

E-FARM

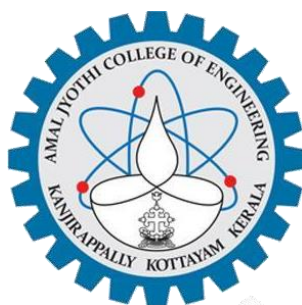
Project Report Submitted By

NEHA ALEX

Reg. No.: AJC20MCA-2055

In Partial fulfillment for the Award of the Degree Of

**MASTER OF COMPUTER APPLICATIONS (2 YEAR)
(MCA)
APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY**



**AMAL JYOTHI COLLEGE OF ENGINEERING
KANJIRAPPALLY**

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE,
Accredited by NAAC with 'A' grade. Koovappally, Kanjirappally, Kottayam, Kerala – 686518]

2020-2022

DEPARTMENT OF COMPUTER APPLICATIONS
AMAL JYOTHI COLLEGE OF ENGINEERING
KANJIRAPPALLY



CERTIFICATE

This is to certify that the Project report, “**E-FARM**” is the bonafide work of **NEHA ALEX (Reg.No.: AJC20MCA-2055)** in partial fulfillment of the requirements for the award of the Degree of Regular Master of Computer Applications under APJ Abdul Kalam Technological University during the year 2021-22.

Ms. Anit James

Internal Guide

Ms. Grace Joseph

Coordinator

Rev.Fr.Dr.Rubin Thottupurathu Jose

Head of the Department

External Examiner

DECLARATION

I hereby declare that the project report “**E-FARM**” is a bonafided work done at Amal Jyothi College of Engineering, towards the partial fulfilment of the requirements for the award of the Degree of Regular Master of Computer Applications (MCA) from APJ Abdul Kalam Technological University, during the academic year 2021-2022.

Date: 25/07/2022

NEHA ALEX

KANJIRAPPALLY

Reg. No: AJC20MCA-2055

ACKNOWLEDGEMENT

First and foremost, I thank God almighty for his eternal love and protection throughout the project. I take this opportunity to express my gratitude to all who helped me in completing this project successfully. It has been said that gratitude is the memory of the heart. I wish to express my sincere gratitude to our manager **Rev. Fr. Dr. Mathew Paikatt** and Principal **Dr. Lillykutty Jacob** for providing good faculty for guidance.

I owe a great depth of gratitude towards our Head of the Department **Rev.Fr.Dr. Rubin Thottupurathu Jose** for helping us. I extend my whole hearted thanks to the project coordinator **Rev.Fr.Dr. Rubin Thottupurathu Jose** and **Ms. Grace Joseph** for his valuable suggestions and for overwhelming concern and guidance from the beginning to the end of the project. I would also like to express sincere gratitude to my guide, **Ms. Anit James** for her inspiration and helping hand.

I thank our beloved teachers for their cooperation and suggestions that helped me throughout the project. I express my thanks to all my friends and classmates for their interest, dedication, and encouragement shown towards the project. I convey my hearty thanks to my family for the moral support, suggestions, and encouragement to make this venture a success.

NEHA ALEX

ABSTRACT

Despite the fact that agriculture is the most common occupation in India, most farmers are from the lower classes and live in abject poverty. Advanced techniques and automated machines that are taking the world to new heights are lagging behind when it comes to agriculture. Either lack of awareness of advanced equipment or non-availability lead to poverty in agriculture. Even after all the hard work and production of the farmers, in today's market, the farmers are cheated by the agents, leading to poverty. Where E-Farm may automate everything, by making it easy to serve as the best solution to all problems. E-Farm will serve as a way for farmers to sell their products across the country with just some basic knowledge of how to use websites.

.

CONTENT

Sl. No	Topic	Page No
1	INTRODUCTION	1
1.1	PROJECT OVERVIEW	2
1.2	PROJECT SPECIFICATION	2
2	SYSTEM STUDY	3
2.1	INTRODUCTION	4
2.2	EXISTING SYSTEM	5
2.3	DRAWBACKS OF EXISTING SYSTEM	5
2.4	PROPOSED SYSTEM	5
2.5	ADVANTAGES OF PROPOSED SYSTEM	5
3	REQUIREMENT ANALYSIS	6
3.1	FEASIBILITY STUDY	7
3.1.1	ECONOMICAL FEASIBILITY	7
3.1.2	TECHNICAL FEASIBILITY	8
3.1.3	BEHAVIORAL FEASIBILITY	8
3.2	SYSTEM SPECIFICATION	9
3.2.1	HARDWARE SPECIFICATION	9
3.2.2	SOFTWARE SPECIFICATION	9
3.3	SOFTWARE DESCRIPTION	9
3.3.1	PHP	9
3.3.2	MYSQL	10
4	SYSTEM DESIGN	12
4.1	INTRODUCTION	13
4.2	UML DIAGRAM	13
4.2.1	USE CASE DIAGRAM	14
4.2.2	SEQUENCE DIAGRAM	16
4.2.3	STATECHART DIAGRAM	19
4.2.4	ACTIVITY DIAGRAM	20
4.2.5	CLASS DIAGRAM	23
4.2.6	OBJECT DIAGRAM	24
4.2.7	COMPONENT DIAGRAM	25
4.2.8	DEPLOYMENT DIAGRAM	26

4.3	USER INTERFACE DESIGN	27
4.4	DATA BASE DESIGN	29
5	SYSTEM TESTING	36
5.1	INTRODUCTION	37
5.2	TEST PLAN	38
5.2.1	UNIT TESTING	38
5.2.2	INTEGRATION TESTING	39
5.2.3	VALIDATION TESTING	39
5.2.4	USER ACCEPTANCE TESTING	39
6	IMPLEMENTATION	45
6.1	INTRODUCTION	46
6.2	IMPLEMENTATION PROCEDURE	46
6.2.1	USER TRAINING	47
6.2.2	TRAINING ON APPLICATION SOFTWARE	47
6.2.3	SYSTEM MAINTENANCE	47
6.2.4	HOSTING	48
7	CONCLUSION & FUTURE SCOPE	49
7.1	CONCLUSION	50
7.2	FUTURE SCOPE	50
8	BIBLIOGRAPHY	51
9	APPENDIX	53
9.1	SAMPLE CODE	54
9.2	SCREEN SHOTS	66
9.3	PLAGIARISM REPORT	71

List of Abbreviation

IDE	-	Integrated Development Environment
HTML	-	Hyper Text Markup Language.
CSS	-	Cascading Style Sheet
SQL	-	Structured Query Language
UML	-	Unified Modeling Language

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

Despite the fact that agriculture is the most common occupation in India, most farmers are from the lower classes and live in abject poverty. Advanced techniques and automated machines that are taking the world to new heights are lagging behind when it comes to agriculture. Either lack of awareness of advanced equipment or non-availability lead to poverty in agriculture. Even after all the hard work and production of the farmers, in today's market, the farmers are cheated by the agents, leading to poverty. Where E-Farm may automate everything, by making it easy to serve as the best solution to all problems. E-Farm will serve as a way for farmers to sell their products across the country with just some basic knowledge of how to use websites.

1.2 PROJECT SPECIFICATION

The proposed system is a website on which farmers can sell and buyers can buy farm vegetables and fruits by logging into the site by registering on it.

The system includes 3 modules. They are:

1. Admin Module

Admin must have a login into this system. He has overall control of the system. Admin can manage users, and products of the site by defining category with particular subcategory. And also, he can view the feedbacks of the buyers with the details of the farmer that they purchased the product on the site.

2. Farmer Module

The farmer can fill the registration form and get his credentials. After that he have to upload the id-proof of him/her in order to add the products that he wants to sell on the website. And the farmer will be one who is fixing the rates of the products that he adding on the site as per quantity. Also, he can have a view and update action on the products that he added.

3. Buyer Module

Buyers can sign-up into site by entering the email-id and password after the registration. And when they have signed in on the site, they will be having product view with cart management which leads them to payment and order summary evaluation of their purchased products, and when the product delivered, they can write their comments about it to the administrator of the site by uploading their order summary.

CHAPTER 2

SYSTEM STUDY

2.1 INTRODUCTION

System analysis is the process of gathering and interpreting facts, diagnosing problems, and information to recommend system improvements. It is a problem-solving activity that requires intensive communication between system users and system developers. System analysis or study is an important phase of any system development process.

The current system study found that there isn't a computerised system in place to help farmers market their farm products. At the moment, a farmer visits the closest market and gives his stuff to a certain representative. The agent instructs the farmer to come back to the market after a predetermined amount of time to receive the money made from the sale of the goods. The product is sold by an agent to a different agent or dealer for the going rate. Each agent makes an effort to reduce their commission from it. The farmer has no access to information about the market or the precise price at which their produce was sold. There is no openness. Farmers lack the chance to research product prices in various markets where they can sell their wares to make a lot of money.

Where E-Farm resolves a way for the farmers to sell their products across them just with some basic knowledge about how to use the website. E-Farm is an web application that will help the farmers to perform the agro-marketing leading to achieve success for their standard of living. In here all a farmer has do is to fill the registration form and get his credentials for logging in into the system. After that he have to upload the id-proof of him/her in order to add the products that he wants to sell on the website. And in E-Farm farmer will be one who is fixing the rates of the products that he/she adding on the site as per quantity. Also, he can have a view and update action on the products that he added, and also, he /she can generate report on their sold products when they needed.

2.2 EXISTING SYSTEM

There is no computerised system for farmers to market their produce. Currently, a farmer goes to the nearest market and hands over his product to a particular agent. The agent asks the farmer to visit the market after a certain period of time to collect the cash earned from the product sold. An agent sells the product to another agent or dealer at the market price. Every agent tries to cut their commission from it. There is no way for the farmer to learn about the trade and the exact amount their product was sold for. There is no transparency. Farmers do not have the opportunity to know the prices of the products in different markets where they can sell their products to earn high profits. Farmers are often not even aware of the schemes and compensation provided by the government. Despite all the opportunities that are banging at the door, farmers are unable to capitalise on them. The current system does not provide a means of e-learning for farmers to provide knowledge about new techniques in agriculture. So, he will not get the maximum profit through the current system.

2.3 DRAWBACKS OF EXISTING SYSTEM

- No proper online management of system
- Human effort is needed.

2.4 PROPOSED SYSTEM

The proposed system is designed to provide all the advantages to farmers. The proposed system is more efficient and better than the existing system. It is designed to overcome all the drawbacks of the present system and to provide a permanent solution to them. The primary aim of the new system is to speed up the transaction.

2.5 ADVANTAGES OF PROPOSED SYSTEM

The system is very simple in design and to implement. The system requires very low system resources, and the system will work in almost all configurations. It has got following features:

- In the proposed system the farmers can add their farm vegetables and fruits behalf of going for market handover to shopkeeper.
- And also, in this system the farmer can decide his/her product price as per quantity they are adding.
- In case of buyers, they can choose the farmers product purchase based on the location preference and price.
- And in this system the farmer will be getting paid directly after the delivery of his/her product to the particular buyer.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDY

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spend on it. The developer can forecast the project's usefulness and potential future thanks to a feasibility study. The viability of the system concept, which includes its impact on the organization, capacity to satisfy user needs, and efficient use of resources, serves as the foundation for a feasibility study. As a result, a feasibility evaluation is frequently performed before a new application is approved for development. The document outlines the project's viability and contains a number of factors that were carefully taken into account throughout this project's feasibility study, Including such as Technical, Economic and Operational feasibilities. It has the following characteristics:

3.1.1 Economical Feasibility

Cost and benefit analyses are required to support the emerging system. criteria to make sure that focus is placed on the project that will yield the best results the earliest. The price that would be involved in developing a new system is one of the variables.

Some significant financial concerns raised during the initial study include the following:

- The costs conduct a thorough system investigation.
- The cost of the hardware and software.
- The benefits manifest themselves as lower costs or less costly errors.

There are no manual costs involved with the suggested system because it was developed as part of a project. Additionally, the fact that all of the resources are already at hand indicates that the system may be developed affordably.

The cost of the project was divided according to the system used, its development cost, and the cost of hosting the project. According to all the calculations the project was developed at a low cost. As it is completely developed using open-source software.

3.1.2 Technical Feasibility

The system must first undergo a technical evaluation. The assessment of its viability must be built upon an overview design of the system's requirements in terms of input, output, programmers, and procedures. The inquiry must next advise the kind of equipment, necessary procedure for constructing the system, and means of operating the system once it has been designed after having identified an outline system. The following technical difficulties came up throughout the investigation:

- Does the proposed technology function with existing technology?
- If the system has been developed, can it expand?

The project should be built in such a way that the required performance and functionality are met within the limits. The system may still be used even though the technology may become outdated after a while because a newer version of the same software still works with an earlier version. Therefore, this project only has a few limitations. The system was developed with PHP for the front end and a MySQL server for the back end; the project can be finished technically. The system was developed with PHP for the front end and a MySQL server for the back end; the project can be finished technically. The system was effective in terms of processor performance.

3.1.3 Behavioral Feasibility

The proposed system includes the following questions:

- Are the users receiving enough support?
- Will anyone be harmed by the proposed system?

The project would be advantageous because, when created and implemented, it would achieve the goals. The project is deemed to be behaviorally feasible after carefully weighing all behavioral factors.

3.2 SYSTEM SPECIFICATION

3.2.1 Hardware Specification

Processor - Intel core i3

RAM - 4 GB

Hard disk - 1 TB

3.2.2 Software Specification

Front End - HTML, CSS

Backend - MYSQL

Client on PC - Windows 7 and above.

Technologies used - JS, HTML5, Bootstrap, J Query, PHP, CSS

3.3 SOFTWARE DESCRIPTION

3.3.1 PHP

PHP is a server-side scripting language used for both web development and general-purpose programming. PHP is now used by 2.1 million web servers and more than 244 million webpages. The PHP group now produces the reference implementation of PHP, which was first developed by Rasmus Lerdorf in 1995. PHP, a recursive acronym that was once meant for personal pages, now stands for PHP-hypertext preprocessor. A web server's PHP processor module interprets PHP code to produce the final web page. Instead of calling an external file, PHP commands can be directly put into an HTML source file to handle data. Additionally, because of restrictions on the usage of the name PHP under the GNU General Public License (GPL), it has evolved to now give a command-line interface and be used in standalone programs. Most web servers support the free deployment of PHP, which is also available as a standalone shell on practically all platforms and operating systems.

3.3.2 MySQL

Oracle Corporation created, distributed, and provided support for MySQL, the most well-known Open-Source SQL database management system. The most recent details regarding MySQL software are available on the MySQL website.

- **MySQL is a database management system.**

A planned collection of data is called a database. Anything might be it, including a straightforward shopping list, a photo gallery, or the enormous amount of data in a company network. Data included in a computer database must be added to, accessed, and processed using a database management system, such as MySQL Server. Because computers are so good at processing enormous amounts of data, database management systems whether employed as standalone programs or as a component of other applications are crucial to computing.

- **MySQL databases are relational.**

Instead of placing all the data in one huge warehouse, a relational database keeps the data in individual tables. Physical files that are designed for speed contain the database structures. A flexible programming environment is offered by the logical model, which includes objects like databases, tables, views, rows, and columns. One-to-one, one-to-many, unique, mandatory, optional, and "pointers" between distinct tables are just a few examples of the rules you may make to control the relationships between various data fields. Since a well-designed database upholds these constraints, your application won't ever run into inconsistent, duplicate, orphan, out-of-date, or missing data. The prefix "SQL" in MySQL stands for "Structured Query Language." SQL is the most widely used standard language for database access. Depending on your programming environment, you might openly enter SQL (for example, to generate reports), embed SQL statements within other languages' code, instead, employ a language-specific API that hides the SQL coding. By way of the ANSI/ISO SQL Standard, SQL is defined. Since its inception in 1986, the SQL standard has undergone multiple revisions. The 1992 standard is referred to in this document as "SQL92," the 1999 standard is referred to as "SQL," and the most recent version of the standard is referred to as "SQL: 2003." "The SQL standard" refers to the SQL Standard as it is at any given time.

- **MySQL software is Open Source.**

Anyone can use and modify the software because it is open source. The MySQL software is available for free download and usage any person online. You have the right to look at the source code and make any necessary changes. The GPL (GNU General Public License), which specifies what you may and cannot do with the programmed under particular circumstances, is followed by the MySQL software. You can purchase a commercially licensed version from us if the GPL makes you uncomfortable or if you need to integrate MySQL code into a for-profit application. For further details, see the MySQL Licensing Overview.

- **The MySQL Database Server is very fast, reliable, scalable, and easy to use.**

If it is what you're after, you should try it. MySQL Server may run easily on a desktop or laptop and needs little to no maintenance in addition to your other apps, web servers, and other software. If you dedicate an entire system to MySQL, you can change the settings to utilize all the RAM, CPU, and I/O power.

- **MySQL Server works in client/server or embedded systems.**

A multi-threaded SQL server, several client programmers and libraries, management tools, and a wide variety of application programming interfaces (APIs) make up the client/server system known as the MySQL Database Software. Additionally, we provide MySQL Server as a built-in multi-threaded library that you can link into your programmed to create a standalone solution that is more manageable, speedy, and simple to use.

CHAPTER 4

SYSTEM DESIGN

4.1 INTRODUCTION

Any engineered system or product's development process begins with design. A creative process is designed. The secret to an efficient system is a decent design. "Design" is the process of using many approaches and concepts to thoroughly outline a process or a system so that it can be physically implemented. One way to describe it is as the process of using different methodologies and concepts to specify a device, a process, or a system in enough detail to allow for its physical reality. Regardless of the development paradigm that is employed, software design forms the technical core of the software engineering process. The architectural detail needed to construct a system or product is developed through the system design. This program has also through the best possible design phase, fine-tuned all efficiency, performance, and accuracy levels, as in the case of any systematic technique. During the design stage, a user-oriented document is transformed into a document for programmers or database employees. The two stages of system design development are logical design and physical design.

4.2 UML DIAGRAM

Software system artefacts can be specified, visualized, built, and documented using the UML standard language. The Object Management Group (OMG) is the organization that developed UML, and the OMG received a draught of the UML 1.0 definition in January 1997.

Unified Modeling Language is known as UML. Compared to other popular programming languages like C++, Java, COBOL, etc., UML is unique. A visual language called UML is used to create software blueprints. A general-purpose visual modeling language for software system visualization, specification, construction, and documentation is what UML is known as. UML is not just used to represent software systems, despite the fact that this is its most common application. It is also used to model systems that are not software-based. For instance, the manufacturing facility's process flow, etc. Although UML is not a programming language, tools can be used to generate code using UML diagrams in a variety of languages. The analysis and design of object-oriented systems are directly related to UML. It has been so well standardized that OMG now recognizes UML as a standard. A complete UML diagram, which depicts a system, is created using all the elements and linkages. The most crucial aspect of the entire procedure is the UML diagram's aesthetic impact. It is completed by using all the additional components. The following nine diagrams are part of UML.

- Class diagram
- Object diagram
- Use case diagram
- Sequence diagram
- Activity diagram
- State chart diagram
- Deployment diagram
- Component diagram

4.2.1 USE CASE DIAGRAM

An illustration of the interactions between system components is a use case diagram. A use case is a method for locating, defining, and arranging system needs. The word "system" in this context refers to a thing that is being built or operated, such as a website for mail-order service and product sales. Use case diagrams are used in UML (Unified Modeling Language), a standard language for modeling actual items and systems.

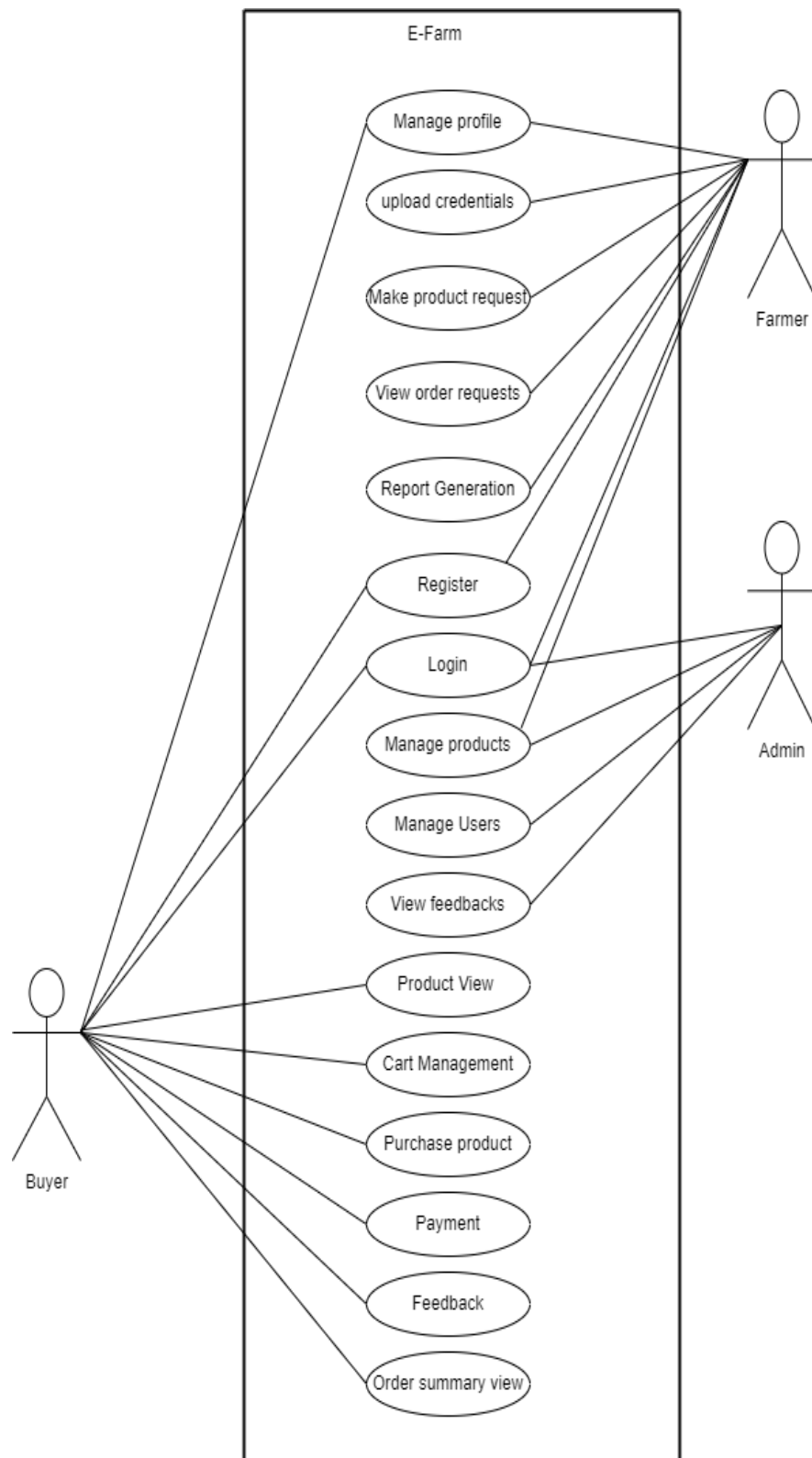
Planning general requirements, validating hardware designs, testing and debugging software products while they are still in development, creating online help resources, and finishing customer support-focused tasks are a few examples of system objectives. For instance, customer support, item ordering, catalog updating, and payment processing are examples of use cases in a setting of product sales. A use case diagram consists of four components.

- The line separating the system of interest from the environment around it.
- The actors are often participants in the system who are identified by their roles.
- The actors within and surrounding the system perform the use cases.
- How the actors and use cases are related to one another and dependent on one another.

Use case diagrams are made to show the functional requirements of a system. To create an effective use case diagram after identifying the aforementioned things, we must adhere to the following rules.

- A use case's name should be selected in a way that makes it clear what functions are being performed.
- Give actors a name that fits them.
- Clearly depict links and dependencies in the diagram.
- Since the primary goal of the diagram is to define the needs, avoid attempting to include all possible relationships.

- Use notes whenever appropriate to clarify some crucial aspects.



4.2.2 SEQUENCE DIAGRAM

A sequence diagram essentially shows how things interact with one another sequentially, or the order in which these interactions occur. A sequence diagram can also be referred to as event diagrams or event scenarios. Sequence diagrams show the actions taken by the components of a system in chronological order. Businesspeople and software engineers frequently use these diagrams to record and comprehend the requirements for new and current systems.

Sequence Diagram Notations –

- i. **Actors** – In a UML diagram, an actor represents a particular kind of role in which it communicates with the system's objects. An actor is always beyond the purview of the system that we want to use the UML diagram to represent. We employ actors to portray a variety of roles, including those of human users and other outside subjects. In a UML diagram, an actor is represented using a stick person notation. A sequence diagram may contain several actors.
- ii. **Lifelines** – A named piece that shows a specific participant in a sequence diagram is called a lifeline. In essence, a lifeline represents each incident in a sequence diagram. The lifeline components in a sequence diagram are at the top.
- iii. **Messages** – Using messages, communication between objects is demonstrated. The messages are displayed on the lifeline in chronological sequence. Arrows are how messages are represented. A sequence diagram's main components are lifelines and messages.

Messages can be broadly classified into the following categories:

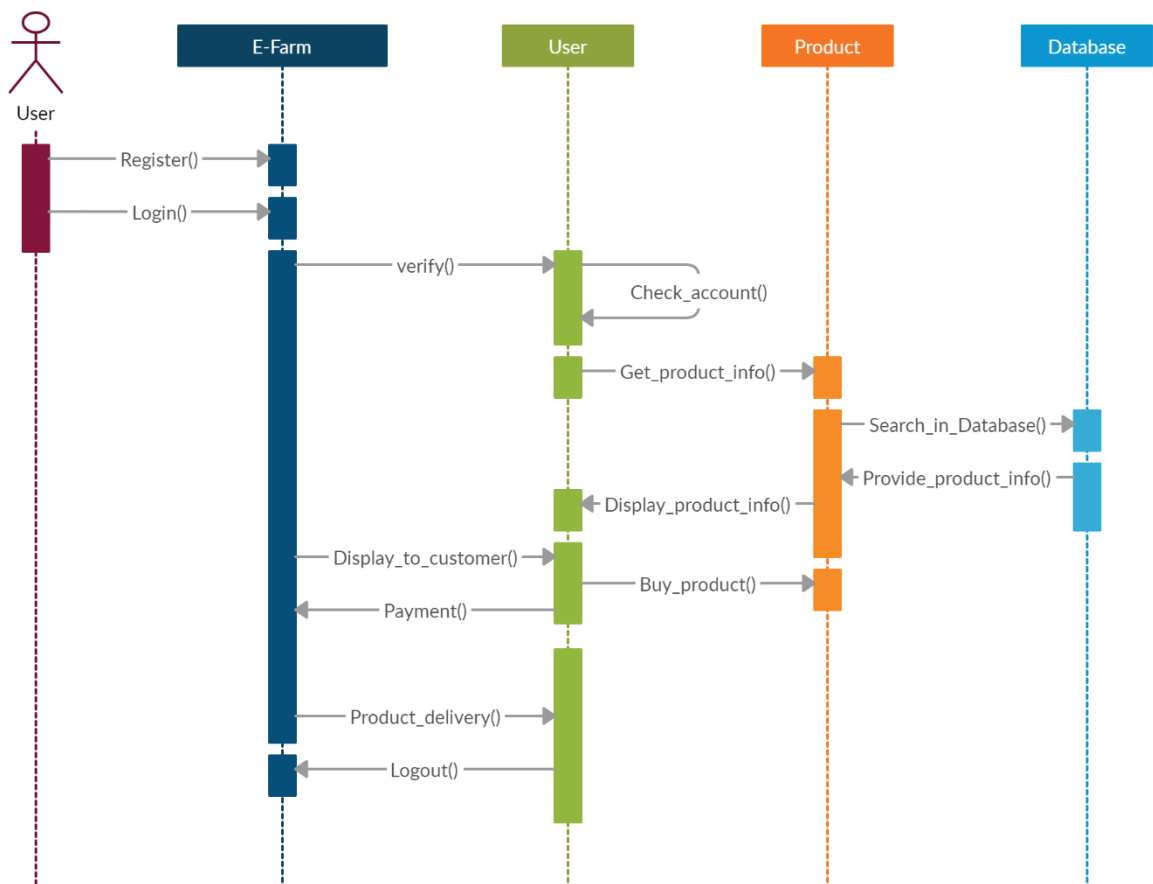
- **Synchronous messages**
- **Asynchronous Messages**
- **Create message**
- **Delete Message**
- **Self-Message**
- **Reply Message**
- **Found Message**
- **Lost Message**

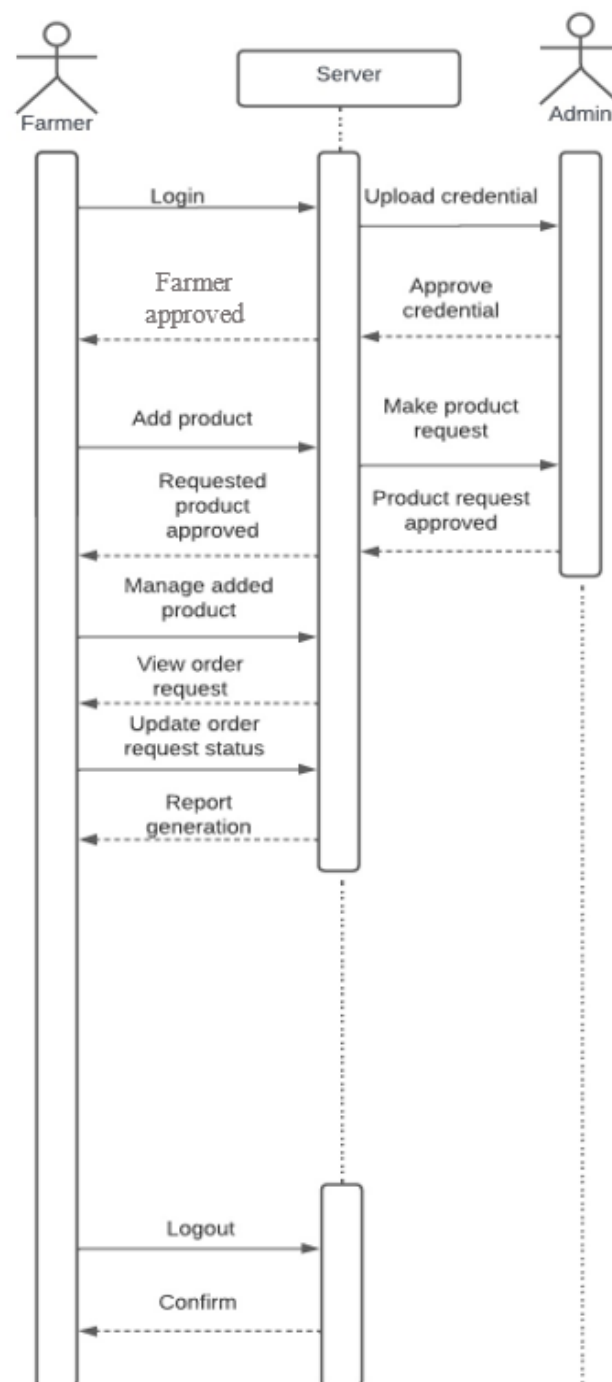
- i. **Guards** – In the UML, we utilise guards to model circumstances. When we need to limit the flow of messages under the guise of a condition being met, we use them. Software engineers rely on guards to inform them of the limitations imposed by a system or specific process.

Uses of sequence diagrams –

- Used to illustrate the reasoning behind a complex function, process, or procedure.
- They're also employed to display the specifics of UML use case diagrams.
- Used to comprehend the precise operation of present or upcoming systems.
- Imagine the flow of information between different system elements or objects.

Buyer Sequence Diagram:



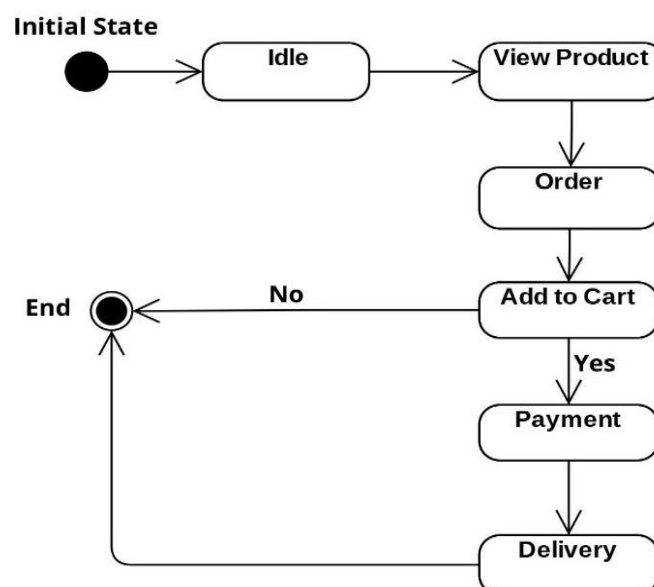
Farmer Sequence Diagram:

4.2.3 STATECHART DIAGRAM

It describes various system components' statuses. The states are unique to a particular system object or component. A state machine is described in a Statechart diagram. A state machine is a device that creates various states for an object and controls these states through external or internal events. Throughout an object's existence, they define several states, and these states are altered by events. The reactive systems can be modelled with statechart diagrams. A system that reacts to internal or external events is known as a reactive system. The transfer of control from one state to another is depicted in a statechart diagram. States are described as a situation in which an object is present and changes in response to an event. The main goal of a statechart diagram is to model an object's lifetime from creation to destruction. For both forward and backward engineering of a system, statechart diagrams are employed. But modelling the reactive system is the fundamental objective.

Following are the main purposes of using Statechart diagrams –

- To model the dynamic aspect of a system.
- To model the lifetime of a reactive system.
- To describe different states of an object during its lifetime.
- Define a state machine to model the states of an object.

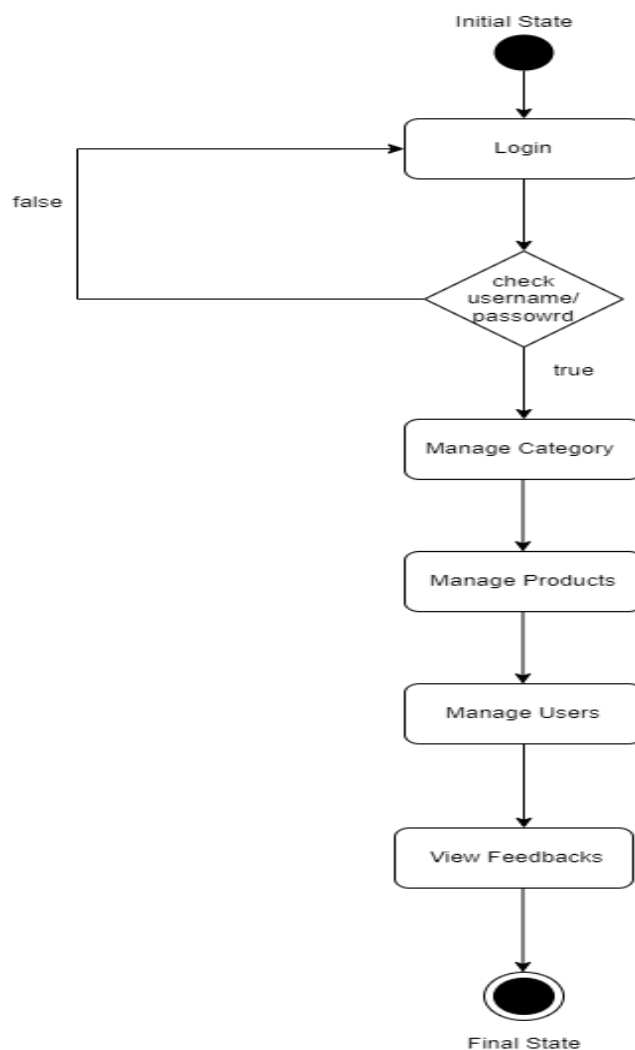


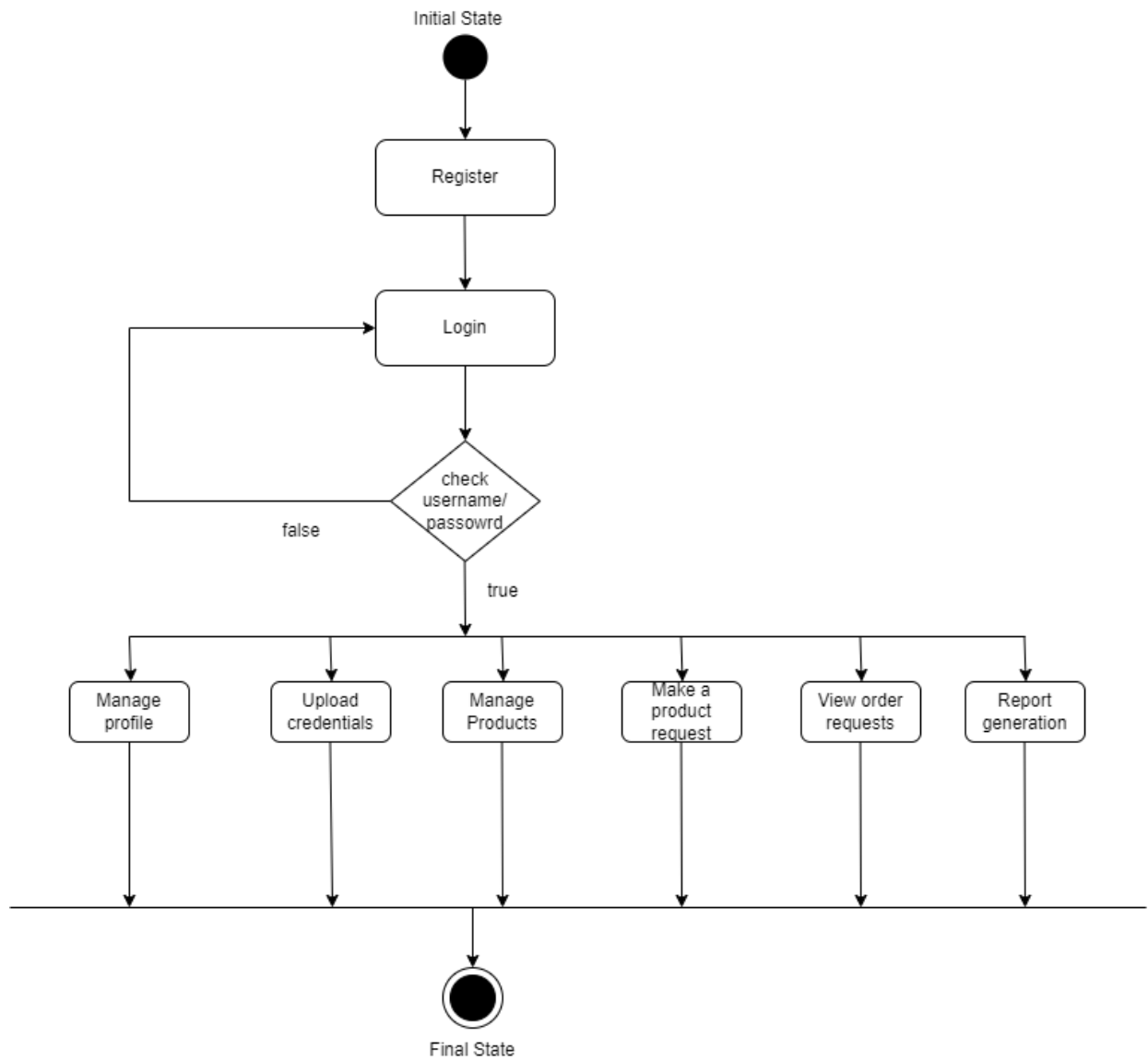
4.2.4 ACTIVITY DIAGRAM

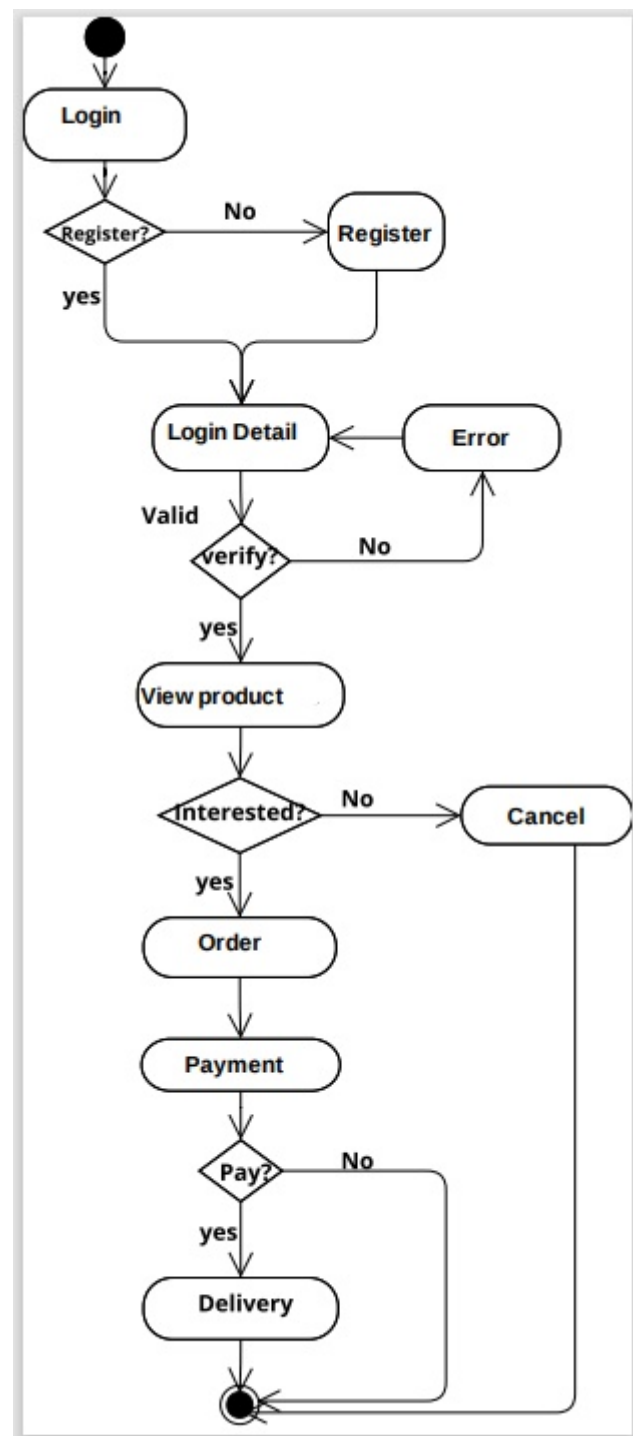
The activity diagram describes the system's dynamic properties. A flowchart that illustrates how one activity leads to another is an activity diagram. A system operation could be used to describe the action. In the control flow, one action leads to the next. This flow may be concurrent, parallel, or branching. Many features, including fork, join, and others, are used in activity diagrams to handle various sorts of flow control.

Activity diagrams are used to build the executable system utilising forward and reverse engineering approaches, as well as to visualise the dynamic nature of a system. The message portion is the only item the activity diagram is missing. No message flow from one activity to another is shown. Occasionally, an activity diagram is used in place of a flowchart. The diagrams are not flowcharts, despite their appearance.

Admin Activity Diagram:

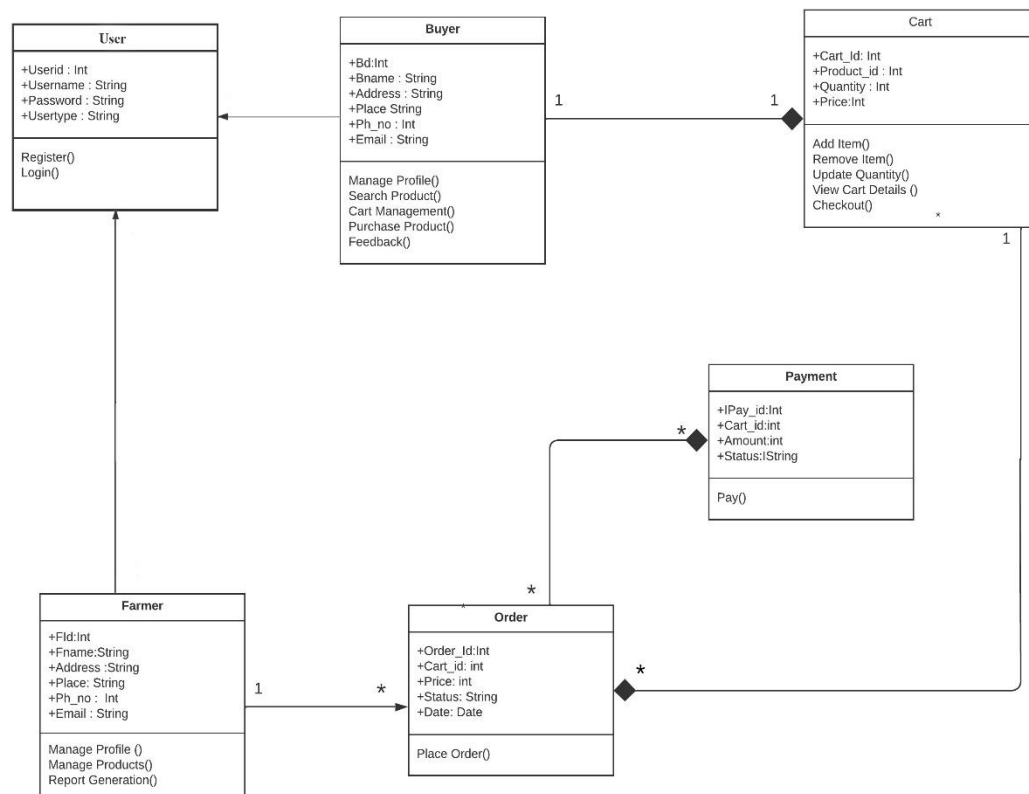


Farmer Activity Diagram:

Buyer Activity Diagram:

4.2.5 CLASS DIAGRAM

Static diagrams include class diagrams. It represents the application's static view. Class diagrams are used to create executable code for software applications as well as for visualizing, explaining, and documenting various elements of systems. The characteristics and functions of a class are described in a class diagram, along with the restrictions placed on the system. Because they are the only UML diagrams that can be directly mapped with object-oriented languages, class diagrams are frequently employed in the modelling of object-oriented systems. A collection of classes, interfaces, affiliations, collaborations, and constraints are displayed in a class diagram. It also goes by the name "structural diagram."

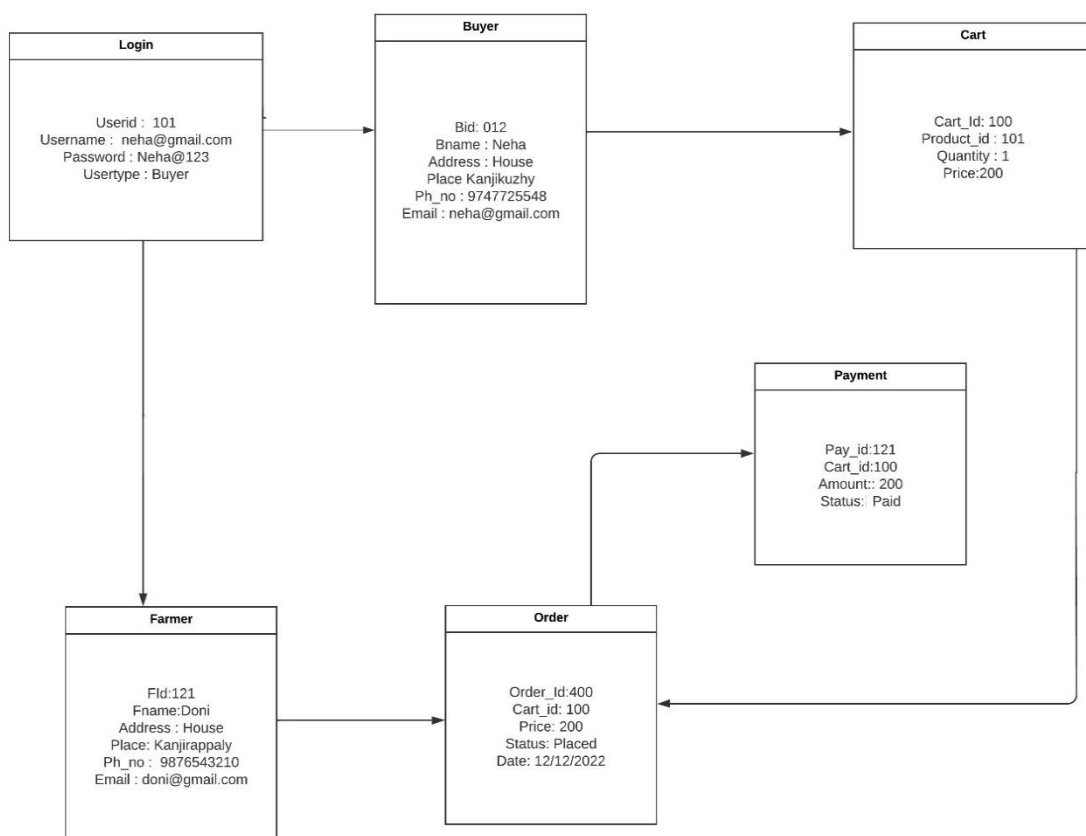


4.2.6 OBJECT DIAGRAM

Class diagrams are a requirement for object diagrams because they are the source of class diagrams. An object diagram illustrates a specific instance of a class diagram. Diagrams of classes and objects share the same underlying concepts. Object diagrams can also be used to describe a system's static view, which is a brief snapshot of the system. Object diagrams are used to illustrate a collection of items and their connections as an example.

The object diagram's goal can be summed up as:

- Reverse and forward engineering.
- System object associations.
- A still image of a dialogue.
- From a practical perspective, understand object behaviour and their relationship.

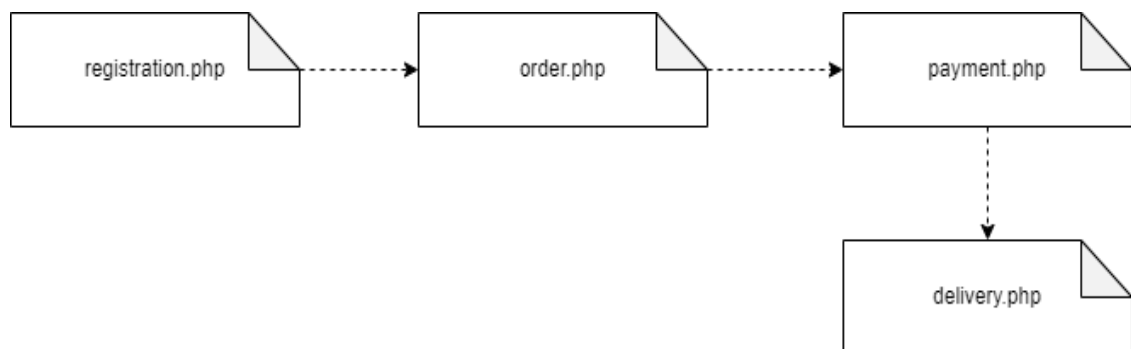


4.2.7 COMPONENT DIAGRAM

A specific type of UML diagram is a component diagram. In addition, the goal is distinct from the previous diagrams mentioned. Instead, than describing the system's functioning, it discusses the parts that go into creating it. Therefore, from that perspective, component diagrams are utilized to represent the actual physical parts of a system. These parts include files, libraries, and packages, among others. Another way to think of component diagrams is as a static implementation perspective of a system. Static implementation depicts how the components are arranged at a specific time. A group of diagrams are utilized to portray the overall system because a single component diagram is unable to do so.

The component diagram's goal can be summed up as:

- Identify the parts of a system visually.
- Utilize both forward and reverse engineering to build an executable.
- Describe how the components are arranged and their connections.



4.2.8 DEPLOYMENT DIAGRAM

The topology of the physical parts of a system, where the software components are installed, is depicted using deployment diagrams.

The static deployment view of a system is described using deployment diagrams. Nodes and their connections are the main components of deployment diagrams.

The aim of the diagram is explained by the word "deployment" itself. For depicting the physical components where software components are delivered, deployment diagrams are employed. Deployment diagrams and component diagrams have a lot in common.

UML is primarily made to concentrate on a system's software artefacts. These two diagrams, however, are unique ones that highlight the hardware and software components.

Deployment diagrams are meant to concentrate on the hardware topology of a system, whereas most UML diagrams are used to handle logical components.

Deployment diagrams serve the following purposes:

- Visualize a system's hardware topology.
- What hardware elements are needed to install software components
- The runtime processing nodes in detail.

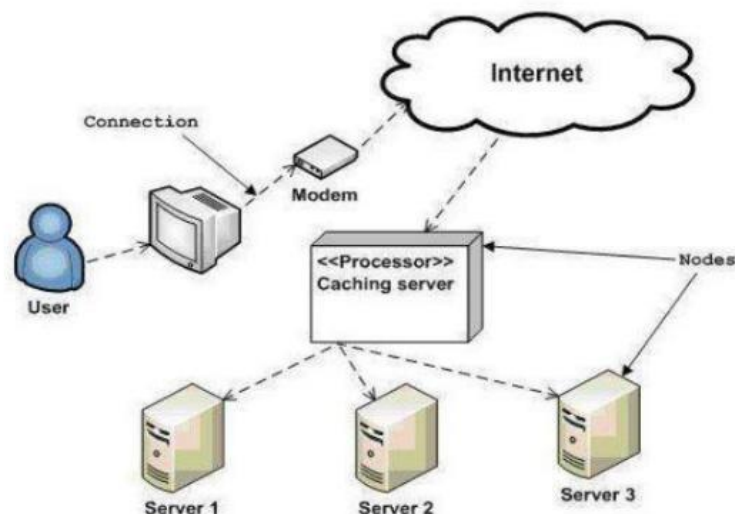
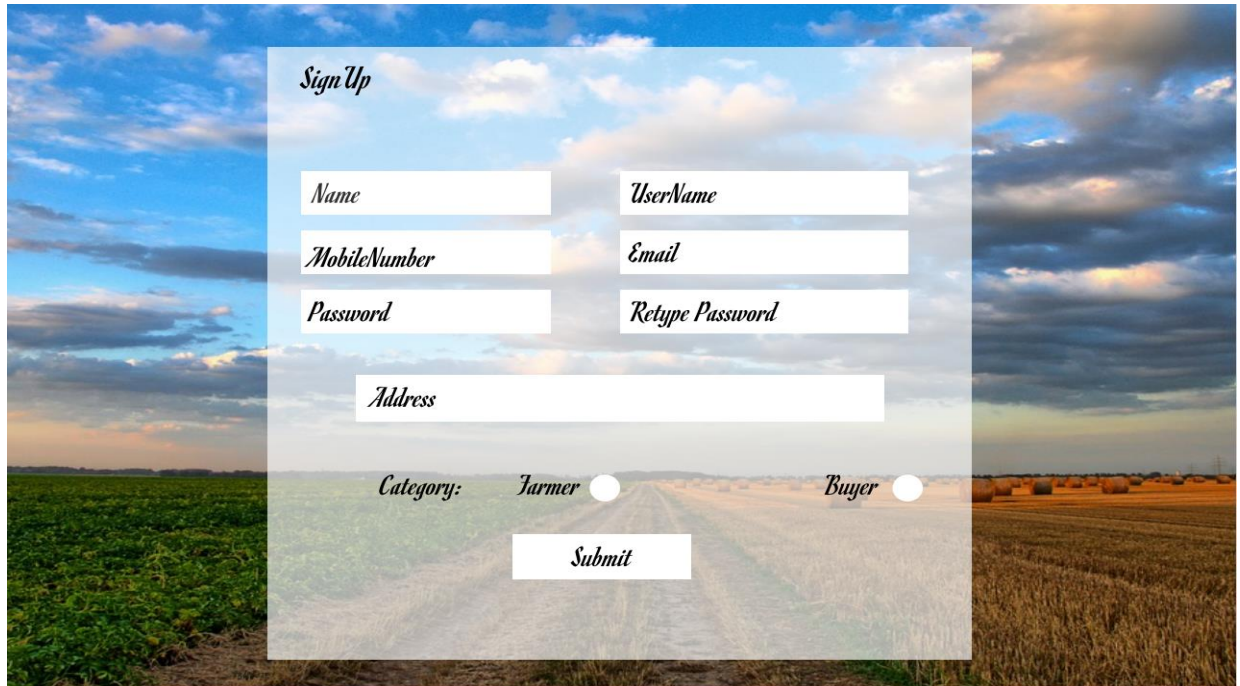


Fig 8: Deployment diagram for Pension Management System

4.3 USER INTERFACE DESIGN

4.3.1-INPUT DESIGN

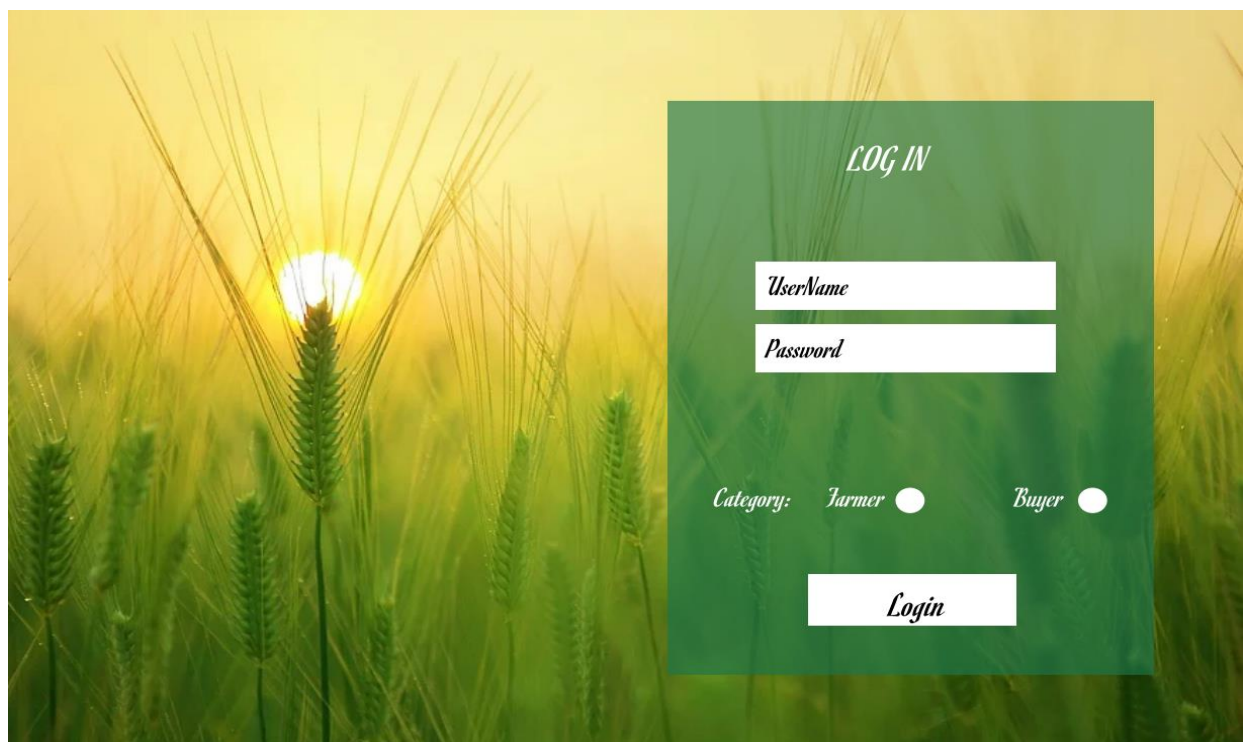
Form Name: User Registration



The User Registration form is displayed over a background image of a field at sunset. The form is titled "Sign Up" and contains the following fields and controls:

- Name* (text input)
- UserName* (text input)
- MobileNumber* (text input)
- Email* (text input)
- Password* (text input)
- Retype Password* (text input)
- Address* (text input)
- Category:* *Farmer* ☐ *Buyer* ☐
- Submit* (button)

Form Name : User Login



The User Login form is displayed over a background image of a field at sunrise. The form is titled "LOG IN" and contains the following fields and controls:

- UserName* (text input)
- Password* (text input)
- Category:* *Farmer* ☐ *Buyer* ☐
- Login* (button)

4.3.2 OUTPUT DESIGN

User Registration

+ 1235 2355 98

ADMIN@GMAIL.COM

3-5 BUSINESS DAYS DELIVERY & FREE RETURNS

VEGEFOODS

HOME SIGN-IN SIGN-UP CONTACT

Create Account

First Name

Last Name

Email Address

Address

place

Phone Number

Password

Confirm Password

Enter Usertype :
select

SIGN UP

User Login

+ 1235 2355 98


ADMIN@GMAIL.COM

3-5 BUSINESS DAYS DELIVERY & FREE RETURNS

VEGEFOODS

HOME SIGN-IN SIGN-UP CONTACT

Login Form



Username

Username

Password

Password

Login

4.4. DATABASE DESIGN

A database is a structured system with the capacity to store information and allows users to retrieve stored information quickly and effectively. Any database's primary goal is its data, which demands protection.

There are two stages to the database design process. The user needs are obtained in the first step, and a database is created to as clearly as possible meet these criteria. This process, known as information level design, is carried out independently of all DBMS.

The design for the specific DBMS that will be used to construct the system in issue is converted from an information level design to a design in the second stage. Physical Level Design is the stage where the characteristics of the particular DBMS that will be used are discussed. Parallel to the system design is a database design. The database's data arrangement aims to accomplish the two main goals listed below.

- Data Integrity
- Data independence

4.4.1 Relational Database Management System (RDBMS)

In a relational model, the database is shown as a set of relations. Each relation resembles a file or table of records with values. A row is referred to as a tuple, a column header is referred to as an attribute, and the table is referred to as a relation in formal relational model language. Numerous tables, each with a unique name, make up a relational database. Each row in a tale reflects a set of related values.

Relations, Domains & Attributes

A relation is a table. Tuples are the units of a table's rows. An ordered group of n elements is a tuple. Attributes are referred to as columns. Every table in the database has relationships already established between them. This guarantees the integrity of both referential and entity relationships. A domain D is a set of atomic values. A common method of defining a domain is by selecting a data type from which the domain's data values are derived. It is also useful to give the domain a name in order to make the values of the domain simpler to comprehend. A relation's values are all atomic and inseparable from one another.

Relationships

- Key is used to create table relationships.
- Primary Key and Foreign Key are the two principal keys that are most crucial.
- With the use of these keys, relationships for entity integrity and referential integrity created.
- Entity Integrity forbids the use of null values for any Primary Key.
- No Primary Key may contain null values, according to Referential Integrity.
- Referential Integrity: A Primary Key value in the same domain must correspond to each unique Foreign Key value. Super Key and Candidate Keys are additional keys.

4.6.2 Normalization

The simplest possible grouping of data is used to put them together so that future changes can be made with little influence on the data structures. The formal process of normalising data structures in a way that reduces duplication and fosters integrity. A technique called normalisation divides superfluous fields and divides a huge table into smaller ones. Anomalies in insertion, deletion, and updating are also prevented by using it. Normal form in data modelling use two concepts, keys and relationships. A row in a table is uniquely identified by a key. The two different types of keys are primary keys and foreign keys. To identify between records from the same table, a primary key is an element, or group of elements, in the table. A column in a table known as a foreign key is used to uniquely identify records from other tables. Up to the third normal form, all tables have been normalised.

It means placing things in their natural form, as the name suggests. By using normalisation, the application developer aims to establish a coherent arrangement of the data into appropriate tables and columns, where names may be quickly related to the data by the user. By removing recurring groups from the data, normalisation prevents data redundancy, which puts a heavy strain on the computer's resources. These consist of:

- ✓ Normalize the data.
- ✓ Choose proper names for the tables and columns.
- ✓ Choose the proper name for the data.

First Normal Form

According to the First Normal Form, each attribute's domain must only include atomic values, and each attribute's value in a tuple must be a single value from that domain. In other words, 1NF forbids using relationships as attribute values within tuples or relations within relations. Single atomic or indivisible values are the only attribute values that are permitted under 1NF. The data must first be entered into First Normal Form. This can be accomplished by separating data into tables of a similar type in each table. Depending on the needs of the project, a Primary Key or Foreign Key is assigned to each table. For each nested relation or non-atomic attribute, new relations are formed in this process. This got rid of data groups that were repeated. If a relation solely meets the constraints that include the primary key, it is said to be in first normal form.

Second Normal Form

In relations when the primary key includes more than one attribute, no non-key attribute should be functionally dependent on a component of the main key. We break down each partial key in this and put up a new relation with its dependent properties. Keep in your database the original primary key and any characteristics that are totally dependent on it. This procedure aids in removing data that depends only on a small portion of the key. If a connection meets all the requirements for first normal form for the main key and every non-primary key feature of the relation is completely dependent on its primary key alone, then and only then is the relation said to be in second normal form.

Third Normal Form

Relation should not have a non-key attribute that is functionally determined by another non-key attribute or by a collection of non-key attributes, according to the Third Normal Form. The primary key should not be transitively dependent, in other words. The non-key attributes that functionally determine other non-key attributes are decomposed in this way put up in relation. This procedure is used to eliminate anything not wholly dependent on the Primary Key. Only when a relation is in second normal form and, more importantly, when its non-key qualities are independent of one another, is it said to be in third normal form.

TABLE DESIGN

Table No: 01

Table Name: login

Primary key: loginid

s.No	Name	Data type	Description
1.	loginid	int(15)	login id
2.	username	varchar(30)	username
3.	password	varchar(30)	password
4.	type	varchar(30)	type
5.	status	int(15)	current status

Table No: 02

Table Name: register

Primary key: id

Foreign key: loginid references table login

s.No	Name	Data type	Description
1.	id	int(15)	registration id
2.	loginid	int(15)	login id
3.	fname	varchar(30)	user first name
4.	lname	varchar(30)	user last name
5.	email	varchar(30)	user email
6.	address	varchar(30)	user address
7.	place	varchar(100)	user place
8.	phone	int(30)	user phone number
9.	estatus	int(30)	current status

Table No: 03

Table Name: tbl_category

Primary key: id

s.No	Name	Data type	Description
1.	id	int(11)	category id
2.	categoryname	varchar(50)	category name
3.	image	varchar(20)	image of the category
4.	status	varchar(50)	current status

Table No: 04**Table Name: tbl_products****Primary key:pro_id**

s.No	Name	Data type	Description
1.	pro_id	int(20)	product id
2.	category	varchar(50)	product category
3.	product_name	varchar(50)	product name
4.	product_image	varchar(20)	product image

Table No: 05**Table Name: farmdetail****Primary key: farmid****Foreign key: loginid references table login**

s.No	Name	Data type	Description
1.	farmid	int(100)	farm id
2.	loginid	int(100)	farmer login id
3.	farm_name	varchar(100)	farm name
4.	email	varchar(100)	farmer email
5.	phone	int(30)	phone number
6.	address	int(100)	address
7.	about	int(100)	about farm
8.	idproof	varchar(100)	id-proof of farmer
9.	status	int(10)	status

Table No: 06**Table Name: farmer_product****Primary key: id****Foreign key: loginid references table login**

s.No	Name	Data type	Description
1.	id	int(10)	product id
2.	loginid	int(20)	farmer login id
3.	fname	varchar(100)	farmer first name
4.	place	varchar(100)	farmer place
5.	pro_id	int(20)	product id
6.	price	int(20)	price of the product
7.	quantity	int(20)	product quantity
8.	image	varchar(100)	product image
9.	status	int(20)	product status

Table No: 07**Table Name: pro_req****Primary key: p_req****Foreign key: loginid references table login**

s.No	Name	Data type	Description
1.	p_req	int(100)	requested product id
2.	loginid	int(100)	farmer login id
3.	fname	varchar(100)	farmer name
4.	email	varchar(100)	farmer email
5.	phone	int(30)	phone number
6.	address	int(100)	address
7.	product_name	int(100)	requested product name
8.	image	varchar(100)	requested product name
9.	status	int(20)	status

Table No: 08**Table Name: tbl_cart****Primary key: id****Foreign key: loginid references table login**

s.No	Name	Data type	Description
1.	id	int(30)	cart id
2.	loginid	int(30)	buyer login id
3.	fname	varchar(100)	farmer name
4.	o_id	int(50)	farmer product id
5.	pro_id	int(30)	product id
6.	quantity	int(30)	purchased quantity
7.	price	int(30)	product price
8.	totalprice	int(30)	product total price
9.	status	int(30)	cart status

Table No: 09

Table Name: order_tbl

Primary key: id

Foreign key: loginid references table login

s.No	Name	Data type	Description
1.	id	int(10)	order id
2.	login_id	int(10)	buyer login id
3.	cartid	int(10)	cart id
4.	price	int(10)	product price
5.	status	varchar(20)	status of the order
6.	date	varchar(50)	ordered date

Table No: 10

Table Name: tbl_payment

Primary key: id

Foreign key: loginid references table login

s.No	Name	Data type	Description
1.	payment_id	int(50)	payment id
2.	login_id	int(50)	buyer login id
3.	cartid	int(50)	buyer cart id
4.	amount	int(50)	product price
5.	status	varchar(50)	status of the payment

Table No: 11

Table Name: feedback

Primary key: id

s.No	Name	Data type	Description
1.	id	int(100)	feedback id
2.	name	varchar(30)	buyer name
3.	comment	varchar(50)	comments on service
4.	Order_proof	varchar(100)	order summary
5.	fdate	varchar(50)	feedback written date

CHAPTER 5

SYSTEM TESTING

5.1 INTRODUCTION

Software testing is the process of carefully controlling the execution of software in order to determine whether it behaves as intended. The words verification and validation are frequently used in conjunction with software testing. Validation is the process of examining or evaluating a product, including software, to determine whether it complies with all relevant specifications. One type of verification, software testing, uses methods including reviews, analyses, inspections, and walkthroughs as well. Checking that what has been specified is what the user requested is the process of validation.

The processes of static analysis and dynamic analysis are additional ones that are frequently related to software testing. Static analysis examines the software's source code, searching for issues and obtaining statistics without actually running the code. Dynamic analysis examines how software behaves while it is running in order to offer data like execution traces, timing profiles, and test coverage details.

Testing is a collection of activities that can be planned ahead of time and carried out in a methodical manner. Testing starts with individual modules and progresses all the way to system integration for computer-based systems. There are many rules that can be used as testing objectives, and testing is necessary for the system testing objectives to be successful. As follows:

A programmed is tested by being run with the goal of identifying any errors.

- A successful test is one that finds an undetected error.
- A good test case is one that has a high likelihood of doing so.

If a test is successfully carried out in accordance with the aforementioned aims, it will reveal software bugs. Additionally, testing shows that the software function seems to be operating in accordance with the specification and that the performance criterion appears to have been satisfied. There are three ways to test program.

- For accuracy
- For effectiveness in implementation
- For the complexity of computing

Test for correctness is supposed to verify that a program does exactly what it was designed to do. This is much more difficult than it may at first appear, especially for large programs

5.2 TEST PLAN

A test plan suggests a number of required steps that need be taken in order to complete various testing methodologies. The activity that is to be taken is outlined in the test plan. A computer programme, its documentation, and associated data structures are all created by software developers. It is always the responsibility of the software developers to test each of the program's separate components to make sure it fulfils the purpose for which it was intended. In order to solve the inherent issues with allowing the builder evaluate what they have developed, there is an independent test group (ITG). Testing's precise goals should be laid forth in quantifiable language. The test plan should include information about the mean time to failure, the cost to locate and correct the defects, the remaining defect density or frequency of occurrence, and the number of test work hours required for each regression test.

The levels of testing include:

- ❖ Unit testing
- ❖ Integration Testing
- ❖ Data validation Testing
- ❖ Output Testing

5.2.1 Unit Testing

Unit testing concentrates verification efforts on the software component or module, which is the smallest unit of software design. The component level design description is used as a guide when testing crucial control paths to find faults inside the module's perimeter. the level of test complexity and the untested area determined for unit testing. Unit testing is white-box focused, and numerous components may be tested simultaneously. To guarantee that data enters and exits the software unit under test properly, the modular interface is tested. To make sure that data temporarily stored retains its integrity during each step of an algorithm's execution, the local data structure is inspected. To confirm that each statement in a module has been executed at least once, boundary conditions are evaluated. Finally, each path for managing errors is examined. Before starting any other test, tests of data flow over a module interface are necessary. All other tests are irrelevant if data cannot enter and depart the system properly. An important duty during the unit test is the selective examination of execution pathways. Error circumstances must be foreseen in good design, and error handling paths must be put up to cleanly reroute or halt work when an error does arise. The final step of unit testing is boundary testing.

5.2.2 Integration Testing

Integration testing is a methodical approach for creating the program's structure while also carrying out tests to find interface issues. The goal is to construct a programmed structure that has been determined by design using unit tested components. The programmed as a whole is tested. Correction is challenging since the size of the overall programmed makes it challenging to isolate the causes. Once these mistakes are fixed, new ones come into being, and the cycle repeats itself indefinitely. All of the modules were integrated after unit testing was completed in the system to check for any interface inconsistencies. In addition, variations in programmed structures were eliminated, and a special programmed structure developed.

5.2.3 Validation Testing or System Testing

The testing process comes to an end here. The complete system, including all forms, code, modules, and class modules, was tested in this. Popular names for this type of testing include system tests and black box testing.

The functional requirements of the software are the main emphasis of the black box testing approach. That example, using Black Box testing, a software engineer can create sets of input conditions that will fully test every programmed requirement.

The following sorts of problems are targeted by black box testing: erroneous or missing functions, interface errors, data structure or external data access errors, performance errors, initialization errors, and termination errors.

5.2.4 Output Testing or User Acceptance Testing

User approval of the system under consideration is tested; in this case, it must meet the needs of the company. When developing, the programmed should stay in touch with the user and perspective system to make modifications as needed. With regard to the following points, this is done:

- Input Screen Designs,
- Output Screen Designs,

The mentioned testing is carried out using a variety of test data. The preparation of test data is essential to the system testing process. The system under investigation is then put to the test using the prepared test data. Errors in the system are once again found during testing, fixed using the methods described above, and logged for use in the future.

Automation Testing

Software and other computer goods are tested automatically to make sure they abide by tight guidelines. In essence, it's a test to ensure that the hardware or software performs exactly as intended. It checks for errors, flaws, and any other problems that might occur throughout the development of the product. Any time of day can be used to do automation testing. It looks at the software using scripted sequences. It then summarises what was discovered, and this data can be compared to results from earlier test runs.

Benefits of Automation Testing

Detailed reporting capabilities - Test cases for different scenarios are carefully built for automation testing. These planned sequences can cover a lot of ground and produce in-depth reports that are simply impossible for a human to produce.

Improved bug detection - Finding bugs and other flaws in a product is one of the key reasons to test it. This procedure can be made simpler with automation testing. Additionally, it can examine a greater test coverage than perhaps people can.

- Simplifies testing - Most SaaS and tech organizations regularly test their products as part of daily operations. The key is to keep things as basic as you can. Automation has a lot of advantages. The test scripts can be reused when automating test tools.
- Speeds up the testing process - Humans cannot keep up with automated technology and machines. This is why we employ them, along with increased accuracy. Your software development cycles are subsequently shortened by this.
- Reduces human intervention - Without the requirement for human supervision, tests can be performed at any time of day, including overnight. Additionally, when done automatically, this can lessen the possibility of human error.

5.2.5 Selenium Testing

An open-source program called Selenium automates web browsers. It offers a single interface that enables you to create test scripts in a number of different programming languages, including Ruby, Java, NodeJS, PHP, Perl, Python, and C#. Web application testing for cross-browser compatibility is automated using the Selenium testing tool. Whether they are responsive, progressive, or standard, it is utilized to assure high-quality web apps. Selenium is a free software program

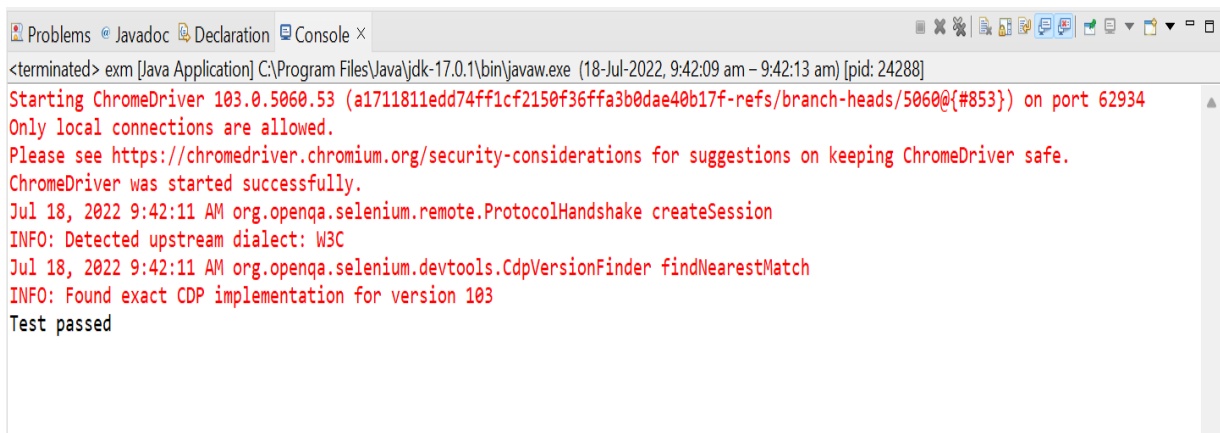
Test case for a Login Page

Test Case ID: test1			Test Designed By: Neha Alex		
Test Priority (Low/Medium/High): High			Test Designed Date: 18-07-2022		
Module Name: Login Page			Test Executed By: Ms Anit James		
Test Title: Verify login with validusername and password			Test Execution Date: 18-07-2022		
Description: Test the Login Page					
Pre-Condition: User has valid username and password					
Step	Test Step	Test Data	ExpectedResult	Actual Result	Status(Pass/Fail)
1	Navigation to LoginPage		Login Page should be displayed	Login page displayed	Pass
2	Provide Valid User Name	User Name: nidhi@gmail.com	User shouldbe able to Login	User Logged in andnavigated to the dashboard with records	Pass
3	Provide Valid Password	Password: nidhi@123			
4	Click on Sign In button				
5	Provide Invalid user name or Password	Email Id: sheeja@gmail.com Password: sheej	User should notbe able to Login	Message for enter valid username or password displayed	Pass
6	Provide Null Username Id orPassword	Email Id: null Password: null			
7	Click on Sign In button				
Post-Condition: User is validated with database and successfully login into account. The Account session details are logged in database					

Code:

```
package test1;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class exm{
public static void main(String[] args) {
System.setProperty("webdriver.chrome.driver","C:\\Users\\susmi\\Downloads\\chromedriver
_win32\\chromedriver.exe" );
WebDriver driver=new ChromeDriver();

driver.get("http://localhost/farmer_corrected/farmer/login.php");
driver.findElement(By.id("username")).sendKeys("nidhi@gmail.com");
driver.findElement(By.id("password")).sendKeys("Nidhi@123");
driver.findElement(By.id("signin")).click();
String actualUrl="http://localhost/farmer_corrected/farmer/farmer.php";
String expectedUrl= driver.getCurrentUrl();
if(actualUrl.equalsIgnoreCase(expectedUrl)) {
System.out.println("Test passed");
} else {
System.out.println("Test failed");
}
}
}
```



```
<terminated> exm [Java Application] C:\Program Files\Java\jdk-17.0.1\bin\javaw.exe (18-Jul-2022, 9:42:09 am - 9:42:13 am) [pid: 24288]
Starting ChromeDriver 103.0.5060.53 (a1711811edd74ff1cf2150f36ffa3b0dae40b17f-refs/branch-heads/5060@{#853}) on port 62934
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
ChromeDriver was started successfully.
Jul 18, 2022 9:42:11 AM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected upstream dialect: W3C
Jul 18, 2022 9:42:11 AM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
INFO: Found exact CDP implementation for version 103
Test passed
```

Test case for E-Farm Registration

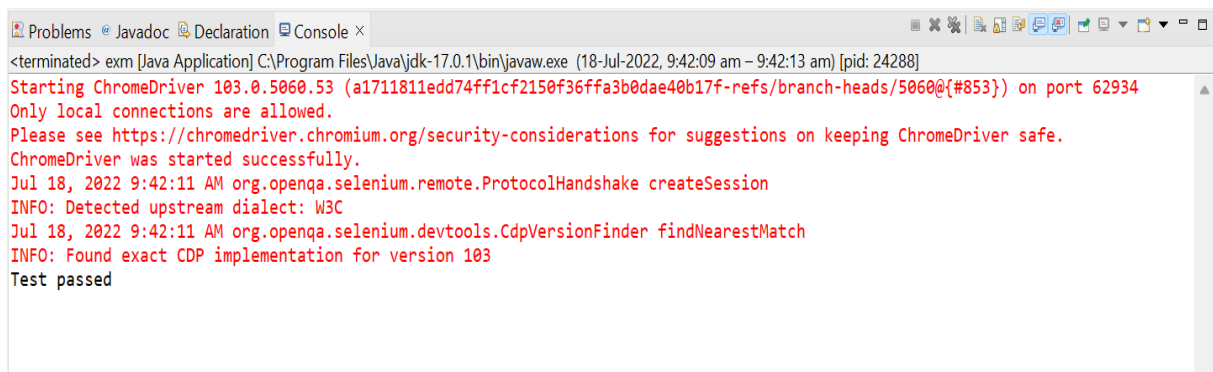
Project Name: E-Farm					
Updation Test Case					
Test Case ID: Registration			Test Designed By: Neha Alex		
Test Priority (Low/Medium/High): High			Test Designed Date: 18-07-2022		
Module Name: Register Screen			Test Executed By: Ms. Anit James		
Test Title: User Registration Details			Test Execution Date: 18-07-2022		
Description: Register to system and Registration is completed then login , if some error occurs, testwill fail					
Pre-Condition: User has valid user name and password					
Step	Test Step	Test Data	Expected Result	Actual Result	Status (Pass/Fail)
1	Navigation to Register Page		Register Page Should be displayed	Registratio n page displayed	Pass
2	Provide Valid Registration details	User Name: Nidhi	User should be able to Register	User registration Completed after go to the login page	Pass
3					
4	Click on Login button				
5	Provide profile details	Input profile details	User will be redirected to Login page	Use will be redirected to Login page	Pass
7	Click on register button				
8	Provide invalid information	Input invalid profile details.	User will be stay in register page	User will be stay on that page showing error message	Pass
9	Click on register button				
Post-Condition: User is validated with database and successfully registered. TheAccount session details are logged in database.					

Code:

```

package test1;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class reg {
public static void main(String[] args) {
System.setProperty("webdriver.chrome.driver", "C:\\Users\\susmi\\Downloads\\chromedriver_win32\\chromedriver.exe" );
WebDriver driver=new ChromeDriver();
driver.get("http://localhost/farmer_corrected/farmer/signup.php");
driver.findElement(By.id("fname")).sendKeys("Susmin");
driver.findElement(By.id("lname")).sendKeys("Radhhh");
driver.findElement(By.id("email")).sendKeys("susminmariam@gmail.com");
driver.findElement(By.id("adress")).sendKeys("Raggg");
driver.findElement(By.id("place")).sendKeys("Arattupuzha");
driver.findElement(By.id("phone")).sendKeys("9087563456");
driver.findElement(By.id("password")).sendKeys("Ancyanu@7");
driver.findElement(By.id("confirm_password")).sendKeys("Ancyanu@7");
driver.findElement(By.id("signup")).click();
String actualUrl="http://localhost/farmer_corrected/farmer/login.php";
String expectedUrl= driver.getCurrentUrl();
if(actualUrl.equalsIgnoreCase(expectedUrl)) {
System.out.println("Test passed");
} else {
System.out.println("Test failed");
}
}
}

```



CHAPTER 6

IMPLEMENTATION

6.1 INTRODUCTION

The project's implementation phase is where the conceptual design is transformed into a functional system. Gaining the users' trust that the new system will function, be efficient, and be accurate can be the most important step in creating a successful new system. User documentation and training are its main concerns. Usually, conversion happens either during or after the user's training. Implementation is the process of turning a newly revised system design into an operational one, and it simply refers to placing a new system design into operation.

At this point, the user department is responsible for the majority of the workload, the most disruption, and the most influence on the current system. If the implementation is not well thought out or managed, confusion and disorder may result.

Implementation encompasses all of the steps used to switch from the old system to the new one. The new system could be entirely different, take the place of an existing manual or automated system, or it could be modified to work better. A reliable system that satisfies organizational needs must be implemented properly. System implementation refers to the process of actually using the built system. This comprises all the processes involved in switching from the old to the new system. Only after extensive testing and if it is determined that the system is operating in accordance with the standards, can it be put into use. The system personnel assess the system's viability. The effort necessary for system analysis and design to implement the three key components of education and training, system testing, and changeover will increase in proportion to how complicated the system being implemented is.

The implementation state involves the following tasks:

- Careful planning.
- Investigation of system and constraints.
- Design of methods to achieve the changeover.

6.2 IMPLEMENTATION PROCEDURES

Software implementation refers to the complete installation of the package in its intended environment, as well as to the system's functionality and satisfaction of its intended applications. A software development project is frequently commissioned by someone who will not be using it. People have initial scepticism against the programme, but we must prevent resistance from growing because one must assure that:

- The active user must be aware of the benefits of using the new system.
- Their confidence in the software is built up.
- Proper guidance is imparted to the user so that he is comfortable in using the application.

Before examining the system, the user must be aware that the server software needs to be running on the server in order to access the results. The actual process won't happen if the server object is not active and functioning on the server.

6.2.1 User Training

The purpose of user training is to get the user ready to test and modify the system. The people who will be involved must have faith in their ability to contribute to the goal and benefits anticipated from the computer-based system. Training is more necessary as systems get more complicated. The user learns how to enter data, reply to error alerts, query the database, and call up routines that will generate reports and carry out other necessary tasks through user training.

6.2.2 Training on the Application Software

The user will need to receive the essential basic training on computer awareness, after which the new application software will need to be taught to them. This will explain the fundamental principles of how to use the new system, including how the screens work, what kind of help is displayed on them, what kinds of errors are made while entering data, how each entry is validated, and how to change the data that was entered. Then, while imparting the program's training on the application, it should cover the information required by the particular user or group to operate the system or a certain component of the system. Depending on the user group and hierarchical level, this training could be different.

6.2.3 System Maintenance

The mystery of system development is maintenance. When a software product is in the maintenance stage of its lifecycle, it is actively working. A system should be properly maintained after it has been effectively implemented. System maintenance is an essential part of the software development life cycle. In order for a system to be flexible to changes in the system environment, maintenance is required. Of course, software maintenance involves much more than just "finding mistakes."

6.2.4 Hosting

app.infinityfree.net is a free website hosting platform that provides an array of valuable features, including a website builder, control panel and file manager.

Hosting Procedures

1. Create an Account or Log In into your account.
2. Create a Hosting Account.
3. Create a Custom Domain or a Subdomain provided by Infinity Free.
4. Manage your account.
5. Upload your Files.
6. Creating Your Database.
7. Changing your PHP connection file configuration.

The image shows two screenshots. The top screenshot is the Infinity Free control panel for the account 'epiz_32206514'. It includes a navigation bar with links like Home, Profile, Accounts, and a 'Go Premium' button. The main content area has buttons for 'Control Panel' and 'File Manager'. Below these are sections for 'Account Details' (Username: epiz_32206514, Password: masked, Status: Active, Label: Website for e-farm.epizy.com, Main Domain: 4vrne830.epizy.com) and 'FTP Details' (FTP Username: epiz_32206514, FTP Password: masked, FTP Hostname: ftpupload.net, FTP Port: 21). There are also links for 'Edit Account', 'Deactivation History', and 'Redirects'. The bottom screenshot shows a sample website for 'VEGEFOODS' with a banner of fresh vegetables and the text '100% FRESH & ORGANIC FOODS'. The website has a green header with a phone number and email, and a navigation menu with links for HOME, SIGN-IN, SIGN-UP, and CONTACT.

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

The proposed system makes all the things automatic which make easier serving as a best solution to all the problems. Where E-farm will serve as a way for the farmers to sell their products across the country just with some basic knowledge about how to use the website. And it is designed to provide all the advantages to farmers. The proposed system is more efficient and better than the existing system. It is designed to overcome all the drawbacks of the present system and to provide a permanent solution to them. The primary aim of the new system is to speed up the transaction.

7.2 FUTURE SCOPE

In future we can expect the modified version of 'E-Farm'. The system is flexible for further up stage with additional requirement of the farmers, the PHP and MYSQL make this modification very easily, it is also possible to involve more functions into the system.

The database and the information can be updated to the latest coming versions. Future scope of the system is that the buyer can purchase the products on the site based on farmer rate based on their product selling profit on the site.

CHAPTER 8

BIBLIOGRAPHY

REFERENCES:

- Gary B. Shelly, Harry J. Rosenblatt, “*System Analysis and Design*”, 2009.
- Roger S Pressman, “*Software Engineering*”, 1994.
- PankajJalote, “*Software engineering: a precise approach*”, 2006.
- James’s lee and Brent ware Addison, “Open-source web development with LAMP”, 2003
- IEEE Std 1016 Recommended Practice for Software Design Descriptions.

WEBSITES:

- www.w3schools.com
- www.jquery.com
- <http://homepages.dcc.ufmg.br/~rodolfo/es-1-03/IEEE-Std-830-1998.pdf>
- www.agilemodeling.com/artifacts/useCaseDiagram.html

CHAPTER 9

APPENDIX

9.1 Sample Code

Signup.php

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>E-Farm</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <link
href="https://fonts.googleapis.com/css?family=Poppins:200,300,400,500,600,700,800&display=swa
p" rel="stylesheet">
  <link href="https://fonts.googleapis.com/css?family=Lora:400,400i,700,700i&display=swap"
rel="stylesheet">
  <link href="https://fonts.googleapis.com/css?family=Amatic+SC:400,700&display=swap"
rel="stylesheet">

  <link rel="stylesheet" href="css/open-iconic-bootstrap.min.css">
  <link rel="stylesheet" href="css/animate.css">

  <link rel="stylesheet" href="css/owl.carousel.min.css">
  <link rel="stylesheet" href="css/owl.theme.default.min.css">
  <link rel="stylesheet" href="css/magnific-popup.css">

  <link rel="stylesheet" href="css/aos.css">

  <link rel="stylesheet" href="css/ionicons.min.css">

  <link rel="stylesheet" href="css/bootstrap-datepicker.css">
  <link rel="stylesheet" href="css/jquery.timepicker.css">

  <link rel="stylesheet" href="css/flaticon.css">
  <link rel="stylesheet" href="css/icomoon.css">
  <link rel="stylesheet" href="css/style.css">
</head>
<body class="goto-here">
  <div class="py-1 bg-primary">
    <div class="container">
      <div class="row no-gutters d-flex align-items-start align-items-center px-md-0">
        <div class="col-lg-12 d-block">
          <div class="row d-flex">
            <div class="col-md pr-4 d-flex topper align-items-center">
              <div class="icon mr-2 d-flex justify-content-center
align-items-center"><span class="icon-phone2"></span></div>
              <span class="text">+ 1235 2355 98</span>
            </div>
            <div class="col-md pr-4 d-flex topper align-items-
center">
              <div class="icon mr-2 d-flex justify-content-center
align-items-center"><span class="icon-paper-plane"></span></div>
```

```

        <span class="text">admin@gmail.com</span>
    </div>
    <div class="col-md-5 pr-4 d-flex top-0 align-items-center text-
lg-right">
        <span class="text">3-5 Business days delivery &
Free Returns</span>
    </div>
</div>
</div>
</div>
</div>
</div>
<div class="collapse navbar-collapse" id="ftco-nav">
    <ul class="navbar-nav ml-auto">
        <li class="nav-item active"><a href="index.php" class="nav-link">Home</a></li>

        <li class="nav-item"><a href="login.php" class="nav-link">sign-in</a></li>
        <li class="nav-item"><a href="signup.php" class="nav-link">sign-up</a></li>
        <li class="nav-item"><a href="contact.html" class="nav-link">Contact</a></li>

    </ul>
</div>
</div>
</nav>

<body>

    <div class="container">
        <div class="inner">
            <div class="image-holder">
                
            </div>
            <form action="signupacc.php" method="POST">
                <h3>Create Account</h3>
                <div class="form-group">
                    <input type="text" name="fname" id="fname"
placeholder="First Name" class="form-control" required onchage="Validate();">
                    <span id="msg1" style="color:red;"></span>
                </div>
            </form>
            <script>
function Validate()
{
    var val = document.getElementById('fname').value;

```

```

if (!val.match(/^[A-Z][A-Za-z]{3,}$/))
{
    document.getElementById('msg1').innerHTML="Start with a Capital letter & Only alphabets
without space are allowed!!";
    document.getElementById('fname').value = "";
    return false;
}
document.getElementById('msg1').innerHTML=" ";
return true;
}
</script>
<br><input type="text" name="lname" id="lname"
placeholder="Last Name" class="form-control" required onchange="Validate1();">
<span id="msg2" style="color:red;"></span>
<script>
function Validate1()
{
    var val = document.getElementById('lname').value;

    if (!val.match(/^[A-Z][a-z]{0,}$/))
    {
        document.getElementById('msg2').innerHTML="Start with a Capital letter & Only alphabets
without space are allowed!!";
        document.getElementById('lname').value = "";
        return false;
    }
    document.getElementById('msg2').innerHTML=" ";
    return true;
}
</script>
</div>
<div class="form-wrapper">
    <input type="text" name="email" id="email"
placeholder="Email Address" class="form-control" required onchange="Validata();">
    <i class="zmdi zmdi-email"></i>
</div>
<span id="msg5" style="color:red;"></span>
<script>
function Validata()
{
    var val = document.getElementById('email').value;

    if (!val.match(/^[a-zA-Z0-9_\.\\-]+\@(((a-zA-Z0-9\\-\\-)+\\-)+([a-zA-Z0-9]{2,4})+$/))
    {
        document.getElementById('msg5').innerHTML="Enter a Valid Email";

        document.getElementById('email').value = "";
        return false;
    }
    document.getElementById('msg5').innerHTML=" ";
    return true;
}

</script>
<div class="form-wrapper">

```



```

        <input type="text" name="adress" id="adress"
placeholder="Address" class="form-control" required onchange="Validname();">
        <i class="zmdi zmdi-home"></i>
    </div>
    <span id="msg3" style="color:red;"></span>

<script>
function Validname()
{
    var val = document.getElementById('adress').value;

    if (!val.match(/^[A-Z][a-z" "]{3,}$/))
    {
        document.getElementById('msg3').innerHTML="Start with a Capital letter & Only alphabets are
allowed";
        document.getElementById('adress').value = "";
        return false;
    }
    document.getElementById('msg3').innerHTML=" ";
    return true;
}
</script>

```

```

<div class="form-wrapper">
        <input type="text" name="place" id="place"
placeholder="place" class="form-control" required onchange="Validname();">
        <i class="zmdi zmdi-home"></i>
    </div>
    <span id="msg3" style="color:red;"></span>

<script>
function Validname()
{
    var val = document.getElementById('place').value;

    if (!val.match(/^[A-Z][a-z" "]{3,}$/))
    {
        document.getElementById('msg3').innerHTML="Start with a Capital letter & Only alphabets are
allowed";
        document.getElementById('place').value = "";
        return false;
    }
    document.getElementById('msg3').innerHTML=" ";
    return true;
}
</script>

```

```

        <div class="form-wrapper">
            <input type="number" name="phone" maxlength="10"
id="phone" placeholder="Phone Number" class="form-control" required onchange="Validat();">
            <i class="zmdi zmdi-phone"></i>
        </div>
        <span id="msg4" style="color:red;"></span>

<script>

```

```

function Validat()
{
    var val = document.getElementById('phone').value;

    if (!val.match(/^[789][0-9]{9}$/))
    {
        document.getElementById('msg4').innerHTML="Only Numbers are allowed and must contain
10 number";

        document.getElementById('phone').value = "";
        return false;
    }
    document.getElementById('msg4').innerHTML=" ";
    return true;
}

</script>

```

```

<div class="form-wrapper">
    <input type="password" name="password"
id="password" placeholder="Password" class="form-control" required onchange="Validp();">
    <i class="zmdi zmdi-lock"></i>
</div>
<span id="msg6" style="color:red;"></span>

```

```

<script>
function Validp()
{
    var val = document.getElementById('password').value;

    if (!val.match(/^(?=.*?[A-Z])(?=.*?[a-z])(?=.*?[0-9])(?=.*?[#?!@$%^&*~])/))
    {
        document.getElementById('msg6').innerHTML="Upper case, Lower case, Special character
and Numeric number are required in Password filed";

        document.getElementById('password').value = "";
        return false;
    }
    document.getElementById('msg6').innerHTML=" ";
    return true;
}

</script>

```

```

<div class="form-wrapper">
    <input type="password"
name="confirm_password" id="confirm_password" placeholder="Confirm Password" class="form-
control" required onchange="check();">
    <i class="zmdi zmdi-lock"></i>
</div>
<span id="msg7" style="color:red;"></span>

<script>
function check()
{
    var pas1=document.getElementById("password");

```

```

                                var
pas2=document.getElementById("confirm_password");

                                if(pas1.value=="")
{
    document.getElementById('msg7').innerHTML="Password can't be null!!!";
    pas1.focus();
    return false;
}
if(pas2.value=="")
{
    document.getElementById('msg7').innerHTML="Please confirm password!!!";
    pass2.focus();
    return false;
}
if(pas1.value!=pas2.value)
{
    document.getElementById('msg7').innerHTML="Passwords does not match!!!";
    pas1.focus();
    return false;
}
document.getElementById('msg7').innerHTML=" ";
return true;
}
</script>
<td align="right" class="form-control" style = width: 239px;>Enter Usertype :</td>
<td><select id="role" name="role" class="form-control" class="tb" required>
    <option value="select" selected disabled style="width: 100%;height: 34px; ">select</option>
    <option value="farmer" class="">FARMER</option>
    <option value="buyer" class="">BUYER</option>
</select></td><br>
                                <center><button>SIGN UP
                                <i class="zmdi zmdi-arrow-right"></i>
                                </button></center>

                                </form>
                                </div>
                                </div>
                                </div>
</body>
</html>

```

signupacc.php

```

<?php
include 'connection.php';

$fname = $_POST['fname'];
$lname = $_POST['lname'];
$email = $_POST['email'];
$adress = $_POST['adress'];
$place = $_POST['place'];
$phone = $_POST['phone'];
$password = $_POST['password'];
$confirm_password = $_POST['confirm_password'];
$role=$_POST['role'];

$s = "SELECT * FROM login WHERE username = '$email'";
$result1 = mysqli_query($con, $s);
$row1=mysqli_fetch_assoc($result1);
$username= isset($row1['username']) ? $row1['username'] : "";

if($username=="")
{
    if($password=== $confirm_password)
    {
        $sq = "INSERT INTO login (username,password,type,status) VALUES
('$email','$password','$role','1')";

        mysqli_query($con, $sq);

        $sql = "SELECT * FROM login WHERE username = '$email' and password = '$password'
and type = '$role'";
        $result = mysqli_query($con, $sql);
        $no=mysqli_num_rows($result);

        if($no > 0)
        {
            $row=mysqli_fetch_assoc($result);
            $loid=$row['loginid'];
            $sql = "INSERT INTO register (loginid,fname,lname,email,adress,place,phone,estatus)
VALUES ('$loid','$fname','$lname','$email','$adress','$place','$phone','1')";
            mysqli_query($con, $sql);
            echo "<script> alert('Registration successfull');
window.location.href='login.php';</script>";
        }
    }
    else
        echo "<script> alert('please enter password correctly');
window.location.href='signup.php';</script>";
}
else
    echo "<script> alert('You are already registered'); window.location.href='index.php';</script>"

?>

```

Login.php

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>E-Farm</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

    <link
href="https://fonts.googleapis.com/css?family=Poppins:200,300,400,500,600,700,800&display=swap"
rel="stylesheet">
    <link href="https://fonts.googleapis.com/css?family=Lora:400,400i,700,700i&display=swap"
rel="stylesheet">
    <link href="https://fonts.googleapis.com/css?family=Amatic+SC:400,700&display=swap"
rel="stylesheet">

    <link rel="stylesheet" href="css/open-iconic-bootstrap.min.css">
    <link rel="stylesheet" href="css/animate.css">

    <link rel="stylesheet" href="css/owl.carousel.min.css">
    <link rel="stylesheet" href="css/owl.theme.default.min.css">
    <link rel="stylesheet" href="css/magnific-popup.css">

    <link rel="stylesheet" href="css/aos.css">

    <link rel="stylesheet" href="css/ionicons.min.css">

    <link rel="stylesheet" href="css/bootstrap-datepicker.css">
    <link rel="stylesheet" href="css/jquery.timepicker.css">

    <link rel="stylesheet" href="css/flaticon.css">
    <link rel="stylesheet" href="css/icomoon.css">
    <link rel="stylesheet" href="css/style.css">
  <style>
body { font-family: Arial, Helvetica, sans-serif; }
form { border: 3px solid #f1f1f1; }

input[type=text], input[type=password] {
  width: 100%;
  padding: 12px 20px;
  margin: 8px 0;
  display: inline-block;
  border: 1px solid #ccc;
  box-sizing: border-box;
}

button {
  background-color: #04AA6D;
  color: white;
  padding: 14px 20px;
  margin: 8px 0;
  border: none;
```

```

    cursor: pointer;
    width: 100%;
}

button:hover {
    opacity: 0.8;
}

.cancelbtn {
    width: auto;
    padding: 10px 18px;
    background-color: #f44336;
}

.imgcontainer {
    text-align: center;
    margin: 24px 0 12px 0;
}

img.avatar {
    width: 5%;
    border-radius: 10%;
}

.container {
    padding: 16px;
}

span.psw {
    float: right;
    padding-top: 16px;
}

/* Change styles for span and cancel button on extra small screens */
@media screen and (max-width: 300px) {
    span.psw {
        display: block;
        float: none;
    }
    .cancelbtn {
        width: 100%;
    }
}
</style>
</head>
<body class="goto-here">
    <div class="py-1 bg-primary">
        <div class="container">
            <div class="row no-gutters d-flex align-items-start align-items-center px-md-0">
                <div class="col-lg-12 d-block">
                    <div class="row d-flex">
                        <div class="col-md pr-4 d-flex topper align-items-center">
                            <div class="icon mr-2 d-flex justify-content-center
align-items-center"><span class="icon-phone2"></span></div>
                            <span class="text">+ 1235 2355 98</span>

```

```

        </div>
        <div class="col-md pr-4 d-flex toppler align-items-center">
            <div class="icon mr-2 d-flex justify-content-center align-
items-center"><span class="icon-paper-plane"></span></div>
            <span class="text">admin@gmail.com</span>
        </div>
        <div class="col-md-5 pr-4 d-flex toppler align-items-center text-
lg-right">
            <span class="text">3-5 Business days delivery &amp;
Free Returns</span>
        </div>
    </div>
</div>
</div>
</div>
</div>
</div>
</div>
<nav class="navbar navbar-expand-lg navbar-dark ftco_navbar bg-dark ftco-navbar-light" id="ftco-
navbar">
    <div class="container">
        <a class="navbar-brand" href="index.html">Vegefoods</a>
        <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#ftco-nav"
aria-controls="ftco-nav" aria-expanded="false" aria-label="Toggle navigation">
            <span class="oi oi-menu"></span> Menu
        </button>

        <div class="collapse navbar-collapse" id="ftco-nav">
            <ul class="navbar-nav ml-auto">
                <li class="nav-item active"><a href="index.php" class="nav-link">Home</a></li>

                <li class="nav-item"><a href="login.php" class="nav-link">sign-in</a></li>
                <li class="nav-item"><a href="signup.php" class="nav-link">sign-up</a></li>
                <li class="nav-item"><a href="contact.html" class="nav-link">Contact</a></li>

            </ul>
        </div>
    </div>
</nav>
<body>

    <body>

<center><h2>Login Form</h2></center>

<form action="loginAcc.php" method="post">
    <div class="imgcontainer">
        
    </div>

    <div class="container">
        <label for="uname"><b>Username</b></label>
        <input type="text" name="username" id="username" placeholder="Username">

        <label for="psw"><b>Password</b></label>

```

```

<input type="password" name="password" id="password" placeholder="Password">

<button type="submit">Login</button>

</div>

</form>

</body>
</html>

loginAcc.php

<?php
    include 'connection.php';
    session_start();

    $myuser = $_POST['username'];
    $mypass = $_POST['password'];

    $sql = "SELECT * FROM login WHERE username = '$myuser' and password =
'mypass' and type='admin' and status='1'";
    $result = mysqli_query($con, $sql);

    $COUNT = mysqli_num_rows($result);
    if ($COUNT > 0) {
        $id = mysqli_query($con, $sql);
        $row=mysqli_fetch_assoc($id);
        // echo $row['loginid'];
        $_SESSION['id']=$row['loginid'];
        $_SESSION['logged_in'] = true;
        header("location:admin1/index.php");
    }

    $sql1 = "SELECT * FROM login WHERE username = '$myuser' and password
= '$mypass' and type='buyer' and status='1'";
    $result1 = mysqli_query($con, $sql1);

    $COUNT1 = mysqli_num_rows($result1);
    if ($COUNT1 > 0) {
        $id = mysqli_query($con, $sql1);
        $row=mysqli_fetch_assoc($id);
        //echo $row['loginid'];
        $_SESSION['id']=$row['loginid'];
        $_SESSION['logged_in'] = true;
        header("location:buyer.php");
    }

```

```
    }

    $sq = "SELECT * FROM login WHERE username = '$myuser' and password =
'mypass' and type='farmer' and status='1'";
    $result2 = mysqli_query($con, $sq);

    $COUNT2 = mysqli_num_rows($result2);
    if ($COUNT2 > 0) {
        $id = mysqli_query($con, $sq);
        $row=mysqli_fetch_assoc($id);
        // echo $row['loginid'];
        $_SESSION['fid']=$row['loginid'];
        $_SESSION['logged_in'] = true;
        header("location: farmer.php");
    }

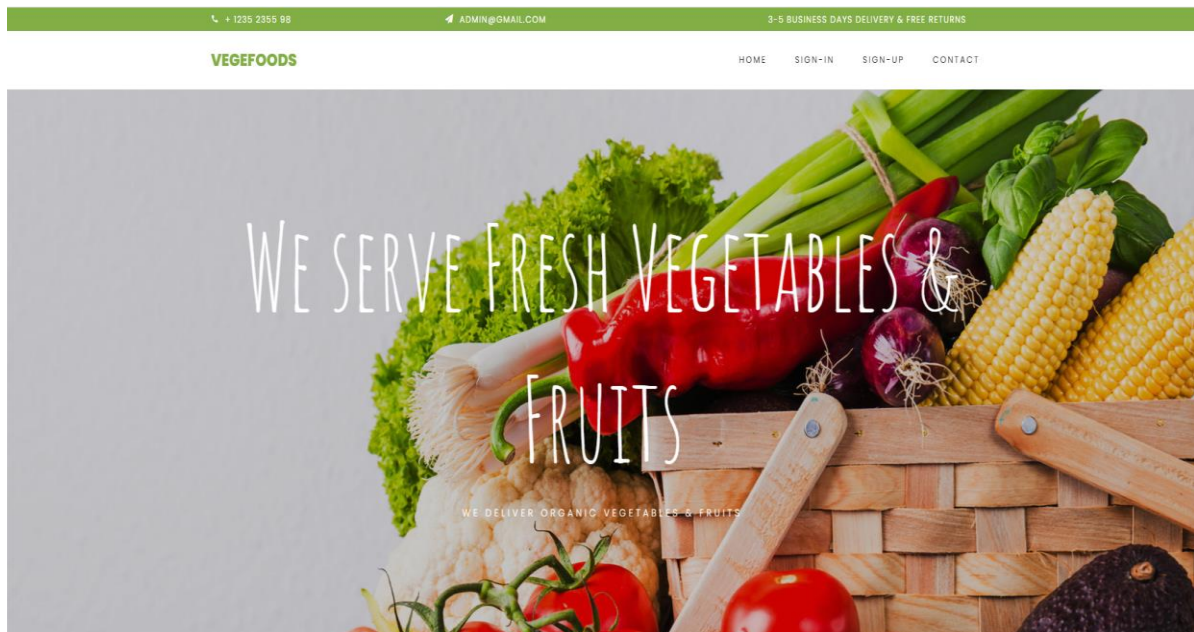
    else {

        echo "<script> alert('incorrect password or email');
window.location.href='login/log.php';</script>";
    }

?>
```

9.2 Screen Shots

Home page



Farmer ID Verification

Admin

- ☒ View Buyers
- ☒ Buyer Approval
- ☒ View Farmers
- ☒ Farmer Approval
- ☒ Add New Category
- ☒ View Category
- ☒ Add Product
- ☒ View Products
- ☒ Feedback
- ☒ Farmer ID_check
- ☒ Product Requests


Farmer ID Verification

Farmer Details	Details	Id-Proof	Action
Name: Anil James Farm-Name: Organic Farm Mail-id: anil@gmail.com Phone: 9605225239 Address: Amal Jyothi About Farm: xxxxxx		Approve	Reject


Farmer Product Request Approval

Admin

- ☒ View Buyers
- ☒ Buyer Approval
- ☒ View Farmers
- ☒ Farmer Approval
- ☒ Add New Category
- ☒ View Category
- ☒ Add Product
- ☒ View Products
- ☒ Feedback
- ☒ Farmer ID_check
- ☒ Product Requests

 ADMIN


Farmer Product Request Approval

Farmer Details	Product Name	Product Image	Action
Name: Nidhi Vinod Mail-id: nidhi@gmail.com Phone: 2147483647 Address: Kattill	Product Name: Green Apple		Approve Reject

Add New Category Page

Admin

- ☒ View Buyers
- ☒ Buyer Approval
- ☒ View Farmers
- ☒ Farmer Approval
- ☒ Add New Category
- ☒ View Category
- ☒ Add Product
- ☒ View Products
- ☒ Feedback
- ☒ Farmer ID_check
- ☒ Product Requests

 ADMIN

ADD NEW CATEGORY

Category Name

enter category

image

Choose File



No file chosen

Add Category

Add New Product Page

Admin

- ☒ View Buyers
- ☐ Buyer Approval
- ☒ View Farmers
- ☐ Farmer Approval
- ☐ Add New Category
- ☒ View Category
- ☐ Add Product
- ☒ View Products
- ☒ Feedback
- ☒ Farmer ID_check
- ☒ Product Requests

 ADMIN

Add products

Category: Select Category

Product name

Upload img

Choose File No file chosen

Add Product

Farmer Product Add Page

1235 2355 56

NEHALEX92@EMAIL.COM

3-5 BUSINESS DAYS DELIVERY & FREE RETURNS

E-FARM

HOME

ADD PRODUCT

VIEW PRODUCTS

VIEW DETAILS

MAKE A REQUEST

ORDER REQUESTS

LOGOUT

Hello Nidhi

ADD PRODUCTS

Category: Select Category

Products: Select Products

Product Images




Choose File No file chosen

Price/kg







Quantity(in kgs)

ADD

Farmer Added Product View

E-FARM						
+ 1235 2355 98		NEHALEX2@EMAIL.COM		3-5 BUSINESS DAYS DELIVERY & FREE RETURNS		
HOME	ADD PRODUCT	VIEW PRODUCTS	VIEW DETAILS	MAKE A REQUEST	ORDER REQUESTS	LOGOUT
Hello Nidhi						
Product View						
Category	Product	Product Image	Quantity	Price	Action	
Vegetable	potato		10	40	Edit	Delete
Vegetable	Onion		10	70	Edit	Delete
Vegetable	Carrots		8	52	Edit	Delete

Buyer Shopping Page

E-FARM						
+ 1235 2355 98						
MY ACCOUNT		HOME	SHOP	YOUR ORDERS	FEEDBACK	LOGOUT
Search..						
OUR PRODUCTS						
						
potato	Onion	Cherry				
Shop Now	Shop Now	Shop Now				
						
Strawberry	Carrots	Tomato				
Shop Now	Shop Now	Shop Now				



Buyer Cart View

+ 1236 2395 98

MY ACCOUNT

HOME SHOP YOUR ORDERS FEEDBACK * LOGOUT

My Cart

Farmer Name	Product Image	Product Name	Price/kg	Quantity/kg	Update	Total	Remove
Nidhi		potato	40	1	Update	40	x
Radhika		Cherry	40	1	Update	40	x

continue shopping

Billing address

Karthika

Email Address

kar@gmail.com

Contact Number

2147483647

Address

Kattill(H),Kottayam

Shopping cart

potato
Price: 40 Qty: 1 Subtotal: 40
Cherry
Price: 40 Qty: 1 Subtotal: 40

Your order


Product	Total
Grand Total	80

NOTE:

Payment on delivery is acceptable
Cash, Card and all type of digital payments are accepted
Once Placed there is no replace option

order now

9.3 PLAGIARISM REPORT

 **Similarity Report ID:** oid:10159:19994073

PAPER NAME
E-farm.pdf

WORD COUNT
8625 Words

CHARACTER COUNT
45236 Characters

PAGE COUNT
51 Pages

FILE SIZE
1.9MB

SUBMISSION DATE
Jul 21, 2022 2:22 PM GMT+5:30

REPORT DATE
Jul 21, 2022 2:24 PM GMT+5:30

● **11% Overall Similarity**


The combined total of all matches, including overlapping sources, for each database.

- 11% Internet database
- 0% Publications database
- Crossref database

● **Excluded from Similarity Report**

- Small Matches (Less than 25 words)

Summary

 **turnitin**

Similarity Report ID: oid:10159:19994073

● **11% Overall Similarity**

Top sources found in the following databases:


- 11% Internet database
- 0% Publications database
- Crossref database

TOP SOURCES

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	docshare.tips	3%
	Internet	
2	coursehero.com	2%
	Internet	
3	seminarprojects.com	1%
	Internet	
4	ijraset.com	<1%
	Internet	
5	dev.mysql.com	<1%
	Internet	
6	ir.jkuat.ac.ke	<1%
	Internet	
7	cloud-hosting.io	<1%
	Internet	
8	pavantestingtools.com	<1%
	Internet	
9	automationtestings.com	<1%
	Internet	

Sources overview

 **turnitin**

Similarity Report ID: oid:10159:19994073

10	classle.net Internet	<1 %
11	Marten Deinum, Koen Serneels, Colin Yates, Seth Ladd, Christophe Van... Crossref	<1 %
12	geeksforgeeks.org Internet	<1 %
13	scribd.com Internet	<1 %

Sources overview