ज्ञानेन प्रकाशते जगत्

INDUS
UNIVERSITY

Grow Green

MINI-PROJECT REPORT

by

Meet Vachhani (IU1941050079)
Saad Kadri (IU2041051092)
Pratham Shah (IU2041051095)

Department of Computer Engineering
Indus Institute of Technology and Engineering
Ahmedabad

July-December & 2022

BONAFIDE CERTIFICATE

This is to certify that this project report entitled **"Grow Green"** submitted to **Department of Computer Engineering, IITE, Ahmedabad**, is a bonafide record of work done by "**Meet Vachhani, Saad Kadri, Pratham Shah"** under my supervision from **"04-07-22"** to **"21-10-22"**

Mr. Abhishek N Vaghela
Assistant Professor,

Place
Date

ACKNOWLEDGEMENT

The final outcome of this report took a lot of help and dedication from our colleagues. We would like to sincerely thank them for their effort. We would also like to extend our graduation toward to our mentor who have helped us throughout this report and through the junction. We are incredibility thankful for our college and the computer department for providing us with all the helpful material. We enjoy and learnt a lot by creating this report and hope that the same will be left from the reader.

Meet Vachhani	**(IU1941050079)**
Saad Kadri	**(IU2041051092)**
Pratham Shah	**(IU2041051095)**

Table of Contents

Table INDEX

Figure INDEX

ABSTRACT

Now a days we are facing problem of global warming day by the temperature is increasing because of cutting of trees. There is big problem of cutting of tress for their own benefit. Now a days no one is aware to grow more plants. They want tools seeds everything for growing plant and now a day's peoples are so busy that they have no such time to buy the equipment's and grow trees.

So, to overcome with this problem we are developing a website. From which user has just to register and then log on to the website. After doing so the user can simply request for plantation from the given trees list. After that user has not to worry for plantation that will be our job to plant the seed for the requested plant at users decided location. After our job there's only user job to take care of the plant which has been planted. The user can also buy fertilizers and tools required for plant maintained from using our website. It will make user effort less and there is also possibility of growing more trees and to overcome the environmental problems.

CHAPTER 1

INTRODUCTION

1.1 Need for the New System: -

- We are on the mission to create a healthy, green and clean planet through tree plantation. Along with our diligent greening efforts while promoting extensive agriculture and making it a happy-green paradise so Idea of online plantation & tree tracking was conceived.
- We are developing a system from which user can request for planation from anywhere in limited area provided by us.
- The user can not only request for plantation but can also request for plant health status, soil testing for plant, for buying fertilizers as well as tools required for plantation.
- User has just to request for the activity which they need to do and has to pay for the following activity.

1.2 Detailed problem definition: -

- Planting a tree is a lifelong investment. How well this investment grows depends on the type of tree selected and the planting location, the care provided during planting, and the follow-up care after planting. Getting your new tree off to a healthy start will help the tree mature to its full size and ensures it will provide environmental, economic, and social benefits throughout its lifetime.
- Now a days many of us are aware of plantation but some of us didn’t know how to plant, where to plant, and what to plant that will be helpful to us in future.
- Our system has overcome with that problem the user need not to worry about the plantation that will be done by us. We also provide

the user suggestion for plant from which they can request for plantation just by following some producers.

- By our system the user of our system or visitor on our system can also get information for the events that are being held for plantation
- We also provide some information about the event to our user so that if they want, they can join the event that is being held.

1.3 Viability of the system: -

- The system saves time and also saves effort of user for planting a plant.
- The user has just to request and plantation will be done at the selected place of user.
- We think that this system will be helpful for the user as well as the environment too.

1.4 Presently Available Systems for the same: -

- Presently there is no system available for Grow Green.

1.5 Future Prospect: -

- If any problem is identified in the system that can be solved or new functionality can be added to the system in future.
- Also, we can link up our system to social media for the awareness and better response of the peoples for environment

CHAPTER-2

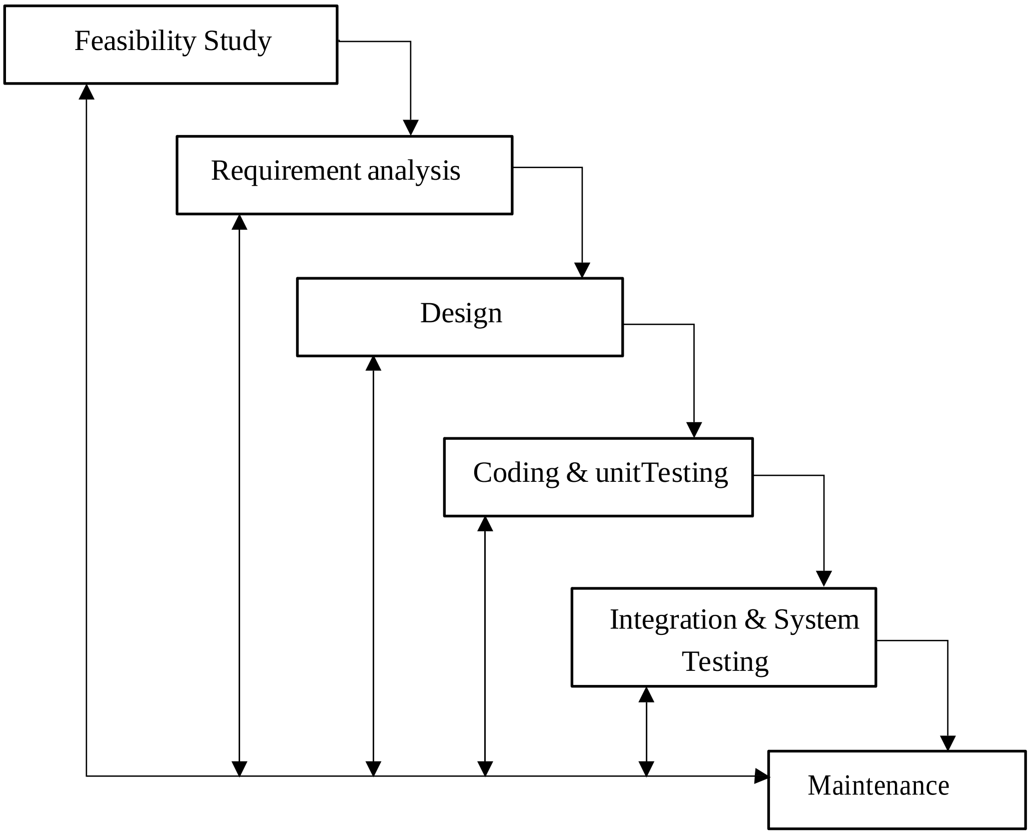
ANALYSIS

2.1 Requirement Analysis

- Our atmosphere is getting hotter, more turbulent, and more unpredictable because of the “boiling and churning” effect caused by the increase in cutting of trees.
- Tree Plantation drives combat many environmental issues like deforestation, erosion of soil, desertification in semi-arid areas, global warming and hence enhancing the beauty and balance of the environment.
- On an average, a single tree emits 260 pounds of oxygen annually. Similarly, a fully-grown tree is sufficient for 18 human beings in one acre of land in one year stressing the importance of tree plantation for mankind.
- Many of us are aware of plantation but some of us didn’t know how to plant, where to plant, and what to plant that will be helpful to us in future.
- We are developing a system from which user can request for plantation from anywhere in limited area provided by us.
- The user can not only request for plantation but can also request for plant health status, soil testing for plant, for buying fertilizers as well as tools required for plantation.

2.2 Project Model

- Classical waterfall model is idealistic it assumes that no defect is introduced during any development activity.
- But in practice defects do get introduced in almost every phase of the life cycle. Even defects may get at much later stage of the life cycle.
- So, solution of this problem is iterative waterfall model.
- Iterative waterfall model is by far the most widely used model. Almost every other model is derived from the waterfall model.
- The principles of detecting errors as close to its point of introduction as possible - is known as phase containment of error.
- Each successive various performing more useful work than previous versions



[Figure 1: Iterative Waterfall Model]

Advantages:

- Each successive various performing more useful work than previous versions.
- The core modules get tested thoroughly, thereby reducing chance of errors in final product.
- The model is more flexible and less costly to change the scope and requirements

2.3 Schedule Representation

- Generalized project scheduling tools and technique can be applied with little modification to software projects.
- Program Evolution and Review Techniques (PERT) and Critical Path Method (CPM) are two project scheduling method that can be applied to software development. Both techniques are driven by information already developed in earlier project planning activities.

- Estimate of effort.
- A decomposition of the product function.
- The selection of appropriate process model and task set.

[Table 1: Schedule Representation]

ACTIVITY	START DATE	FINISH DATE
Requirement Analysis		
System Analysis		
System Design		
System Coding		
Testing and Integration		

2.4 Feasibility Study

1. Technical Feasibility

- This includes the study of function, performance and restrictions that may affect the ability to achieve an efficient system.
- For this, we studied complete functionality to be provided in the system as per the needs of the user that in turn provides support for different platforms and a user-friendly environment.

2. Operational Feasibility

- The proposed system is completely Web based and users with little to no knowledge can easily go through the website.
- The proposed system will be beneficial only if it can be turned into a system which will meet the requirements of the user.

3. Economic Feasibility

- This is a very important aspect to be considered while developing a project. We decided the technology based on minimum possible cost factor.
- All hardware and software cost must be borne by the organization.

- We have estimated that the benefits the organization is going to receive from the proposed system will surely overcome the initial costs and later the running cost for the system.

4. **Environmental Feasibility**

- An evaluation of the probability that the organization has sufficient motivation to support the development and implementation of the application with necessary user participation, resources, training etc.

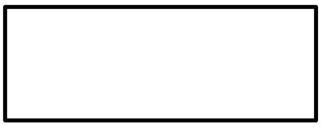
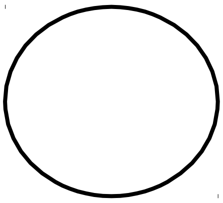
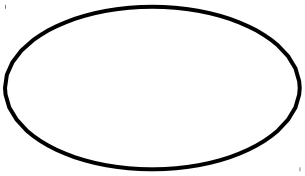
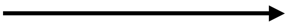
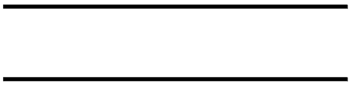
5. **Behavioural Feasibility**

- This includes how the system reacts and how it works.
- The system should be working such that all the functions react correctly.

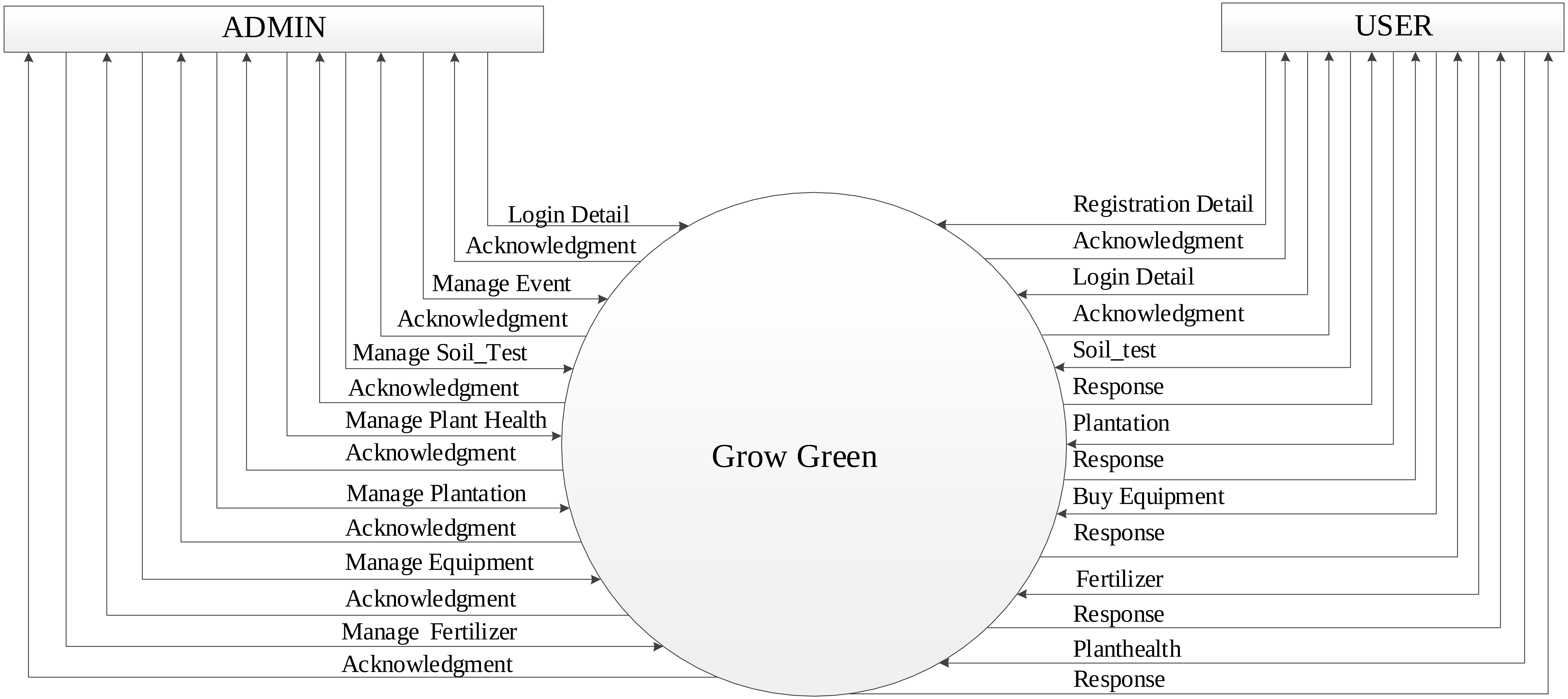
3.2 Data Flow Diagram

- DFD (data flow diagram) is also known as bubble chart or data flow graph.
- DFD’s are very useful in understanding the system and can be effectively used during analysis. It shows flow of data through a system visually. The DFD is a hierarchical graphical model of a system the different processing activities or functions that the system performs and the data interchange among these functions.
- It views a system as a function that transforms the inputs into desired output.
- Each function is considered as a process that consumes some input data and produces some output data.
- Function model can be represented using DFD.

[Table 2: Data Flow Diagram Symbols]

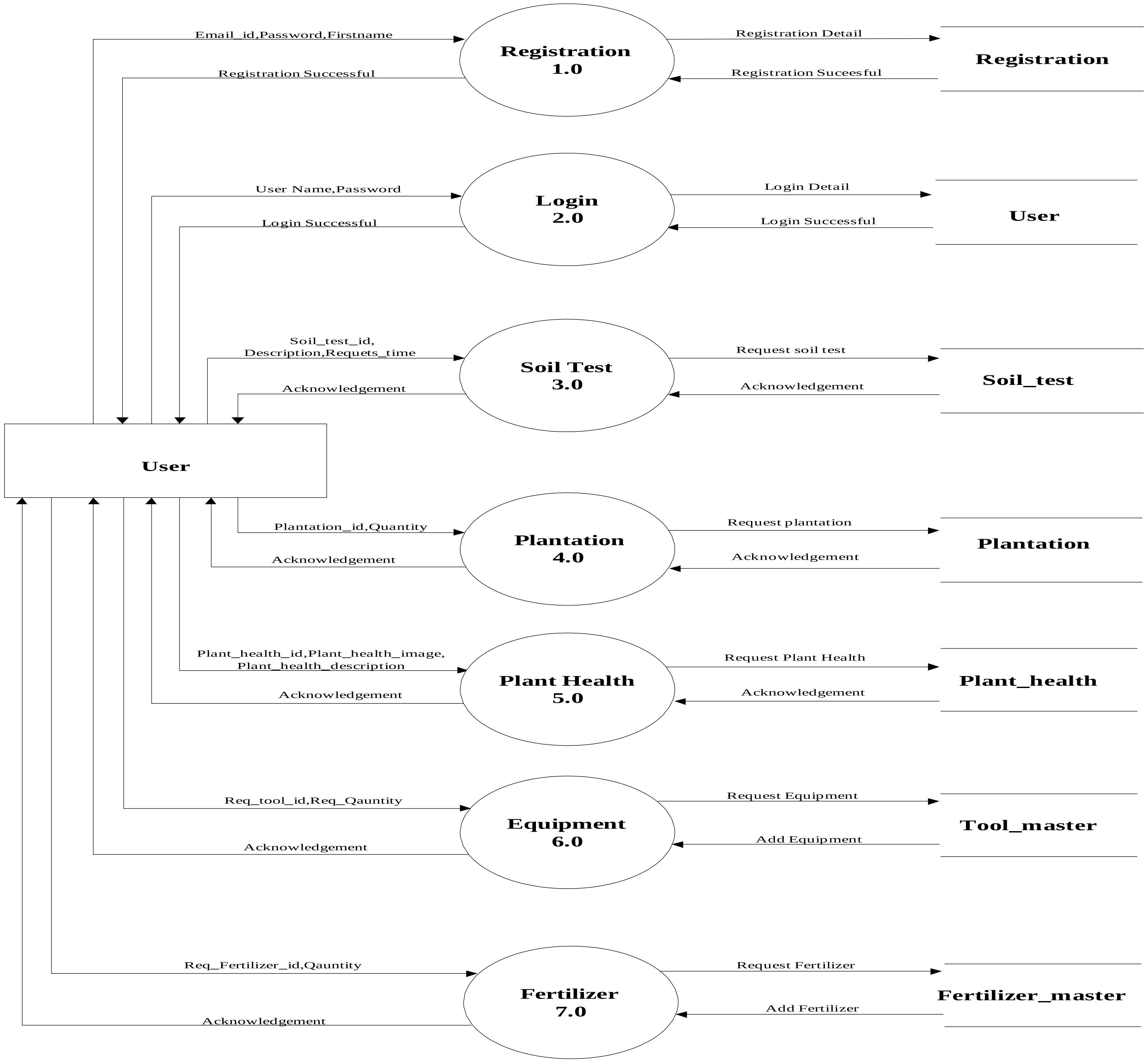
Symbols	Description
	**Entity:** Entities are external to the system which interacts by inputting the data.
	**System:** It shows the system name.
	**Process:** It shows the part of the system that transforms into outputs.
	**Data Flow:** It passes the data from one part to another.
	**Data Store:** Data store is represented by two parallel lines. It is generally logical file or database.

LEVEL 0 for Grow GREEN



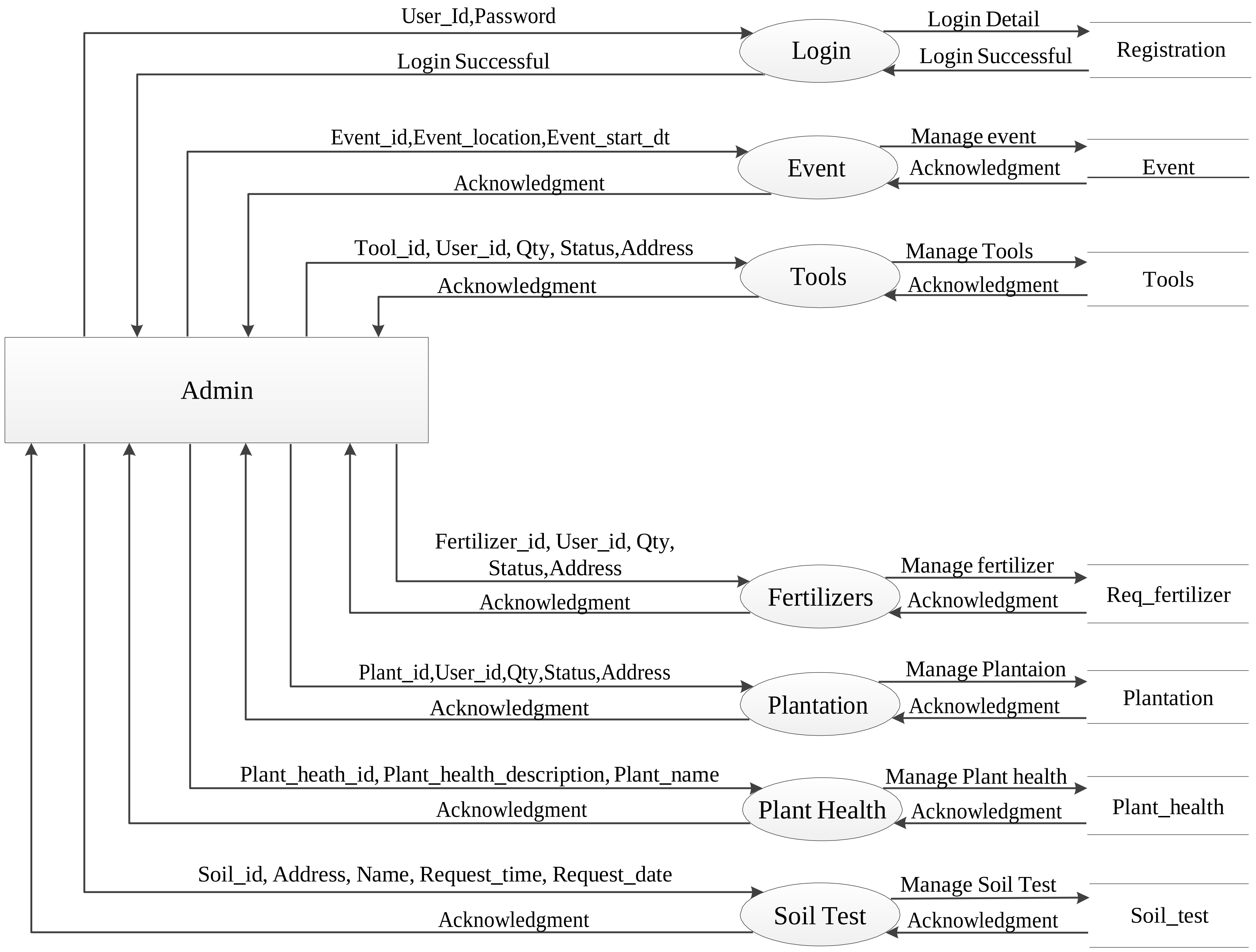
[Figure-3 DFD Level 0]

LEVEL 1 for USER:



[Figure-4 DFD Level User]

Level 1 for ADMIN

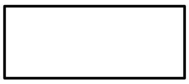
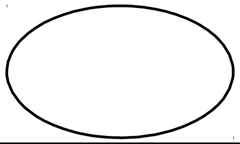
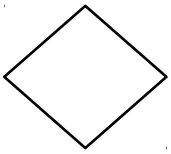
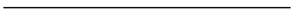
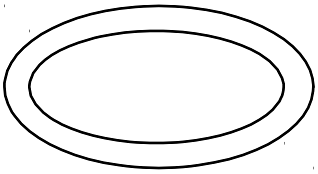


[Figure-5 DFD Level Admin]

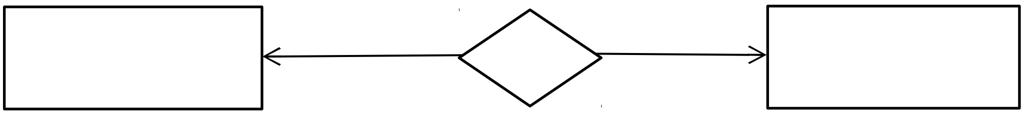
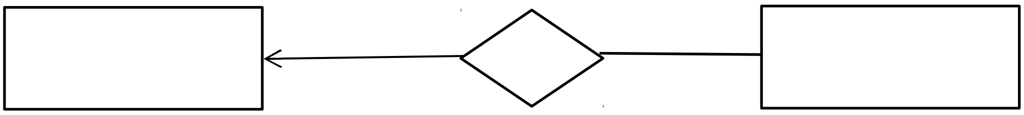
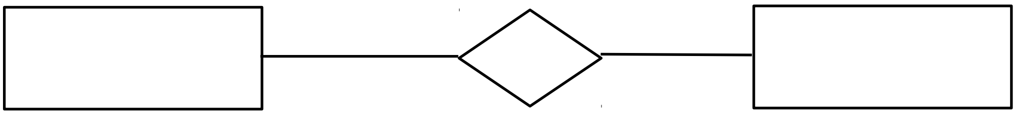
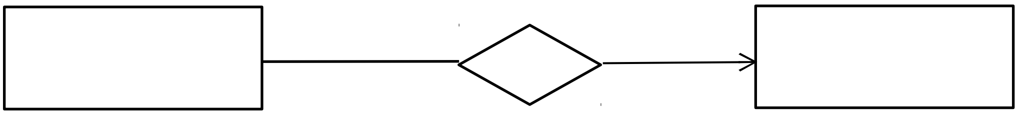
3.1 ER-Diagram

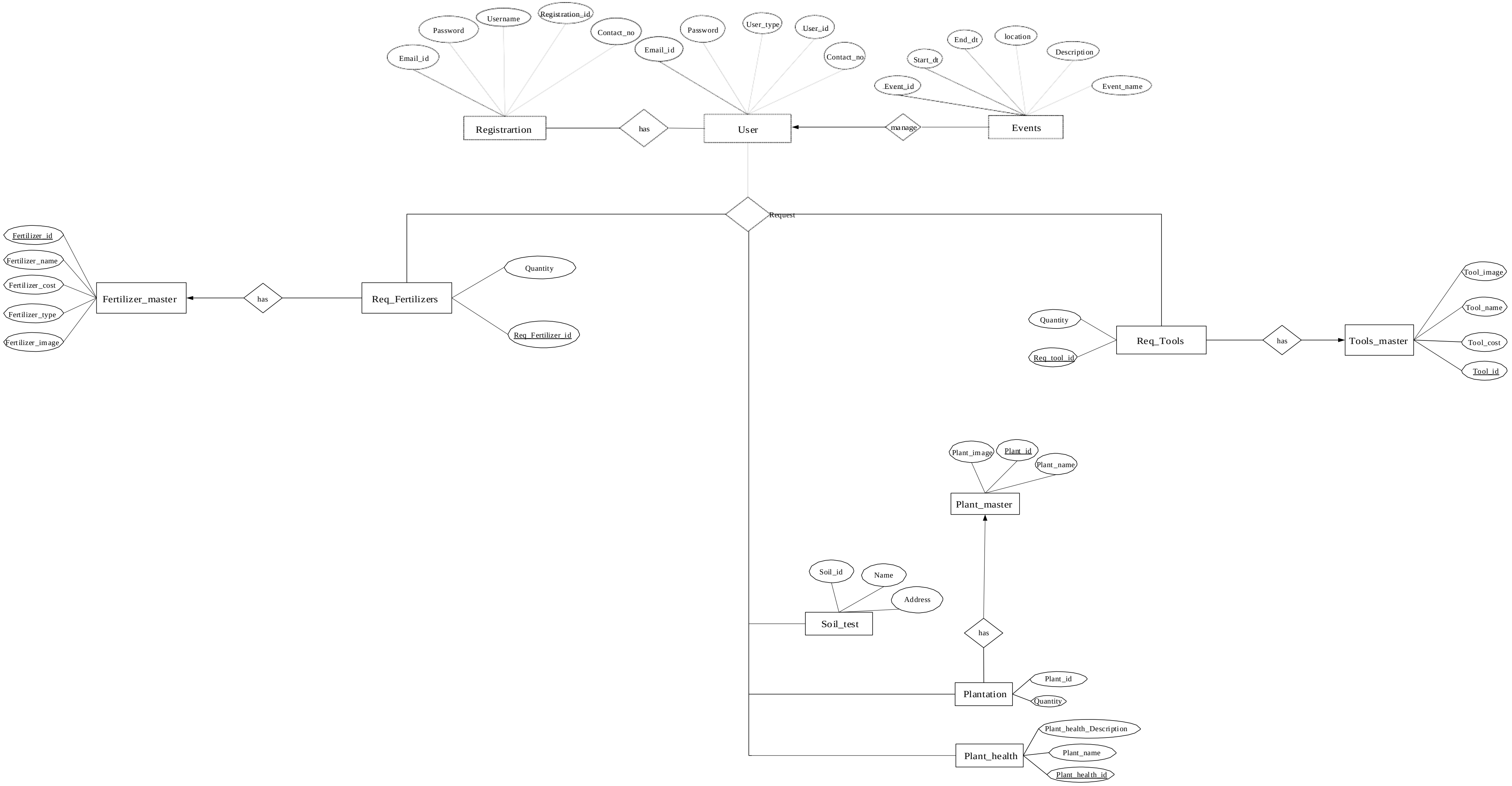
- An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research. Also known as ERDs or ER Models, they use a defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the interconnectedness of entities, relationships and their attributes. They mirror grammatical structure, with entities as nouns and relationships as verb

[Table-2 ER Diagram]

Symbols	Description
	**Entity:** Data object is real world entity or thing.
	**Attributes:** An attribute is property of characteristic of an entity.
	**Relationship:** Entity are connected each other via relations. Generally, relationships in binary because there are two entities are related to each other.
	**Data flow:** Link entity set to attributes & entity set to relationship.
	**Multivalued attributes:** Multivalued attributes are depicted by double ellipse.

[Table 3: Relationship of entities]

One to One	
One to Many	
Many to Many	
Many to One	



[Figure ER 2: Diagram]

Table: Registration

Primary key: Registration_id

[Table: 5: Registration]

Column name	Data-type	Size	Constraints	Description
Registration_id	Int	4	Primary key	To store register id
Name	Varchar	20	Not Null	To store Name
Email_id	Varchar	20	Not Null	To store Email id
Password	Varchar	10	Not Null	For Authentication
Contact_no	Int	10	Not Null	To store Contact-no
Username	Varchar	20	Not Null	To store Username
Address	Varchar	100	Not Null	To store address
Created_date	datetime	-	Not Null	To store check date
Created_by	datetime	-	Not Null	To store check by

Table: Plant_master

Primary key: Plant_id

Foreign key: User_id

[Table: 6: Plant_master]

Column name	Data-type	Size	Constraints	Description
Plant_id	Int	4	Primary key	To store plant id
Plant_cost	Int	20	Not Null	To store plant cost
Plant_name	Varchar	20	Not Null	To store plant name
Plant_image	Varchar	-	Not Null	To store plant image
Created_date	Varchar	10	Not Null	To store check date
Created_by	Varchar	10	Not Null	To store check by

Table: Tool_master

Primary: Key: Tool_id

Foreign: key: User_id

[Table 7: Tool_master]

Column name	**Data-type**	**Size**	**Constraints**	**Description**
Tool_id	Int	4	Primary Key	To store tool id
User_id	Int	4	Foreign Key	To manage user_id
Tool_image	Varchar	-	Not Null	To store tool image
Tool_name	Varchar	20	Not Null	To store tool name
Tool_cost	Varchar	10	Not Null	To store tool cost
Created_date	Varchar	10	Not Null	To store check date
Created_by	Varchar	10	Not Null	To store check by

Table: Fertilizer_master

Primary: Key: Fertilizer_id

Foreign: key: User_id

[Table 8: Fertilizer_master]

Column name	**Data-type**	**Size**	**Constraints**	**Description**
Fertilizer _id	Int	4	Primary Key	To store fertilizer id
User_id	Int	4	Foreign Key	To manage user_id
Fertilizer _image	Varchar	-	Not Null	To store fertilizer image
Fertilizer _type	Varchar	20	Not Null	To store fertilizer type
Fertilizer _name	Varchar	20	Not Null	To store fertilizer name
Fertilizer _cost	Varchar	10	Not Null	To store fertilizer cost
Created_date	Varchar	10	Not Null	To store check date
Created_by	Varchar	10	Not Null	To store check by

Table: Soil_test

Primary key: Soil_id

[Table 9: Soil_test]

Column name	Data-type	Size	Constraints	Description
Soil_id	Int	4	Primary Key	To Store id
Address	Varchar	500	Not Null	To store Address
Name	Varchar	50	Not Null	To Store name
Request_time	Varchar	50	Not Null	To Store Request time
Request_date	Varchar	50	Not Null	To Store Request date
Created_date	Varchar	10	Not Null	To store check date
Created_by	Varchar	10	Not Null	To store check by

Table: Plantation

Primary key: Id

Foreign key: Plant_id

[Table 10: Plantation]

Column name	Data-type	Size	Constraints	Description
Id	Int	4	Primary key	To store id
User_id	int	4	Not Null	To store id
Plant_id	Int	4	Foreign key	To store plant id
RequestDate	Varchar	50	Not Null	To store Request Date
Total	Varchar	50	Not Null	To store total no of plant
PaymentMode	Varchar	50	Not Null	To store payment mode
Quantity	Varchar	20	Not Null	To store Plant quantity
Address	Varchar	50	Not Null	To store Address
Contact_no	Varchar	10	Not Null	To store Contact-no
Status	Varchar	50	Not Null	To store Status
Created_date	Varchar	10	Not Null	To store check date
Created_by	Varchar	10	Not Null	To store check by

Table: Plant_health

Primary key: Plant_health_id

Foreign key: User_id

[Table 11: Plant_health]

Column name	**Data-type**	**Size**	**Constraints**	**Description**
Plant_health_id	Int	4	Primary key	To store plant health id
Plant_health_Description	Varchar	100	Not Null	To store plant status
Plant_name	Varchar	20	Not Null	To store name
Created_date	Varchar	10	Not Null	To store check date
Address	Varchar	50	Not Null	To store Address
Request_date	Varchar	50	Not Null	To store Request Date
Request_time	Varchar	50	Not Null	To store Request Time
Created_by	Varchar	10	Null	To store check by

Table: Req_Tools

Primary key: Req_tool_id

Foreign key: User_id, Tool_id

[Table 12: Req_Tools]

Column name	**Data-type**	**Size**	**Constraints**	**Description**
Req_tool_id	Int	4	Primary Key	To store req tool id
Tool_id	Int	4	Foreign Key	To store id
User_id	Int	4	Foreign Key	To store user_id
Quantity	varchar	20	Not Null	To Store quantity of tool
Created_date	Varchar	10	Not Null	To store check date
Created_by	Varchar	10	Not Null	To store check by

Table: Req_fertilizer

Primary key: Req_fertilizer_id

Foreign key: User_id, Fertilizer_id

[Table 13: Req_fertilizer]

Column name	**Data-type**	**Size**	**Constraints**	**Description**
Req_fertilizer_id	Int	4	Primary Key	To store id
Fertilizer_id	Int	4	Foreign key	To store id
User_id	Int	4	Foreign key	To store user_id
Quantity	varchar	20	Not Null	To store quantity of fertilizers
Created_date	Varchar	10	Not Null	To store check date
Created_by	Varchar	10	Not Null	To store check by

Table: Event

Primary key Event_id

[Table 14: Event]

Column name	**Data-type**	**Size**	**Constraints**	**Description**
Event_id	Int	4	Primary key	To store Event id
Event_name	Varchar	20	Not Null	To store name
Event_location	Varchar	100	Not Null	To store location
Event_Start_date	Date	8	Not Null	To store Start_date
Event_End_date	Date	8	Not Null	To store End_date
Event_Description	varchar	500	Not Null	To store Description
Created_date	Varchar	10	Not Null	To store check date
Created_by	Varchar	10	Not Null	To store check by

CHAPTER – 5

TECHNICAL SPECIFICATION

5.1 Hardware Specification

- **5.1.1 RAM:** 4 GB
- **5.1.2 Hard Drive Storage needed:** 10 GB
- **5.1.3 Other Hardware requirements:** No

5.2 Platform

- **5.2.1 Supported Operating System:** Window XP and above.

5.3 Framework

- **5.3.1 Markup Language:** HTML5
- **5.3.2 Programming Language:** PHP 7.3.9
- **5.3.3 Scripting Language:** Java

5.4 Technical Specification

- **5.4.1 Front-End:** HTML5, CSS 3
- **5.4.2 Back-End:** ASP.net 2017, SQL 5.7.23
- **5.4.3 IDE:** ASP.net 2017
- **5.4.4 UML Tools:** Microsoft Visio 2010 Professional Version.
- **5.4.5 SRS Tools:** Microsoft Word 2019 Professional Plus Version.

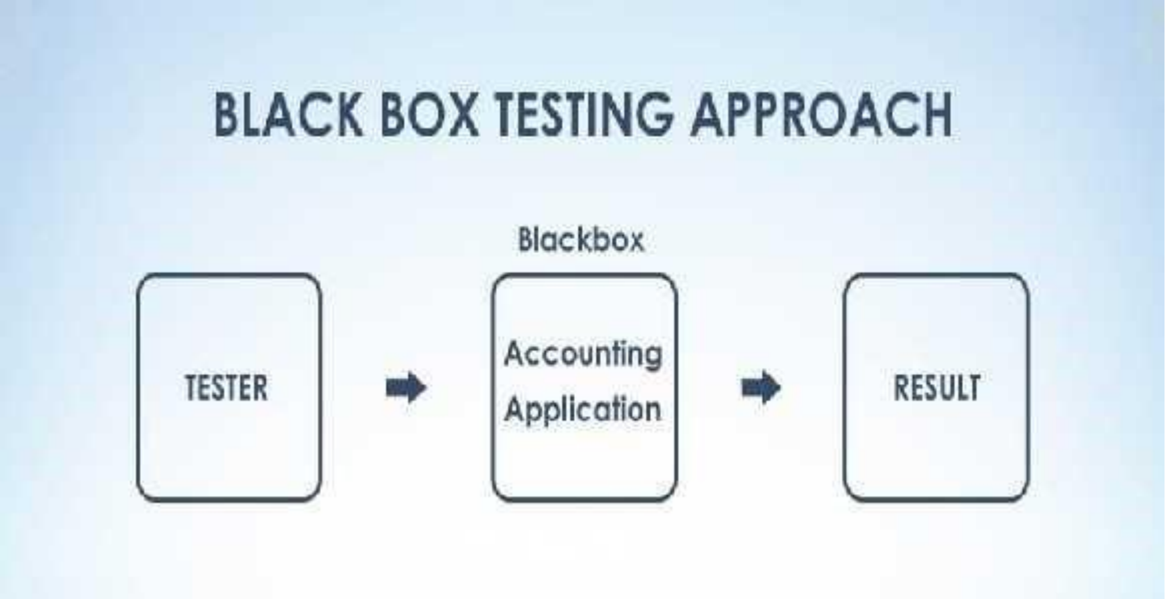
CHAPTER-6

TESTING

6.1 Testing Methods

6.1.1 Black Box Testing

Black Box Testing is also known as Behavioural Testing or Functional Testing. It is a technique of testing without having any knowledge of the internal working of the application. Black Box Testing treats the software as a "Black Box"- without any knowledge of internal working and it only examines the fundamental aspects of the system. This method of test can be applied to each and every level of software testing such as unit, integration, system and acceptance testing.



[Figure 6: Black Box Testing]

This method attempts to find errors in the following cases:

- Incorrect or missing functions
- Interface Errors
- Errors in structures or external database access
- Behaviour or performance errors
- Initialization and termination errors

Advantages:

- Unbiased tests because the designer and tester work independently.
- Tester is free from any pressure of knowledge of specific programming languages to test the reliability and functionality of an application / software.
- Test is performed from a user's point-of-view and not of the designer's.
- Test cases can be designed immediately after the completion of specifications.

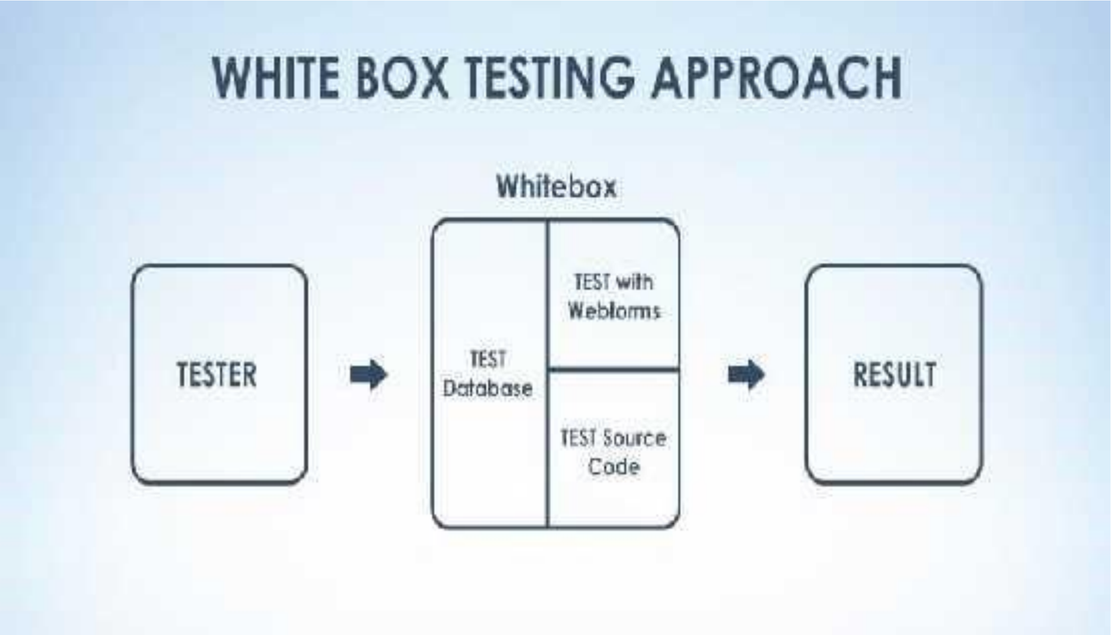
Disadvantages:

- Testing every possible input stream is not possible because it is time- consuming and this would eventually leave many program paths untested.
- Test cases are extremely difficult to be designed without clear and concise specifications.
- Results might be overestimated at time.
- Cannot be used for testing complex segments of code.

6.1.2 White Box Testing

White Box Testing is a software testing method in which the

internal structure/design/implementation of the item being tested is known to the tester. The tester chooses inputs to exercise paths through the code and determines the appropriate outputs. This method is named so because the software program, in the eyes of the tester, is like a white/transparent box; inside which one clearly sees.



[Figure 7: White Box Testing]

The aim of this testing is to investigate the internal logic and structure of the code. That is why white box testing is also known as Structural Testing.

Test Cases generated using White Box Testing can:

- Guarantee that all independent paths within a module have been exercised at least once.
- Exercise all decisions whether they are true or false.
- Exercise external data structure of the program.

Advantages:

- Code optimization by revealing hidden errors.
- Transparency of the internal coding structure which is helpful in deriving the type of input data needed to test an application effectively.
- Covers all possible paths of a code thereby, empowering a software engineering team to

conduct thorough application testing.

- Enables programmer to introspect because developers can carefully describe any new implementation.
- Gives engineering-based rules to stop testing an application.

Disadvantages:

- Since tests can be very complex, highly skilled resources are required with a thorough knowledge of programming and implementation.
- Test script maintenance can be a burden if the implementation changes too frequently.
- Necessity to create full range of inputs to test each path and condition make the white box testing method time-consuming.
- Exhaustive testing becomes even more complex using the white box testing method if the application is of large size.

Conclusion

- This report gives information about our project “GROW GREEN” features like soil testing, Plantation, buy Equipment’s, buy fertilizers and show events. It will be so much easy for Customer to plant the trees, buy Equipment’s, Fertilizers by using our website.

- This project will spread awareness about importance of the trees and planting them.

Bibliography

Book reference

[1] “Database System Concepts” by Abraham Silberschatz, Henry F. Korth & S. Sudarshan, McGraw Hill.

[2] “Software Engineering: A Practitioner’s Approach” by Roger S. Pressman.

[3] “MySQL Novice TO Ninja” by Kevin Yank.

[4] “C# in Depth, Fourth Edition” by Jon Skeet.

[5] “JavaScript: The Definitive Guide” by David Flanagan.

Web Reference

[1] https://www.w3schools.com/html/default.asp

[2] https://dev.mysql.com/doc/refman/8.0/en/

[3] https://en.wikipedia.org/wiki/Main_Page

[4] https://www.javatpoint.com/c-sharp-tutorial\

[5] http://www.ecoindia.com/flora/trees/