CHAPTER 1.0
INTRODUCTION

1.1 OVER VIEW OF THE PROJECT

The problem at hand is to create a static and user-friendly website for a nursery plant business. This website should not only serve as an online platform for showcasing and sellingnursery plants but also provide valuable information and features to both customers and the business owners. The primary goal is to enhance the online presence and functionality of the nursery plant business, catering to the evolving needs of the digital age.

- **E-commerce Integration**

  Develop a secure and efficient e-commerce platform that allows customers to browse and purchase nursery plants with ease. Ensure secure payment processing and a seamless shopping experience.

- **Product Catalog Management**

  Implement a system to efficiently manage the plant catalog. This should include details about each plant, such as name, description, care instructions, pricing, and availability. The catalog should be easy to update as new plants arrive and existingones are sold.

- **Search and Filter Functionality**

  Create robust search and filtering options so customers caneasily find plants based on criteria like plant type, size, sun/shade preferences, and more.

- **User Accounts and Profiles**

  Enable customers to create accounts with profiles, order history,and wish lists. Implement user authentication and security measures to protect customer data.

- **Responsive Design**

  Ensure that the website is responsive and functions seamlessly across various devices, including desktops, tablets, and smartphones.

- **Information and Education**

  Provide valuable plant care information, gardening tips, and guides to educate customers. This content should be regularly updated to engage and retain visitors.

- **7.Customer Support**

  Integrate customer support features such as chatbots, FAQs, and contact forms to assist customers with enquiries and issues.

1.2 MODULES

- Admin login
- User login
- User registration form
- Feedback form
- Contact us
- Purchase history

1.2.1 MODULE DESCRIPTION

- **Admin Login**

  Admin login refers to the functionality provided for administrators or site owners to access the backend or administrative area of a website or system. It typically requires entering valid credentials, such as a username and password, to authenticate and gain access to administrative features. Admin login allows administrators to manage various aspects of the website or system, such as adding or editing content, managing users, configuring settings, and viewing analytics.

- **User Login**

  User login is a feature that enables registered users to access their accounts on a website or system. Users typically enter their credentials, such as a username/email and password, to authenticate and gain access to personalized features and content. User login allows users to perform actions like viewing their profile, accessing saved preferences, making purchases, or interacting with other users, depending on the functionality provided by the website or system.

- **User Registration Form**

  The user registration form is a web form provided to new users who wish to create an account on a website or system. It typically collects essential information from users, such as username, email address, password, and any additional details required for registration. Upon submitting the

registration form, the user's information is usually stored securely in a database, and the user is granted access to the website's features and services.

- **Feedback Form**

  A feedback form is a web form used to gather feedback and opinions from users about their experience with a website, product, or service. It typically includes fields for users to provide comments, ratings, suggestions, or specific feedback on various aspects of their experience. Feedback forms help website owners or businesses gather valuable insights from users, identify areas for improvement, and enhance user satisfaction and engagement.

- **Contact Us**

  The "Contact Us" page or feature provides users with a way to reach out to the website or business for inquiries, support, or general communication. It typically includes contact information such as email addresses, phone numbers, physical addresses, and sometimes a contact form where users can submit their messages directly through the website.

- **Purchase History**

  Purchase history refers to a record of past transactions or purchases made by a user on a website or system. It typically includes details such as the date of purchase, item(s) purchased, quantity, price, payment method, and order status. Purchase history allows users to track their previous purchases, view order details, reorder items, and monitor their transaction history.

1.3 COMPANY PROFILE

E-Growth Solutions, established in 2018, uniquely combines expertise in CCTV security solutions with a deep commitment to sustainable agriculture. Specializing in the installation and maintenance of cutting-edge CCTV systems for residential, commercial, and industrial clients, we also offer a diverse range of high-quality plants, seeds, fruits, and vegetables. Our mission is to create a harmonious balance between security and nature, delivering state-of-the-art security systems while promoting environmentally friendly agricultural practices. Our core values of expertise, sustainability, and innovation drive us to provide exceptional service quality, reliable security solutions, and premium agricultural products that cater to the needs of our clients and support sustainable living.

CHAPTER 2.0

SYSTEM STUDY AND ANALYSIS

2.1 Existing System

The current system for shopping at nursery plants usually involves visiting physical stores with limited plant options available. Customers often struggle to find the specific plants they're looking for and may not have access to detailed information about them. Additionally, it's hard for customers to share feedback or ask questions easily because there aren't convenient communication channels in place. This lack of accessibility and communication can lead to lower customer satisfaction and engagement with the nursery.

2.1.1 Disadvantages of Existing System

- Limited accessibility: Physical stores require customers to visit in person, restricting access for those unable to travel.
- Lack of comprehensive management features: Online platforms may lack advanced inventory control, user authentication, and order processing capabilities, leading to operational inefficiencies.
- Inconvenient shopping experience: Customers may face challenges navigating through limited product listings or dealing with cumbersome checkout processes, resulting in a subpar shopping experience.

2.2 Proposed System

The proposed system is an nursery shopping website will feature user-friendly navigation and a visually appealing interface to enhance the browsing experience for customers. The "About Us" page will offer insights into the nursery's background, values, and commitment to providing high-quality products and exceptional service. The categorized product catalog will allow customers to easily browse through different plant varieties, seeds, gardening tools, and accessories, ensuring they find exactly what they're looking for. The feedback and gift section will enable customers to share their thoughts, suggestions, and requests, fostering a sense of community and engagement. Furthermore, the contact page will provide multiple channels for communication, including an enquiry submission form, email, and phone number, ensuring that customers can reach out with any inquiries or concerns they may have. Overall, the website aims to revolutionize the nursery shopping experience, offering convenience, accessibility, and a seamless platform for customers to explore and connect with the nursery's offerings.

2.2.1 Advantages of Proposed System

- Enhanced accessibility to a wide range of nursery plants and related products.
- Detailed information about the shop's background, values, and product offerings.
- Convenient channels for customer feedback, suggestions, and gift requests.
- Improved customer satisfaction and engagement.
- Time and cost savings for customers by eliminating the need for physical visits to stores.
- Streamlined checkout process for faster and hassle-free transactions.
- Regular updates on new arrivals, promotions, and special offers to keep customers informed.
- Integration with social media platforms for enhanced engagement and community building.

CHAPTER 3.0

SYSTEM SPECIFICATION

3.1 HARDWARE REQUIREMENTS

COMPONENTS	REQUIREMENTS
PROCESSOR	AMD RYZEN 3(5000)
Memory	4GB or above (RAM)
Storage	50 GB (Minimum)

3.2 SOFTWARE REQUIREMENTS

COMPONENTS	REQUIREMENTS
OPERATING SYSTEM	Windows 11(version 23H2)
FRONT-END	HTML,Java script,CSS
BACK-END	PHP(version: 8.2.0)
FRAMEWORK	Bootstrap
Database	MYSQL(version: 8.3.0)
WEB BROWSER	Google/Chrome/Edge/Firefox/Brave (Etc.)

SOFTWARE SPECIFICATION

This project is developed by using HTML,CSS,JavaScript front-end and with back-end of MY SQL. Server and PHP and this application is tested in Windows platform.

3.3.1 PHP

- **DEFINITION OF PHP**

PHP can be defined as a programming language for Database access from the web's browser.

In other words, it is an HTML-embedded scripting language. It focuses on the logic of how a page responds to user input and not how the page looks that i.e. not the primary appearance of the page.

PHP runs on the server side, which means that the web server that sends an HTML file to a user's browser, will carry out the instructions found in the embedded PHP code first, and then send the output of the PHP code along with the HTML code. The result is a webpage with dynamic content.

➢ BRIEF HISTORY ON PHP

PHP is a language for creating website that can be more or less interactive. It 350was created in 1994 by Rasmus Lerdorf who was a software engineer and who was part 15 of the Apache Team. In the same year, he created a package, added some database support and called it PHP/FI (Form Interpretation).

In 1995, it was called the Personal Home Page Tool then was released as version2 with a name called PHP/FI (a form interpreter responsible for analyzing queries). In mid of 1997, more than 50,000 websites began using PHP and in October. 1998, there was an increase in the number of websites using PHP which was about 100,000.In 2000, there was a release of PHP 4.0.2. And currently over 1,000,000 sites in the whole world are using PHP.

➢ PHP AND ITS USES

PhP can help read and write files. It also can do basic files and directory maintenance; therefore it basically can help one in editing documents. It can also take content that can be used in the generation of files in various formats which can include HTML (Hypertext Markup Language) and PDF.

It also can help manage graphical content which include charts. Not only can it do the above but can it also read, write information in a database. You can make a PHP script to run it without any server or browser. You only need the PHP interpreter to use it. PHP's abilities include outputting images, PDF files, and even Flash movies. PHP can help also output easily any text, such as XML.

➢ ADVANTAGES AND DISADVANTAGES OF PHP

It is more or less cost-free in other words, PHP is an Open Source solution, freely available for a wide variety of platforms. It is also easy, as it's a combination of Cand Perl. The strongest and

most significant feature of PHP is its native database supportfor a wide range of databases for example (MySQL, mSQL., Oracle), which allows access to the databases directly through SQL statements.

There is a cross-platform compatibility (Windows, Macintosh, or a version of Unix.): Compiled and is built on more than 25 platforms. With PHP, you have 'freedom of choice regarding an operating system and a Web server. The error handling is not as sophisticated as in ASP (Active Server Pages).

3.3.2 CASCADING STYLE SHEET(CSS)

- **What are Cascading Style Sheets?**

Cascading Style Sheets (CSS) are a collection of rules we use to define and modify web pages. CSS are similar to styles in Word. CSS allow Web designers to have much more control over their pages look and layout. For instance, you could create a style that defines the body text to be Verdana, 10 point. Later on, you may easily change the body text to Times New Roman, 12 point by just changing the rule in the CSS. Instead of having to change the font on each page of your website, all you need to do is redefine the style on the style sheet, and it will instantly change on all of the pages that the style sheet has been applied to. With ITMI. styles, the font change would be applied to each instance of that font and have to be changed in each spot. CSS can control the placement of text and objects on your pages as well as the look of those objects.

HTML information creates the objects (or gives objects meaning), but styles describe how the objects should appear. The IHTML gives your page structure, while the CSS creates the "presentation". An external CSS is really just a text file with a .css extension.These files can be created with Dreamweaver, a CSS editor, or even Notepad,The best practice is to design your web page on paper first so you know where you will want to use styles on your page. Then you can create the styles and apply them to your page.

- **Css Rules**

A Style Sheet is made up of Rules. A Rule is one or more properties that will be applied to one or more elements. Each rule will have a selector and a definition. A Style Sheet can be collection of Styles; however, a style sheet could have just one style. A style rule consists of a selector and the properties or attributes of that selector that are to be affected. These properties or attributes are contained within curly brackets and called the style definition. For example in the style rule: h2 {color: red; font-family: Arial;), h2 is the selector and the rest (within the curly brackets) is the style

definition. The standards of the style rules are that each attribute is followed by a colon (:) followed by the value for that attribute. Then each set (of attribute and value) is followed by a semicolon (:). Several style attributes can be assigned at one time to a tag or selector or multiple tags. For example you could specify the font family and color of all the heading tags at once (with a comma between each one) and then give each one a separate size
in a different rule.

Several tags being defined at once:

```
h1,h2,h3,h4
{
        color:#ff3300;
        font-family:Arial;
}
```

Then define the size of this one tag separately:

```
hl
{
        font-size: 160%;
}
```

To define a style using Dreamweaver, you can either redefine the predefined tags such as the Heading 1 tag. Heading 2 tag, the paragraph tag, etc. or define a custom style that you create. The tag or the custom style is the selector. One custom style is called a class. Classes can be used many times. Another custom style is an ID. An ID is similar to a class, but it can only be used once. You can also create specific styles by combining selectors. For example you could set up a sidebar for News and specify that only paragraph text within that section should be a particular color or size

```
news p {
font-size: 80%;
color: blue;
}
```

- **CSS Styles**

There are three kinds of Styles: Inline, Embedded (or Internal), and External (or Linked). One or more types of styles can be used for each web page.Inline styles are very similar to HTML styles in that they are placed in the <body> of your document and good for only that one instance. The style

definition must be marked by **<style=attributes:values;>** tags. This type is not commonly used because they are specific to each instance and would be time consuming to change and could not be used for other pages in your site.

Embedded / Internal styles are placed in the <head> of your document with the tag **<styletype="text/css">tags.**

The selector and the definition will be enclosed by comment markers **<!--→**and end with **</style>** and will be good for the whole page. Comment markers are good for documentation or to hide information from older browsers.

An External (Linked) Style Sheet is a totally separate document from the HTML, document and can be linked to and utilized by many pages Create the link in this manner:

<link rel="stylesheet" type="text/css"

href="full/path/to/stylesheet.css">

This is telling the browser that the link relates to a CSS style sheet, it is in text and the path to the sheet follows the href tag.

The external style sheet contains no HTML tags. HTML tags in a style sheet can cause it to not function properly. It even has it's own comment tag:

/* CSS Comment */

Using External Style Sheets takes advantage of one of the most powerful features of CSS which is the ability to make changes to the style and have it immediately change all of your web pages. For example if you have a particular color scheme that needs to be changed you would just have to change the properties in the style sheet and all of the pages linked to that sheet would be changed. Also if you create a style and reuse it several times you shrink the size of your file. Therefore, the External style sheet is the type of style sheet used most often.

3.3.3 JavaScript

- **Introduction to JavaScript**

JavaScript is a dynamic and versatile programming language widely used for web development. It runs directly in web browsers, enabling developers to create interactive and responsive user interfaces.

- **Key Features and Capabilities**

JavaScript empowers developers to add functionality to web pages, manipulate HTML and CSS, handle events, and interact with servers asynchronously. Its ability to run on the client-side allows for real-time updates without page reloads.

- **Applications and Versatility**

JavaScript is utilized not only for client-side development but also for server-side development through platforms like Node.js. Its flexibility makes it a popular choice for building full-stack applications, enabling developers to write both client-side and server-side code in the same language. JavaScript's wide adoption and robust ecosystem make it indispensable in modern web development.

3.3.4 MySQL

- **ABOUT MYSQL**

MYSQL is an open-source relational database management systems (RDBMS), is developed, distributed and supported by MYSQL AB. MYSQL is a popular choice of database for use in web applications. MYSQL can be scaled by deploying it on more powerful hardware, such asa multi-processor server with gigabytes of memory. MYSQL is easy to use, yet extremely powerful, secure, and scalable. And because of its small size and speed, it is the idea database solution for Web sites.

- **MYSQL IS A DATABASE MANAGEMENT SYSTEM**

A database is a structured collection of data. It may be anything from a simple shopping list toa picture gallery or the vast amount of information in a corporation network. To add, access and process data stored in a computer database we need a database management system such as MYSQL server. Since computers are very good at handling large amount of data, database management system plays a central role in computing.

- **MYSQL IS A RELATIONAL DATABASE MANAGEMENT SYSTEM**

A relational database stores separate data in separate tables rather than putting all the data in

one big storeroom. This adds speed and SQL is the most common standardize language used to access database and is defined by the ANSI/ISO SQL standard. The SQL standard has beenevolving since 1986 and several versions exist.

- **MYSQL SOFTWARE IS OPEN SOURCE**

Open source means that it is possible for anyone to use modify the software. Anybody can download the MYSQL software uses the GPL (GNU General Public License), to define what we may and may not use do with the software.

- **FEATURES OF MYSQL**

- **Client/server Architecture:** MYSQL is a client/server system. There is a database server (MYSQL) and arbitrarily many clients (application programs), which communicate with the server. The clients can run on the same computer as the server or on another computer.
- **Speed:** MYSQL is considered a very fast database program.
- **SQL Compatibility:** As before said SQL is a standardized language for querying and updating data and for the administration of a database. Through the configuration setting solmode we can make the MYSQL server behave for the most part compatibly with various database systems.
- **Stored procedures:** Stored procedures (SPs for short) are generally used to simplify steps such as inserting or deleting a data record.
- **Triggers**: Triggers are SQL commands that are automatically executed by the server in certain database operations INSERT, UPDATE, and DELETE, MYSQL has supported triggers.
- **Replication:** Replication allows the contents of a database to be copied (replicated) onto a number of computers to increase protection against system and to improve the speed of database queries.

CHAPTER 4.0

SYSTEM DESING AND DEVELOPMENT

4.1 INPUT DESIGN

- **User Forms**
  - **Registration Form:** Users provide necessary information to create an account, including username, password, email, etc.
  - **Login Form:** Users input their credentials to access their accounts.
  - **Checkout Form:** During the purchasing process, users input shipping details, payment information, etc.
  - **Search Bar:** Allows users to input keywords for product searches.
- **Shopping Cart Actions**
  - **Add to Cart:** Users input the quantity of desired products to add them to their shopping carts.
  - **Remove from Cart:** Allows users to delete items from their carts.
  - **Update Quantity:** Users may adjust the quantity of items in their cart.
- **Search and Filtering**
  - **Search Input:** Users provide keywords or phrases to search for specific products.
  - **Filter Inputs:** Users can input criteria like price range, brand, and features to refine product searches.

4.2 OUTPUT DESIGN

- **Dynamic Content**
  - **Product Listings:** Display information about CCTV cameras, including images, descriptions, and prices.
  - **User Account Information:** Display user details, order history, and account settings.
  - **Shopping Cart Summary:** Show selected items, quantities, and total prices.

- **Feedback and Reviews**
  - **Product Reviews:** Display user-generated reviews and ratings for each product.
  - **Confirmation Messages:** Output messages confirming actions like successful login, purchase, or account creation.
- **Responsive Design**
  - **Adaptation to Different Devices:** Output changes in layout and design based on the user's device, ensuring a consistent experience.
- **Error Messages**
  - **Validation Errors:** Output messages for incomplete or incorrect form submissions.
  - **Transaction Errors:** Display messages for issues during the payment process.
- **Admin Panel Outputs**
  - **Product Management:** Output product details, inventory status, and administrative options.
  - **User Management:** Display user information and administrative actions.

4.3 TABLE DESIGN

User Table

Foreign key: id, Email

Column Name	Data Type	Constraints	Description
Id	INT	FOREIGN KEY	User id
full_name	VARCHAR(20)	NOT NULL	User full name
Email	VARCHAR(25)	FOREIGN KEY	User email
Password	VARCHAR(20)	NOT NULL	User password
Login_time	Int	NOT NULL	User login time

Products Table

Column Name	Data Type	Constraints	Description
Date	VARCHAR(20)	NOT NULL	Purchase date
Time	INT	NOT NULL	Purchase time
Item	VARCHAR(100)	NOT NULL	Item purchased
Price	DECIMAL(10, 2)	NOT NULL	Item price

Enquirie Table

Foreign key: id ,Email

Column Name	Data Type	Constraints	Description
Id	INT	FOREIGN KEY	User id
Name	VARCHAR(20)	NOT NULL	User full name
Address	VARCHAR(100)	NOT NULL	User address
Email	VARCHAR(25)	FOREIGN KEY	User email
Phone	INT(10)	NOT NULL	User phone number
Enquiry	VARCHAR(100)	NOT NULL	User enquiry
created_at	INT	NOT NULL	User login time

Admin User Table

Primary key: user_name

Foreign key: id

Column Name	Data Type	Constraints	Description
Id	INT	FOREIGN KEY	Admin id
user_name	VARCHAR(20)	PRIMARY KEY	Admin full name
Password	VARCHAR(20)	NOT NULL	Admin password
Name	VARCHAR(20)	NOT NULL	Admin name

Feedback Table:

Primary key: name.email

Foreign key: id

Column Name	Data Type	Constraints	Description
id	INT	FOREIGN KEY	User id
name	VARCHAR(20)	PRIMARY KEY	User full name
email	VARCHAR(25)	FOREIGN KEY	User email
phone	INT(10)	NOT NULL	User phone number
feedback	VARCHAR(100)	NOT NULL	User feedback

4.4 ENTITY RELATIONSHIP DIAGRAM

ER Diagram stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases. ER diagrams are created based on three basic concepts:entities, attributes and relationships. It is a graphical representation that depicts relationships among people, objects, places, concepts or events within an information technology (IT) system. Entity relationship diagrams provide a visual starting point for database design that can also be used to help determine information system requirements throughout an organization. An ERD usesdata modelling techniques that can help define business processes and serve as the foundation fora relational database.

Following are the main components and its symbols in ER Diagrams:

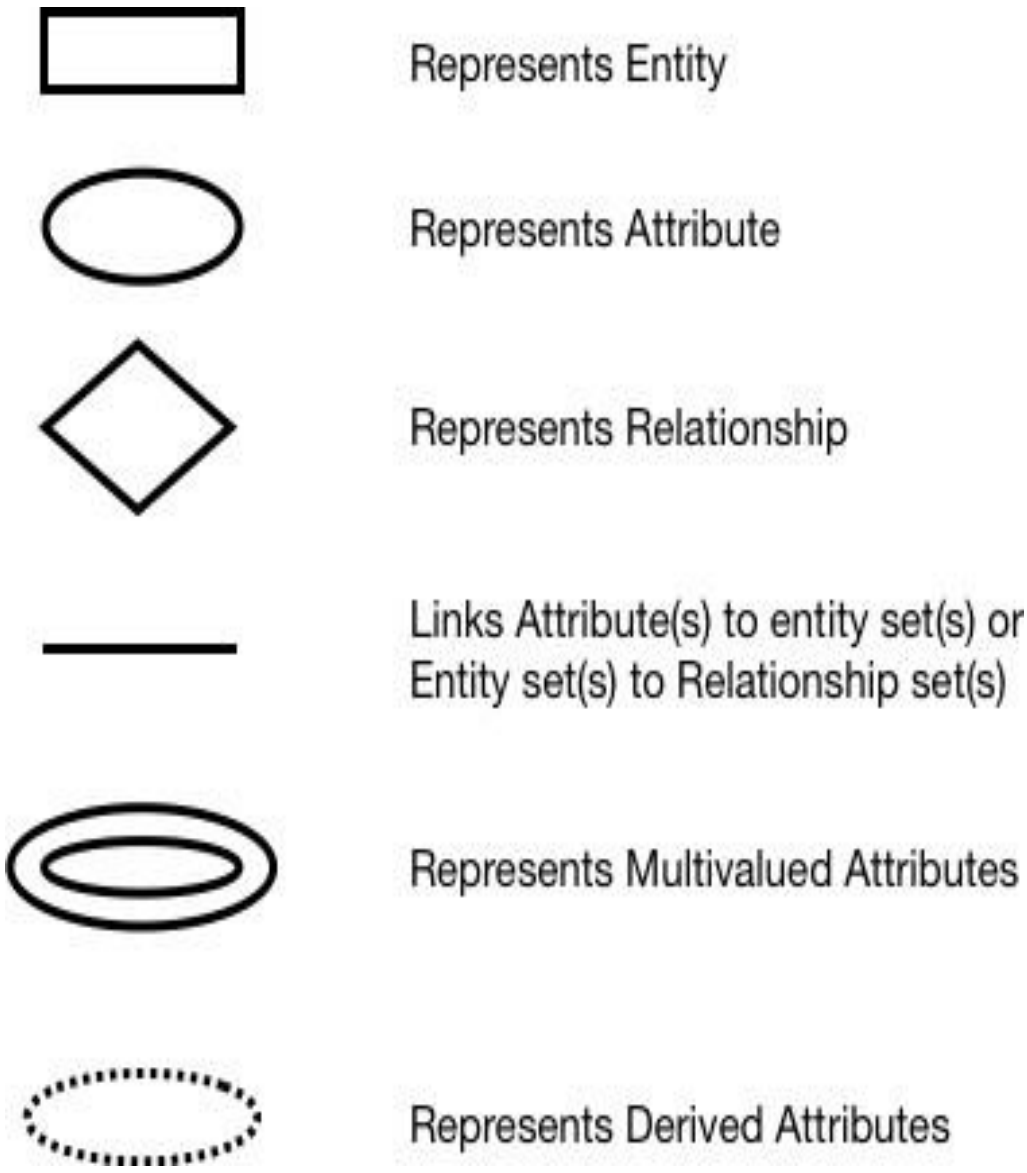


Fig 4.4.1 ER Symbols

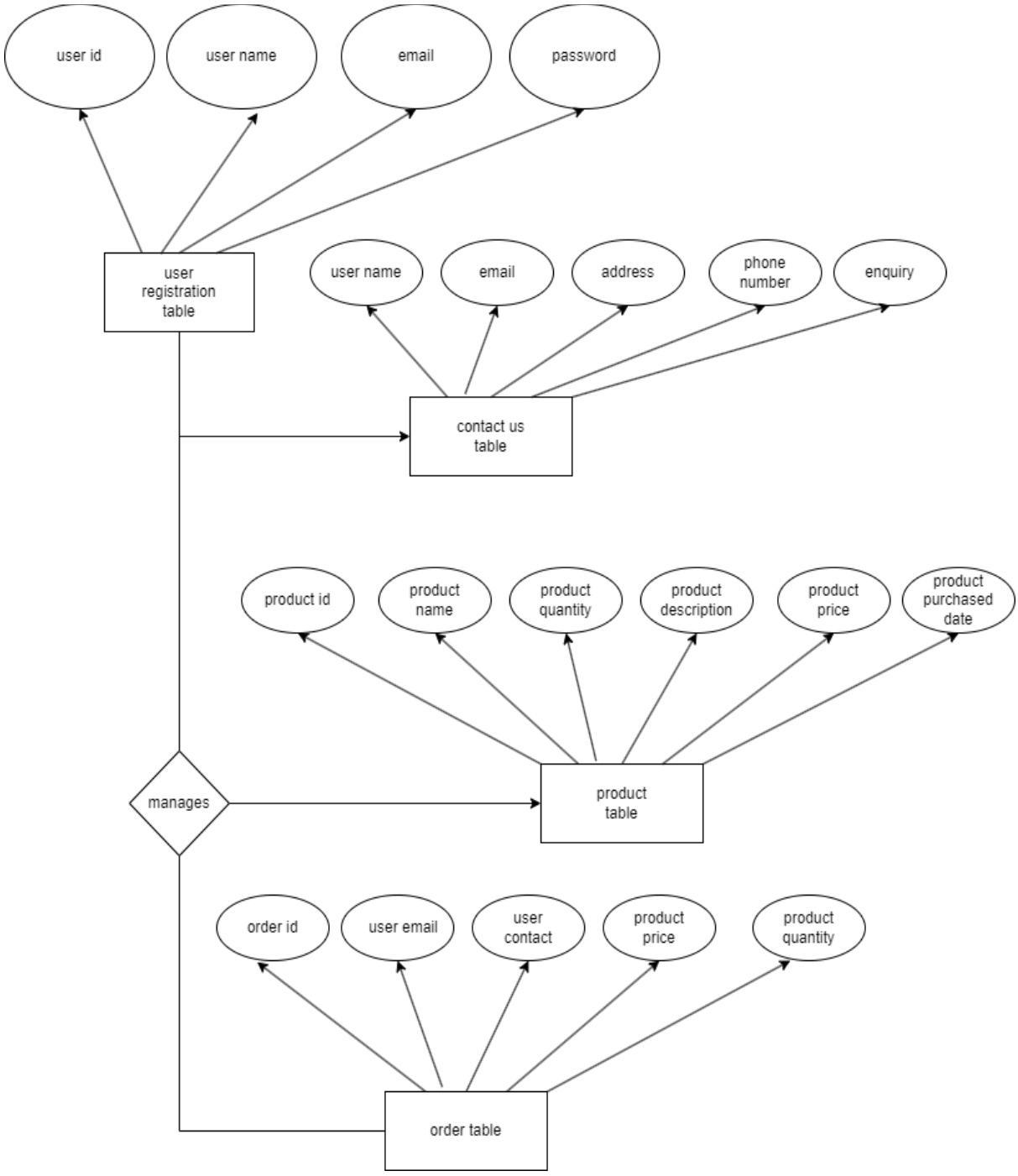


Fig 4.4.2 ER diagram

4.5 DATA FLOW DIAGRAM

A data flow diagram also known as “bubble chart” has the purpose of clarifying system requirement and identifying major transformation that will become program in system design. Therefore, it is the starting point of the phase that functionally decomposes the requirement specification down to the lowest level of details. A DFD contains series of bubbles joined by lines.

DFD SYMBOLS:

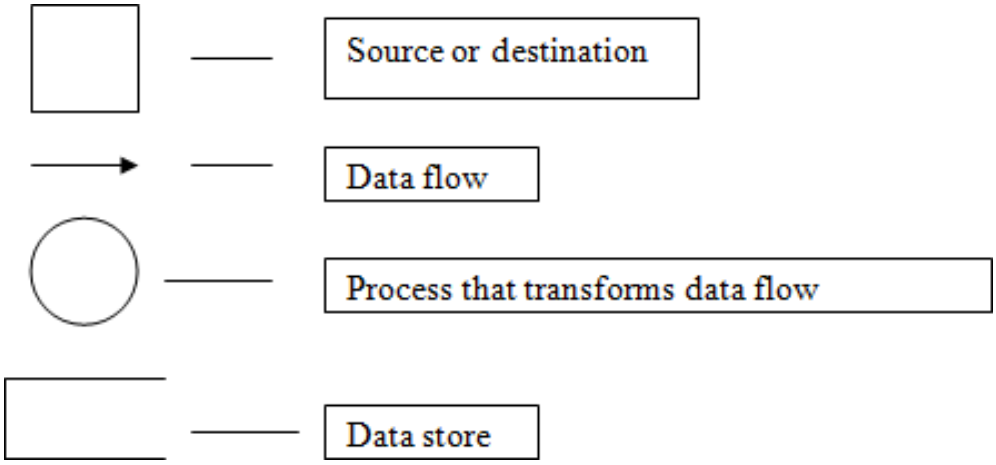


Fig 4.5.1 Data flow diagram symbols

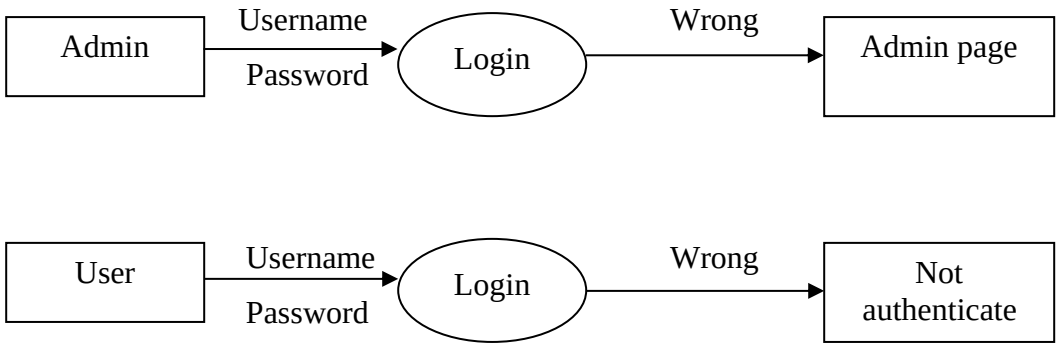


Fig 4.5.2 Level 0

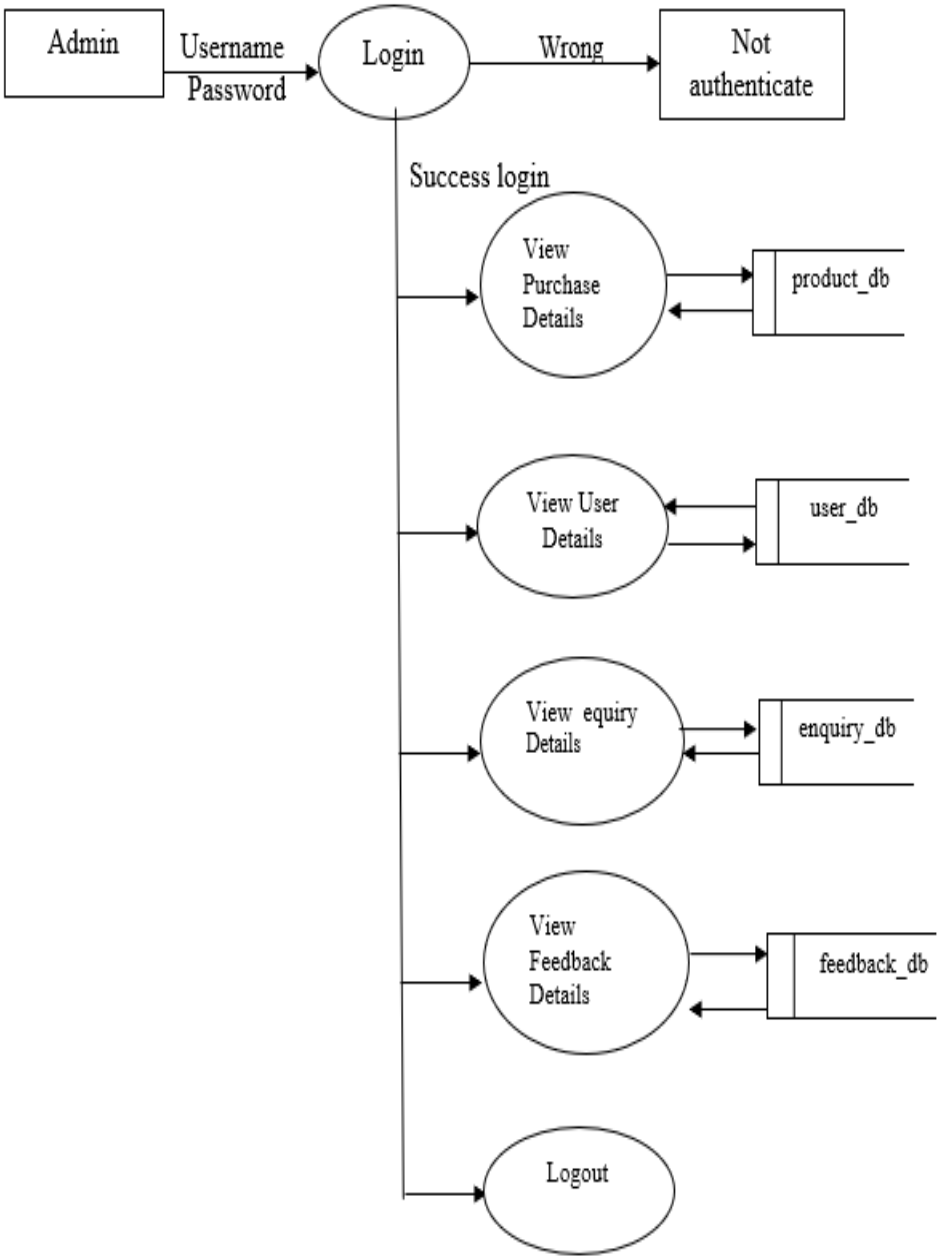


Fig 4.5.3 Level 1

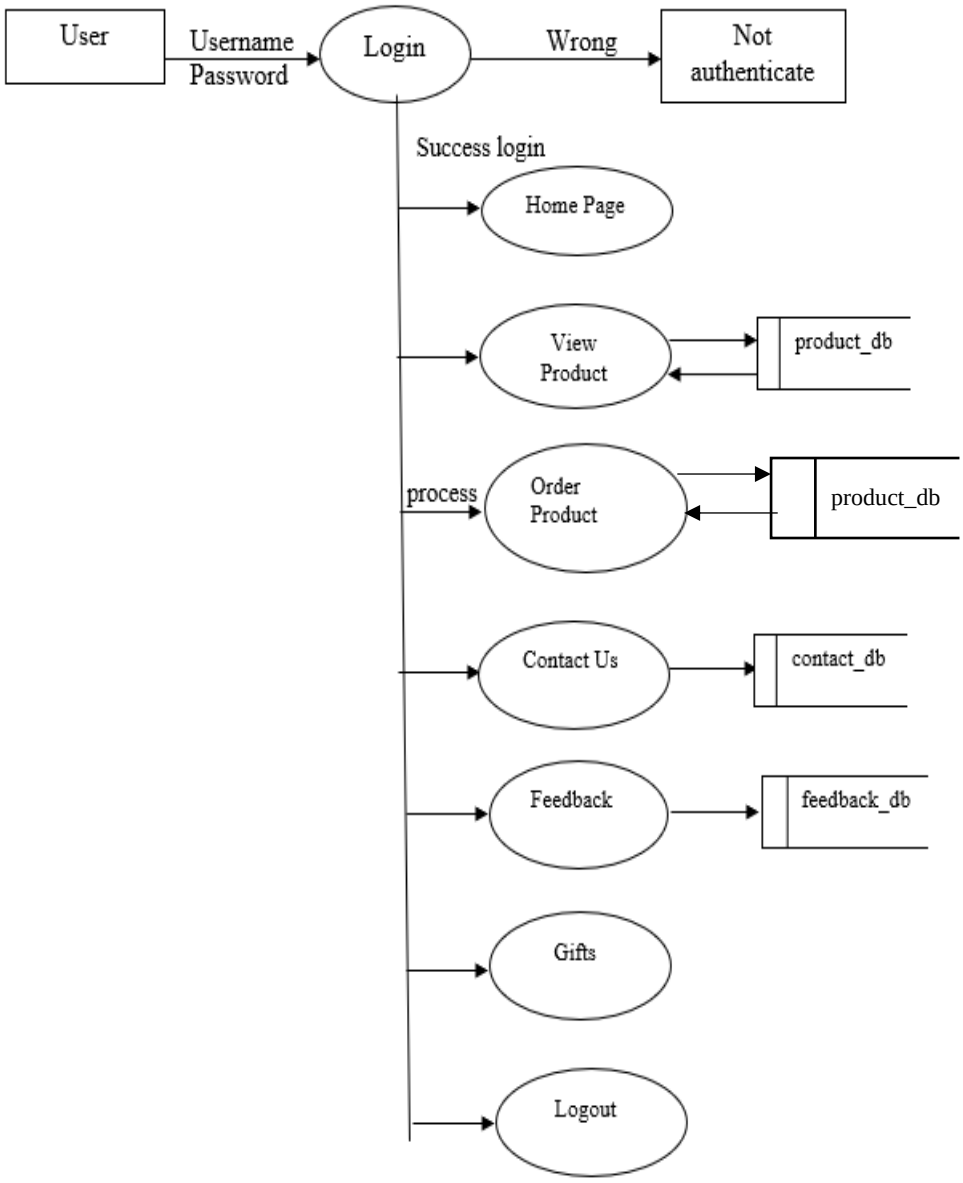


Fig 4.5.4 Level 2

4.6 SYSTEM FLOW DIAGRAM

A flowchart is a type of diagram that represents a workflow or process. A flowchart can also be defined as a diagrammatic representation of an algorithm, a step-by-step approach to solving atask. Flowcharts are widely used in various fields, including software development, engineering, business process management, and project planning, to provide a clear and structured overview ofhow a particular task or process is organized and executed.

FLOW CHART SYMBOLS

Flow chart symbols and their descriptions are given as follows:

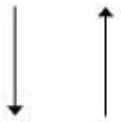
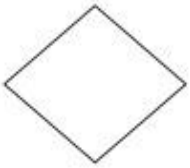
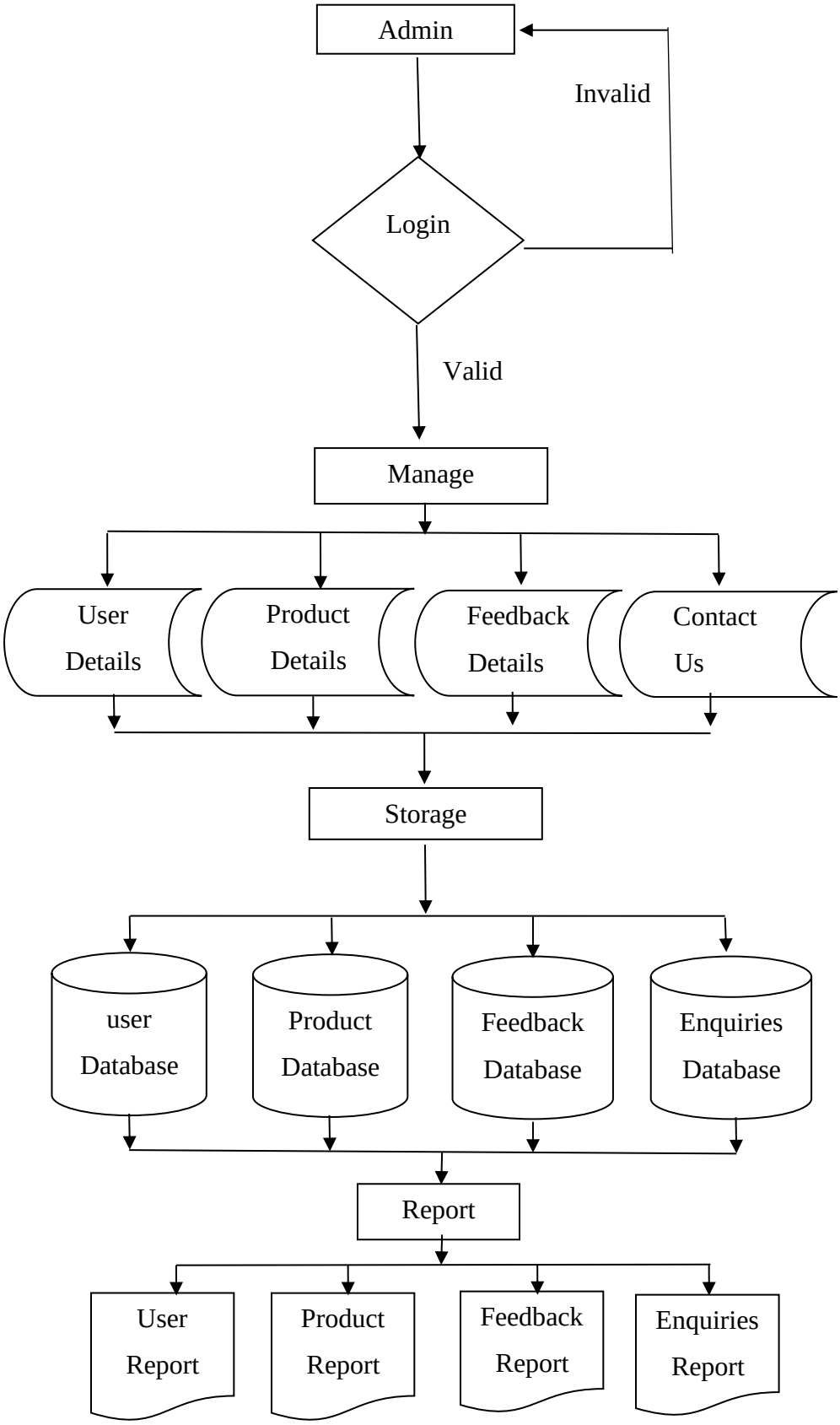
Flowchart Symbol	Symbol Name	Description
	Terminal (Start or Stop)	Terminals (Oval shapes) are used to represent start and stop of the flowchart.
	Flow Lines or Arrow	Flow lines are used to connect symbols used in flowchart and indicate direction of flow.
	Input / Output	Parallelograms are used to read input data and output or display information
	Process	Rectangles are generally used to represent process. For example, Arithmetic operations, Data movement etc.
	Decision	Diamond shapes are generally used to check any condition or take decision for which there are two answers, they are, yes (true) or no (false).
	Connector	It is used connect or join flow lines.

Fig 4.6.1 System Flow chart symbols



+

Fig 4.6.2 System Flow chart diagram

4.7 USECASE DIAGRAM

A use case diagram is a graphical representation in the Unified Modeling Language (UML) that illustrates the functional requirements and interactions of a system from the perspective of itsusers (or "actors"). Use case diagrams are widely used in software engineering and system designto document and visualize the various ways users interact with a system or software application.

Following are the main components and its symbols in Use Case Diagrams:

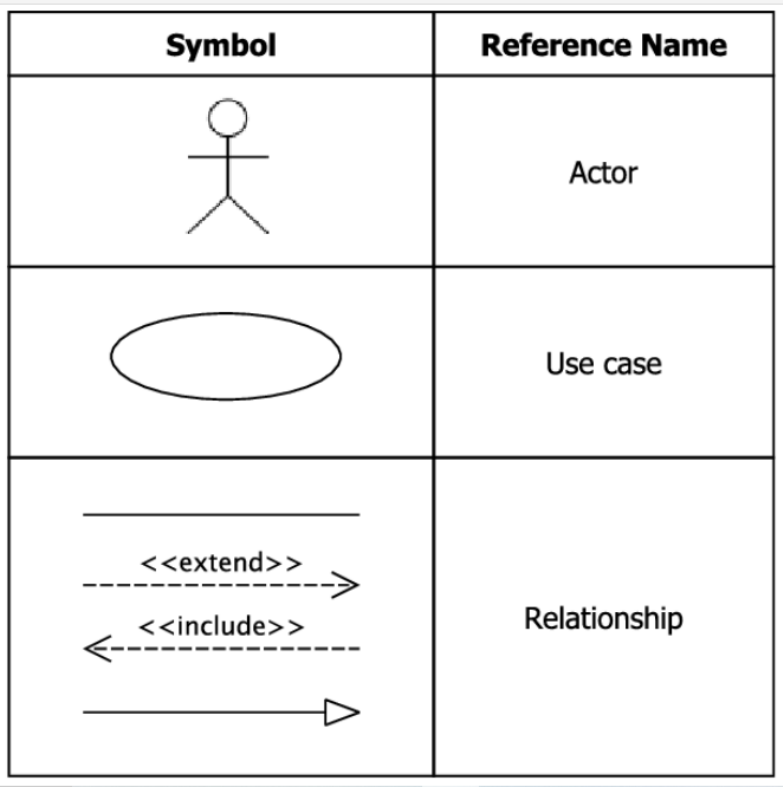


Fig 4.7.1 Use case symbols

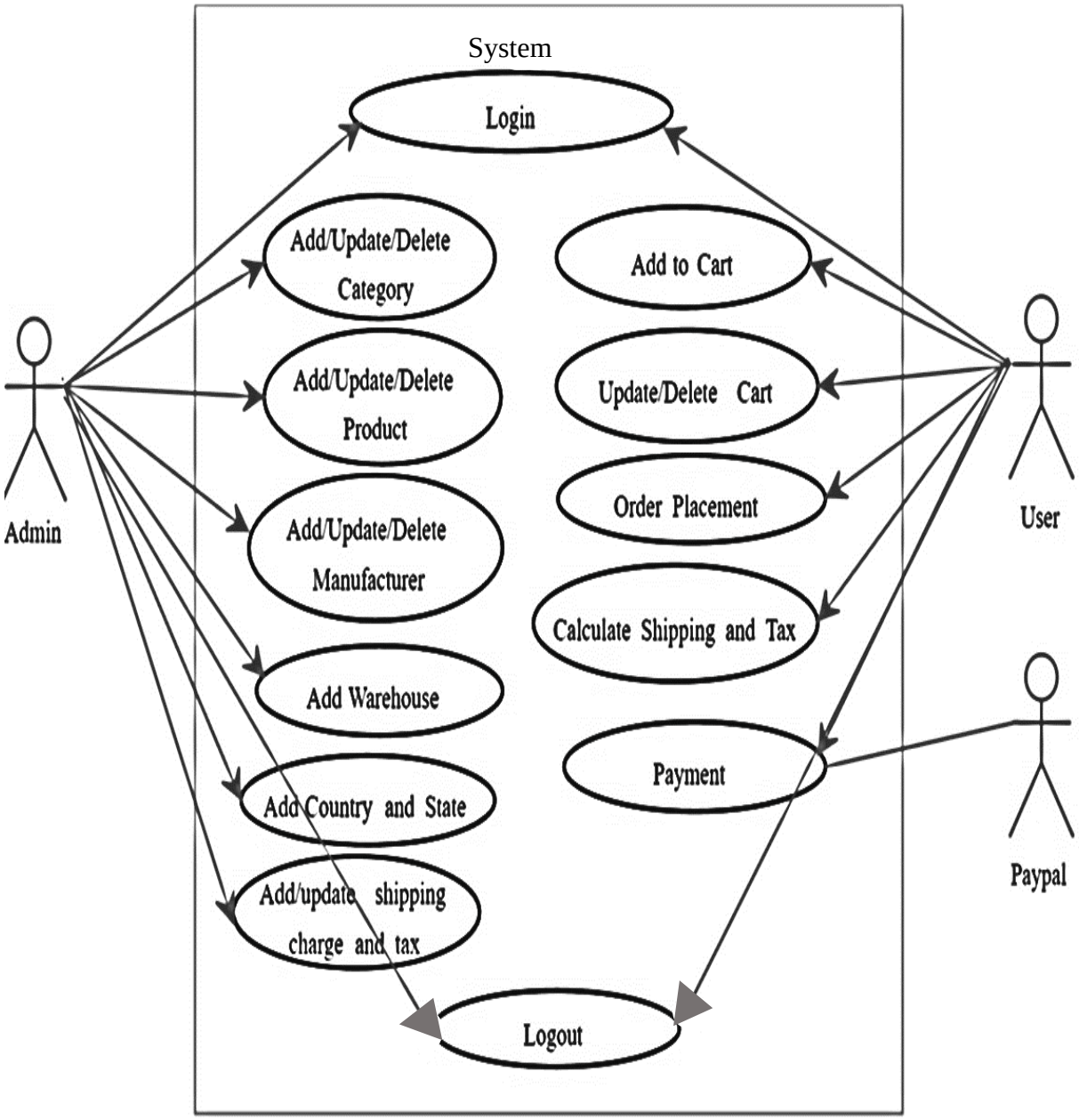


Fig 4.7.2 Use case Diagram

CHAPTER 5.0

SYSTEM TESTING AND IMPLENTATION

5.1 SYSTEM TESTING

Testing is carried out after the development of the proposed system. The principle activity of system development is preparing the source code. In this system the source code is developed for each module separately. The source code is prepared for master files and they are compiled and corrected. Then the source code for the transaction files are prepared, compiled and corrected. Then the modules are combined and corrected as a whole module.

A strategy for software testing must accommodate low-level tests that are necessary to verify that all small source code segments has been correctly implemented as well as high-level tests that validate major system functions against customer requirements. Testing is a process of executing program with the intent of finding error. A good test case is one that has high probability of finding an undiscovered error. If testing is conducted successfully it uncovers the errors in the software. Testing cannot show the absence of defects, it can only show that software defects present. Test configuration includes test plan and test cases and test tools.

- **TESTING OBJECTIVES**

  Software Testing has different goals and objectives. The major objectives of Software testing are as follows:

  - Finding defects which may get created by the programmer while developing the software.
  - Gaining confidence in and providing information about the level of quality.
  - To prevent defects.
  - To make sure that the end result meets the business and user requirements.
  - To ensure that it satisfies the BRS that is Business Requirement Specification and SRS that is System Requirement Specifications.
  - To gain the confidence of the customers by providing them a quality product

- **Testing methodologies**

  Testing methodologies are the strategies and approaches used to test a particular product to ensure it is fit for purpose. Testing methodologies usually involve testing that the product works in accordance with its specification, has no undesirable side effects when used in ways outside of its design parameters and worst case will fail-safely (e.g. a nuclear reactor will shut down on failure).

5.1.1 Unit testing

Unit testing is essential for the verification of the code produced during the coding phase and hence the goal is to test the internal logic of the modules. Using the detailed design description as a guide, important paths are tested to uncover errors within the boundary of the modules. These tests were carried out during the programming stage itself.

5.1.2 Integration testing

Integration testing is a systematic technique for constructing the program structure while at the same time conducting tests to uncover error associated within the interface. The objective is to take unit tested modules and build a program structure that has been dictated by design. All modules are combined in this step. The entire program is tested as whole. And chaos in interfaces may usually result. A set of errors is encountered in such a case.

5.1.3 Validation testing

Here in the validation testing we want to check whether the given conditions to the text box are working correctly. Because in the name place we want to enter the characters and the special symbols only we should not enter the numbers in the name field. Here while on runtime we entered numeric values in the string specified columns of product inwards. It raises error. In this phase each module has been tested by wrong inputs, for example Employee Name should be a character as well as their age should be in numbers.

- **Functional testing**

  The functional testing part of a testing methodology is typically broken down into four components - unit testing, integration testing, system testing and acceptance testing – usually executed in this order. Entire system is working properly or not will be tested here, and specified path connection IS correct or not, and giving output or not are tested here these verifications and validations are done by giving input values to the system and by comparing with expected output.

5.1.4 White box testing

White box testing (also known as Clear Box Testing, Open Box Testing, Glass Box Testing, Transparent Box Testing, Code-Based Testing or Structural Testing) is a software testing method in which the internal structure/design/implementation of the item being tested is known to the tester. The tester chooses inputs to exercise paths through the code and determines the appropriate outputs.

Programming know-how and the implementation knowledge is essential. White box testing is testing beyond the user interface and into the nitty-gritty of a system.

This method is named so because the software program, in the eyes of the tester, is like a white/transparent box; inside which one clearly sees.

Definition by ISTQB

- **White-box testing:** Testing based on an analysis of the internal structure of the component or system.
- **White-box test design technique:** Procedure to derive and/or select test cases based on an analysis of the internal structure of a component or system.

5.1.5 Black box testing

Black box testing, also known as Behavioral Testing, is a software testing method in which the internal structure/design/implementation of the item being tested is not known to the tester. These tests can be functional or non-functional, though usually functional.

This method is named so because the software program, in the eyes of the tester, is like a black box; inside which one cannot see. This method attempts to find errors in the following categories:

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

Definition by **ISTQB**

- **Black box testing:** Testing, either functional or non-functional, without reference to the internal structure of the component or system.
- **Black box test design technique:** Procedure to derive and/or select test cases based on an analysis of the specification, either functional or non-functional, of a component or system without reference to its internal structure.

- **Acceptance testing**

  This testing is done to verify the readiness of the system for the implementation. Acceptance testing begins when the system is complete. Its purpose is to provide the end user with the confidence that the system is ready for use. It involves planning and execution of functional tests, performance tests and stress tests in order to demonstrate that the implemented system satisfies its requirements. Tools to special importance during acceptance testing include:

- **Test coverage Analyzer**

  Records the control paths followed for each test case.

- **Timing Analyzer**

  Also called a profiler, reports the time spent in various regions of the code are areas to concentrate on to improve system performance.

5.2 SYSTEM IMPLEMENTATION

Implementation is the stage in the project where the theoretical design is turned into a working system and is giving confidence on the new system for the users that it will work efficiently and effectively. It involves careful planning, investigation of the current system and its constraints on implementation, design of methods to achieve the change over, an evaluation of change over methods. Apart from planning major task of preparing the implementation are education and training of users. The implementation process begins with preparing a plan for the implementation of the system.

According to this plan, the activities are to be carried out, discussions made regarding the equipment and resources and the additional equipment has to be acquired to implement the new system. In network backup system no additional resources are needed. Implementation is the final and the most important phase. The most critical stage in achieving a successful new system is giving the users confidence that the new system will work and be effective. The system can be implemented only after thorough testing is done and if it is found to be working according to the specification. This method also offers the greatest security since the old system can take over if the errors are found or inability to handle certain type of transactions while using the new system. As the part of system testing we execute the program with the intent of finding errors and missing operations and also a complete verification to determine whether the objectives are met and the user requirements are satisfied. The ultimate aim is quality assurance.

5.3 SYSTEM MAINTENANCE

According to this plan, the activities are to be carried out, discussions made regarding the equipment and resources and the additional equipment has to be acquired to implement the new system. In network backup system no additional resources are needed. Implementation is the final and the most important phase. The most critical stage in achieving a successful new system is giving the users confidence that the new system will work and be effective. The system can be implemented only after thorough testing is done and if

it is found to be working according to the specification. This method also offers the greatest security since the old system can take over if the errors are found or inability to handle certain type of transactions while using the new system. As the part of system testing we execute the program with the intent of finding errors and missing operations and also a complete verification to determine whether the objectives are met and the user requirements are satisfied. The ultimate aim is quality assurance.

CHAPTER 6.0

CONCLUSION AND FUTURE ENHENCEMENT

6.1 CONCLUSION

The nursery eCommerce website operates smoothly within the constraints of various browsers, meeting both the company's and customers' needs by minimizing errors in input. Transactions are conducted efficiently, enhancing the overall speed of the website. The website creation project focuses on designing an interface that effectively communicates information about the nursery eCommerce portal using HTML and CSS. The user interface is designed to prevent users from making mistakes, ensuring that certain operations are only enabled when appropriate actions are taken, such as saving or cancelling current operations.

6.2 FUTURE ENHANCEMENT

Future enhancements could further improve the interactivity and utility of the application. This could involve refining the user experience to better meet the needs of both the company and customers. Additionally, the system could be optimized to further enhance transaction speeds.While the current project has been successfully developed and implemented according to customer requirements, there is always room for improvement. Future information requirements may necessitate updates or expansions to the existing modules. For instance, additional modules could be developed to include details about instructors or other relevant information.

BIBLIOGRAPHY

BOOKS

1. Matt Doyle, **Beginning PHP 5.3**, Published by Wiley Publishing, Inc. 10475 Cross point *Boulevard, Indianapolis, IN 46256.
2. Larry Ullman, **PHP and MySQL for Dynamic Web Sites** Fourth Edition, Peach pit Press, 1249 Eighth Street, Berkeley, CA 94710.
3. Luke Welling and Laura Thompson **PHP and MySQL Web Development** Second Edition, Same Publishing, 201 West 103rd Street, Indianapolis, Indiana 46290.
4. Hasin Hayder, **Object-Oriented Programming with PHP5** Learn to leverage PHP5's OOP features to write manageable applications with ease, Published by Packt Publishing Ltd. 32 Lincoln Road Olton Birmingham, B27 6PA, UK.
5. Jack Franklin, **query Beginner** write better and more efficient java script with jury, published by Après.

WEBSITES

- http://www.php.net/manual/en/
- www.w3schools.com
- www.tutorialpoint.com
- www.bootstrap.com
- www.mysql.com

APPENDICES

APPENDIX - A

INPUT DESIGNS

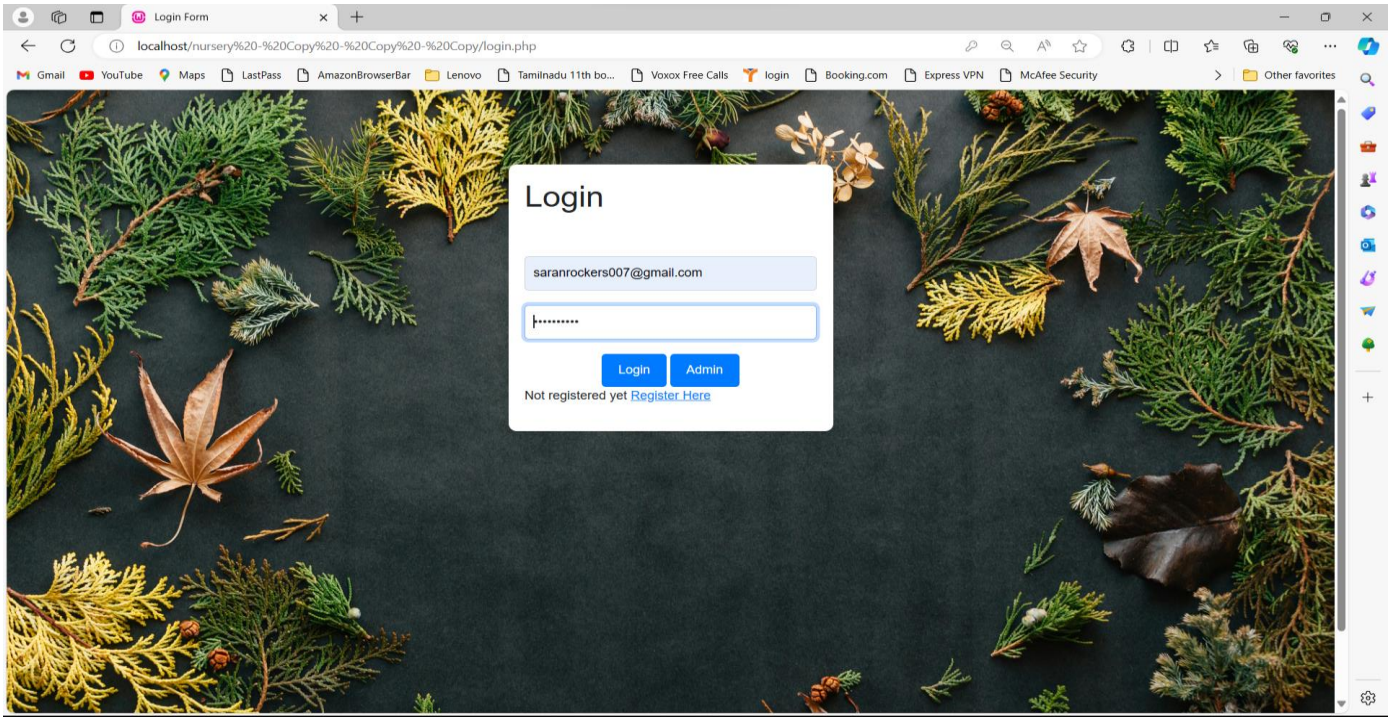


Fig A1.User login page

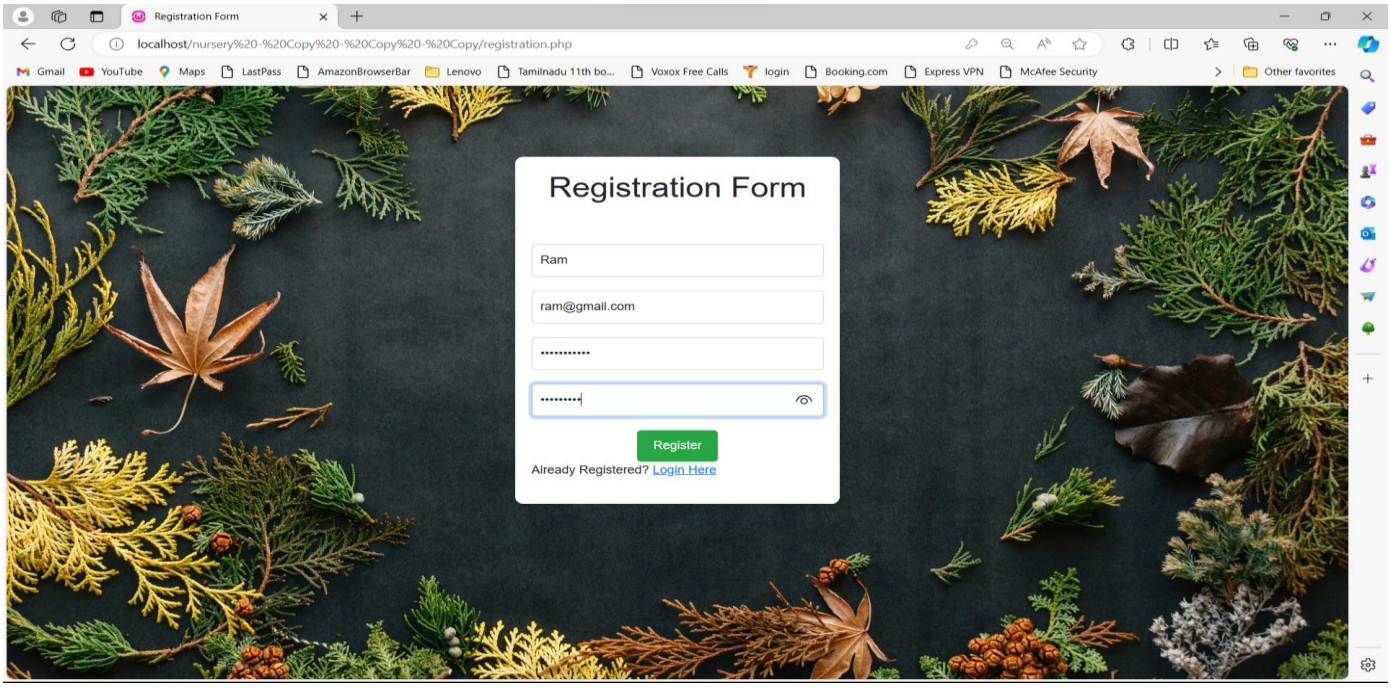


Fig A2.User registration page

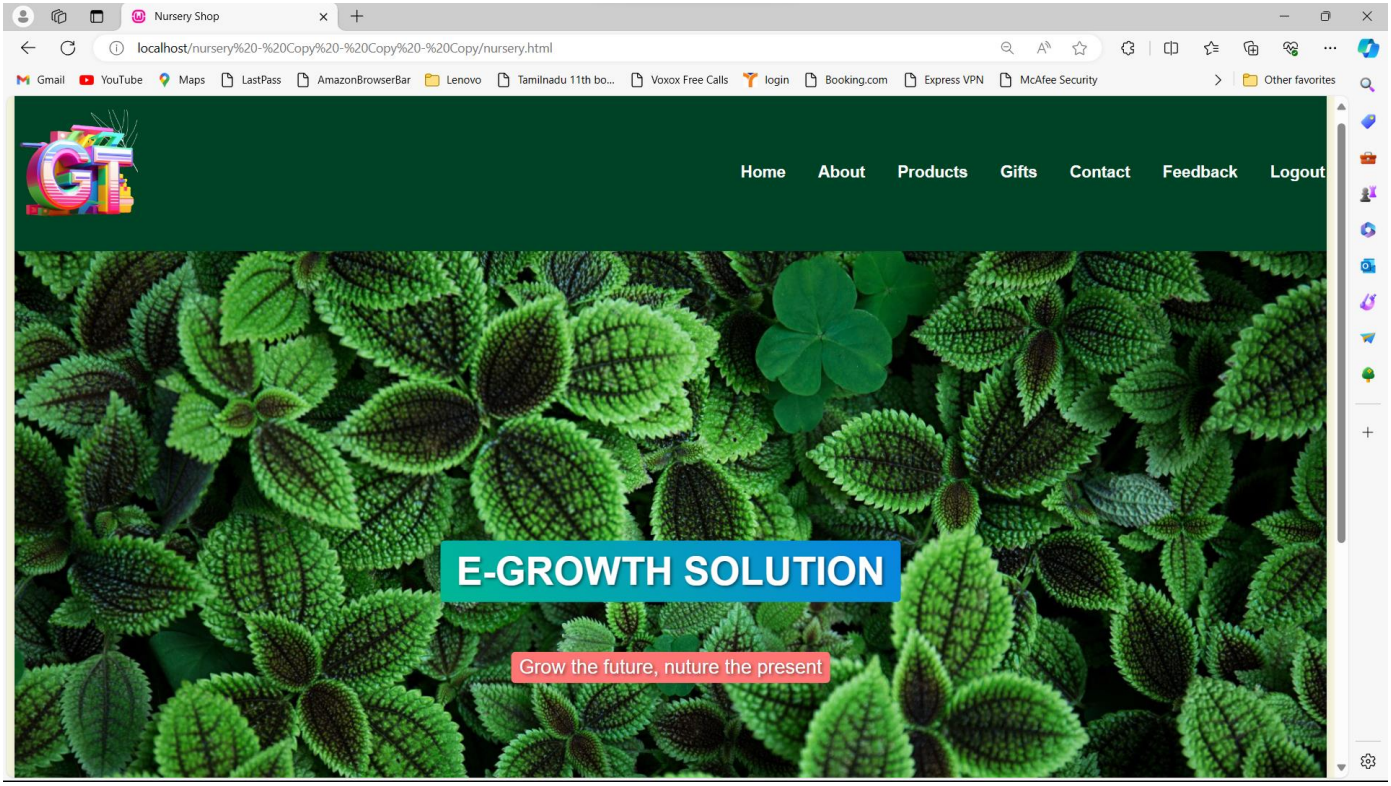


Fig A3.Home page

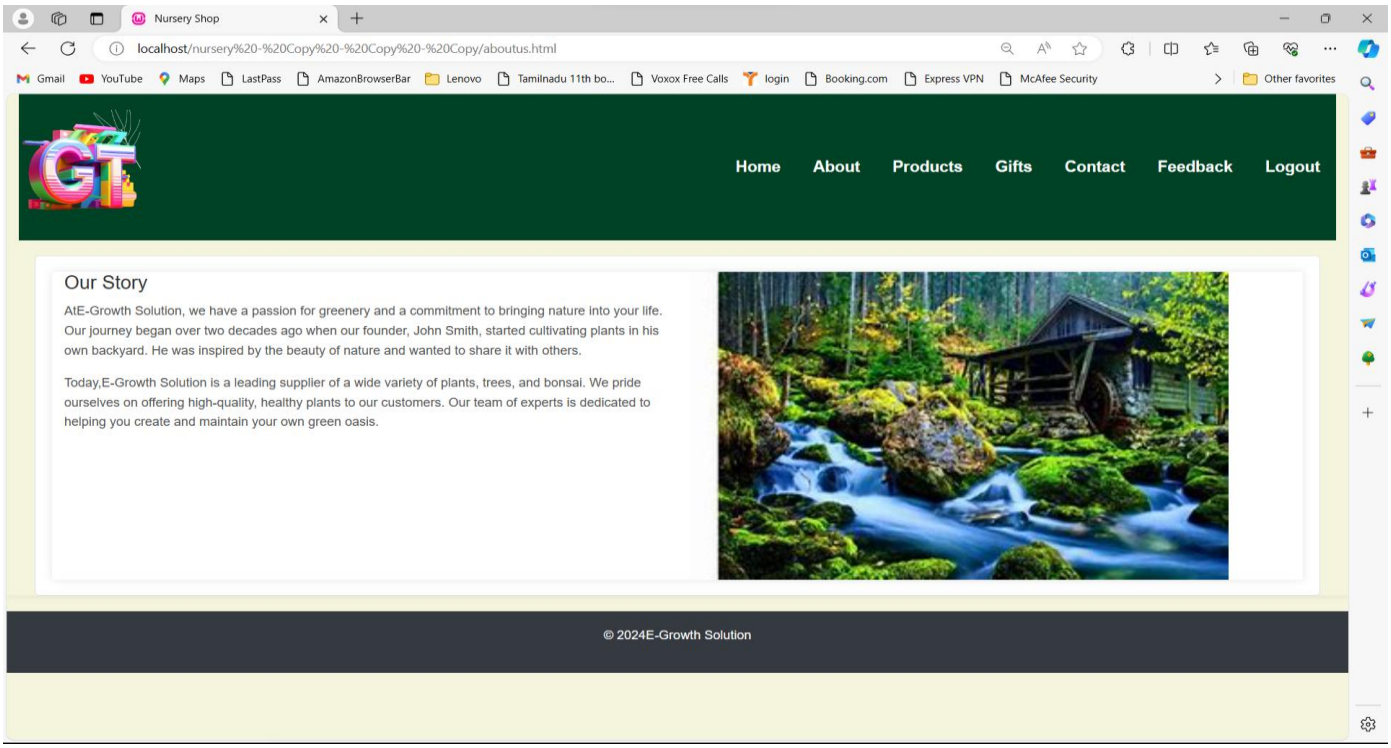


Fig A4.About Us page

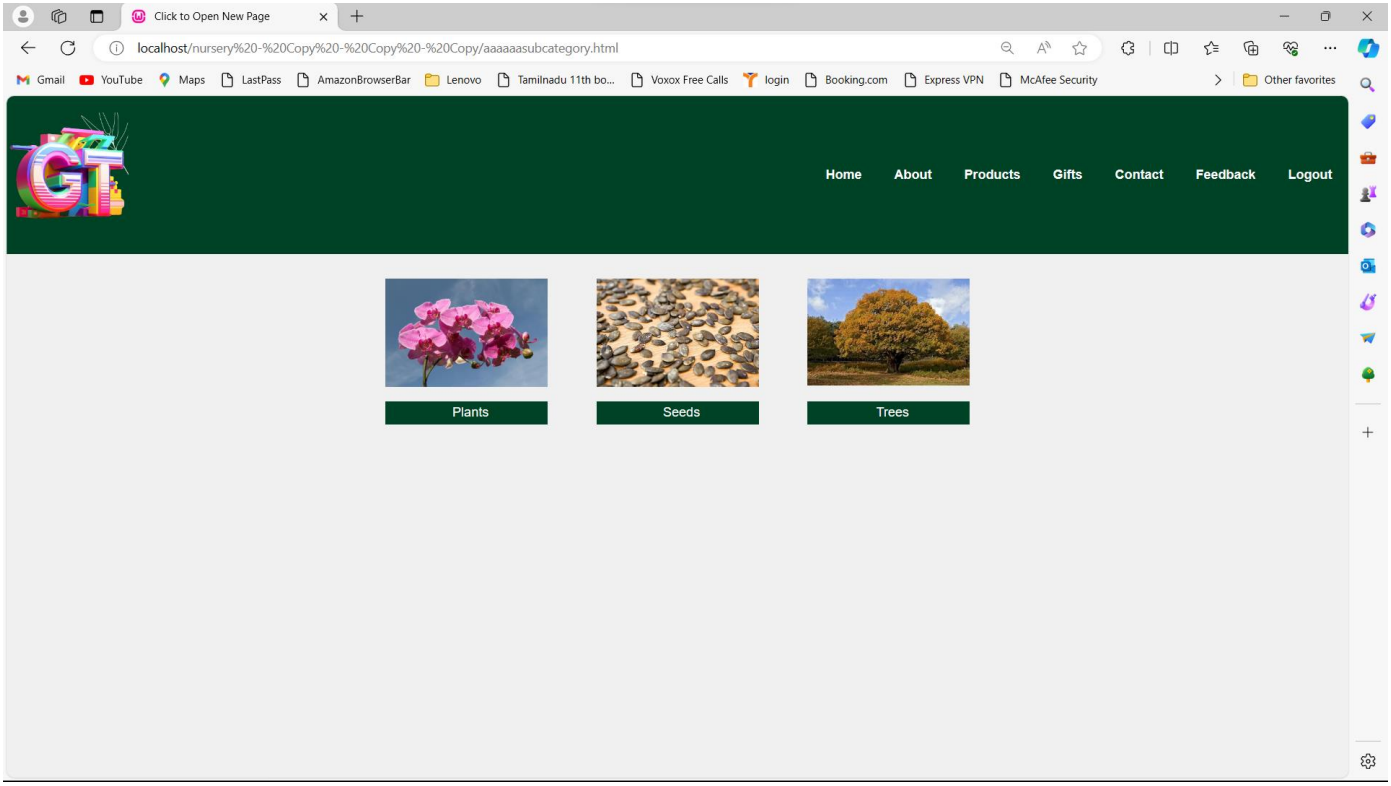


Fig A5.Product page

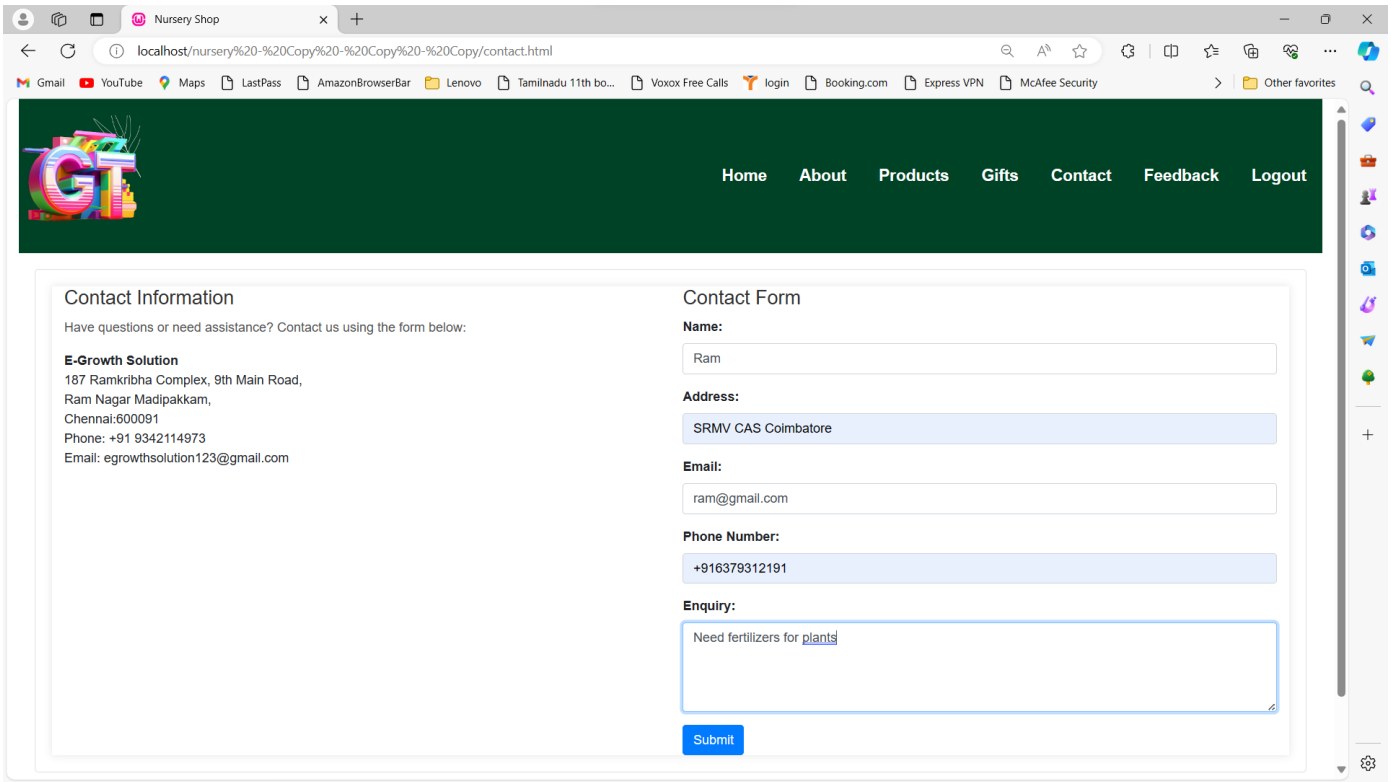


Fig A6.Contact Us page

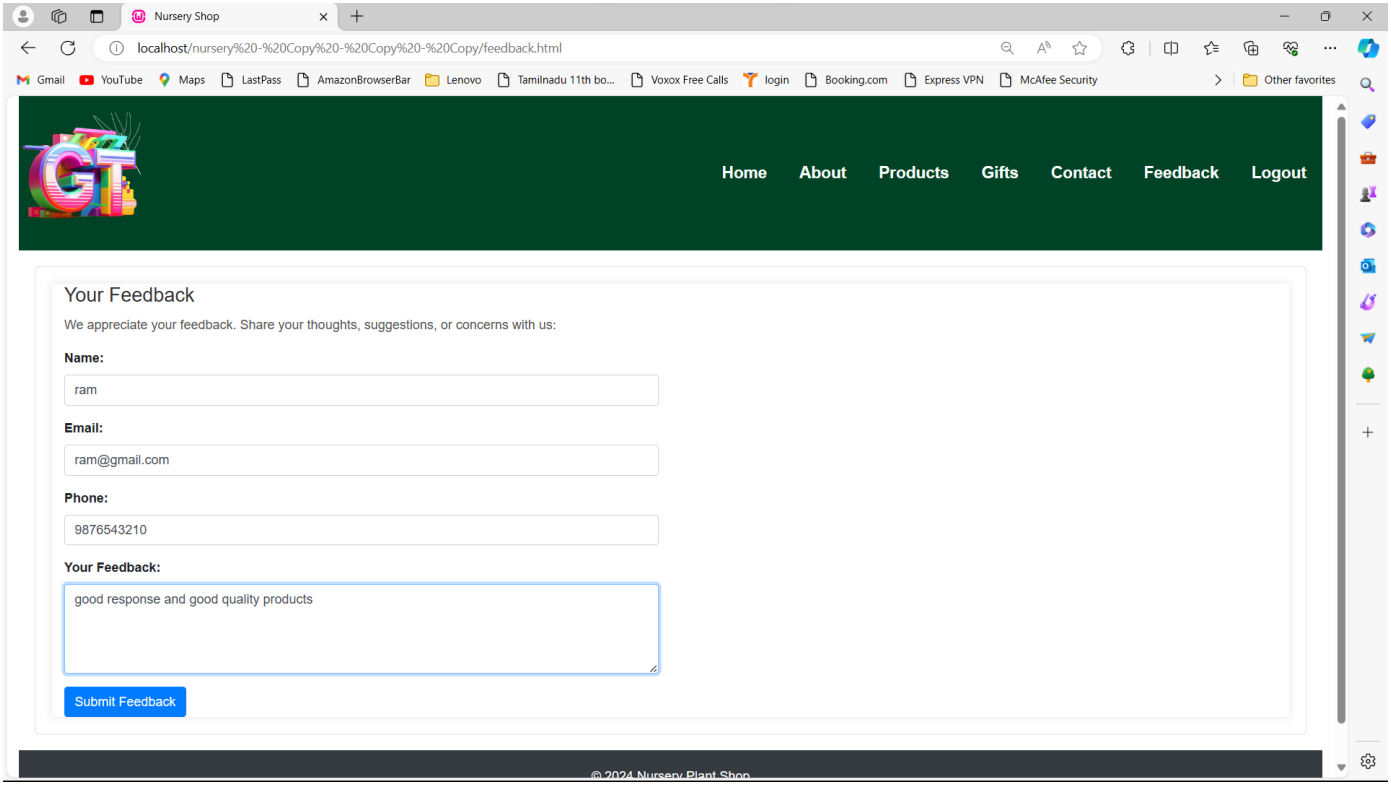


Fig A6.Feedback page

APPENDIX - B

OUTPUT DESIGNS

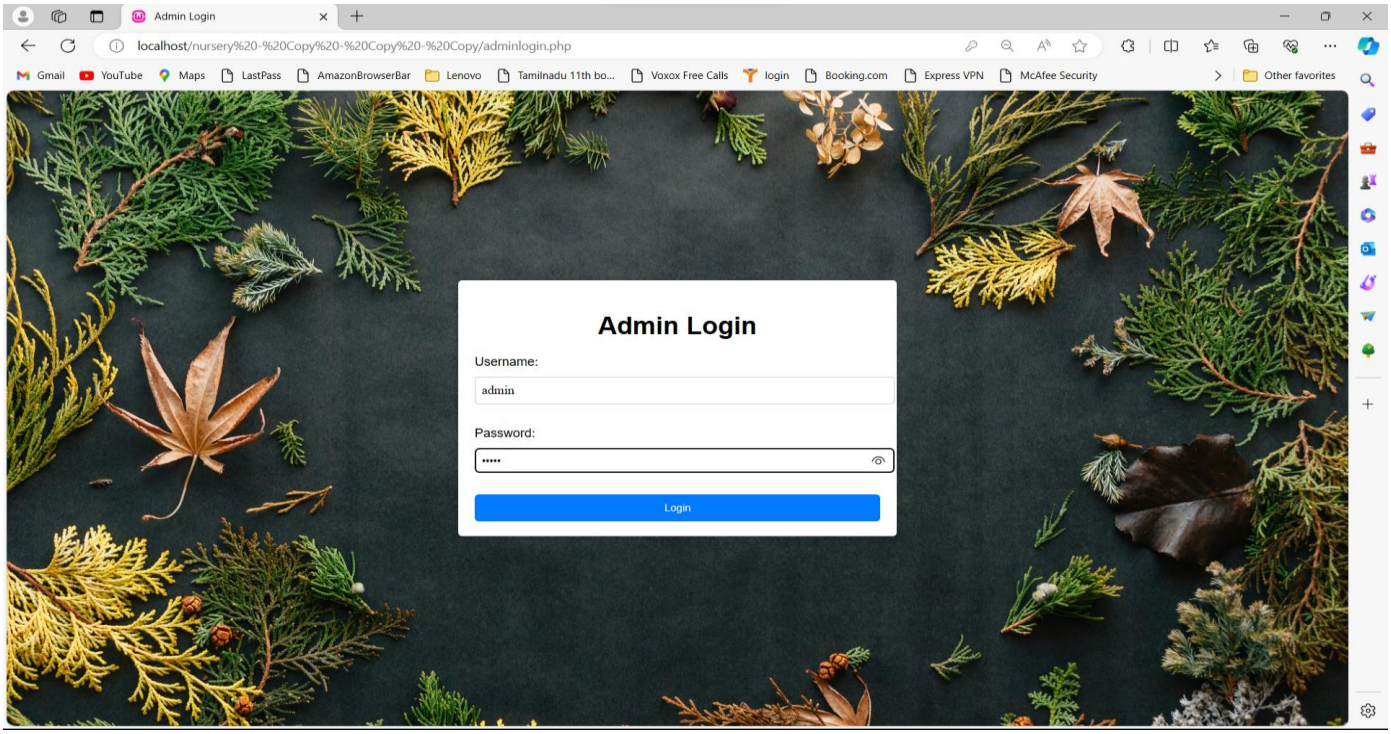
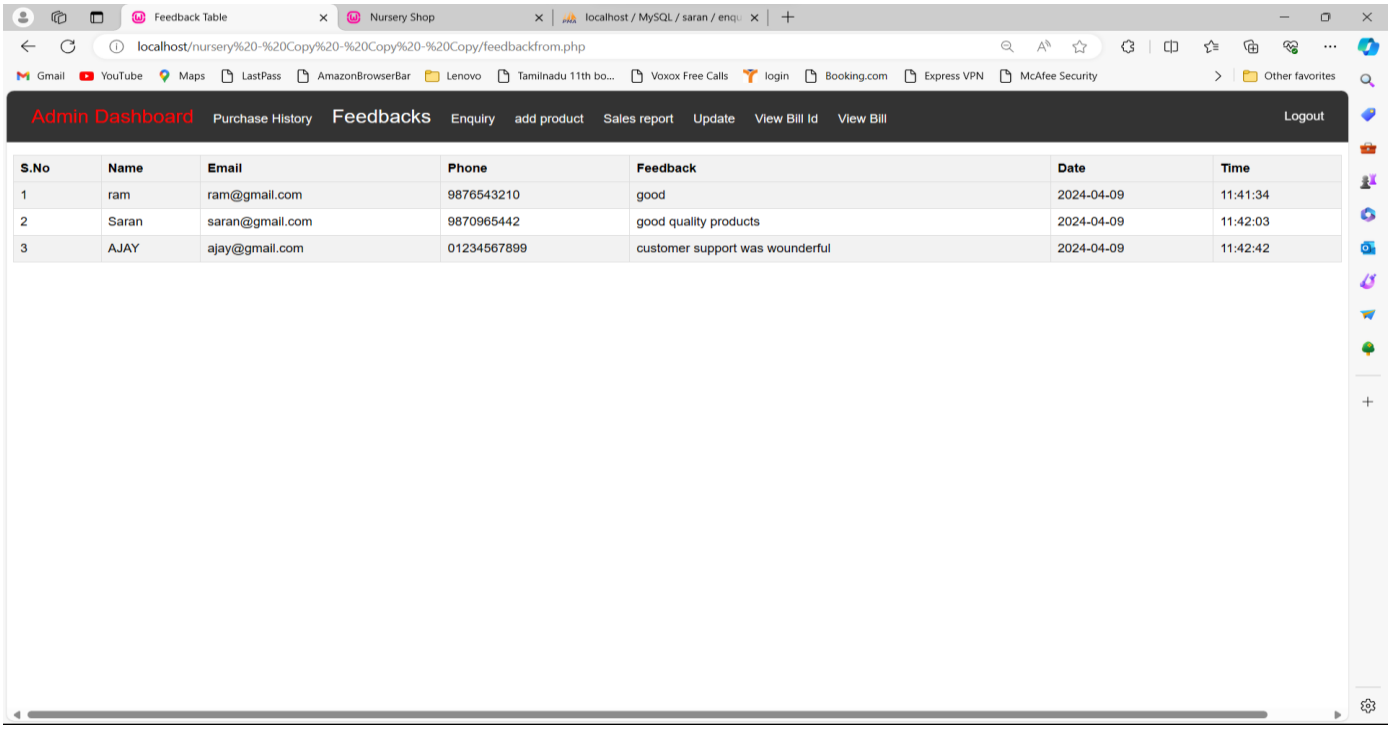


Fig B1.Admin login page



S.No	Name	Email	Phone	Feedback	Date	Time
1	ram	ram@gmail.com	9876543210	good	2024-04-09	11:41:34
2	Saran	saran@gmail.com	9870965442	good quality products	2024-04-09	11:42:03
3	AJAY	ajay@gmail.com	01234567899	customer support was wounderful	2024-04-09	11:42:42

Fig B2.View feedback

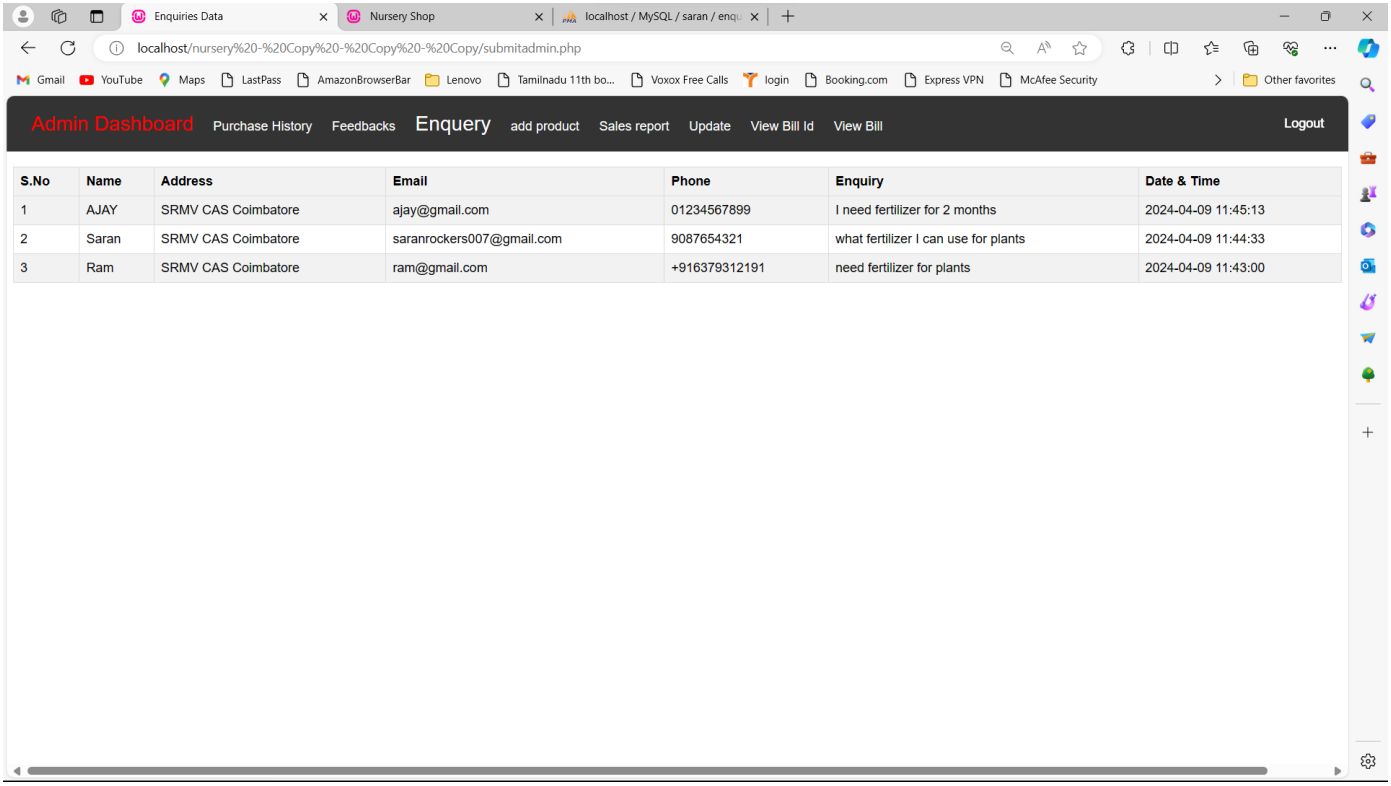


Fig B3.View enquiry

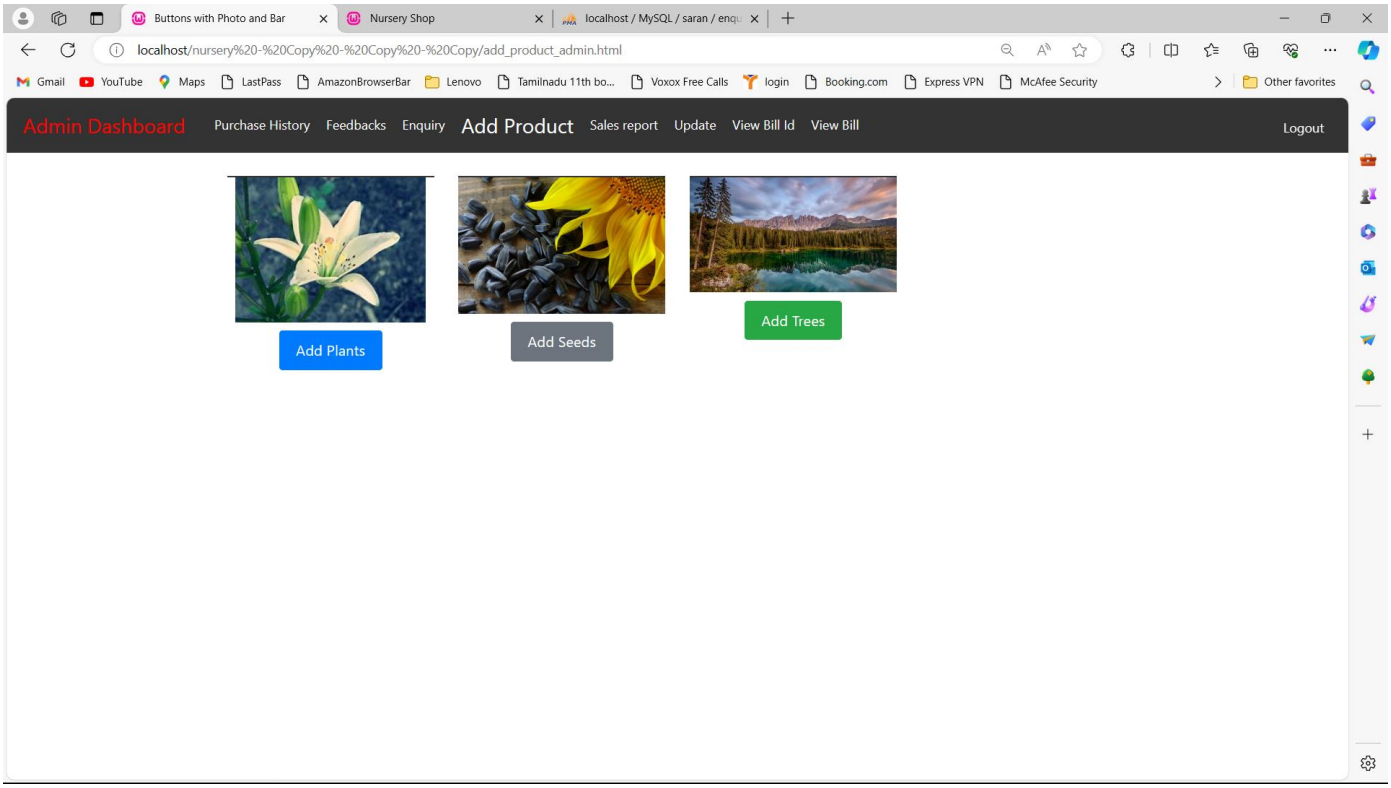


Fig B4.Add products

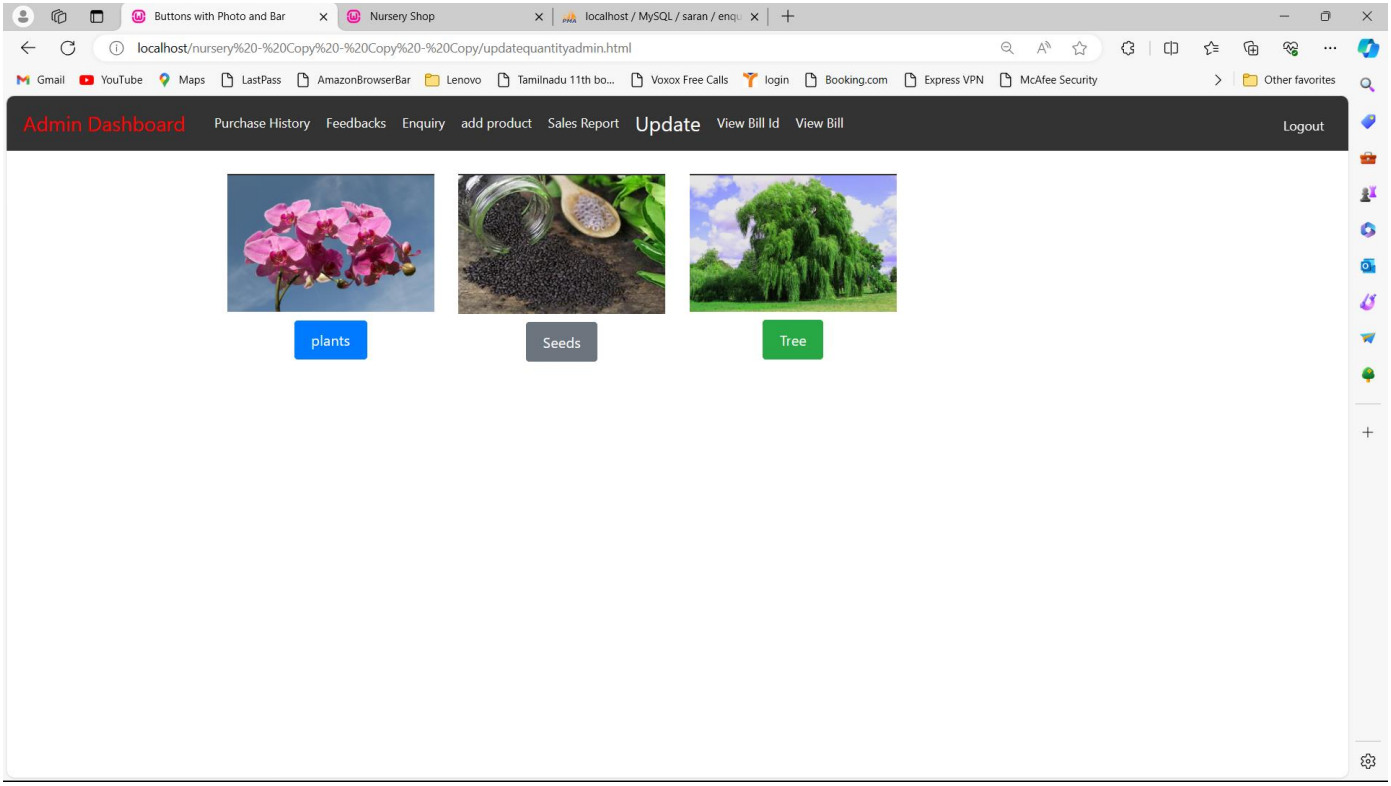


Fig B5.Update Quantity

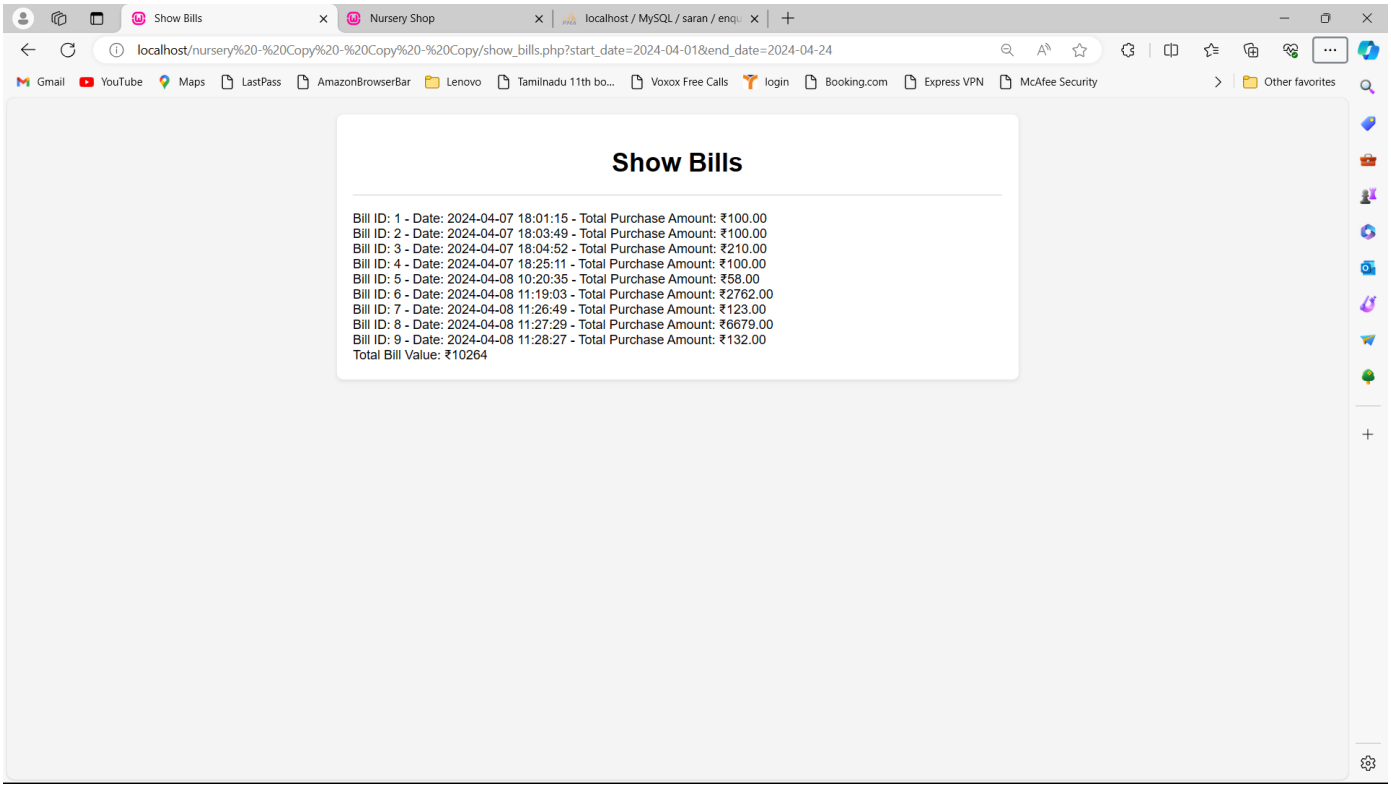


Fig B6.View bill

APPENDIX - C

SAMPLE CODING

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Nursery Shop</title>
    <link rel="stylesheet" href="style.css">
        <style>
            body {
            font-family: Arial, sans-serif;
            background-color: #F5F5DC;
            margin: 10px;
            padding: 0;
        }

        .container {
            width: 100%;
            max-width: none;
            background-color: #FFF;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
        }

        header {
            background-color: #004225;
            color: #FFF;
            padding: 20px 2px;
            display: flex;
            justify-content: space-between;
            align-items: center;
        }
```

```
        .logo img {
            width: 150px;
            height: auto;
        }

        .navigation ul {
            list-style: none;
            padding: 0;
            margin: 0;
            display: flex;
        }

        .navigation ul li {
            margin: 0 20px;
        }

        .navigation ul li a {
            color: #FFF;
            text-decoration: none;
            font-weight: bold;
        }
        .navigation a {
    font-size: 20px;
}

        .navigation ul li a:hover {
            text-decoration: underline;
        }

        section {
            margin: 20px;
            padding: 20px;
```

```
    background-color: #fff;
    border: 1px solid #dee2e6;
    border-radius: 5px;
}

form label {
    font-weight: bold;
}

section h2 {
    font-size: 24px;
    margin-bottom: 10px;
    color: #333;
}

section p {
    font-size: 16px;
    line-height: 1.6;
    color: #555;
}

img.img-fluid {
    max-width: 100%;
    height: 400px;
}

.banner {
    position: relative;
    overflow: hidden;
    image-rendering: optimizeQuality;
    image-orientation: from-image;
    width: 100%;
```

```
    height: auto;
}

#heading {
    position: absolute;
    top: 40%;
    left: 50%;
    transform: translate(-50%, -50%);
    font-size: 3em;
    color: #FFF;
    text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.5);
    background: linear-gradient(45deg, #00b894, #0984e3); /* Gradient background */
    padding: 10px 20px;
    border-radius: 5px;
}

#slogan {
    position: absolute;
    top: 54%;
    left: 50%;
    transform: translate(-50%, -50%);
    font-size: 1.5em;
    color: #FFF;
    text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.5);
    background-color: #ff7675; /* Stylish background color */
    padding: 5px 10px;
    border-radius: 5px;
}

.banner img {
    width: 100%;
    height: 900px;
```

```
}

.categories ul {
    list-style: none;
    padding: 0;
    margin: 20px 0;
    text-align: center;
}

.categories ul li {
    display: inline;
    margin-right: 20px;
    cursor: pointer;
    color: #004225;
}

.categories ul li.active {
    font-weight: bold;
}

.items {
    display: flex;
    flex-wrap: wrap;
    justify-content: space-around;
    padding: 20px;
}

.item {
    width: 30%;
    margin-bottom: 20px;
    background-color: #FFF;
    box-shadow: 0 0 5px rgba(0, 0, 0, 0.1);
```

```
    text-align: center;
    padding: 10px;
}

.item img {
    max-width: 100%;
    height: 300px;
}

.item h3 {
    margin: 10px 0;
    color: #004225;
}

.item .price {
    color: #FFB000;
    font-weight: bold;
}

footer {
    background-color: #004225;
    color: #FFF;
    padding: 20px 0;
    text-align: center;
}

.footer-container {
    max-width: 50%;
    margin: 0 auto;
}

/* Cart and Proceed to Payment Styles */
```

```
        .cart {
            background-color: #004225;
            color: #FFF;
            padding: 20px;
            text-align: center;
        }

        .cart-button {
            background-color: #FFB000;
            color: #FFF;
            border: none;
            padding: 10px 20px;
            cursor: pointer;
            margin-top: 10px;
        }

        .proceed-to-payment {
            background-color: #507062;
            color: #FFF;
            border: none;
            padding: 10px 20px;
            cursor: pointer;
            margin-top: 10px;
        }
    </style>
</head>
<body>
<div class="container">
    <header>
        <div class="logo">
            <img src="nursery_logo.png" alt="Nursery Shop Logo">
```

```
    </div>
    <nav class="navigation">
        <ul class="menu">
            <li><a href="nursery.html">Home</a></li>
            <li><a href="aboutus.html">About</a></li>
            <li><a href="aaaaaasubcategory.html">Products</a></li>
            <li><a href="gift.html">Gifts</a></li>
            <li><a href="contact.html">Contact</a></li>
            <li><a href="feedback.html">Feedback</a></li>
            <li><a href="logout.php">Logout</a></li>
           <!-- <li><a href="aaaaaasubcategory.html">product</a></li>-->

        </ul>
    </nav>
</header>
<div class="banner">

    <img src="banner_image.png" alt="Banner Image">
    <h1 id="heading">E-GROWTH  SOLUTION</h1>
    <p id="slogan">Grow the future, nuture the present</p>

</div>
<main>
  <!--  <div class="categories">
        <ul>
            <li class="active" data-category="all">All</li>
            <li data-category="plants">Plants</li>
            <li data-category="seeds">Seeds</li>
            <li data-category="trees">Trees</li>
        </ul>
    </div>
    <div class="items">
```

```
Plants -->
 <!--  <div class="item" data-category="plants">
      <img src="plant1.jpg" alt="Orchid Delight">
      <h3>Orchid Delight</h3>
      <p class="price">₹20.00</p>
      <input type="number" id="plant1-quantity" placeholder="Quantity">
      <button class="add-to-cart" data-item="plant 1" data-price="20.00">Add to Cart</button>
  </div>
  <div class="item" data-category="plants">
      <img src="plant2.jpg" alt="Fern Fantasy">
      <h3>Fern Fantasy</h3>
      <p class="price">₹18.00</p>
      <input type="number" id="plant2-quantity" placeholder="Quantity">
      <button class="add-to-cart" data-item="plant 2" data-price="18.00">Add to Cart</button>
  </div>
  <div class="item" data-category="plants">
      <img src="plant3.jpg" alt="Lily of the Valley">
      <h3>Lily of the Valley</h3>
      <p class="price">₹25.00</p>
      <input type="number" id="plant3-quantity" placeholder="Quantity">
      <button class="add-to-cart" data-item="plant 3" data-price="25.00">Add to Cart</button>
  </div>
  <div class="item" data-category="plants">
      <img src="plant4.jpg" alt="Golden Pothos">
      <h3>Golden Pothos</h3>
      <p class="price">₹22.00</p>
      <input type="number" id="plant4-quantity" placeholder="Quantity">
      <button class="add-to-cart" data-item="plant 4" data-price="22.00">Add to Cart</button>
  </div>
  <div class="item" data-category="plants">
      <img src="plant5.jpg" alt="Scarlet Begonia">
      <h3>Scarlet Begonia</h3>
```

```
    <p class="price">₹30.00</p>
    <input type="number" id="plant5-quantity" placeholder="Quantity">
    <button class="add-to-cart" data-item="plant 5" data-price="30.00">Add to Cart</button>
  </div>-->

  <!-- Seeds -->
<!--  <div class="item" data-category="seeds">
    <img src="seed1.jpg" alt="Sunflower Radiance">
    <h3>Sunflower Radiance</h3>
    <p class="price">₹5.00</p>
    <input type="number" id="seed1-quantity" placeholder="Quantity">
    <button class="add-to-cart" data-item="seed 1" data-price="5.00">Add to Cart</button>
  </div>
  <div class="item" data-category="seeds">
    <img src="seed2.jpg" alt="Basil Bliss">
    <h3>Basil Bliss</h3>
    <p class="price">₹4.00</p>
    <input type="number" id="seed2-quantity" placeholder="Quantity">
    <button class="add-to-cart" data-item="seed 2" data-price="4.00">Add to Cart</button>
  </div>
  <div class="item" data-category="seeds">
    <img src="seed3.jpg" alt="Wildflower Wonderland">
    <h3>Wildflower Wonderland</h3>
    <p class="price">₹6.00</p>
    <input type="number" id="seed3-quantity" placeholder="Quantity">
    <button class="add-to-cart" data-item="seed 3" data-price="6.00">Add to Cart</button>
  </div>
  <div class="item" data-category="seeds">
    <img src="seed4.jpg" alt="Tomato Tango">
    <h3>Tomato Tango</h3>
    <p class="price">₹3.00</p>
    <input type="number" id="seed4-quantity" placeholder="Quantity">
```

```
    <button class="add-to-cart" data-item="seed 4" data-price="3.00">Add to Cart</button>
</div>
<div class="item" data-category="seeds">
    <img src="seed5.jpg" alt="Pumpkin Spice">
    <h3>Pumpkin Spice</h3>
    <p class="price">₹7.00</p>
    <input type="number" id="seed5-quantity" placeholder="Quantity">
    <button class="add-to-cart" data-item="seed 5" data-price="7.00">Add to Cart</button>
</div>-->

<!-- Trees -->
<!-- <div class="item" data-category="trees">
    <img src="tree1.jpg" alt="Majestic Oak">
    <h3>Majestic Oak</h3>
    <p class="price">₹50.00</p>
    <input type="number" id="tree1-quantity" placeholder="Quantity">
    <button class="add-to-cart" data-item="tree 1" data-price="50.00">Add to Cart</button>
</div>
<div class="item" data-category="trees">
    <img src="tree2.jpg" alt="Royal Maple">
    <h3>Royal Maple</h3>
    <p class="price">₹45.00</p>
    <input type="number" id="tree2-quantity" placeholder="Quantity">
    <button class="add-to-cart" data-item="tree 2" data-price="45.00">Add to Cart</button>
</div>
<div class="item" data-category="trees">
    <img src="tree3.jpg" alt="Evergreen Pine">
    <h3>Evergreen Pine</h3>
    <p class="price">₹60.00</p>
    <input type="number" id="tree3-quantity" placeholder="Quantity">
    <button class="add-to-cart" data-item="tree 3" data-price="60.00">Add to Cart</button>
</div>
```

```
        <div class="item" data-category="trees">
            <img src="tree4.jpg" alt="Willow Whisper">
            <h3>Willow Whisper</h3>
            <p class="price">₹55.00</p>
            <input type="number" id="tree4-quantity" placeholder="Quantity">
            <button class="add-to-cart" data-item="tree 4" data-price="55.00">Add to Cart</button>
        </div>
        <div class="item" data-category="trees">
            <img src="tree5.jpg" alt="Redwood Majesty">
            <h3>Redwood Majesty</h3>
            <p class="price">₹70.00</p>
            <input type="number" id="tree5-quantity" placeholder="Quantity">
            <button class="add-to-cart" data-item="tree 5" data-price="70.00">Add to Cart</button>
        </div>
    </div>-->

<!-- Cart Section -->
<!--<div class="cart">
    <h2>Your Cart</h2>
    <div id="cart-items" class="cart-items"></div>
    <div class="cart-total">
        <p>Total Amount:</p>
        <span id="cart-total-amount">₹0.00</span>
    </div>
    <button class="proceed-to-payment">Proceed to Payment</button>
</div>-->
    </main>
    <footer>
        <div class="">
            <div>
                <p><strong>Contact Info:</strong>187 Ramkribha Complex, 9th Main Road, Ram Nagar Madipakkam, Chennai:600091
```

```
            </p>
            <b> <p>Phone: </b>+91 9342114973</p>
            <b> <p>Email:</b>egrowthsolution123@gmail.com
          </div>
        </div>
    </footer>
</div>
<script>
// Cart data structure
const cart = [];

// Function to add an item to the cart
function addToCart(itemName, itemPrice) {
    const quantityInput = document.querySelector(`#${itemName.replace(/ /g, "").toLowerCase()}-quantity`);
    const quantity = quantityInput.value;

    if (quantity > 0) {
        const totalItemPrice = (itemPrice * quantity).toFixed(2);
        const newItem = {
            name: itemName,
            quantity: quantity,
            price: itemPrice,
            total: `₹${totalItemPrice}`,
        };

        cart.push(newItem);

        // Update the cart display
        displayCart();
    }
}
```

```
// Function to remove an item from the cart
function removeFromCart(itemName) {
    const itemIndex = cart.findIndex(item => item.name === itemName);

    if (itemIndex !== -1) {
        cart.splice(itemIndex, 1);
    }

    // Update the cart display
    displayCart();
}

// Function to display the cart
function displayCart() {
    const cartItemsElement = document.getElementById('cart-items');
    cartItemsElement.innerHTML = '';

    let totalAmount = 0;

    cart.forEach(item => {
        const cartItem = document.createElement('div');
        cartItem.classList.add('cart-item');

        const cartItemName = document.createElement('div');
        cartItemName.classList.add('cart-item-name');
        cartItemName.textContent = `${item.name} (${item.quantity} x ₹${item.price})`;

        const cartItemPrice = document.createElement('div');
        cartItemPrice.classList.add('cart-item-price');
        cartItemPrice.textContent = item.total;
```

```
        const cartRemoveButton = document.createElement('button');
        cartRemoveButton.classList.add('cart-remove-button');
        cartRemoveButton.textContent = 'Remove';
        cartRemoveButton.addEventListener('click', () => {
            removeFromCart(item.name);
        });

        cartItem.appendChild(cartItemName);
        cartItem.appendChild(cartItemPrice);
        cartItem.appendChild(cartRemoveButton);

        cartItemsElement.appendChild(cartItem);

        // Calculate the total amount
        totalAmount += parseFloat(item.price) * item.quantity;
    });

    // Update the total amount
    const cartTotalAmount = document.getElementById('cart-total-amount');
    cartTotalAmount.textContent = `₹${totalAmount.toFixed(2)}`;
}

// Add a click event listener to the "Add to Cart" buttons
const addToCartButtons = document.querySelectorAll('.add-to-cart');
addToCartButtons.forEach(button => {
    button.addEventListener('click', () => {
        const itemName = button.getAttribute('data-item');
        const itemPrice = parseFloat(button.getAttribute('data-price'));
        addToCart(itemName, itemPrice);
    });
});
```

```
// Add a click event listener to the "Proceed to Payment" button
const proceedToPaymentButton = document.querySelector('.proceed-to-payment');
proceedToPaymentButton.addEventListener('click', () => {
    // You can implement the checkout functionality here
    // For now, let's just clear the cart
    cart.length = 0;

    // Update the cart display
    displayCart();

    // Redirect the user to payment.html
    window.location.href = 'payment.html';
});

// Get the category list items and the items container
const categoryItems = document.querySelectorAll('.categories li');
const itemsContainer = document.querySelector('.items');

// Add a click event listener to each category item
categoryItems.forEach(item => {
  item.addEventListener('click', () => {
    // Remove the "active" class from all category items
    categoryItems.forEach(item => {
      item.classList.remove('active');
    });
// Add the "active" class to the clicked category item
    item.classList.add('active');

    // Get the selected category
    const selectedCategory = item.getAttribute('data-category');

    // If the selected category is "all", show all items
```

```
    if (selectedCategory === 'all') {
      itemsContainer.querySelectorAll('.item').forEach(item => {
        item.style.display = 'block';
      });
    } else {
      // Otherwise, show only the items that belong to the selected category
      itemsContainer.querySelectorAll('.item').forEach(item => {
        if (item.getAttribute('data-category') === selectedCategory) {
          item.style.display = 'block';
        } else {
          item.style.display = 'none';
        }
      });
    }
  });
});
</script>
</body>
</html>
```