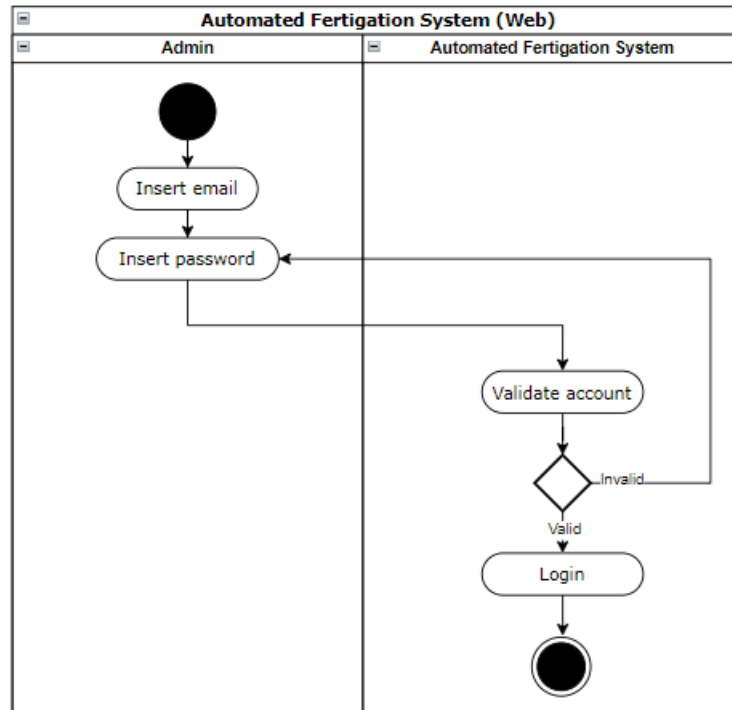


Figure 3.14 Manage Login Activity Diagram

## Manage Farm

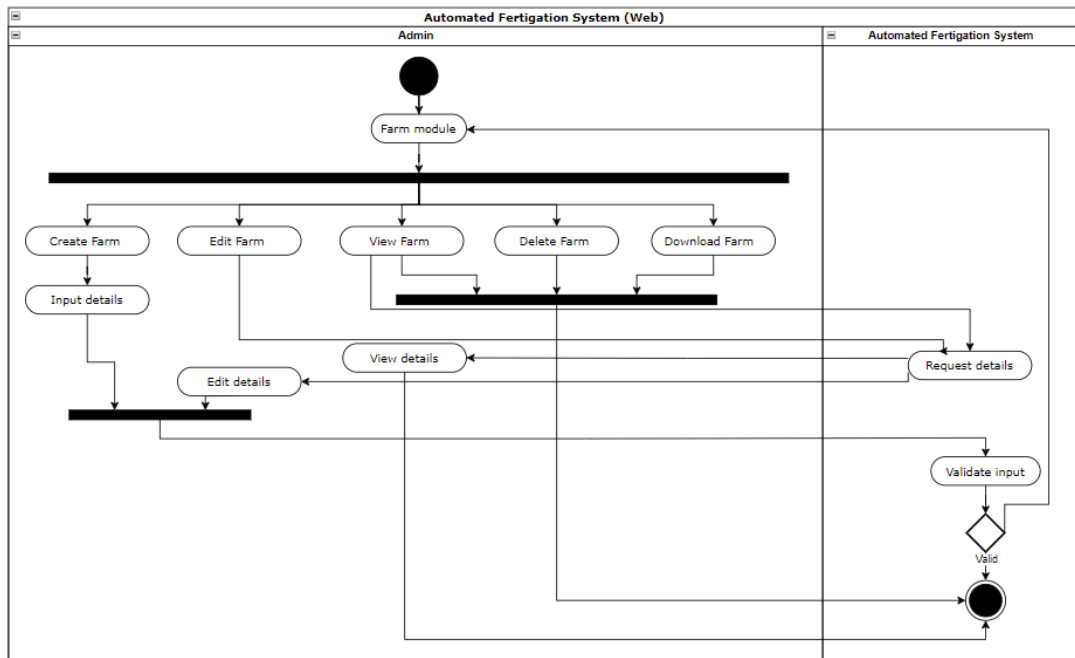


Figure 3.15 Manage Farm Activity Diagram

## Manage Farmer

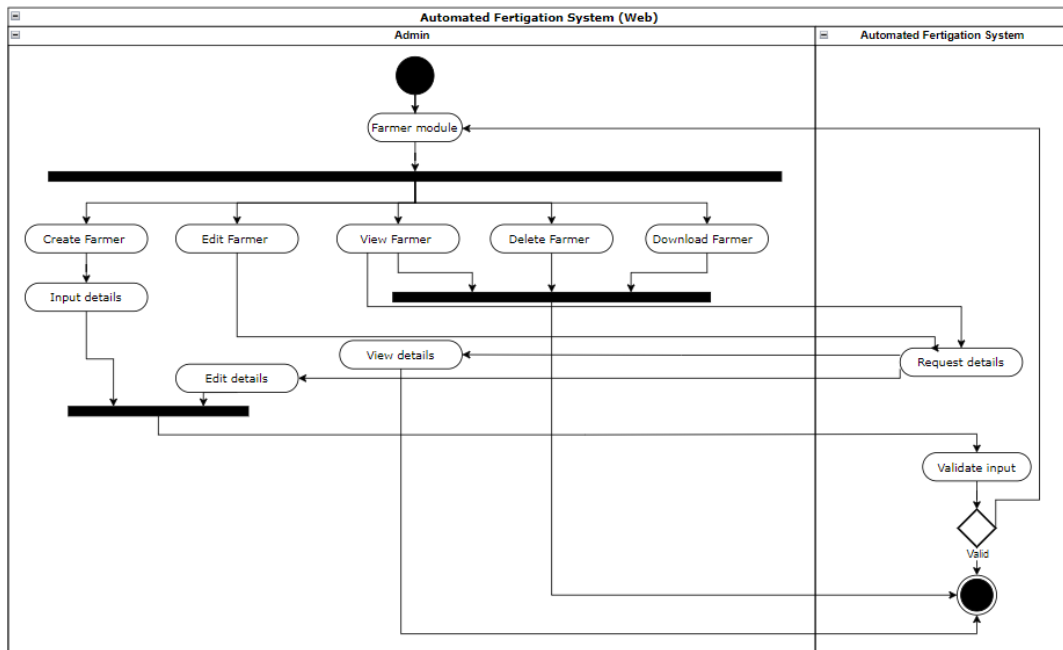


Figure 3.16 Manage Farmer Activity Diagram

## Manage Project

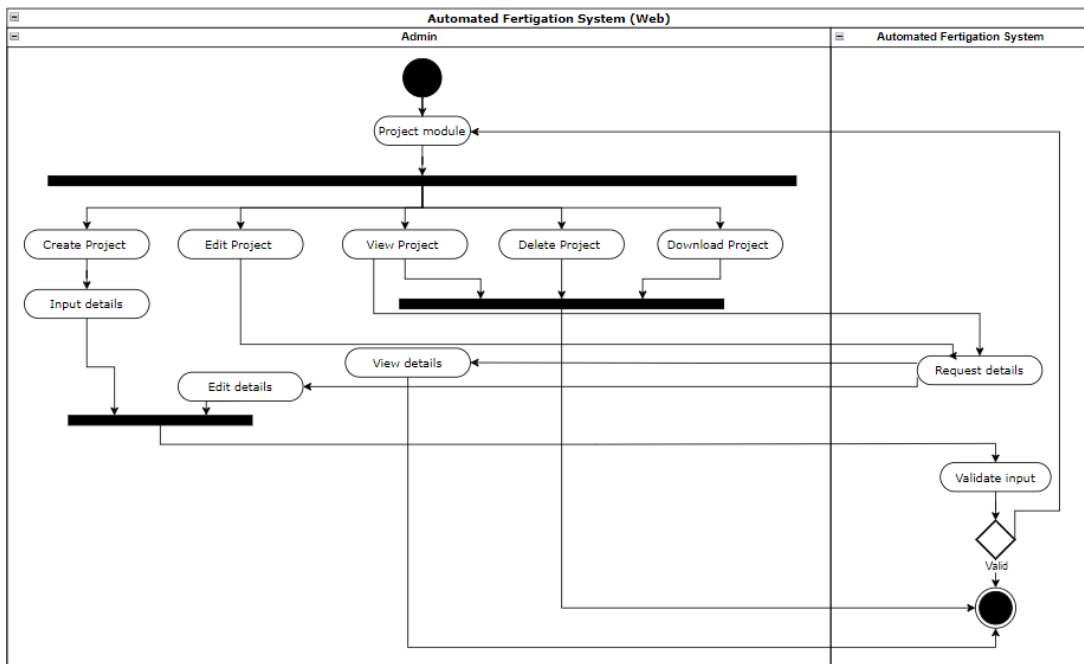


Figure 3.17 Manage Project Activity Diagram

## Manage Purchase

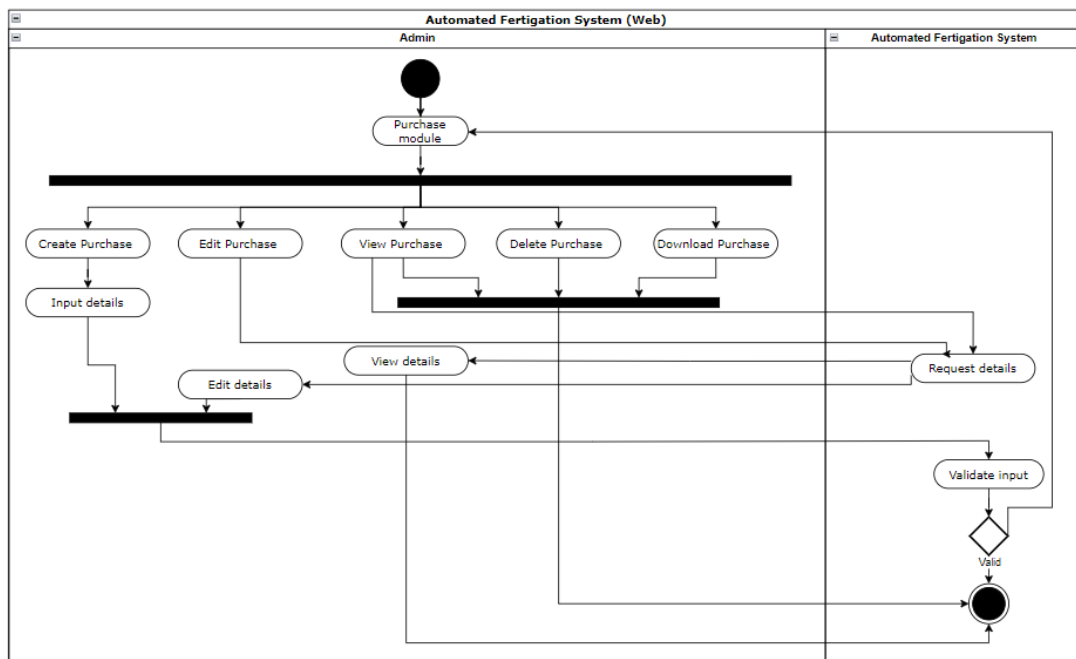


Figure 3.18 Manage Purchase Activity Diagram

## Manage Inventory

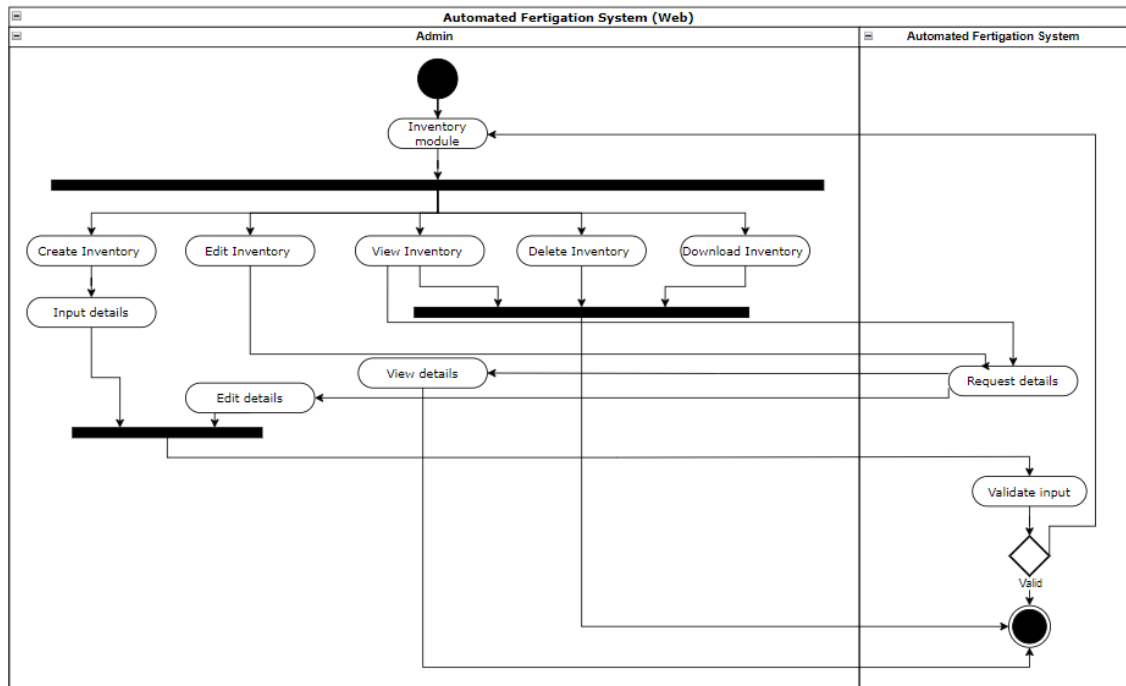


Figure 3.19 Manage Inventory Activity Diagram

## Manage Sales

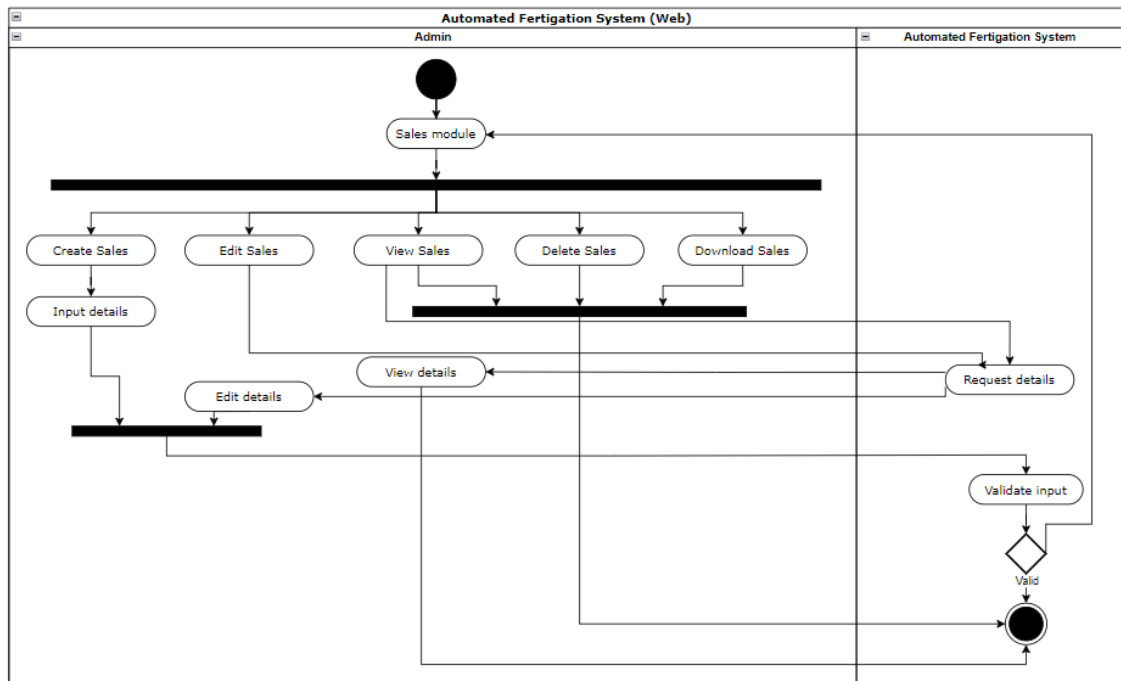


Figure 3.20 Manage Sales Activity Diagram

## Manage Schedule

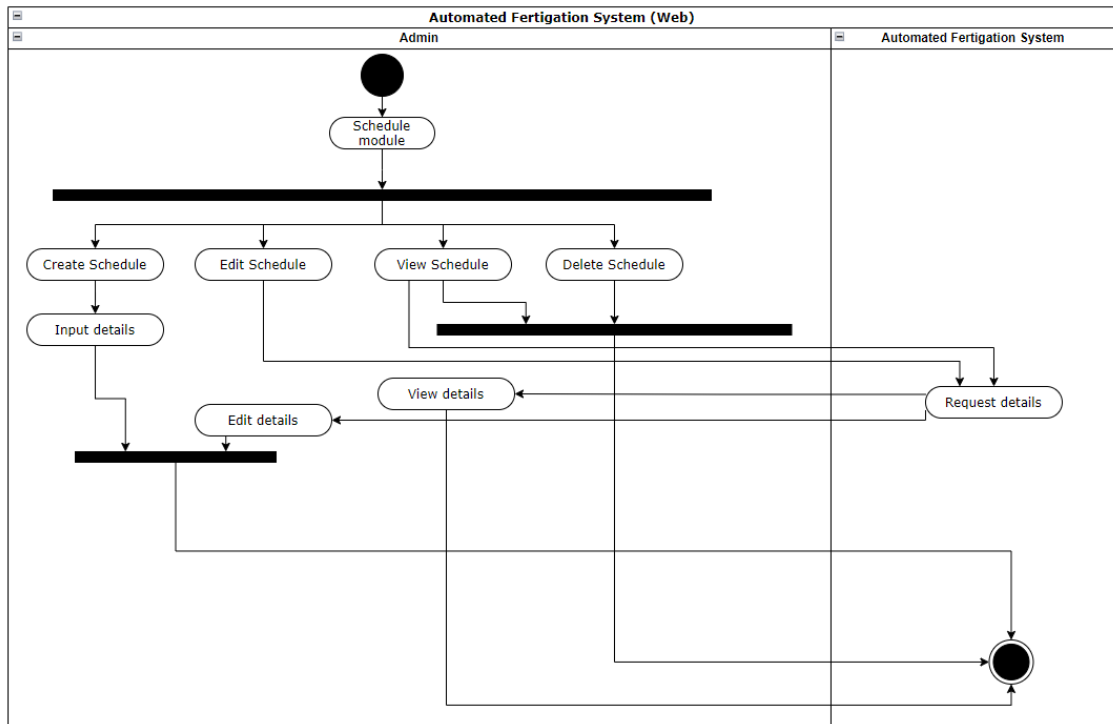


Figure 3.21 Manage Schedule Activity Diagram

## Manage Report

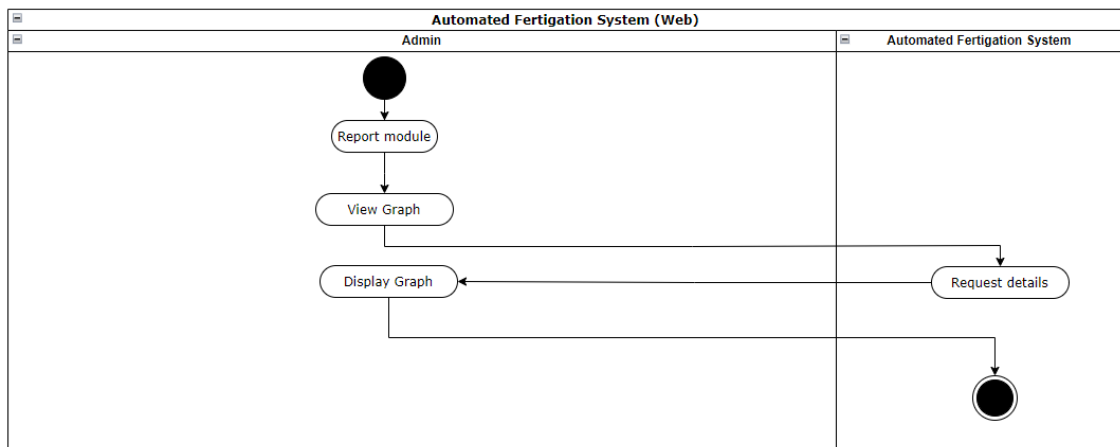


Figure 3.22 Manage Report Activity Diagram



### 3.4.6 Storyboard

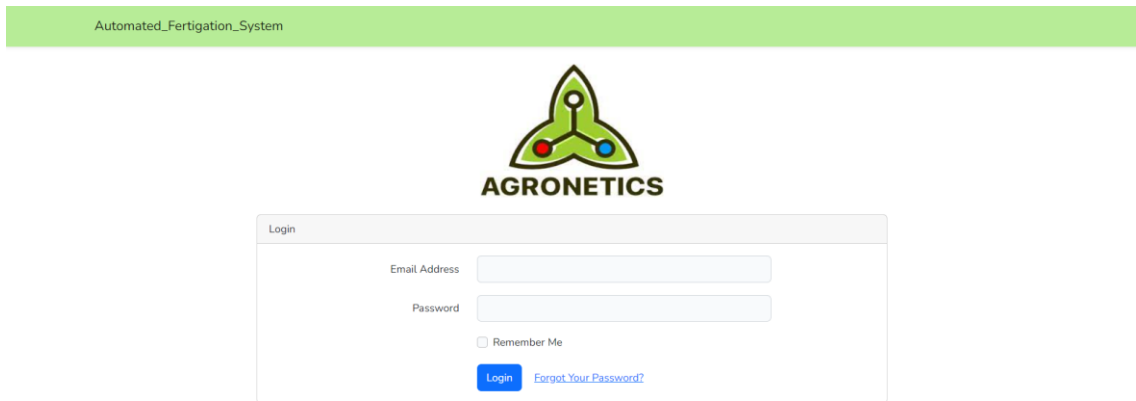


Figure 3.23 Login

This is the login module, admin can login into the system by entering the email address and password. After done filling, the login button is clicked and it will login the admin.

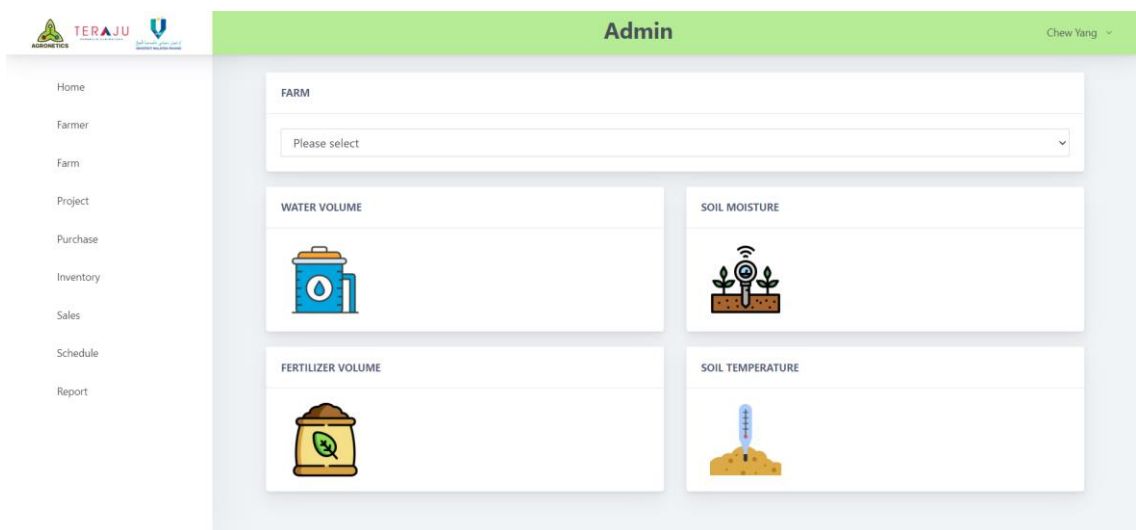


Figure 3.24 Home

This is the home page of system. Admin can use the navigation bar on the left hand side to navigate to other module. Admin can select the farm and then system will automatically show the farm data out at below.

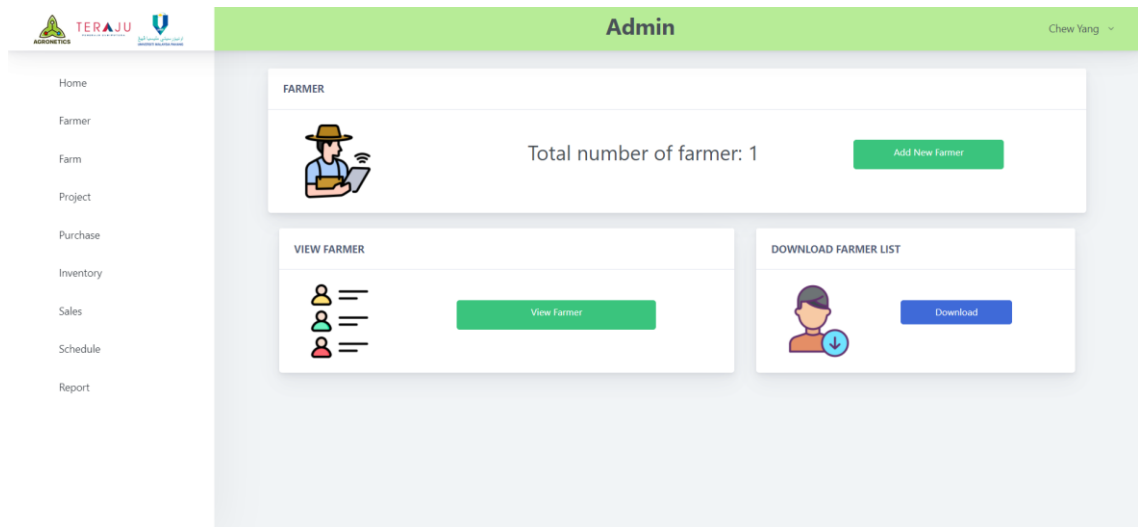


Figure 3.25 Farmer

This is the Farmer module. Admin can perform create, view, update and delete functions for the farmer.

- Admin can press the 'Add New Farmer' and system will redirect to create farmer page.
- Admin can press the 'View Farmer' button and system will redirect to view farmer page.
- Admin can press the 'Download' button and system will auto download all farmer information in a excel file.

The screenshot displays the 'Admin' dashboard with a 'REGISTER FARMER' form. The form contains five input fields: Name, Email, Password, Phone, and Location. At the bottom right of the form are two buttons: a green 'Register' button and a grey 'Cancel' button. The left sidebar lists navigation options: Home, Farmer, Farm, Project, Purchase, Inventory, Sales, Schedule, and Report. The top header features the 'Admin' title and the user name 'Chew Yang'.

Figure 3.26 Create Farmer

This is the Create Farmer page.

- Admin need to fill in the 'Name', 'Email', 'Password', 'Phone' and 'Location'.
- Admin need to click 'Register' button to register farmer.

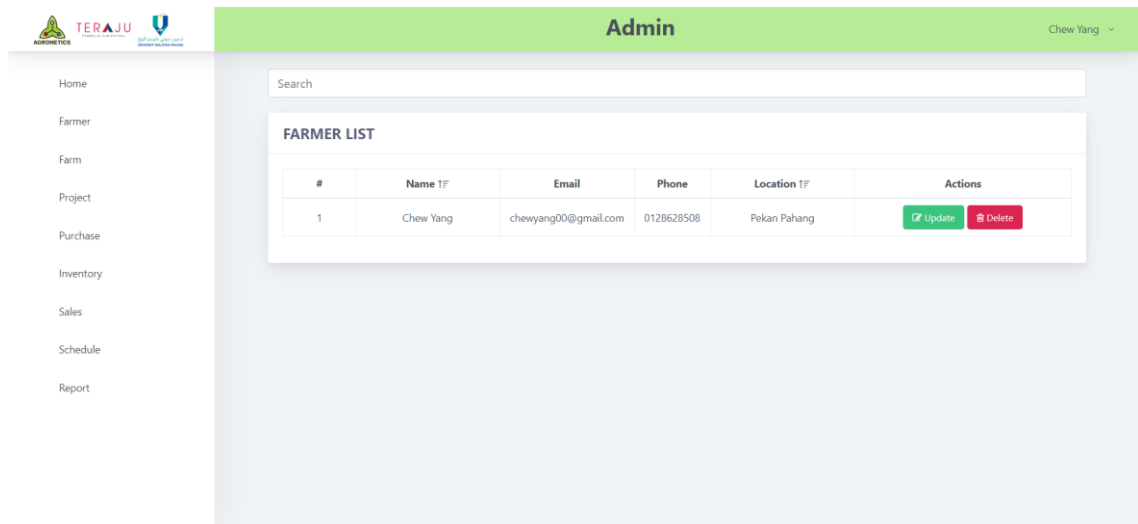
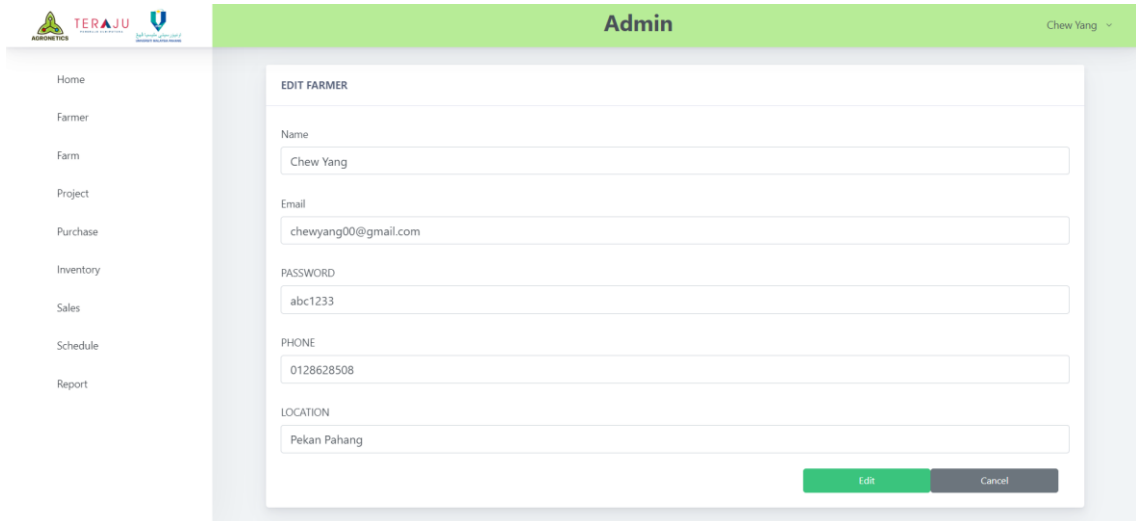


Figure 3.27 View Farmer

This is the View Farmer page.

- Admin can see all the farmer at this page.
- Admin can click 'Update' button to update the farmers' details.
- Admin can click 'Delete' button to delete the farmer.
- Admin can search for farmer by their name using the search bar above.



Admin Chew Yang

EDIT FARMER

Name  
Chew Yang

Email  
chewyang00@gmail.com

PASSWORD  
abc1233

PHONE  
0128628508

LOCATION  
Pekan Pahang

Edit Cancel

Figure 3.28 Edit Farmer

This is the Edit Farmer page.

- Admin can make changes to the old data.
- Admin can click on 'Edit' button to edit the data.
- Admin can click on 'Cancel' button to go back to View Farmer page.

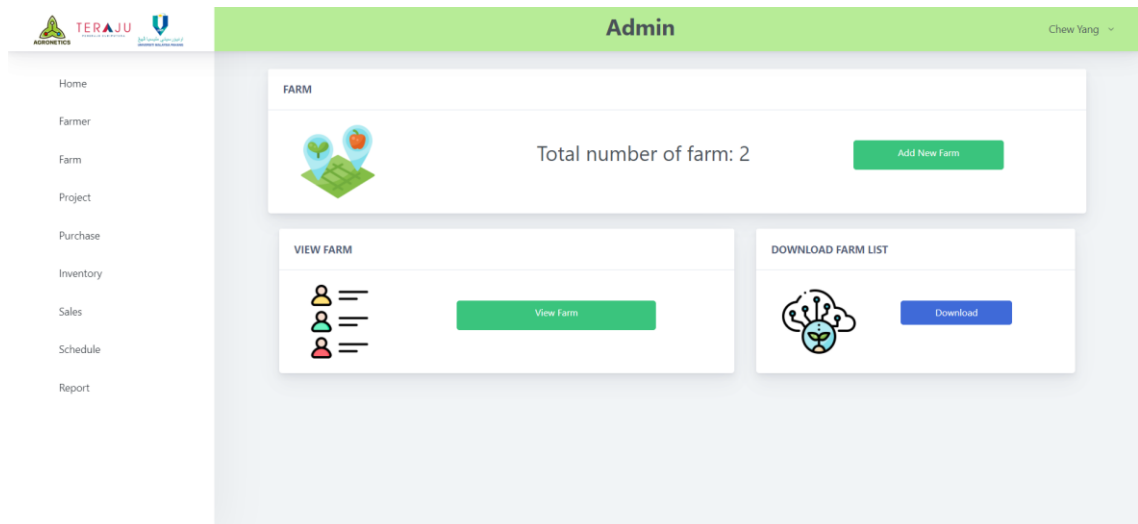


Figure 3.29 Farm

This is the Farm page.

- Admin can click 'Add New Farm' button and system will navigate to Create Farm page.
- Admin can click on 'View Farm' button and system will navigate to View Farm page.
- Admin can click on 'Download' button to download all farm data in excel file.

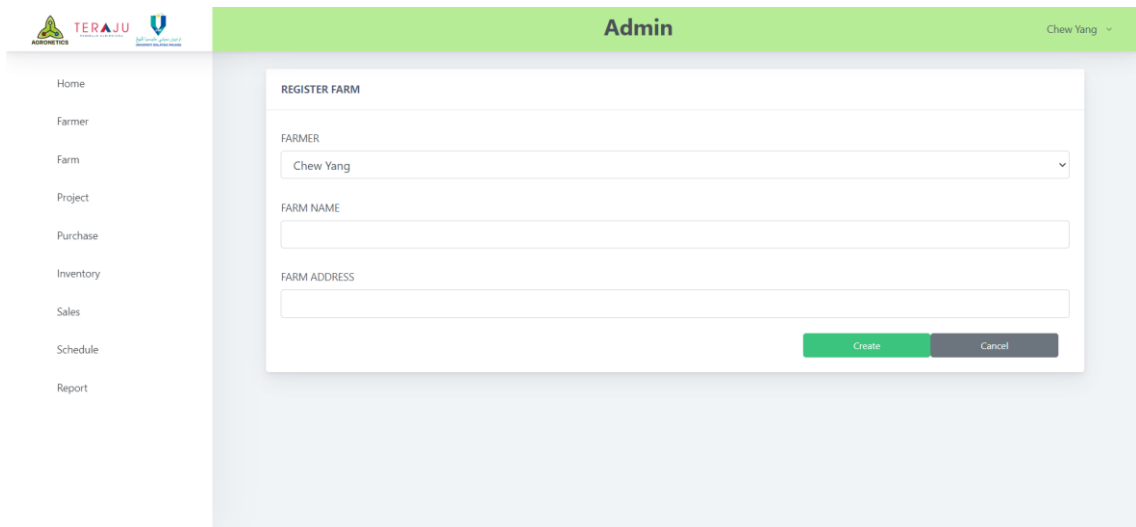


Figure 3.30 Register Farm

This is the Register Farm page.

- Admin can choose the farmer, fill the farm name and farm address.
- Admin can click on 'Create' button and the data is stored.

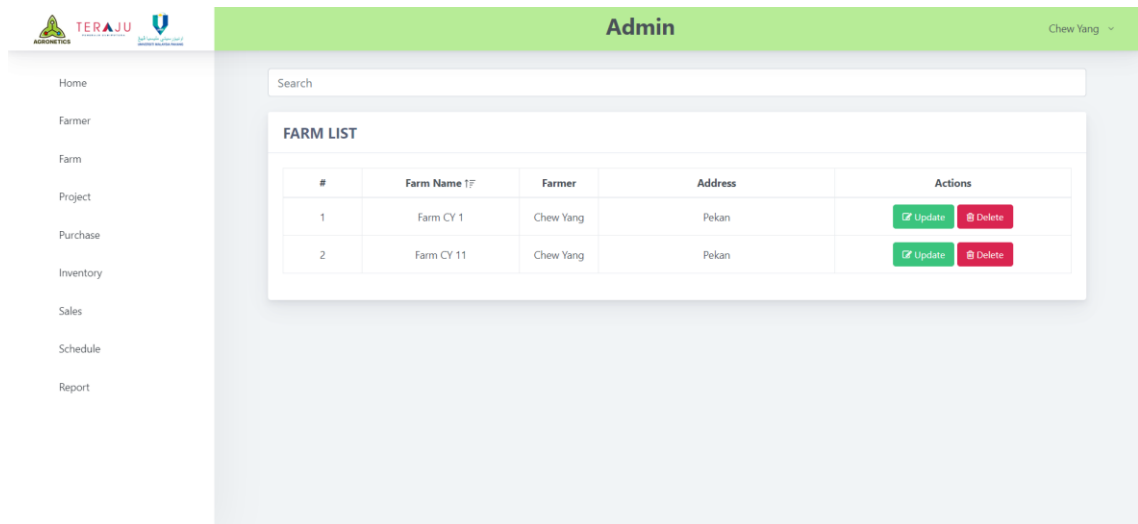


Figure 3.31 View Farm

This is the Register Farm page.

- Admin can view the farm details.
- Admin can click on 'Update' button and the system will navigate to Edit Farm page.
- Admin can click on 'Delete' button to delete the selected farm.
- Admin can search for the farm on the search bar above.



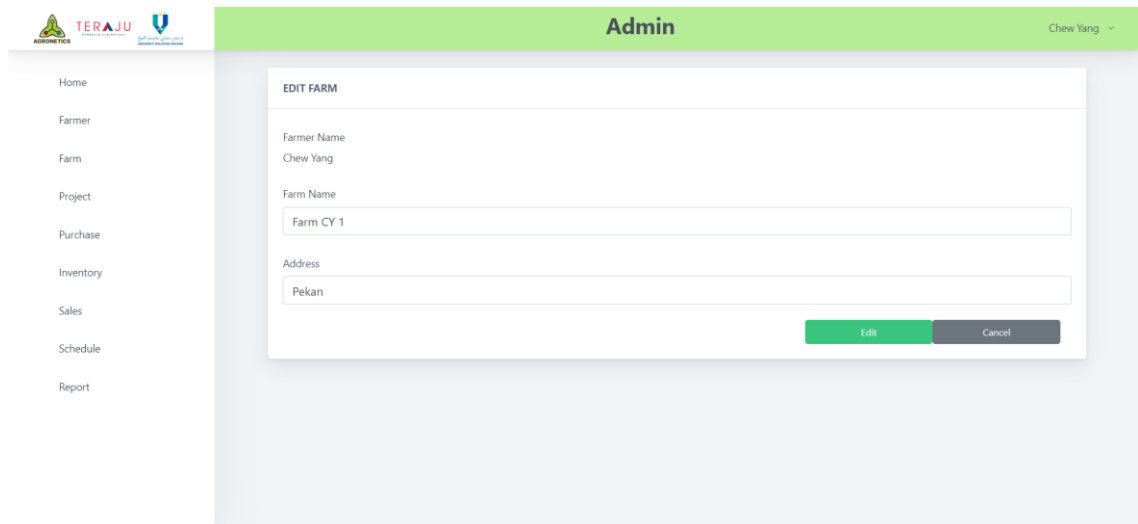


Figure 3.32 Edit Farm

This is the Edit Farm page.

- Admin can edit the farm name and farm address in the input textbox.
- Admin can click on 'Edit' button and the data is updated in database.

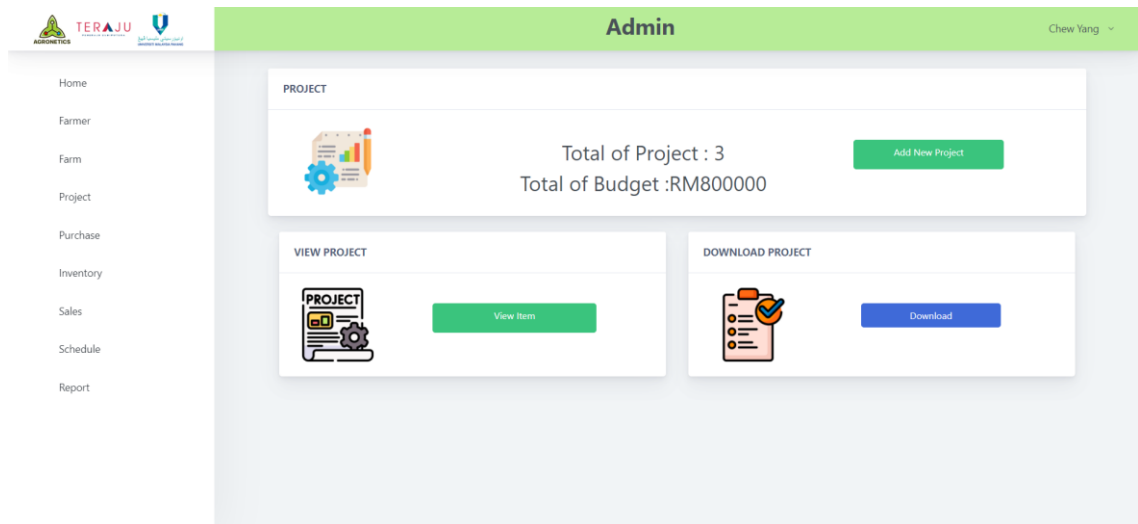


Figure 3.33 Project

This is the Project page.

- Admin can click ‘Add New Project’ button and system will navigate to Create Project page.
- Admin can click on ‘View Project’ button and system will navigate to View Project page.
- Admin can click on ‘Download’ button to download all project data in excel file.

Figure 3.34 Add New Project

This is the Add New Project page.

- Admin can fill for project name, budget and select the correct input for farm, start date, end date and status.
- Admin can click on ‘Add New’ button and system will save the project into database.
- Admin can click on ‘Cancel’ button to go back to the project page.

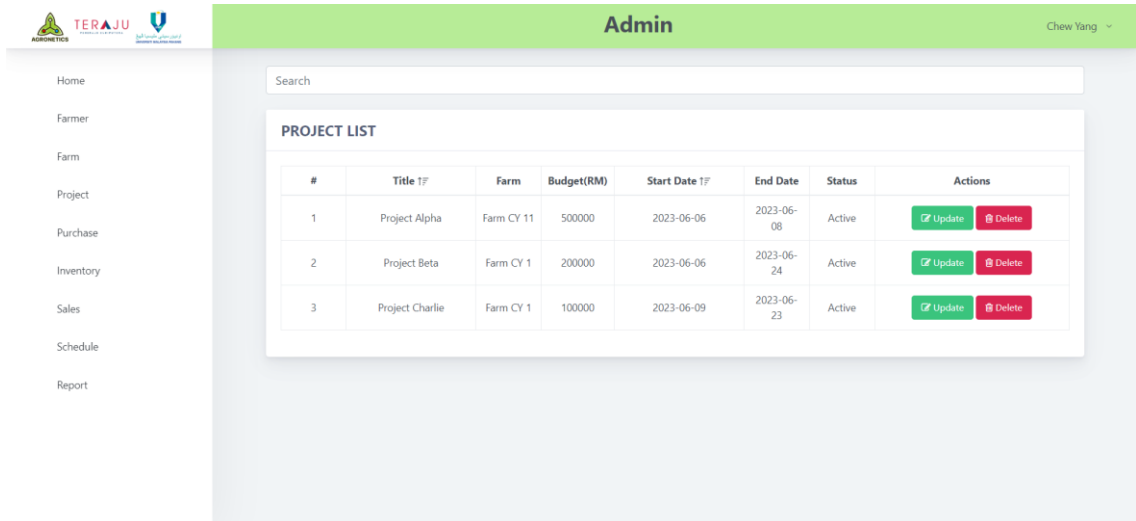


Figure 3.35 View Project

This is the View Project page.

- Admin can view the project details.
- Admin can click on 'Update' button and the system will navigate to Edit Project page.
- Admin can click on 'Delete' button to delete the selected project.
- Admin can search for the project on the search bar above.

The screenshot shows a web application interface for editing a project. At the top, there is a green header with the word 'Admin' and a user profile 'Chew Yang'. On the left, a sidebar menu lists various options: Home, Farmer, Farm, Project, Purchase, Inventory, Sales, Schedule, and Report. The main content area is titled 'EDIT PROJECT' and contains several form fields: 'PROJECT NAME' with the value 'Project Alpha', 'FARM' with a dropdown menu showing 'Farm CY 11', 'BUDGET (RM)' with the value '500000', 'START DATE' with the value '06/06/2023', 'END DATE' with the value '08/06/2023', and 'STATUS' with a dropdown menu showing 'Active'. At the bottom right of the form, there are two buttons: a green 'Edit' button and a grey 'Cancel' button.

Figure 3.36 Edit Project

This is the Edit Project page.

- Admin can edit the project details by changing the display data in input box.
- Admin can click on 'Update' button and the system updated the data from database.
- Admin can click on 'Cancel' button to go back to the Project page.

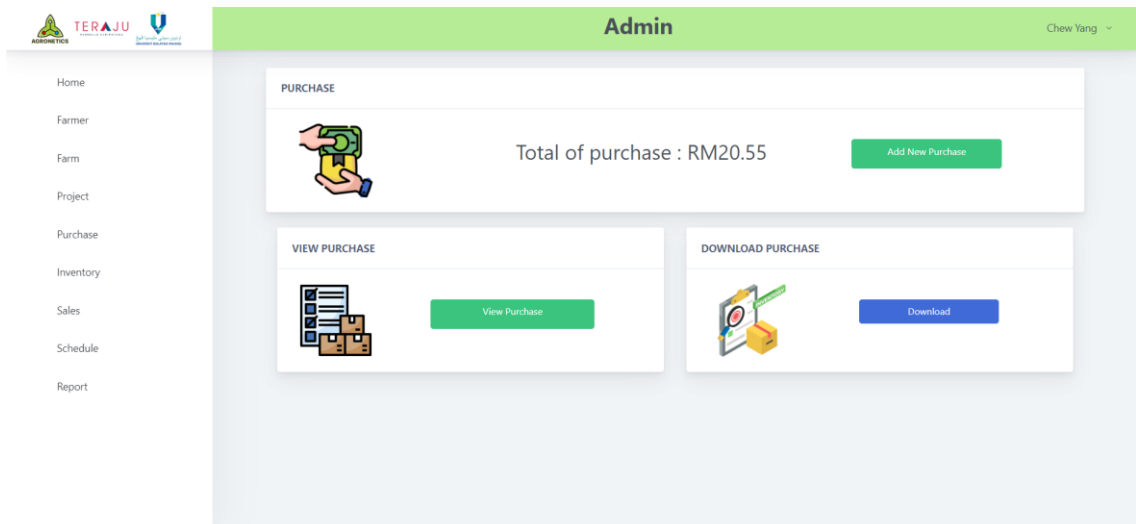


Figure 3.37 Purchase

This is the Purchase page.

- Admin can click 'Add New Purchase' button and system will navigate to Create Purchase page.
- Admin can click on 'View Purchase' button and system will navigate to View Purchase page.
- Admin can click on 'Download' button to download all purchase data in excel file.

The screenshot shows a web application interface for an administrator. At the top, there is a green header bar with the word "Admin" in the center and the user name "Chew Yang" on the right. On the left side, there is a vertical navigation menu with the following items: Home, Farmer, Farm, Project, Purchase, Inventory, Sales, Schedule, and Report. The main content area is titled "ADD NEW PURCHASE" and contains the following form fields:

- PIC Name: A text input field.
- Project: A dropdown menu with "Project Alpha" selected.
- Purchase Title: A text input field.
- PRICE (RM): A text input field.
- DATE: A text input field with a calendar icon on the right.
- UPLOAD RECEIPT: A section with a "Choose File" button and the text "No file chosen".

At the bottom right of the form, there are two buttons: "Add New" (highlighted in green) and "Cancel" (grey).

Figure 3.38 Create Purchase

This is the Create Purchase page.

- Admin can fill for pic name, purchase title, price and select the correct input for project, date and upload the file.
- Admin can click on 'Add New' button and system will save the purchase into database.
- Admin can click on 'Cancel' button to go back to the purchase page.

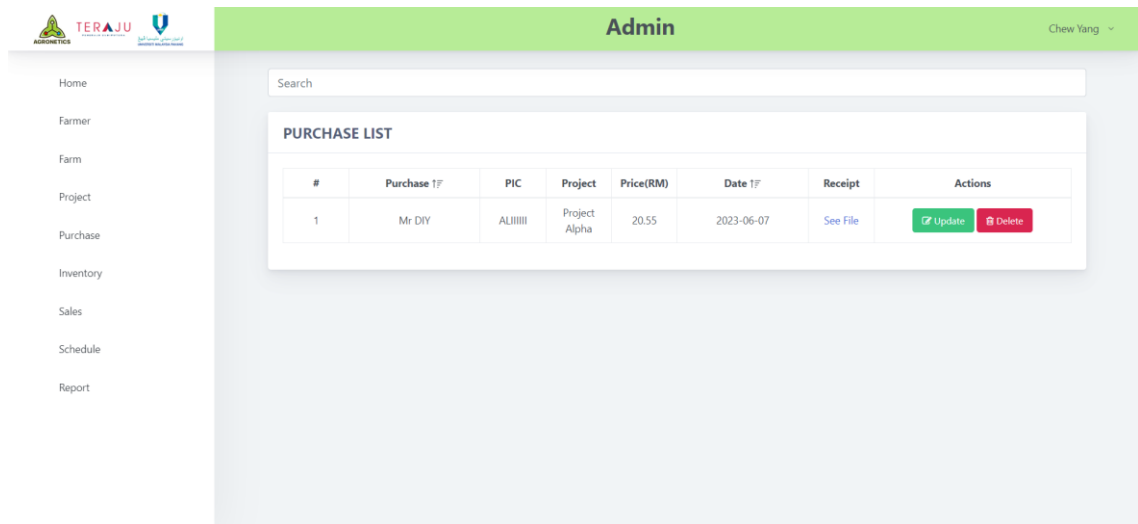


Figure 3.39 View Purchase

This is the View Purchase page.

- Admin can view the purchase details.
- Admin can click on 'Update' button and the system will navigate to Edit Purchase page.
- Admin can click on 'Delete' button to delete the selected purchase.
- Admin can search for the purchase on the search bar above.





Figure 3.40 Edit Purchase

This is the Edit Purchase page.

- Admin can edit the project details by changing the display data in input box.
- Admin can reupload new file to be updated in the system.
- Admin can click on 'Update' button and the system updated the data from database.
- Admin can click on 'Cancel' button to go back to the Purchase page.

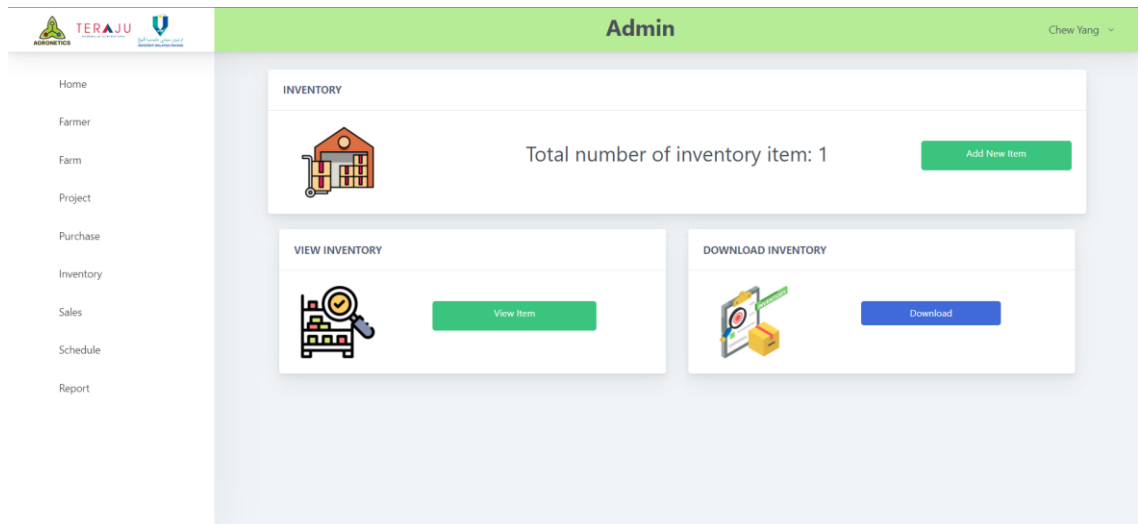


Figure 3.41 Inventory

This is the Inventory page.

- Admin can click ‘Add New Item’ button and system will navigate to Create Inventory page.
- Admin can click on ‘View Item’ button and system will navigate to View Inventory page.
- Admin can click on ‘Download’ button to download all inventory data in excel file.

The screenshot shows a web application interface for an administrator. At the top, there is a green header with the word 'Admin' and a user profile 'Chew Yang'. On the left, a sidebar menu lists various options: Home, Farmer, Farm, Project, Purchase, Inventory (highlighted), Sales, Schedule, and Report. The main content area is titled 'ADD NEW INVENTORY' and contains the following form fields:

- Purchase:** A dropdown menu with 'Mr DIY' selected.
- Name:** A text input field.
- Purchase PIC:** A text input field.
- Quantity:** A text input field.
- PRICE (RM):** A text input field.
- DATE:** A date picker showing 'dd/mm/yyyy'.
- STATUS:** A dropdown menu with 'In Stock' selected.

At the bottom right of the form, there are two buttons: a green 'Add New' button and a grey 'Cancel' button.

Figure 3.42 Create Inventory

This is the Create Inventory page.

- Admin can fill for name, purchase pic, quantity, price and select the correct input for purchase, date and status.
- Admin can click on 'Add New' button and system will save the inventory into database.
- Admin can click on 'Cancel' button to go back to the inventory page.

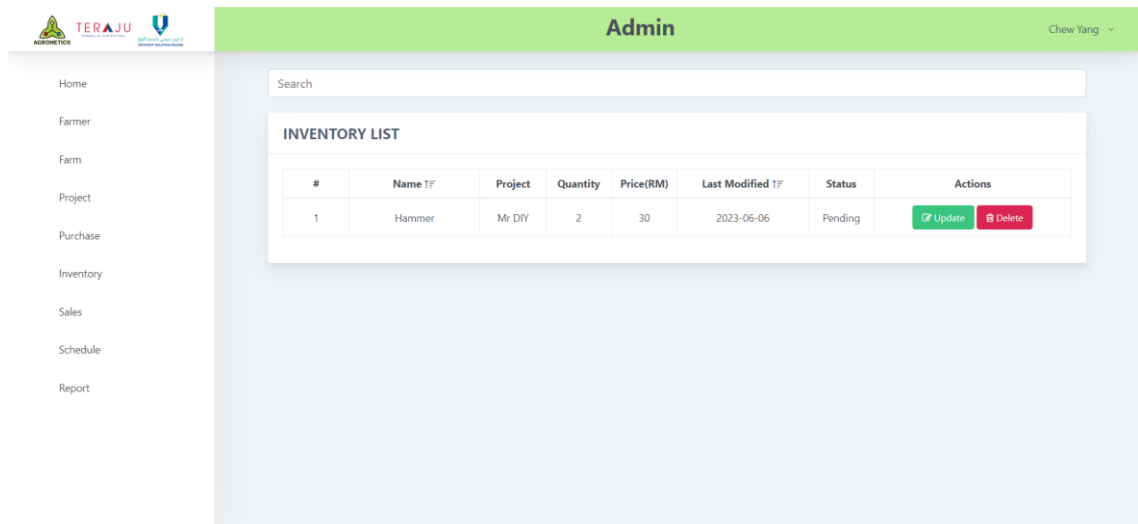


Figure 3.43 View Inventory

This is the View Inventory page.

- Admin can view the inventory item details.
- Admin can click on 'Update' button and the system will navigate to Edit Inventory page.
- Admin can click on 'Delete' button to delete the selected inventory item.
- Admin can search for the inventory item on the search bar above.

The screenshot shows the 'Edit Inventory' page in an admin dashboard. The page has a green header with 'Admin' and a user name 'Chew Yang'. A sidebar on the left lists navigation options: Home, Farmer, Farm, Project, Purchase, Inventory, Sales, Schedule, and Report. The main content area is titled 'EDIT INVENTORY' and contains several input fields: 'Purchase' (dropdown menu with 'Mr DIY'), 'Name' (text box with 'Hammer'), 'Purchase PIC' (text box with 'Ali'), 'Quantity' (text box with '2'), 'PRICE (RM)' (text box with '30'), 'DATE' (calendar icon with '06/06/2023'), and 'STATUS' (dropdown menu with 'Pending'). At the bottom right, there are two buttons: a green 'Edit' button and a grey 'Cancel' button.

Figure 3.44 Edit Inventory

This is the Edit Inventory page.

- Admin can edit the inventory item details by changing the display data in input box.
- Admin can click on 'Update' button and the system updated the data from database.
- Admin can click on 'Cancel' button to go back to the view inventory page.

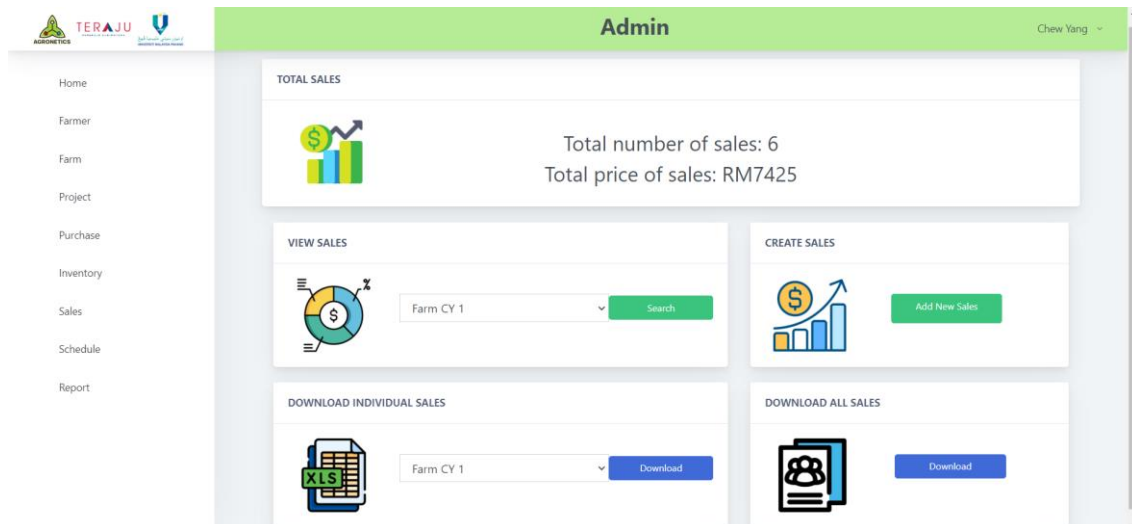


Figure 3.45 Sales

This is the Sales page.

- Admin can click 'Add New Sales' button and system will navigate to Create Sales page.
- Admin can select farm and then click on 'Search' button and system will navigate to View Sales page with the data of selected farm only.
- Admin can click on 'Download' button to download all sales data or individual sales data in excel file.

Figure 3.46 Create Sales

This is the Create Sales page.

- Admin can fill for sales title, crop type, weight, grade, market price, total sales price and date.
- Admin can click on ‘Add New’ button and system will save the sales into database.
- Admin can click on ‘Cancel’ button to go back to the sales page.

The screenshot shows the 'Admin' interface with a sidebar on the left containing navigation options: Home, Farmer, Farm, Project, Purchase, Inventory, Sales, Schedule, and Report. The main content area features a search bar and a table titled 'SALES LIST FOR FARM CY 1'. The table has the following data:

#	Name	Type	Weight(KG)	Price(RM/KG)	TOTAL(RM)	Grade	Date	Actions
1	cili	cili	20	20	400	A	2022-08-07	[Update] [Delete]
2	cili	cili	50	20	1000	A	2023-06-06	[Update] [Delete]
3	eggplant	eggplant	50	25	125	A	2023-03-05	[Update] [Delete]
4	cili	cili	20	20	500	A	2023-05-09	[Update] [Delete]
5	cili	vege	10	20	400	A	2023-06-10	[Update] [Delete]

Figure 3.47 View Sales

This is the View Sales page.

- Admin can view the sales details.
- Admin can click on 'Update' button and the system will navigate to Edit Sales page.
- Admin can click on 'Delete' button to delete the selected sales item.
- Admin can search for the sales on the search bar above.



The screenshot shows a web application interface for an administrator. At the top, there is a green header bar with the word "Admin" and a user profile "Chew Yang". On the left, a sidebar menu lists various options: Home, Farmer, Farm, Project, Purchase, Inventory, Sales, Schedule, and Report. The main content area is titled "EDIT SALES" and contains several input fields with pre-filled values: "FARMER" (Farm CY 1), "SALES TITLE (PLANT)" (cili), "CROP TYPE" (cili), "WEIGHT (KG)" (20), "GRADE" (A), "MARKET PRICE (RM/KG)" (20), "TOTAL PRICE (RM)" (400), and "DATE" (07/08/2022). At the bottom right of the form, there are two buttons: a green "Edit" button and a grey "Cancel" button.

Figure 3.48 Edit Sales

This is the Edit Sales page.

- Admin can edit the sales details by changing the display data in input box.
- Admin can click on 'Update' button and the system updated the data from database.
- Admin can click on 'Cancel' button to go back to the view sales page.

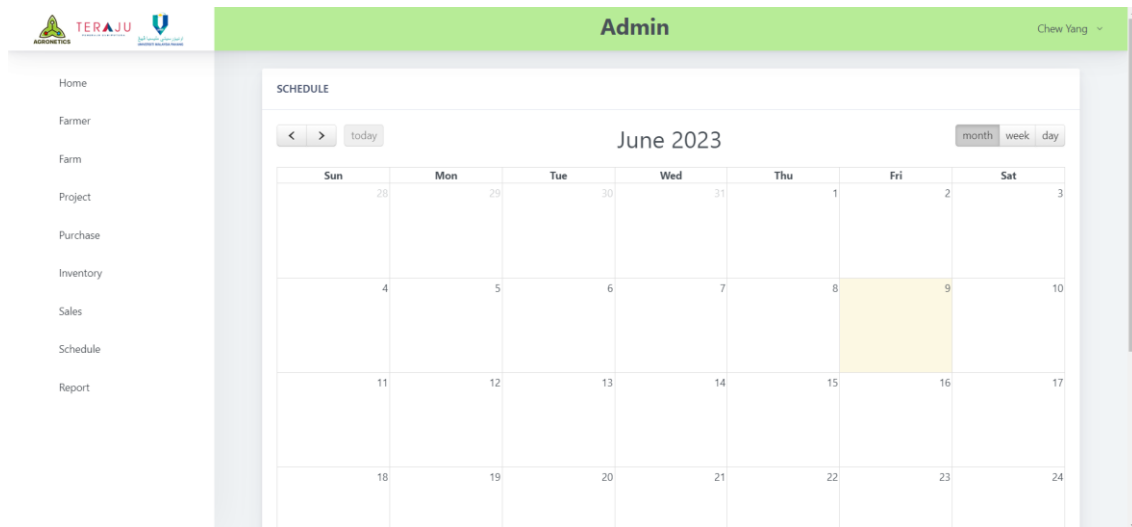


Figure 3.49 Schedule

This is the Schedule page.

- Admin can add event into the calendar by click on the date.
- Admin can drag and drop the event freely in the calendar and the data is updated in database.
- Admin can click on the event to delete the event.
- Admin can select the view mode on month, week and day.
- Admin can change the month of the calendar freely.

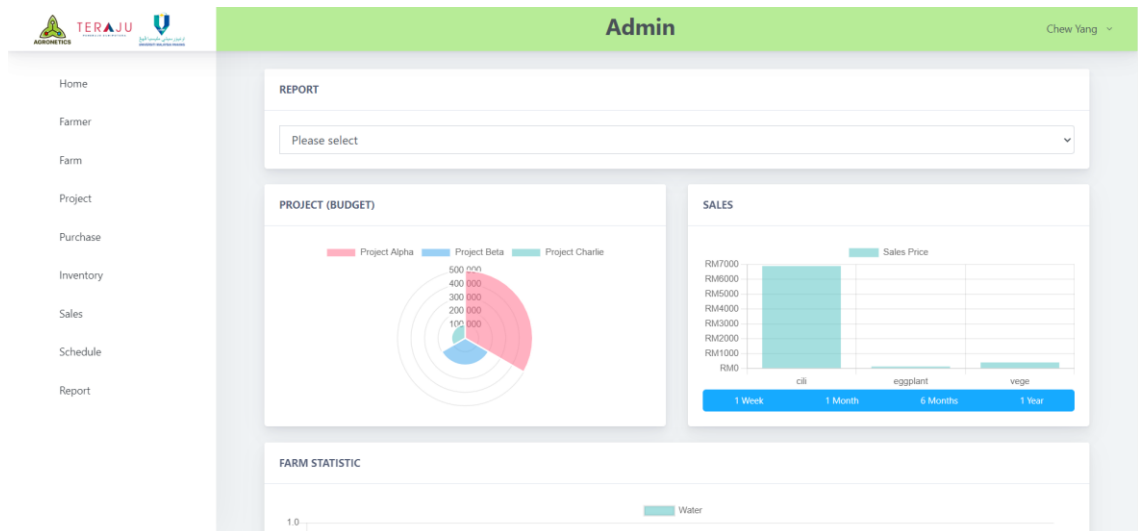


Figure 3.50 Report

This is the Report page.

- Admin can view the default all data graph.
- Admin can select the farm at above to see individual graph.
- Admin can select timeframe at the graph to show specific graph.

### 3.5 Data Design

#### 3.5.1 ERD

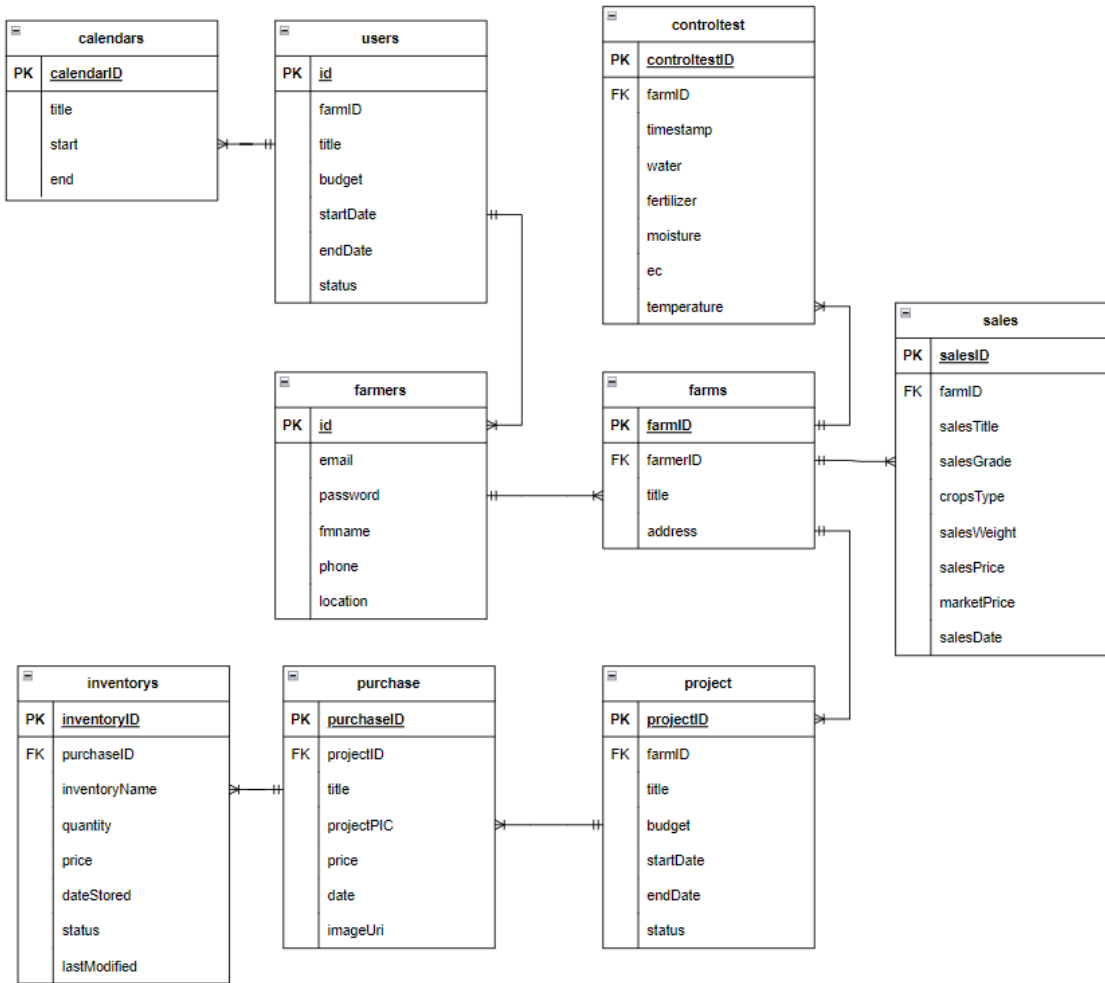


Figure 3.51 ERD

### 3.5.2 Data Dictionary

#### 3.5.2.1 calendars

Table 3.10 Data Dictionary of calendars

Field Name	Description	Data Type	Constraint
calendarID	The ID number for calendar.	bigint(20)	PK, Auto increment
title	The title for calendar event.	varchar(255)	Not Null
start	The start date of calendar event.	datetime	Not Null
end	The end date of calendar event.	datetime	Not Null

#### 3.5.2.2 controltest

Table 3.11 Data Dictionary of controltest

Field Name	Description	Data Type	Constraint
controltestID	The ID number for farm data.	bigint(20)	PK, Auto increment
farmID	The ID of farm.	varchar(255)	FK, Not Null
timestamp	The timestamp of the data.	int(11)	Not Null
water	The water data of farm.	varchar(255)	Not Null

fertilizer	The fertilizer data of farm.	VARCHAR(255)	Not Null
moisture	The moisture data of farm.	int(11)	Not Null
ec	The ec data of farm.	int(11)	Not Null
temperature	The temperature data of farm.	double	Not Null

### 3.5.2.3 farms

Table 3.12 Data Dictionary of farms

Field Name	Description	Data Type	Constraint
farmID	The ID for farm.	bigint(20)	PK, Not Null
farmerID	The ID for farmer.	varchar(255)	FK, Not Null
title	The title for farm.	varchar(255)	Not Null
address	The address for farm.	varchar(255)	Not Null

### 3.5.2.4 farmers

Table 3.13 Data Dictionary of farmers

Field Name	Description	Data Type	Constraint
id	The ID for farmer.	bigint(20)	PK, Not Null

email	The email for farmer.	varchar(255)	Not Null
password	The password for farmer.	varchar(255)	Not Null
fmname	The name for farmer.	varchar(255)	Not Null
phone	The phone number for farmer.	varchar(255)	Not Null
location	The location for farmer.	varchar(255)	Not Null

### 3.5.2.5 inventories

Table 3.14 Data Dictionary of inventories

Field Name	Description	Data Type	Constraint
inventoryID	The ID for inventory.	bigint(20)	PK, Not Null
purchaseID	The ID for purchase.	int(110)	FK, Not Null
inventoryName	The name for inventory.	varchar(255)	Not Null
inventoryPIC	The person in charge for inventory.	varchar(255)	Not Null
quantity	The quantity for inventory.	int(11)	Not Null

price	The price for inventory.	double	Not Null
dateStored	The date for storing inventory.	date	Not Null
status	The status for inventory.	varchar(255)	Not Null
lastModified	The last modified date for inventory.	date	Not Null

### 3.5.2.6 project

Table 3.15 Data Dictionary of project

Field Name	Description	Data Type	Constraint
projectID	The ID for project	bigint(20)	PK, Not Null
farmID	The ID for farm	int(11)	FK, Not Null
title	The title for project	varchar(255)	Not Null
budget	The budget for project	double	Not Null
startDate	The start date for project	date	Not Null
endDate	The end date for project	date	Not Null



status	The status for project	tinyint(1)	Not Null
--------	------------------------	------------	----------

### 3.5.2.7 purchase

Table 3.16 Data Dictionary of purchase

Field Name	Description	Data Type	Constraint
purchaseID	The ID for purchase	bigint(20)	PK, Not Null
projectID	The ID for project	int(11)	FK, Not Null
title	The title for purchase	varchar(255)	Not Null
projectPIC	The person in charge for purchase	varchar(255)	Not Null
price	The price for purchase	double	Not Null
date	The date for purchase	date	Not Null
imageUri	The file name for purchase	varchar(255)	Not Null

### 3.5.2.8 sales

Table 3.17 Data Dictionary of sales

Field Name	Description	Data Type	Constraint
------------	-------------	-----------	------------

salesID	The ID for sales.	bigint(20)	PK, Not Null
farmID	The ID for farm.	int(11)	FK, Not Null
salesTitle	The title for sales.	varchar(50)	Not Null
salesGrade	The grade for sales.	varchar(5)	Not Null
cropsType	The type of crop for sales.	varchar(15)	Not Null
salesWeight	The weight for sales.	double(7,2)	Not Null
salesPrice	The price for sales.	double(8,2)	Not Null
marketPrice	The market price for sales.	double(8,2)	Not Null
salesDate	The date for sales.	date	Not Null

### 3.5.2.9 users

Table 3.18 Data Dictionary of users

Field Name	Description	Data Type	Constraint
id	The ID for user.	bigint(20)	PK, Not Null
name	The name for user.	varchar(255)	Not Null
email	The email for user.	varchar(255)	Not Null
email_verified_at	The time verified for user.	timestamp	Nullable

password	The password for user.	varchar(255)	Not Null
option	The option for user.	varchar(255)	Not Null
remember_token	The token for user.	varchar(100)	Nullable
created_at	The timestamp for user.	timestamp	Nullable
updated_at	The timestamp for user.	timestamp	Nullable

### 3.6 Proof of Initial Concept



Figure 3.52 Sample IoT concept for Automated Fertigation System

The picture above is showing the project that involve UMP and Agronetic to develop an IoT system which named as Automation Fertigation System.



Figure 3.53 Sample IoT concept for Automated Fertigation System

This is the demo prototype that is done developing and it is usable. From the right picture, we can see that the water and fertilizer will be send to each plant through the silicon pipe and there are sensors in the soil to check the temperature and moisture level.

### 3.7 Testing / Validation Plan

The testing plan that would be use in this project is User Acceptance test. There will be a User Acceptance Test carry out together with the client and make sure the testing process is satisfying and only will deliver the system to the client once they agreed all the developed function in the system. The example User Acceptance form is shown as below Figure 3.41.

## USER ACCEPTANCE TEST

### Automated Fertigation System

No	Acceptance Requirement	Test Result		Comment
<b>Test for Admin</b>				
	<b>Login</b>			
1.	Login	Yes	No	
2.	Forgot Password	Yes	No	
3.	Logout	Yes	No	
	<b>Home</b>			
1.	Select farm	Yes	No	
2.	Display farm data	Yes	No	
	<b>Farmer</b>			
1.	Add New Farmer	Yes	No	
2.	View Farmer	Yes	No	
3.	Update Farmer	Yes	No	
4.	Delete Farmer	Yes	No	
5.	Download Farmer data			
	<b>Farm</b>			
1.	Add New Farm	Yes	No	
2.	View Farm	Yes	No	
3.	Update Farm	Yes	No	
4.	Delete Farm	Yes	No	
5.	Download Farm Data	Yes	No	
	<b>Project</b>			
1.	Add New Project	Yes	No	
2.	View Project	Yes	No	
3.	Update Project	Yes	No	
4.	Delete Project	Yes	No	
5.	Download Project Data	Yes	No	
	<b>Purchase</b>			
1.	Add New Purchase	Yes	No	
2.	View Purchase	Yes	No	
3.	Update Purchase	Yes	No	
4.	Delete Purchase	Yes	No	
5.	Download Purchase Data	Yes	No	
6.	View uploaded file	Yes	No	
	<b>Inventory</b>			
1.	Add New Inventory	Yes	No	
2.	View Inventory	Yes	No	
3.	Update Inventory	Yes	No	
4.	Delete Inventory	Yes	No	
5.	Download Inventory Data	Yes	No	

<b>Sales</b>			
1.	Add New Sales	Yes	No
2.	Select User and View Sales	Yes	No
3.	Update Sales	Yes	No
4.	Delete Sales	Yes	No
5.	Download All Sales Data	Yes	No
6.	Download Individual Sales Data	Yes	No
<b>Schedule</b>			
1.	Add New Event	Yes	No
2.	View Event	Yes	No
2.	Update Event	Yes	No
3.	Delete Event	Yes	No
<b>Report</b>			
1.	View All User Graph	Yes	No
2.	View User Graph	Yes	No

Comment (Improve/ Design/ Bugs):

Name:

Date:

I, \_\_\_\_\_ had conducted the User Acceptance Test as requested. I admit that the information that filled is my true personal opinion.

\_\_\_\_\_  
Name:

Figure 3.54 Sample User Acceptance Test Form

### 3.8 Potential Use of Proposed Solution

The potential use of proposed solution will be benefited to the admin who is managing the farm and transform their management into the smart farm management. The potential people would be the admin from UMP. We will provide all the training services such as provide training until the admin or person in charge are familiar with the system.

### 3.9 Gantt Chart

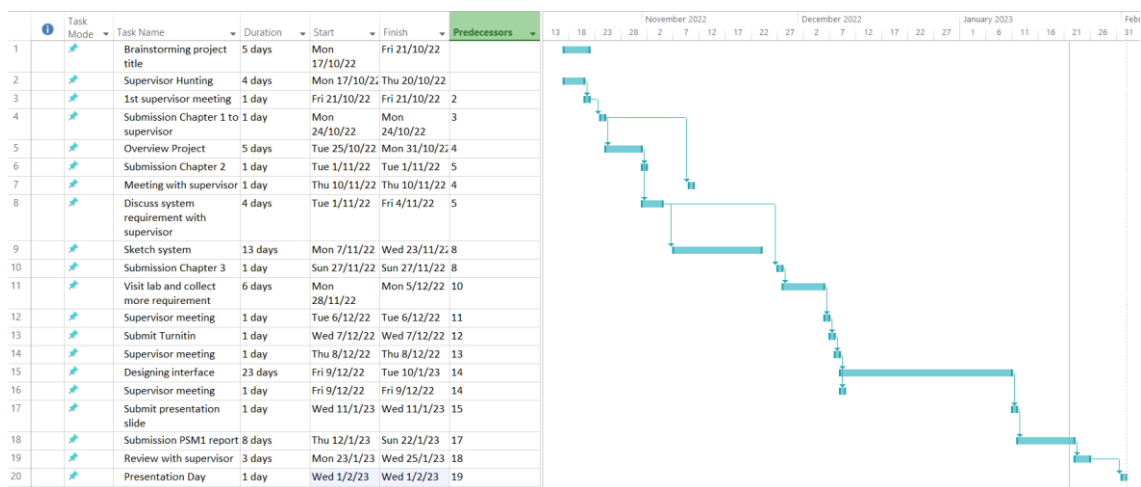


Figure 3.55 Gantt Chart for PSM1

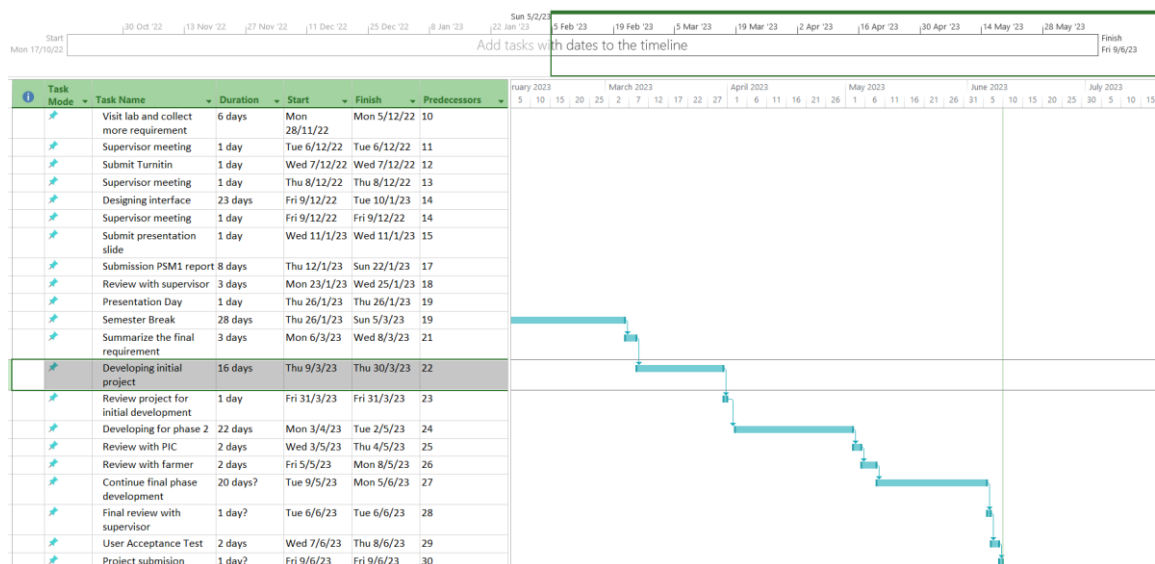


Figure 3.56 Gantt Chart for PSM2

## CHAPTER 4

### 4.1 Introduction

In this chapter, there are several things discussed such as implementation process, testing and result discussion. The implementation process means that there will involve the initial proposed solution and methodology into action. Furthermore, there will be explanations on the programming languages, database and software used for the project development. Throughout the chapter, the stages of implementation are explained such as data collection, preprocessing, model development and deployment. The challenges and difficulties are addressed and the way to overcome these challenges are well explained. Moreover, the suitable testing method is chosen to test out the system to make sure that all the requirements are matched and agreed by the stakeholders.

### 4.2 Implementation process

Before the project starting I have implemented the brainstorming session with the stakeholders and to have a discussion together and collect the main requirements from them. After that, we will have the discussion on the design and sketch the interface for the project. Next, the prototype is produced and present it to the stakeholders. A review session together with the stakeholders is carried out to identify out the things that need to be modified. After the prototype phase has been agreed by the stakeholder, the development phase is started immediately. The interface and coding for CRUD function in the project is shown below.



REGISTER FARMER

---

Name

Email

Password

Phone

Location

Figure 4.1 Register Farmer

```
public function store(Request $request)
{
    $input = new farmers();
    //Request input from POST
    $input->email = $request->farmerEmail;
    $input->password = $request->farmerPassword;
    $input->fmname = $request->farmerName;
    $input->phone = $request->farmerPhone;
    $input->location = $request->farmerLocation;

    $input->save();
    return redirect('farmer')->with('success', 'Farmer Registered Successfully!');;
}
```

Figure 4.2 Register Farmer Code

This is the register farmer function that is under farmer module. The function will use the request function to request for the input that is submitted from the frontend. Then all the submitted data will be store into the model and then redirected back to the module default page.

EDIT FARMER

---

Name

Email

PASSWORD

PHONE

LOCATION

Figure 4.3 Edit Farmer

```
public function edit($id)
{
    //POST farmerID to edit page
    $farmerData = farmers::find($id);
    return view('ManageFarmer.editFarmer',compact('farmerData'));
}

public function update(Request $request, $id)
{
    //Request and post new data to update in db
    $farmerData = farmers::find($id);
    $farmerData->fmname = $request->input('fmname');
    $farmerData->email = $request->input('email');
    $farmerData->password = $request->input('password');
    $farmerData->phone = $request->input('phone');
    $farmerData->location = $request->input('location');

    $farmerData->save();

    return redirect('/farmer')->with('success', 'Farmer Edited Successfully!');
}
```

Figure 4.4 Edit Farmer Code

This is the edit farmer function that is under farmer module. The function will get the selected farmer id and then redirect to the edit page and showing the selected data to user. Then it uses the request function to request for the input that is submitted from the frontend and then all the submitted edit data will be update into the model and then redirected back to the module default page.

## FARMER LIST

#	Name 17	Email	Phone	Location 17	Actions
1	Chew Yang	chewyang00@gmail.com	0128628508	Pekan Pahang	<a href="#">Update</a> <a href="#">Delete</a>

Figure 4.5 View & Delete Farmer

```
public function show()
{
    //
    $farmerData = farmers::all();
    return view('ManageFarmer.viewFarmer')->with('farmerData', $farmerData);
}
```

Figure 4.6 View Farmer Code

```
public function destroy($farmerID)
{
    //Delete farmer by id
    farmers::destroy($farmerID);
    return redirect('farmer')->with('success', 'Farmer Deleted Successfully!');
}
```

Figure 4.7 Delete Farmer Code

This is the view and delete farmer function that is under farmer module. The view function will get all the data and show the data at the interface. The delete function will get the selected id and then will delete the data from the database and will redirected back to the module default page.

## DOWNLOAD FARMER LIST



Download

Figure 4.8 Download Farmer List

```
public function exportFarmer()
{
    $farmerData = farmers::all();

    $headers = [
        "Content-type" => "text/csv",
        "Content-Disposition" => "attachment; filename=All_Farmers.csv",
        "Pragma" => "no-cache",
        "Cache-Control" => "must-revalidate, post-check=0, pre-check=0",
        "Expires" => "0",
    ];

    $columns = ['Farmer ID', 'Name', 'Email', 'Phone', 'Location'];

    $callback = function() use ($farmerData, $columns) {
        $file = fopen('php://output', 'w');
        fputcsv($file, $columns);

        foreach ($farmerData as $farmerList) {
            $row['Farmer ID'] = $farmerList->id;
            $row['Name'] = $farmerList->fmname;
            $row['Email'] = $farmerList->email;
            $row['Phone'] = $farmerList->phone;
            $row['Location'] = $farmerList->location;

            fputcsv($file, array($row['Farmer ID'], $row['Name'], $row['Email'], $row['Phone'], $row['Location']));
        }

        fclose($file);
    };

    return Response::stream($callback, 200, $headers);
}
```

Figure 4.9 Download Farmer List Code

This is the download function that is under farmer module. The download function will get all the data and print the data in excel file and deliver to the user.

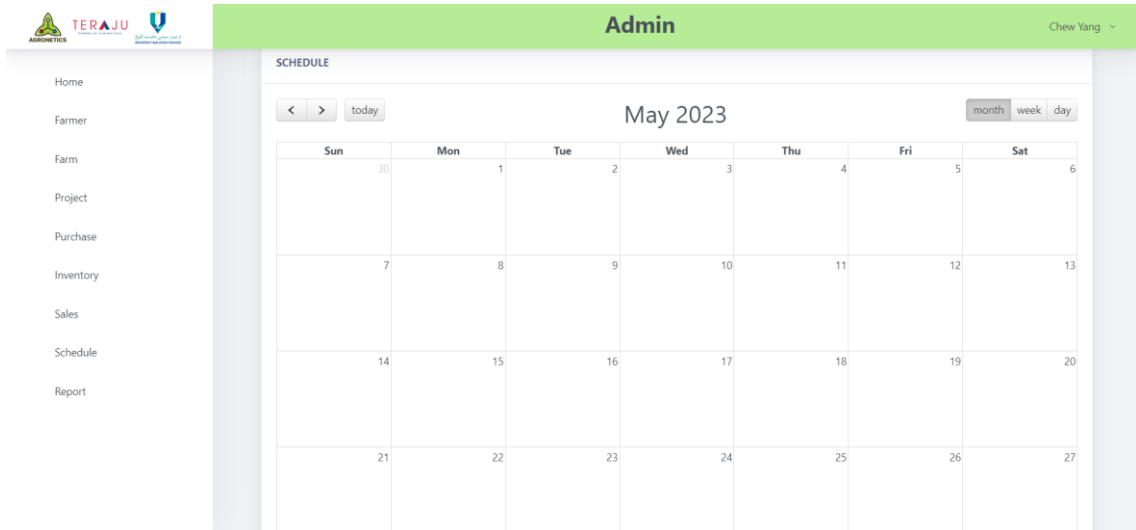


Figure 4.10 Schedule

```
@extends('layouts.adminNav')
@section('content')
<head>
<meta charset="UTF-8">
<meta name="csrf-token" content="{{ csrf_token() }}">
<title>FullCalendar Example</title>

<!-- jQuery library -->
<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

<!-- FullCalendar CSS -->
<link href="https://cdnjs.cloudflare.com/ajax/libs/fullcalendar/3.10.2/fullcalendar.min.css" rel="stylesheet">

<!-- FullCalendar JavaScript -->
<script src="https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.29.1/moment.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/fullcalendar/3.10.2/fullcalendar.min.js"></script>
</head>

<link rel="stylesheet" href="{{ asset('css/mycss.css') }}">
<div class="container">
  <div class="row justify-content-center">
    <div class="col-md-12">
      <div class="card">
        <div class="card-header">{{ __('Schedule') }}</div>

        <div class="card-body">
          <div id="calendars"></div>
        </div>
      </div>
    </div>
  </div>
</div>
</div><br>
```

```

<script>
$(document).ready(function() {
  var calendar = $('#calendars').fullCalendar({
    // Your FullCalendar configuration options
    defaultView: 'month',
    //... other options

    // Header buttons
    header: {
      left: 'prev,next today',
      center: 'title',
      right: 'month,agendaWeek,agendaDay'
    },

    // Event sources
    eventSources: [
      {
        url: '/calendar', // Make sure this matches your route for fetching events
        method: 'GET',
        failure: function() {
          alert('Error fetching events from the server!');
        }
      }
    ]
  },

```

```

// Create event function
selectable: true,
select: function(start, end) {
  var title = prompt('Event Title:');
  if (title) {
    // Send the event data to the server for saving
    $.ajax({
      url: '/calendar',
      data: {
        title: title,
        start: start.format(),
        end: end.format()
      },
      method: 'POST',
      headers: {
        'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')
      },
      success: function(data) {
        // Refresh the calendar to display the new event
        calendar.fullCalendar('refetchEvents');
      },
      error: function() {
        alert('Error saving the event!');
      }
    });
  }
  calendar.fullCalendar('unselect');
},

```

```

// Edit event function
editable: true,
eventDrop: function(event) {
    // Send the updated event data to the server for editing
    $.ajax({
        url: '/calendar/' + event.calendarID, // Access the event ID using event.event.id
        data: {
            start: event.start.format(),
            end: event.end.format()
        },
        method: 'PUT',
        headers: {
            'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')
        },
        success: function(data) {
            alert('Event updated successfully!');
        },
        error: function() {
            alert('Error updating the event!');
        }
    });
},

```

```

// Delete event function
eventClick: function(event) {
    if (confirm("Are you sure you want to delete this event?")) {
        // Send the event ID to the server for deletion
        $.ajax({
            url: '/calendar/' + event.calendarID,
            method: 'DELETE',
            headers: {
                'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')
            },
            success: function(data) {
                // Refresh the calendar to remove the deleted event
                alert('Event deleted successfully!');
                calendar.fullCalendar('refetchEvents');
            },
            error: function() {
                alert('Error deleting the event!');
            }
        });
    }
}
});
});

```

Figure 4.11 Schedule Code

This is the calendar function that is under schedule module. The calendar function will be able to perform create, view, update and delete function. There is fullCalendar library implemented in this module through javascript to perform the calendar function.

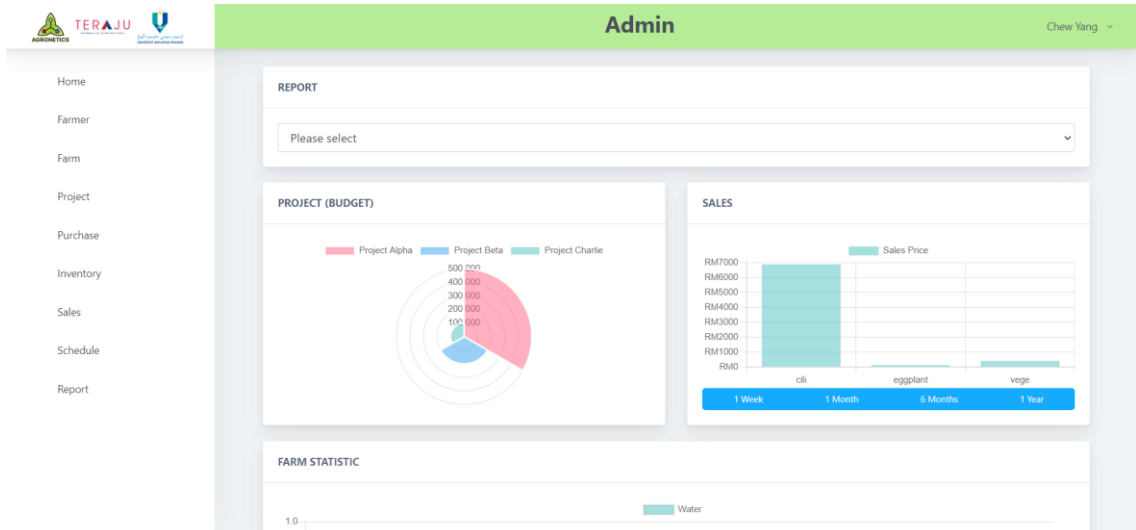


Figure 4.12 Report Code

```

@if ($projectBudget && $projectLabels && $farmData)
    var budgetData = {!! json_encode($projectBudget) !!};
    var projectLabels = {!! json_encode($projectLabels) !!};

    new Chart(document.getElementById("projectChart"), {
        type: "polarArea",
        data: {
            labels: projectLabels,
            datasets: [
                {
                    data: budgetData,
                    backgroundColor: [
                        "rgba(255, 99, 132, 0.5)",
                        "rgba(54, 162, 235, 0.5)",
                        "rgba(75, 192, 192, 0.5)"
                    ]
                }
            ]
        },
        options: {
            responsive: true,
            aspectRatio: 1.3,
            maintainAspectRatio: false,
            width: 100, // Adjust the width as desired
            height: 100 // Adjust the height as desired
        }
    });
@endif

```

Figure 4.13 Report Chart Code




This is the report module that allow the user to see the graph of all data or individual data. There are polar chart, bar chart and line chart in this module. With the help of Chart JS library, we are able to visualize the data easily.

## 4.3 User Manual

### 4.3.1 Login Function

Automated\_Fertigation\_System



**AGRONETICS**

Login

Email Address

Password

Remember Me

[Forgot Your Password?](#)

Figure 4.14 Login

For login, Admin need to input the email in the 'Email Address' column and password in the 'Password' column. After done input, Admin need to press 'Login' button to perform the login action.

### 4.3.2 Home module

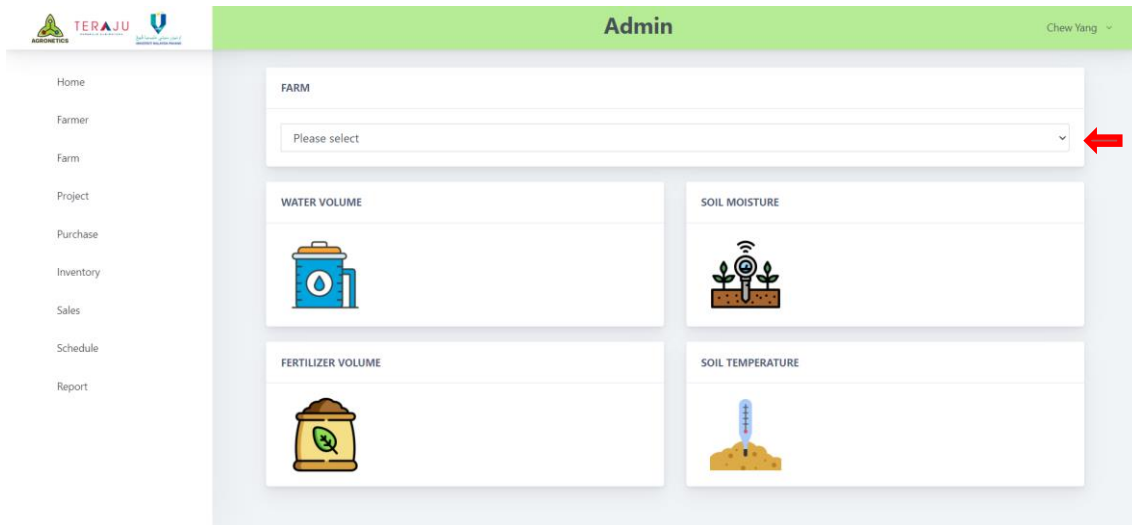


Figure 4.15 Home

In home module, Admin can select the farm by clicking on the select box to choose the farm. After the farm is chosen, the farm status will display at below.

### 4.3.3 Farmer Module

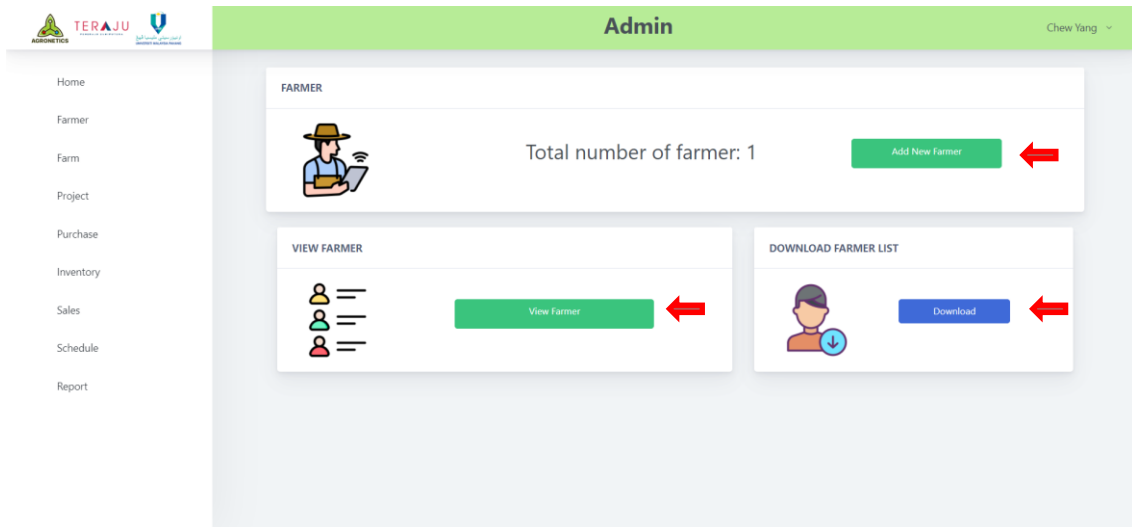


Figure 4.16 Farmer

In Farmer module, Admin can press the 'Add New Farmer' and system will redirect to create farmer page. Next, Admin can press the 'View Farmer' button and system will redirect to view farmer page. Then, Admin can press the 'Download' button and system will auto download all farmer information in a excel file.

### 4.3.4 Register Farmer module

The screenshot shows the 'REGISTER FARMER' form within an 'Admin' interface. The form has a title 'REGISTER FARMER' and five input fields: 'Name', 'Email', 'Password', 'Phone', and 'Location'. At the bottom right of the form are two buttons: a green 'Register' button and a grey 'Cancel' button. Red arrows point to each of the five input fields and the 'Register' button. The interface also features a left sidebar with navigation options (Home, Farmer, Farm, Project, Purchase, Inventory, Sales, Schedule, Report) and a top header with the 'Admin' title and a user profile 'Chew Yang'.

Figure 4.17 Create Farmer

In create farmer module, Admin need to input 'Name', 'Email', 'Password', 'Phone' and 'Location' in order to create new farmer. Next, Admin need to click 'Register' button to register new farmer.

### 4.3.5 View Farmer module

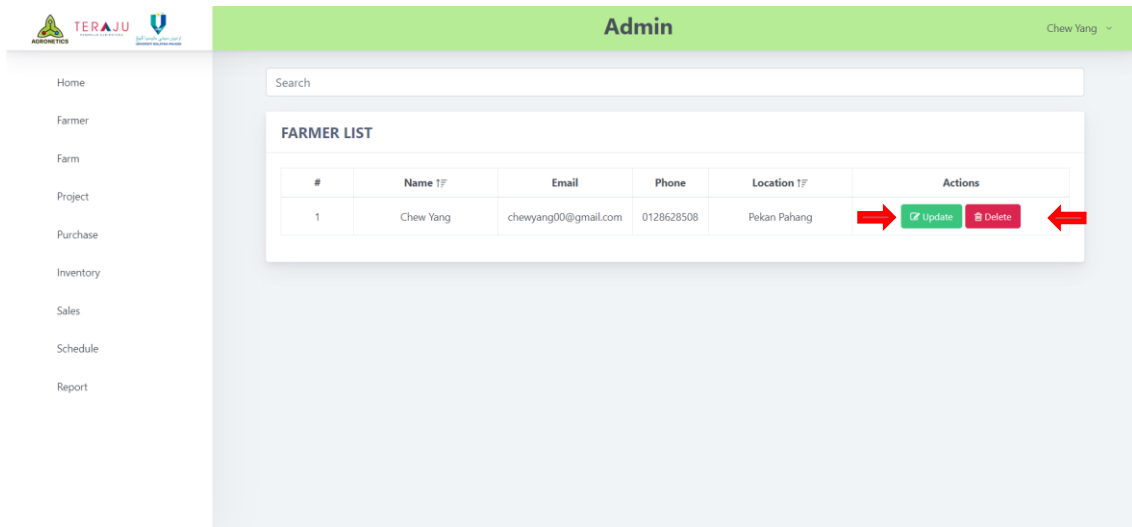


Figure 4.18 View Farmer

This is the View Farmer page. Admin can see all the farmer details in this page. Next, Admin can click 'Update' button to update the farmers' details. Then, Admin can click 'Delete' button to delete the farmer. Lastly, Admin can search for farmer by their name using the search bar above.

### 4.3.6 Edit Farmer module

The screenshot displays the 'Edit Farmer' module within an Admin interface. The interface includes a sidebar with navigation options: Home, Farmer, Farm, Project, Purchase, Inventory, Sales, Schedule, and Report. The main content area is titled 'Admin' and shows the user 'Chew Yang'. The 'EDIT FARMER' form contains the following fields and values:

Field	Value
Name	Chew Yang
Email	chewyang00@gmail.com
PASSWORD	abc1233
PHONE	0128628508
LOCATION	Pekan Pahang

Red arrows indicate the focus on each input field and the 'Edit' button.

Figure 4.19 Edit Farmer

In edit farmer module, Admin can modify 'Name', 'Email', 'Password', 'Phone' and 'Location' from the displayed old data. Next, Admin can click 'Edit' button to update farmer data.

### 4.3.7 Farm module

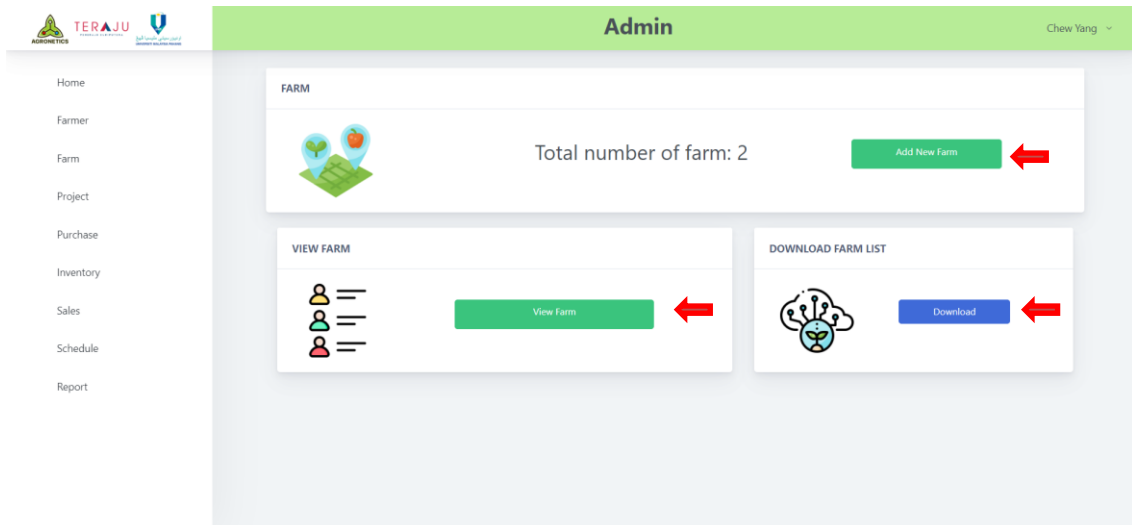


Figure 4.20 Farm

This is the Farm page. Admin can click 'Add New Farm' button and system will navigate to Create Farm page. Next, Admin can click on 'View Farm' button and system will navigate to View Farm page. Lastly, Admin can click on 'Download' button to download all farm data in excel file.



### 4.3.8 Register Farm module

The screenshot shows the 'REGISTER FARM' form within the 'Admin' interface. The form has three input fields: 'FARMER' (a dropdown menu with 'Chew Yang' selected), 'FARM NAME' (a text input field), and 'FARM ADDRESS' (a text input field). At the bottom right of the form are two buttons: 'Create' (green) and 'Cancel' (grey). Red arrows point to the 'FARMER' dropdown, the 'FARM NAME' field, the 'FARM ADDRESS' field, and the 'Create' button. The left sidebar contains a navigation menu with items: Home, Farmer, Farm, Project, Purchase, Inventory, Sales, Schedule, and Report. The top header shows 'Admin' and a user profile 'Chew Yang'.

Figure 4.21 Register Farm

In register farm module, Admin can input 'Farmer', 'Farm Name' and 'Farm Address' in order to create new farm. Next, Admin can click 'Create' button to create new farm data.

### 4.3.9 View Farm module

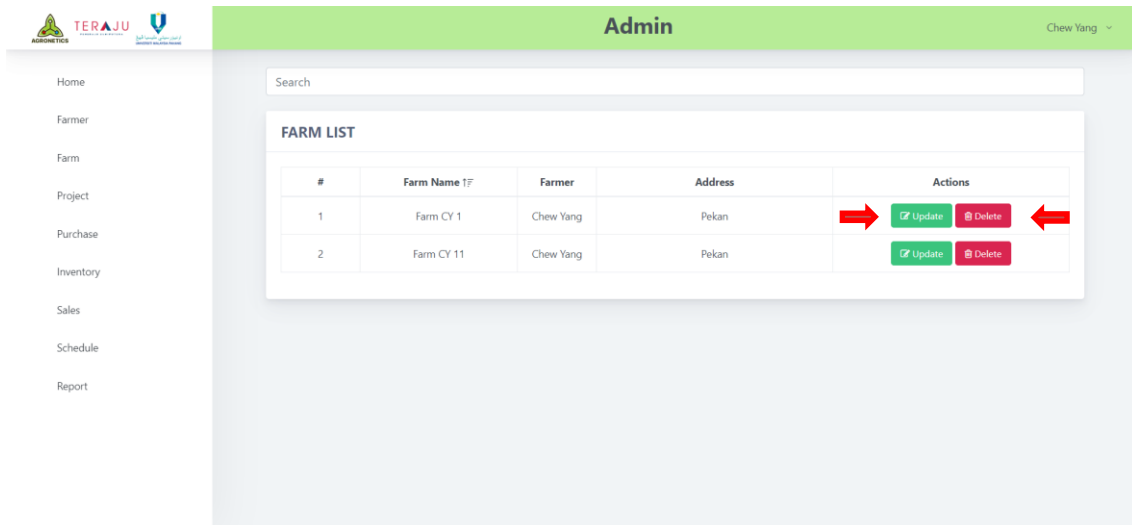


Figure 4.22 View Farm

This is the Register Farm page. When Admin navigate to this page, Admin can view the farm details. Next, Admin can click on 'Update' button and the system will navigate to Edit Farm page. Then, Admin can click on 'Delete' button to delete the selected farm. Lastly, Admin can search for the farm on the search bar above.

### 4.3.10 Edit Farm module

The screenshot displays the 'Edit Farm' module within an Admin interface. The interface includes a sidebar with navigation options: Home, Farmer, Farm, Project, Purchase, Inventory, Sales, Schedule, and Report. The main content area is titled 'Admin' and shows the 'EDIT FARM' form. The form contains the following fields and values:

- Farmer Name: Chew Yang
- Farm Name: Farm CY 1
- Address: Pekan

At the bottom of the form, there are two buttons: 'Edit' (green) and 'Cancel' (grey). Red arrows point to the 'Farm Name' and 'Address' input fields, and another red arrow points to the 'Edit' button.

Figure 4.23 Edit Farm

This is Edit farm page and Admin can edit 'Farm Name' and 'Address' from the displayed old data. Next, Admin can click 'Edit' button to update farmer data.

### 4.3.11 Project module

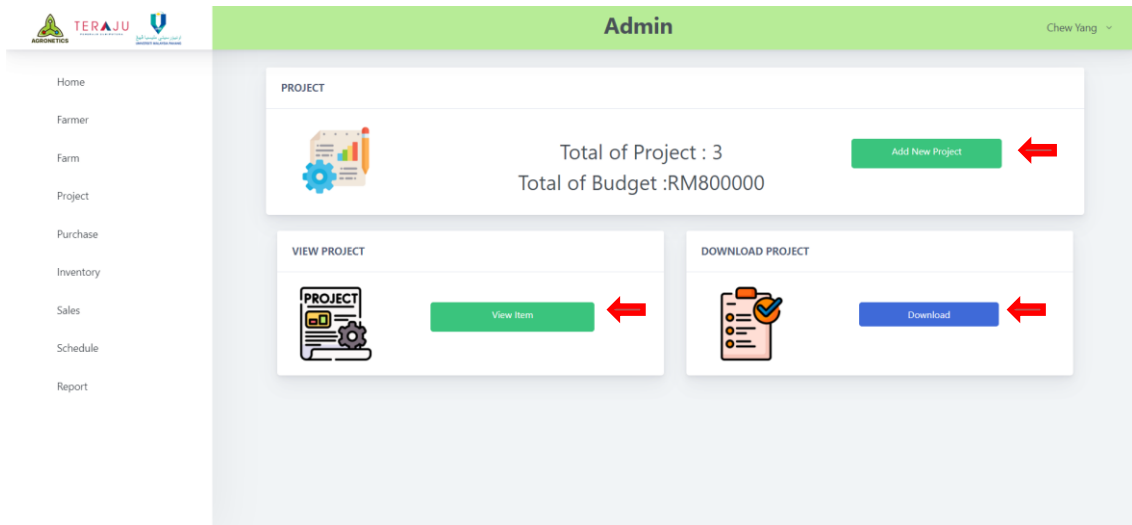


Figure 4.24 Project

This is the Project page. Admin can click 'Add New Project' button and system will navigate to Create Project page. Next, Admin can click on 'View Project' button and system will navigate to View Project page. Then, Admin can click on 'Download' button to download all project data in excel file.

### 4.3.12 Add Project module

The screenshot shows the 'ADD NEW PROJECT' form in the Admin module. The form is titled 'ADD NEW PROJECT' and is located in the 'Admin' section. The user is identified as 'Chew Yang'. The form contains the following fields:

- PROJECT NAME: A text input field.
- FARM: A dropdown menu with 'Farm CY 1' selected.
- BUDGET (RM): A text input field.
- START DATE: A date picker with the format 'dd/mm/yyyy'.
- END DATE: A date picker with the format 'dd/mm/yyyy'.
- STATUS: A dropdown menu with 'Active' selected.

At the bottom of the form, there are two buttons: 'Add New' (green) and 'Cancel' (grey). Red arrows point to each of these fields and buttons, indicating where the user should input data or click.

Figure 4.25 Add New Project

This add new project page and Admin can input 'Project Name', 'Farm', 'Budget', 'Start Date', 'End Date' and 'Status' in order to add new project. Next, Admin can click 'Add New' button to create new project data.

### 4.3.13 View Project module

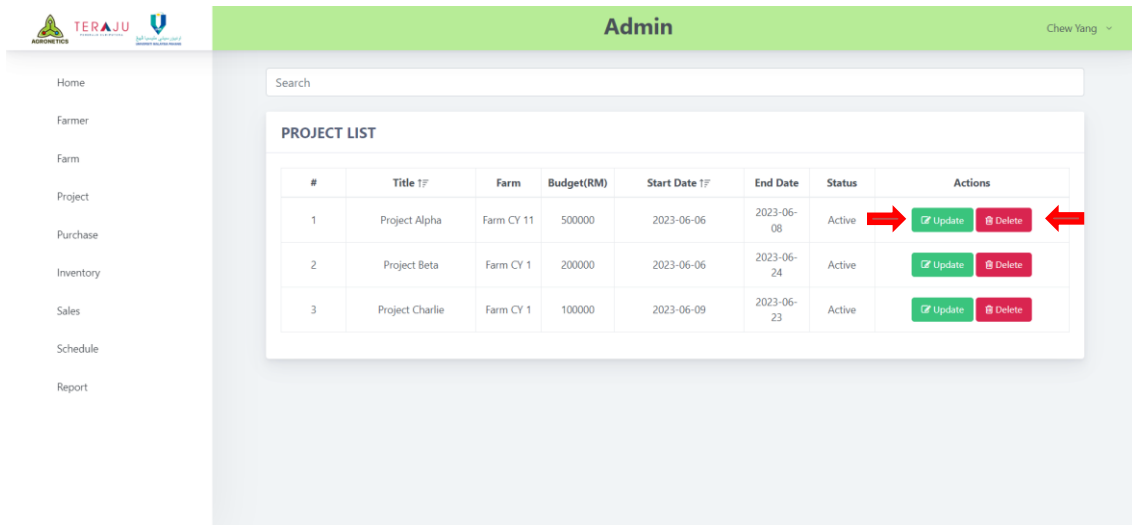


Figure 4.26 View Project

This is the View Project page. Admin can view the project details. Next, Admin can click on 'Update' button and the system will navigate to Edit Project page. Then, Admin can click on 'Delete' button to delete the selected project. Lastly, Admin can search for the project on the search bar above.

### 4.3.14 Edit Project module

The screenshot displays the 'Admin' interface for editing a project. The page title is 'Admin' and the user is 'Chew Yang'. The main content area is titled 'EDIT PROJECT' and contains the following form fields:

- PROJECT NAME: Project Alpha
- FARM: Farm CY 11
- BUDGET (RM): 500000
- START DATE: 06/06/2023
- END DATE: 08/06/2023
- STATUS: Active

At the bottom right, there are two buttons: 'Edit' (green) and 'Cancel' (grey). Red arrows point to each of these fields and buttons, indicating they are the focus of the edit operation.

Figure 4.27 Edit Project

This is edit project page and Admin can edit 'Project Name', 'Farm', 'Budget', 'Start Date', 'End Date' and 'Status' from displayed old data. Next, Admin can click 'Edit' button to update the project data.

### 4.3.15 Purchase module

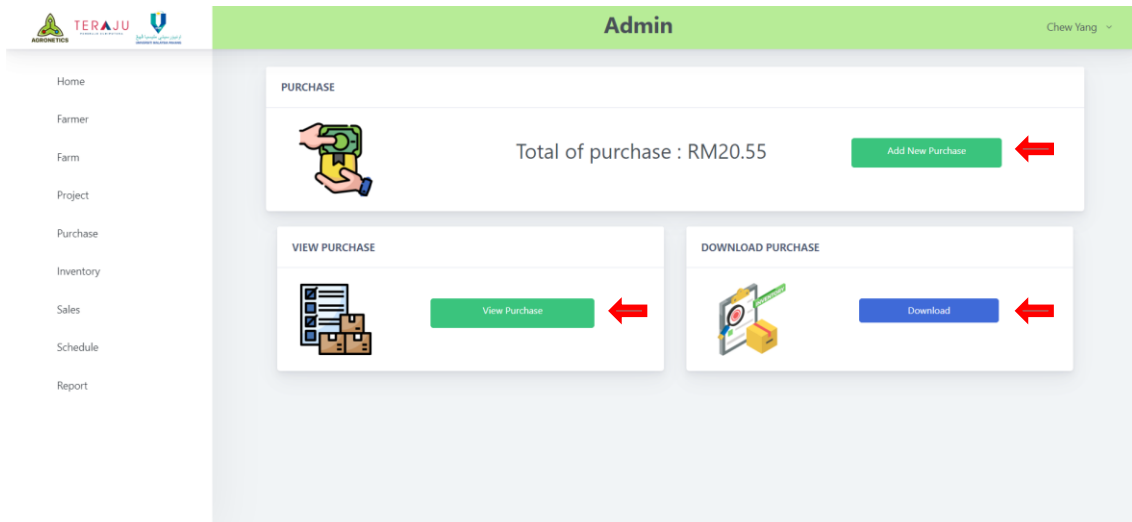


Figure 4.28 Purchase

This is the Purchase page. Admin can click 'Add New Purchase' button and system will navigate to Create Purchase page. Next, Admin can click on 'View Purchase' button and system will navigate to View Purchase page. Lastly, Admin can click on 'Download' button to download all purchase data in excel file.



### 4.3.16 Create Purchase module

The screenshot shows the 'Admin' interface with a sidebar menu on the left containing 'Home', 'Farmer', 'Farm', 'Project', 'Purchase', 'Inventory', 'Sales', 'Schedule', and 'Report'. The main content area is titled 'ADD NEW PURCHASE' and contains the following form fields:

- PIC Name: A text input field with a red arrow pointing to it from the right.
- Project: A dropdown menu showing 'Project Alpha' with a red arrow pointing to it from the right.
- Purchase Title: A text input field with a red arrow pointing to it from the right.
- PRICE (RM): A text input field with a red arrow pointing to it from the right.
- DATE: A date picker showing 'dd/mm/yyyy' with a calendar icon and a red arrow pointing to it from the right.
- UPLOAD RECEIPT: A file upload area with a 'Choose File' button and the text 'No file chosen'. A red arrow points to the 'Add New' button below this field.

At the bottom right of the form, there are two buttons: 'Add New' (green) and 'Cancel' (grey). A red arrow points to the 'Add New' button.

Figure 4.29 Create Purchase

This is create purchase page and Admin can input 'PIC Name', 'Project', 'Budget', 'Purchase Title', 'Date' and 'File'. Next, Admin can click 'Add New' button to create new purchase data.

### 4.3.17 View Purchase module

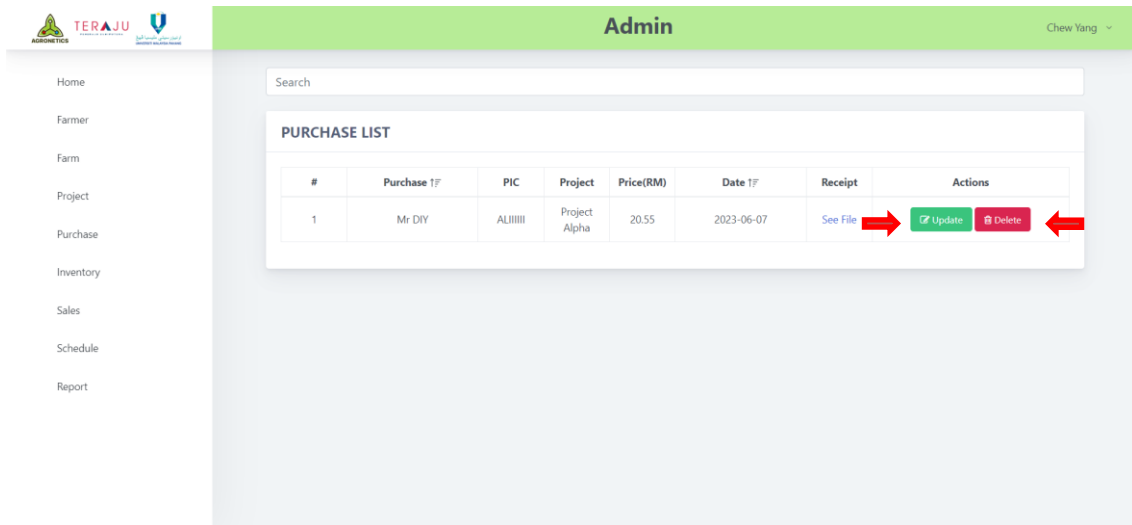


Figure 4.30 View Purchase

This is the View Purchase page. Admin can view the purchase details. Next, Admin can click on 'Update' button and the system will navigate to Edit Purchase page. Then, Admin can click on 'Delete' button to delete the selected purchase. Lastly, Admin can search for the purchase on the search bar above.

### 4.3.18 Edit Purchase module

The screenshot displays the 'Edit Purchase' interface. On the left is a sidebar with menu items: Home, Farmer, Farm, Project, Purchase, Inventory, Sales, Schedule, and Report. The main header is green with 'Admin' and 'Chew Yang'. The form fields are: PIC Name (text input: ALIIIII), Project (dropdown: Project Alpha), PURCHASE TITLE (text input: Mr DIY), PRICE (RM) (text input: 20.55), and DATE (calendar input: 07/06/2023). Below these is an 'UPLOAD RECEIPT' section showing 'Current File: NmIzMMRdQC7pvFsBandMCHpAUkswKASirMClb0KSO.jpg' and a 'Choose File' button with 'No file chosen' text. Red arrows point to each of these fields and the 'Choose File' button.

Figure 4.31 Edit Purchase

This is edit purchase page and Admin can edit 'PIC Name', 'Project', 'Budget', 'Purchase Title', 'Date' and 'File' from the old displayed data. Lastly, Admin can click 'Edit' button to update purchase data.

### 4.3.19 Inventory module

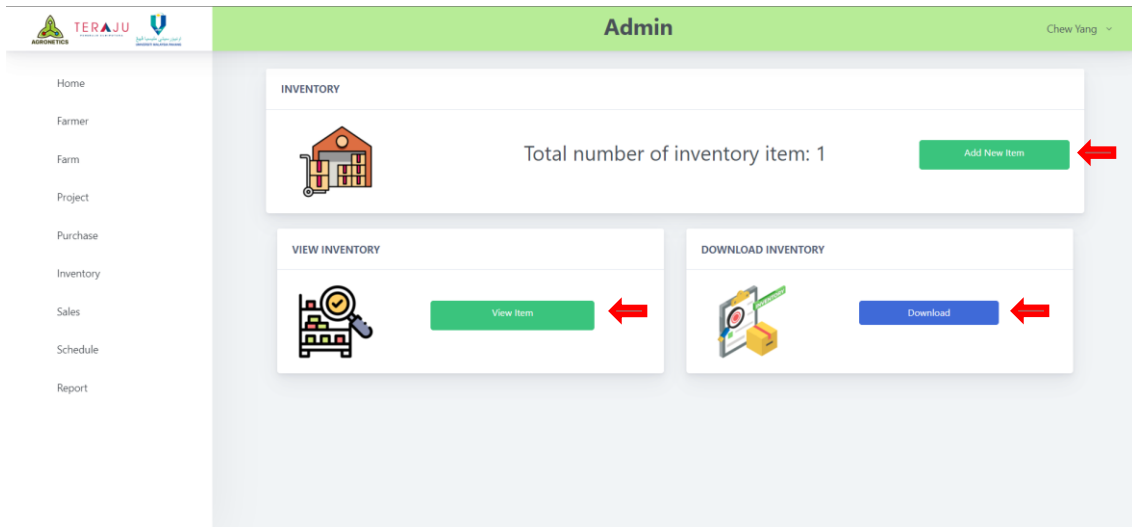


Figure 4.32 Inventory

This is the Inventory page. Admin can click 'Add New Item' button and system will navigate to Create Inventory page. Next, Admin can click on 'View Item' button and system will navigate to View Inventory page. Lastly, Admin can click on 'Download' button to download all inventory data in excel file.

### 4.3.20 Create Inventory module

The screenshot shows the 'ADD NEW INVENTORY' form in the Admin module. The form is titled 'ADD NEW INVENTORY' and is located in the 'Admin' section. The user is identified as 'Chew Yang'. The form contains the following fields:

- Purchase: A dropdown menu with 'Mr DIY' selected.
- Name: A text input field.
- Purchase PIC: A text input field.
- Quantity: A text input field.
- PRICE (RM): A text input field.
- DATE: A date picker field with the format 'dd/mm/yyyy'.
- STATUS: A dropdown menu with 'In Stock' selected.

At the bottom of the form, there are two buttons: 'Add New' (green) and 'Cancel' (grey). Red arrows point to each of these fields and buttons, indicating where the user should input data or click.

Figure 4.33 Create Inventory

This is the create inventory page and Admin can input 'Purchase', 'Name', 'Purchase PIC', 'Quantity', 'Price', 'Date' and 'Status' in order to create inventory. Next, admin can click 'Add New' button to create new inventory data.

### 4.3.21 View Inventory module

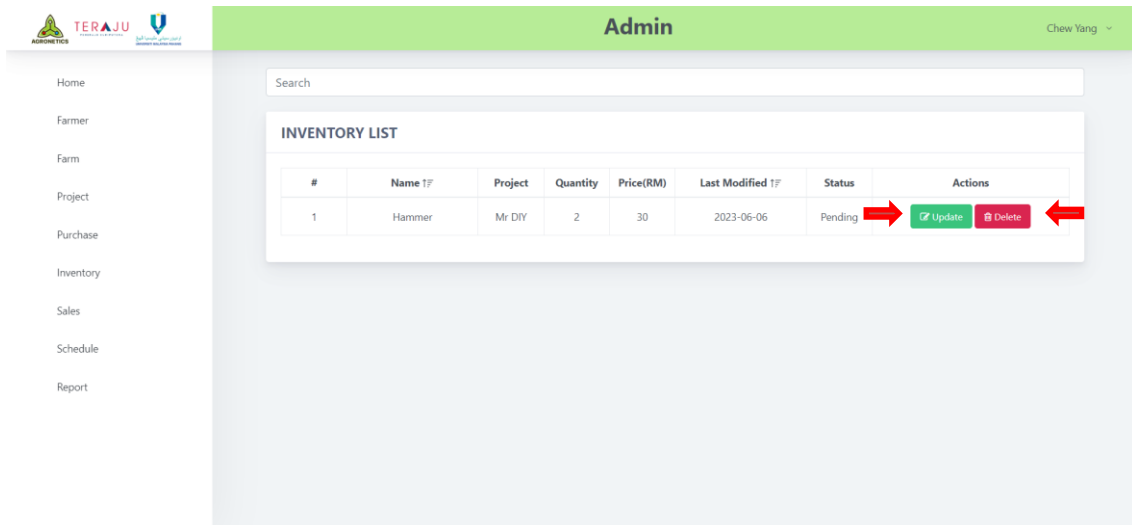


Figure 4.34 View Inventory

This is the View Inventory page. Admin can view the inventory item details. Next, Admin can click on 'Update' button and the system will navigate to Edit Inventory page. Then, Admin can click on 'Delete' button to delete the selected inventory item. Lastly, Admin can search for the inventory item on the search bar above.

### 4.3.22 Edit Inventory module

The screenshot displays the 'Admin' interface for the 'Edit Inventory' module. The page title is 'Admin' and the user is 'Chew Yang'. The left sidebar contains navigation options: Home, Farmer, Farm, Project, Purchase, Inventory, Sales, Schedule, and Report. The main content area is titled 'EDIT INVENTORY' and contains the following form fields:

- Purchase:** A dropdown menu with 'Mr DIY' selected.
- Name:** A text input field containing 'Hammer'.
- Purchase PIC:** A text input field containing 'Ali'.
- Quantity:** A text input field containing '2'.
- PRICE (RM):** A text input field containing '30'.
- DATE:** A date picker field showing '06/06/2023'.
- STATUS:** A dropdown menu with 'Pending' selected.

At the bottom right of the form, there are two buttons: a green 'Edit' button and a grey 'Cancel' button. Red arrows point to each of these fields and buttons, indicating they are editable.

Figure 4.35 Edit Inventory

This is edit inventory page and Admin can edit 'Purchase', 'Name', 'Purchase PIC', 'Quantity', 'Price', 'Date' and 'Status' in order to edit the inventory. Lastly, Admin can click 'Edit' button to update inventory data.

### 4.3.23 Sales module

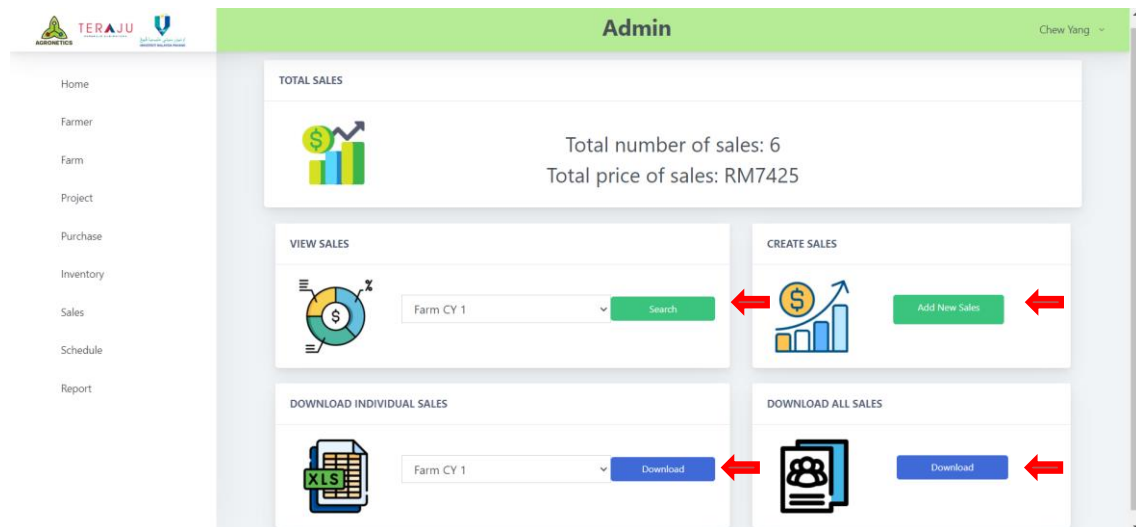


Figure 4.36 Sales

This is the Sales page. Admin can click ‘Add New Sales’ button and system will navigate to Create Sales page. Next, Admin can select farm and then click on ‘Search’ button and system will navigate to View Sales page with the data of selected farm only. Lastly, Admin can click on ‘Download’ button to download all sales data or individual sales data in excel file.



### 4.3.24 Create Sales module

The screenshot shows the 'Admin' interface for creating new sales. The page title is 'Admin' and the user is 'Chew Yang'. The main heading is 'CREATE NEW SALES'. The form contains the following fields:

- FARMER: A dropdown menu with 'Farm CV 1' selected.
- SALES TITLE (PLANT): A text input field.
- CROP TYPE: A text input field.
- WEIGHT (KG): A text input field.
- GRADE: A text input field.
- MARKET PRICE (RM/KG): A text input field.
- TOTAL SALES PRICE (RM): A text input field.
- DATE: A date input field with the format 'dd/mm/yyyy' and a calendar icon.

At the bottom right, there are two buttons: 'Add New' (green) and 'Cancel' (grey). Red arrows point to each of these fields and buttons.

Figure 4.37 Create Sales

This is create sales page and Admin can input 'Farmer', 'Sales Title', 'Crop Type', 'Weight', 'Grade', 'Market Price', 'Total Sales' and 'Date' in order to add new sales. Lastly, Admin can click 'Add New' button to create new sales data.

### 4.3.25 View Sales module

The screenshot displays the 'View Sales' module in the Admin interface. The page features a search bar at the top and a sidebar on the left with navigation options: Home, Farmer, Farm, Project, Purchase, Inventory, Sales, Schedule, and Report. The main content area shows a table titled 'SALES LIST FOR FARM CY 1' with the following data:

#	Name	Type	Weight(KG)	Price(RM/KG)	TOTAL(RM)	Grade	Date	Actions
1	cili	cili	20	20	400	A	2022-08-07	<a href="#">Update</a> <a href="#">Delete</a>
2	cili	cili	50	20	1000	A	2023-06-06	<a href="#">Update</a> <a href="#">Delete</a>
3	eggplant	eggplant	50	25	125	A	2023-03-05	<a href="#">Update</a> <a href="#">Delete</a>
4	cili	cili	20	20	500	A	2023-05-09	<a href="#">Update</a> <a href="#">Delete</a>
5	cili	vege	10	20	400	A	2023-06-10	<a href="#">Update</a> <a href="#">Delete</a>

Figure 4.38 View Sales

This is the View Sales page. Admin can view the sales details. Next, Admin can click on 'Update' button and the system will navigate to Edit Sales page. Then, Admin can click on 'Delete' button to delete the selected sales item. Lastly, Admin can search for the sales on the search bar above.

### 4.3.26 Edit Sales Module

The screenshot displays the 'Admin' interface for the 'Edit Sales' module. The page title is 'Admin' and the user is 'Chew Yang'. The left sidebar contains navigation links: Home, Farmer, Farm, Project, Purchase, Inventory, Sales, Schedule, and Report. The main content area is titled 'EDIT SALES' and contains the following form fields:

- FARMER: Farm CY 1
- SALES TITLE (PLANT): cili
- CROP TYPE: cili
- WEIGHT (KG): 20
- GRADE: A
- MARKET PRICE (RM/KG): 20
- TOTAL PRICE (RM): 400
- DATE: 07/08/2022

At the bottom of the form, there are two buttons: 'Edit' (green) and 'Cancel' (grey). Red arrows point to each of the form fields and the 'Edit' button.

Figure 4.39 Edit Sales

This is edit sales page and Admin can perform edit 'Farmer', 'Sales Title', 'Crop Type', 'Weight', 'Grade', 'Market Price', 'Total Sales' and 'Date' in order to edit the sales. Next, Admin can click 'Edit' button to update sales data.

### 4.3.27 Schedule Module

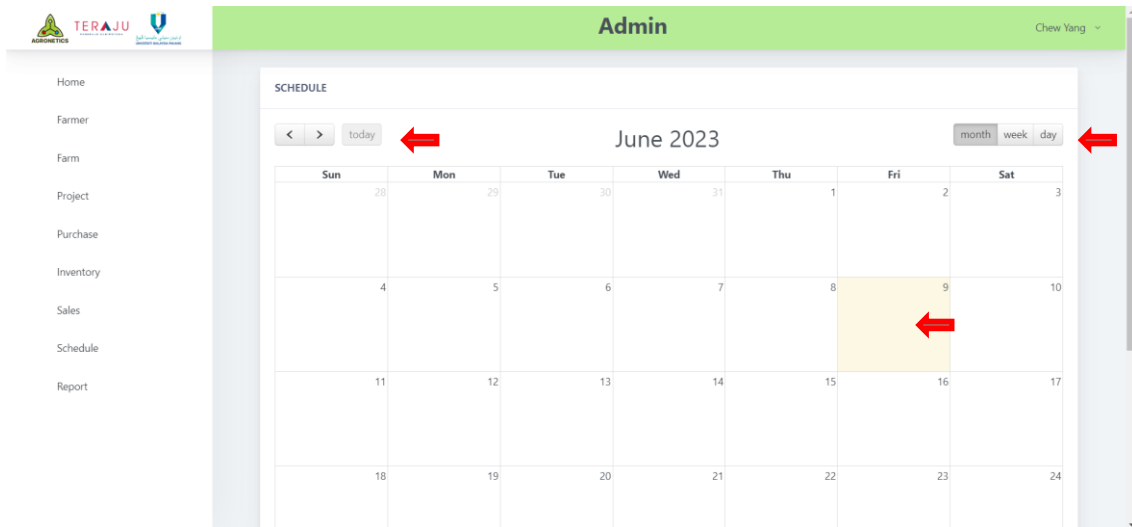


Figure 4.40 Schedule

This is the Schedule page. Admin can add event into the calendar by click on the date. Next, Admin can drag and drop the event freely in the calendar and the data is updated in database. Then, Admin can click on the event to delete the event. Furthermore, Admin can select the view mode on month, week and day. Lastly, Admin can change the month of the calendar freely.

### 4.3.28 Report module

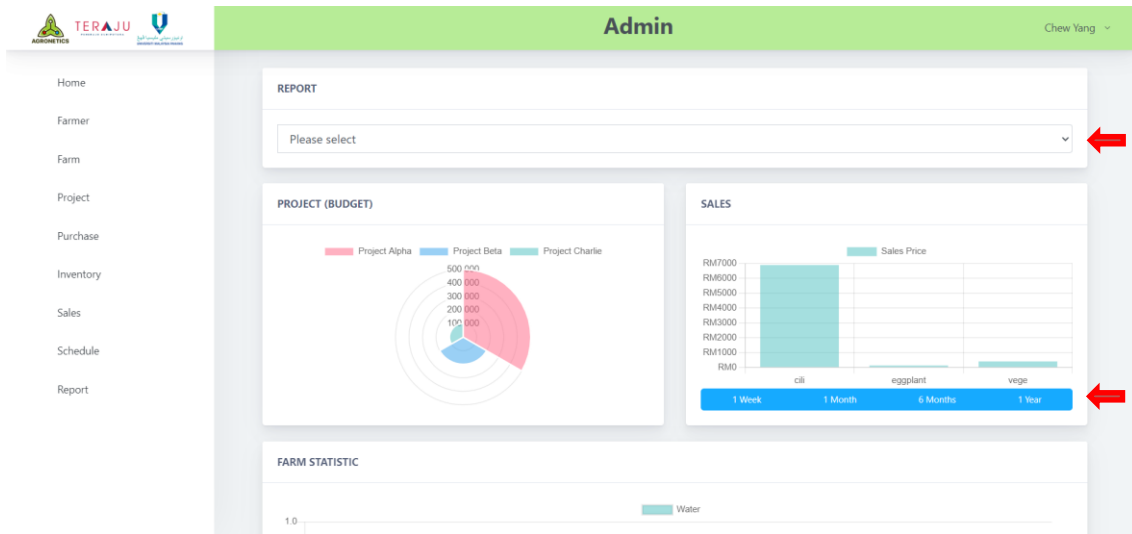


Figure 4.41 Report

This is the Report page. Admin can view the default all data graph. Next, Admin can select the farm at above to see individual graph. Lastly, Admin can select timeframe at the graph to show specific graph.

## 4.4 Database Implementation

Table	Action	Rows	Type	Collation	Size	Overhead
calendars	★ Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_unicode_ci	16.0 K1B	-
controltest	★ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_unicode_ci	16.0 K1B	-
farm	★ Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_unicode_ci	16.0 K1B	-
farmers	★ Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_unicode_ci	16.0 K1B	-
inventorys	★ Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_unicode_ci	16.0 K1B	-
migrations	★ Browse Structure Search Insert Empty Drop	12	InnoDB	utf8mb4_unicode_ci	16.0 K1B	-
password_resets	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	32.0 K1B	-
password_reset_tokens	★ Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_unicode_ci	16.0 K1B	-
project	★ Browse Structure Search Insert Empty Drop	4	InnoDB	utf8mb4_unicode_ci	16.0 K1B	-
purchase	★ Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_unicode_ci	16.0 K1B	-
sales	★ Browse Structure Search Insert Empty Drop	7	InnoDB	utf8mb4_unicode_ci	16.0 K1B	-
users	★ Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_unicode_ci	32.0 K1B	-
12 tables	Sum	39	InnoDB	utf8mb4_general_ci	224.0 K1B	0 B

Figure 4.42 Database table

There are 12 tables that is implemented in this project and each table will handle their own data according to their module.

### 4.4.1 calendars table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	calendarID 🔑	bigint(20)		UNSIGNED	No	None		AUTO_INCREMENT
2	title	varchar(255)	utf8mb4_unicode_ci		No	None		
3	start	datetime			No	None		
4	end	datetime			No	None		

Figure 4.43 calendars table

This table is handling the data of calendar and there is calendarID as primary key and the other attributes are title, start and end.

#### 4.4.2 controltest table


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	<b>controltestID</b> 	bigint(20)		UNSIGNED	No	None		AUTO_INCREMENT
2	<b>farmID</b>	varchar(255)	utf8mb4_unicode_ci		No	None		
3	<b>timestamp</b>	int(11)			No	None		
4	<b>water</b>	varchar(255)	utf8mb4_unicode_ci		Yes	NULL		
5	<b>fertilizer</b>	varchar(255)	utf8mb4_unicode_ci		Yes	NULL		
6	<b>moisture</b>	int(11)			Yes	NULL		
7	<b>ec</b>	int(11)			Yes	NULL		
8	<b>temperature</b>	double			Yes	NULL		

Figure 4.44 controltest table

This table is handling the data of farm sensor and there is controltestID as primary key, farmID as foreign key and the other attributes are timestamp, water, fertilizer, moisture, ec and temperature.

#### 4.4.3 farm table


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	<b>farmID</b> 	bigint(20)		UNSIGNED	No	None		AUTO_INCREMENT
2	<b>farmerID</b>	varchar(255)	utf8mb4_unicode_ci		No	None		
3	<b>title</b>	varchar(255)	utf8mb4_unicode_ci		Yes	NULL		
4	<b>address</b>	varchar(255)	utf8mb4_unicode_ci		Yes	NULL		

Figure 4.45 farm table

This table is handling the data of farm and there is farmID as primary key, farmerID as foreign key and the other attributes are title and address.

#### 4.4.4 farmers table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	<b>id</b> 🔑	bigint(20)		UNSIGNED	No	None		AUTO_INCREMENT
2	<b>email</b>	varchar(255)	utf8mb4_unicode_ci		Yes	NULL		
3	<b>password</b>	varchar(255)	utf8mb4_unicode_ci		Yes	NULL		
4	<b>fmname</b>	varchar(255)	utf8mb4_unicode_ci		Yes	NULL		
5	<b>phone</b>	varchar(255)	utf8mb4_unicode_ci		Yes	NULL		
6	<b>location</b>	varchar(255)	utf8mb4_unicode_ci		Yes	NULL		

Figure 4.46 farmers table

This table is handling the data of farmer and there is id as primary key and the other attributes are email, password, fmname, phone and location.

#### 4.4.5 inventories table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	<b>inventoryID</b> 🔑	bigint(20)		UNSIGNED	No	None		AUTO_INCREMENT
2	<b>purchaseID</b>	int(11)			No	None		
3	<b>inventoryName</b>	varchar(255)	utf8mb4_unicode_ci		No	None		
4	<b>inventoryPIC</b>	varchar(255)	utf8mb4_unicode_ci		No	None		
5	<b>quantity</b>	int(11)			No	None		
6	<b>price</b>	double			No	None		
7	<b>dateStored</b>	date			No	None		
8	<b>status</b>	varchar(255)	utf8mb4_unicode_ci		No	None		
9	<b>lastModified</b>	date			No	None		

Figure 4.47 inventories table

This table is handling the data of inventory and there is inventoryID as primary key, purchaseID as foreign key and the other attributes are email, password, fmname, phone and location.



#### 4.4.6 migrations table


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	<b>id</b> 	int(10)		UNSIGNED	No	None		AUTO_INCREMENT
2	<b>migration</b>	varchar(255)	utf8mb4_unicode_ci		No	None		
3	<b>batch</b>	int(11)			No	None		

Figure 4.48 migrations table

This table is handling the data migration and there is id as primary key and the other attributes are migration and batch.

#### 4.4.7 password\_resets table


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	<b>email</b> 	varchar(255)	utf8mb4_unicode_ci		No	None		
2	<b>token</b>	varchar(255)	utf8mb4_unicode_ci		No	None		
3	<b>created_at</b>	timestamp			Yes	NULL		

Figure 4.49 password\_resets table

This table is handling the password reset data and there is email as primary key and the other attributes are token and created\_at.

#### 4.4.8 password\_reset\_tokens table


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	<b>email</b> 	varchar(255)	utf8mb4_unicode_ci		No	None		
2	<b>token</b>	varchar(255)	utf8mb4_unicode_ci		No	None		
3	<b>created_at</b>	timestamp			Yes	NULL		

Figure 4.50 password\_reset\_token table

This table is handling the password reset token and there is email as primary key and the other attributes are token and created\_at.

#### 4.4.9 project table


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	<b>projectID</b> 	bigint(20)		UNSIGNED	No	None		AUTO_INCREMENT
2	<b>farmID</b>	int(11)			No	None		
3	<b>title</b>	varchar(255)	utf8mb4_unicode_ci		No	None		
4	<b>budget</b>	double			No	None		
5	<b>startDate</b>	date			No	None		
6	<b>endDate</b>	date			No	None		
7	<b>status</b>	tinyint(1)			No	None		

Figure 4.51 project table

This table is handling the project data and there is projectID as primary key, farmID as foreign key and the other attributes are title, budget, startDate, endDate and status.

#### 4.4.10 purchase table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	<b>purchaseID</b> 🔑	bigint(20)		UNSIGNED	No	None		AUTO_INCREMENT
2	<b>projectID</b>	int(11)			No	None		
3	<b>title</b>	varchar(255)	utf8mb4_unicode_ci		No	None		
4	<b>projectPIC</b>	varchar(255)	utf8mb4_unicode_ci		No	None		
5	<b>price</b>	double(8,2)			No	None		
6	<b>date</b>	date			No	None		
7	<b>imageUri</b>	varchar(255)	utf8mb4_unicode_ci		No	None		

Figure 4.52 purchase table

This table is handling the purchase data and there is purchaseID as primary key, projectID as foreign key and the other attributes are title, projectPIC, price, date and imageUri.

#### 4.4.11 sales table


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	<b>salesID</b> 	bigint(20)		UNSIGNED	No	None		AUTO_INCREMENT
2	<b>farmID</b>	int(11)			No	None		
3	<b>salesTitle</b>	varchar(50)	utf8mb4_unicode_ci		No	None		
4	<b>salesGrade</b>	varchar(5)	utf8mb4_unicode_ci		No	None		
5	<b>cropsType</b>	varchar(15)	utf8mb4_unicode_ci		No	None		
6	<b>salesWeight</b>	double(7,2)			No	None		
7	<b>salesPrice</b>	double(8,2)			No	None		
8	<b>marketPrice</b>	double(8,2)			No	None		
9	<b>salesDate</b>	date			No	None		

Figure 4.53 sales table

This table is handling the sales data and there is salesID as primary key, farmID as foreign key and the other attributes are salesTitle, salesGrade, cropsType, salesWeight, salesPrice, marketPrice and salesDate.

#### 4.4.12 users table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id 🔑	bigint(20)		UNSIGNED	No	None		AUTO_INCREMENT
2	name	varchar(255)	utf8mb4_unicode_ci		No	None		
3	email 🔑	varchar(255)	utf8mb4_unicode_ci		No	None		
4	email_verified_at	timestamp			Yes	NULL		
5	password	varchar(255)	utf8mb4_unicode_ci		No	None		
6	option	varchar(255)	utf8mb4_unicode_ci		No	admin		
7	remember_token	varchar(100)	utf8mb4_unicode_ci		Yes	NULL		
8	created_at	timestamp			Yes	NULL		
9	updated_at	timestamp			Yes	NULL		

Figure 4.54 users table

This table is handling the users data and there is id as primary key and the other attributes are name, email, email\_verified\_at, password, option, remember\_token, created\_at and updated\_at.

#### 4.5 Testing and Result Discussion

The testing technique that used in this project is User Acceptance Test. The reason that I choose this testing technique is because through this method, we can understand well that whether our system meets the requirement of the system. In addition, we will carry out the testing process with our stakeholders especially the clients to make sure that they whether they satisfy the project or not. From the analysis of UAT, we can conclude that the developed functions meet with the requirements from the client and the UAT session is successful and happy for both of us. The example form of UAT is shown as below.

**USER ACCEPTANCE TEST**  
**Automated Fertigation System**

No	Acceptance Requirement	Test Result		Comment
<b>Test for Admin</b>				
<b>Login</b>				
1.	Login	(Yes)	No	
2.	Forgot Password	(Yes)	No	
3.	Logout	(Yes)	No	
<b>Home</b>				
1.	Select farm	(Yes)	No	
2.	Display farm data	(Yes)	No	
<b>Farmer</b>				
1.	Add New Farmer	(Yes)	No	
2.	View Farmer	(Yes)	No	
3.	Update Farmer	(Yes)	No	
4.	Delete Farmer	(Yes)	No	
5.	Download Farmer data			
<b>Farm</b>				
1.	Add New Farmer	(Yes)	No	
2.	View Farmer	(Yes)	No	
3.	Update Farmer	(Yes)	No	
4.	Delete Farmer	(Yes)	No	
5.	Download Farmer Data	(Yes)	No	
<b>Project</b>				
1.	Add New Project	(Yes)	No	
2.	View Project	(Yes)	No	
3.	Update Project	(Yes)	No	
4.	Delete Project	(Yes)	No	
5.	Download Project Data	(Yes)	No	
<b>Purchase</b>				
1.	Add New Purchase	(Yes)	No	
2.	View Purchase	(Yes)	No	
3.	Update Purchase	(Yes)	No	
4.	Delete Purchase	(Yes)	No	
5.	Download Purchase Data	(Yes)	No	
6.	View uploaded file	(Yes)	No	
<b>Inventory</b>				
1.	Add New Inventory	(Yes)	No	
2.	View Inventory	(Yes)	No	
3.	Update Inventory	(Yes)	No	
4.	Delete Inventory	(Yes)	No	
5.	Download Inventory Data	(Yes)	No	

Figure 4.55 UAT Form 1 Page 1

Sales			
1.	Add New Sales	<input checked="" type="radio"/>	No
2.	Select User and View Sales	<input checked="" type="radio"/>	No
3.	Update Sales	<input checked="" type="radio"/>	No
4.	Delete Sales	<input checked="" type="radio"/>	No
5.	Download All Sales Data	<input checked="" type="radio"/>	No
6.	Download Individual Sales Data	<input checked="" type="radio"/>	No
Schedule			
1.	Add New Event	<input checked="" type="radio"/>	No
2.	View Event	<input checked="" type="radio"/>	No
2.	Update Event	<input checked="" type="radio"/>	No
3.	Delete Event	<input checked="" type="radio"/>	No
Report			
1.	View All User Graph	<input checked="" type="radio"/>	No
2.	View User Graph	<input checked="" type="radio"/>	No

Comment (Improve/ Design/ Bugs):

All functioning well. Can improve for faster loading time and improve UI design.

Name: Nurul Syahiqah Binti Zaidi

Date: 09/06/2023

I, Nurul Syahiqah Binti Zaidi had conducted the User Acceptance Test as requested. I admit that the information that filled is my true personal opinion.



Name: Nurul Syahiqah Binti Zaidi

Figure 4.56 UAT Form 1 Page 2

**USER ACCEPTANCE TEST**  
**Automated Fertigation System**

No	Acceptance Requirement	Test Result		Comment
<b>Test for Admin</b>				
<b>Login</b>				
1.	Login	(Yes)	No	
2.	Forgot Password	(Yes)	No	
3.	Logout	(Yes)	No	
<b>Home</b>				
1.	Select farm	(Yes)	No	
2.	Display farm data	(Yes)	No	
<b>Farmer</b>				
1.	Add New Farmer	(Yes)	No	
2.	View Farmer	(Yes)	No	
3.	Update Farmer	(Yes)	No	
4.	Delete Farmer	(Yes)	No	
5.	Download Farmer data			
<b>Farm</b>				
1.	Add New Farmer	(Yes)	No	
2.	View Farmer	(Yes)	No	
3.	Update Farmer	(Yes)	No	
4.	Delete Farmer	(Yes)	No	
5.	Download Farmer Data	(Yes)	No	
<b>Project</b>				
1.	Add New Project	(Yes)	No	
2.	View Project	(Yes)	No	
3.	Update Project	(Yes)	No	
4.	Delete Project	(Yes)	No	
5.	Download Project Data	(Yes)	No	
<b>Purchase</b>				
1.	Add New Purchase	(Yes)	No	
2.	View Purchase	(Yes)	No	
3.	Update Purchase	(Yes)	No	
4.	Delete Purchase	(Yes)	No	
5.	Download Purchase Data	(Yes)	No	
6.	View uploaded file	(Yes)	No	
<b>Inventory</b>				
1.	Add New Inventory	(Yes)	No	
2.	View Inventory	(Yes)	No	
3.	Update Inventory	(Yes)	No	
4.	Delete Inventory	(Yes)	No	
5.	Download Inventory Data	(Yes)	No	

Figure 4.57 UAT Form 2 Page 1



Sales			
1.	Add New Sales	<input checked="" type="radio"/> Yes	<input type="radio"/> No
2.	Select User and View Sales	<input checked="" type="radio"/> Yes	<input type="radio"/> No
3.	Update Sales	<input checked="" type="radio"/> Yes	<input type="radio"/> No
4.	Delete Sales	<input checked="" type="radio"/> Yes	<input type="radio"/> No
5.	Download All Sales Data	<input checked="" type="radio"/> Yes	<input type="radio"/> No
6.	Download Individual Sales Data	<input checked="" type="radio"/> Yes	<input type="radio"/> No
Schedule			
1.	Add New Event	<input checked="" type="radio"/> Yes	<input type="radio"/> No
2.	View Event	<input checked="" type="radio"/> Yes	<input type="radio"/> No
2.	Update Event	<input checked="" type="radio"/> Yes	<input type="radio"/> No
3.	Delete Event	<input checked="" type="radio"/> Yes	<input type="radio"/> No
Report			
1.	View All User Graph	<input checked="" type="radio"/> Yes	<input type="radio"/> No
2.	View User Graph	<input checked="" type="radio"/> Yes	<input type="radio"/> No

Comment (Improve/ Design/ Bugs):

Can improve UI design for admin website and faster loading time.

Name: DR. MOHD AZRAAI BIN MOHD RAZMAN

Date: 09/06/2023

I, DR. MOHD AZRAAI BIN MOHD RAZMAN had conducted the User Acceptance Test as requested. I admit that the information that filled is my true personal opinion.



Name: DR. MOHD AZRAAI BIN MOHD RAZMAN

DR. MOHD AZRAAI BIN MOHD RAZMAN  
 PERSOANAL KAWAN  
 FAKULTI TEKNOLOGI KOMPUTER  
 HUBUNGAN & MUPATRAH  
 UNIVERSITI MALAYSIA BANGI  
 43600 SEREMBAN, NEGERI SEMBILAN

Figure 4.58 UAT Form 2 Page 2

## CHAPTER 5

### 5.1 Introduction

Chapter 5 of this project focuses on the conclusion, project constraints, and future work. It provides a concise summary of the project's outcomes, emphasizing the successful development of Automated Fertigation System (Web-Based). The conclusion section highlights the positive impact of the system on optimizing farm operations and resource utilization. The chapter also discusses the constraints encountered during the project, including technical limitations and budgetary restrictions. Finally, it explores future possibilities for enhancing the system, such as incorporating advanced analytics, IoT sensors, and mobile compatibility, ensuring continuous improvement and adaptability to evolving agricultural needs. Overall, Chapter 5 offers a comprehensive overview of the project's conclusion, constraints, and future directions.

### 5.2 Project Constraint

During the development process of the project, several constraints were encountered that influenced its execution and outcomes. These constraints included:

- **Technical Limitations:** The project faced challenges related to the technical infrastructure and software capabilities. Integrating and managing a large volume of farm data required robust database systems and efficient data processing algorithms.
- **Time Constraints:** The project had specific timelines and deadlines to meet, which required efficient project management and adherence to a structured development plan. Time constraints necessitated effective task prioritization and resource allocation to ensure timely completion.
- **User Requirements:** Meeting the diverse needs and preferences of farm administrators proved to be a challenge. Balancing different user requirements and expectations required careful consideration and effective communication between the development team and stakeholders.

As a conclusion, I need to overcome various challenges and work hardly so that I am able to deliver a functional and effective farm management system. The constraints provided valuable insights and opportunities for innovation, leading to the development of creative solutions and future improvement possibilities.

### **5.3 Future Work**

Based on the feedback gathered from the user acceptance test, the future work for my system will primarily focus on improving both the design aspect and performance aspect. In terms of design, incorporating the user feedback will be essential to enhance the overall user interface (UI) design, ensuring it is intuitive, visually appealing, and tailored to the specific needs and preferences of the users. This may involve refining the layout, navigation, and visual elements to create a more engaging and user-friendly experience. Simultaneously, efforts will be directed towards optimizing system performance, particularly addressing any identified issues related to loading times and responsiveness. By leveraging user feedback and applying iterative design and optimization processes, the future work aims to create an improved system that delivers a seamless and efficient user experience while meeting the performance expectations of the users.

## REFERENCES

- Differences Between Granular Fertiliser and Foliar Fertiliser. (2022). 检索来源: SERBAJADI: <https://serbajadi.com.my/blog/differences-between-granular-fertiliser-and-foliar-fertiliser/>
- HamiltonThomas. (2023 年 January 月 7 日). What is User Acceptance Testing (UAT)? Examples. 检索来源: GURU99: <https://www.guru99.com/user-acceptance-testing.html#:~:text=The%20UAT%20test%20plan%20outlines,approach%20and%20timelines%20of%20testing.>
- IbrahimAbd Hafiz Zainal Abidin and S.NoorjannahShah. (2015). Web-based Monitoring of an Automated Fertigation.
- INSTITUTEFERTILIZERTHE. (2014). Fertilizer 101: The Big 3 - Nitrogen, Phosphorus and Potassium. 检索来源: <https://www.tfi.org/the-feed/fertilizer-101-big-3-nitrogen-phosphorus-and-potassium>
- JosephCyril. (2017). Automated fertigation system for efficient. Automated fertigation system for efficient.
- ParadigmVisual. (无日期). What is Activity Diagram. 检索来源: Visual Paradigm: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/>
- SohngenTess. (2017 年 AUGUST 月 31 日). The World Could Run Out of Food by 2023. 检索来源: GLOBAL CITIZEN: <https://www.globalcitizen.org/en/content/world-running-out-of-food-by-2023/>
- The SDLC Models & Methodology. (无日期). 检索来源: FINOIT: <https://www.finoit.com/blog/sdlc-models-methodologies/>
- World Population Clock. (2022). 检索来源: WORLDMETER: <https://www.worldometers.info/world-population/>
- Yaser Gamila. Abdullah, Ismail Abd Rahman, Muhammad Mujtaba AsadMajid. (2020). Internet of things in construction industry revolution 4.0: Recent trends and challenges in the Malaysian context. Journal of Engineering, Design and Technology.