



Kristu Jayanti College

AUTONOMOUS **Bengaluru**

Reaccredited A++ Grade by NAAC | Affiliated to Bengaluru North University

BLOOMING HEARTS

Project Report submitted in partial fulfilment of the requirements for the
award of the degree of

BACHELOR OF COMPUTER APPLICATIONS (BCA)

Department of Computer Science (UG)



Submitted by

ROSE MARIA JAMES
23BCAE53

Under the guidance of

Prof. Sumalatha A

DEPARTMENT OF COMPUTER SCIENCE (UG)

BCA PROGRAMME

KRISTU JAYANTI COLLEGE (Autonomous)

K. Narayanapura, Kothanur P.O., Bangalore – 560077



Kristu Jayanti College

AUTONOMOUS Bengaluru

Reaccredited A++ Grade by NAAC | Affiliated to Bengaluru North University

DEPARTMENT OF COMPUTER SCIENCE (UG)

CERTIFICATE OF COMPLETION

This is to certify that the project entitled “**BLOOMING HEARTS – Bouquet and Gift Card Management System**” has been satisfactorily completed by in partial fulfilment of the award of the Bachelor of Computer Applications degree requirements prescribed by Kristu Jayanti College (Autonomous) Bengaluru (Affiliated to Bangalore North University) during the academic year 2024-25.

Internal Guide

Head of the Department

Valued by Examiners

1: _____

Centre: Kristu Jayanti College

2: _____

Date:

DECLARATION

I, Rose Maria James , 23BCAE53 hereby declare that the project work entitled “Blooming Hearts” is an original project work carried out by me, under the guidance of **Prof. Sumalatha A .**

This project work has not been submitted earlier either to any University / Institution or any other body for the fulfillment of the requirement of a course of study.

Signature

Rose Maria James

Bengaluru

Date:

ACKNOWLEDGEMENT

The success of the project depends upon the efforts invested. It's my duty to acknowledge and thank the individuals who has contributed to the successful completion of the project.

I take this opportunity to express my profound and wholehearted thanks to **Rev. Fr. Dr. AUGUSTINE GEORGE, PRINCIPAL, AND Rev. Fr. Dr. LIJO P. THOMAS, VICE PRINCIPAL & CHIEF FINANCE OFFICER IN KRISTU JAYANTI COLLEGE, BANGALORE** for providing ample facilities made to undergo my project successfully.

I express my deep sense of gratitude and sincere thanks to our **PROF. CALISTUS JUDE AL , Dean ,Faculty of Sciences** for his valuable support

I express my deep sense of gratitude and sincere thanks to our Head of the Department **Prof. SEVUGA PANDIAN** for his valuable advice.

I feel immense pleasure to thank my respected guide **Prof. Sumalatha A** for sustaining Interest and providing dynamic guidance in aiding me to complete this project immaculately and impeccably and for being the source of my strength and confidence.

It is my duty to express my thanks to all Teaching and Non-Teaching Staff members of Computer science department who offered me help directly or indirectly by their suggestions. The successful completion of my project would not have been possible without my parent's Sacrifice, guidance, and prayers. I take this opportunity to thank everyone for their continuous Encouragement. I convey my thankfulness to all my friends who were with me to share my happiness and agony.

Last but not the least I thank almighty God for giving me strength and good health throughout my project and enabling me to complete it successfully.

SYNOPSIS

Blooming Hearts is a web-based application designed to enhance the experience of gifting by providing a digital platform for ordering floral bouquets and personalized gift cards. In today's fast-paced world, where physical presence is not always possible, Blooming Hearts bridges the emotional gap by offering users a meaningful way to express love, gratitude, and celebration through elegant and heartfelt gifts. The platform features a visually captivating interface inspired by nature and romance, using soft color palettes, floral backgrounds, and a glassmorphism-inspired layout to deliver a luxurious and calming user experience. It is accessible to both regular users and administrators, with role-based access ensuring a streamlined and secure interaction for each type of user. From browsing beautifully arranged bouquets to customizing greeting cards, Blooming Hearts simplifies the gifting process while preserving the emotional value behind each gesture. Users can create accounts, manage their profiles, explore the curated catalog, and purchase items with ease. The system also includes wishlist and cart functionality to support thoughtful planning and easy management of selected gifts. The front-end of the application is built using HTML, CSS and JavaScript, designed with a floral-themed background to create a warm and inviting visual appeal. The backend is developed in PHP, all the data stored and managed in MySQL database. Development was carried out using VS Code and locally hosted in Xampp. Blooming Hearts ultimately aims to blend emotion, design, and technology to create a beautiful digital gifting experience—helping people connect, celebrate, and share love, no matter the distance.

TABLE OF CONTENTS

S L. No	Topic	Page No
1.	Introduction	1-4
1.1	Problem Definition	2
1.2	Scope of the Project	3-4
2	System Study	5-8
2.1	Existing System	6
2.2	Feasibility Study	6-8
2.3	Proposed System	8
3	System Design	9-71
3.1	ER Diagram	13-16
3.2	DFD [lev0, lev1]	17-20
3.3	Gantt Chart	21-24
3.4	Input / Output Design	26-71
4	System Configuration	72
4.1	Hardware Requirements	72
4.2	Software Requirements	72
5.	Details of Software	73-74
5.1	Overview of Frontend	73
5.2	Overview of Backend	73
5.3	About the Platform	74
6	Testing	75-76
7	Conclusion and Future Enhancement	77
8	Bibliography	78
9	APPENDICES A-Table Structure	79-81
10	APPENDICES B-Screenshots	82-87
11	APPENDICES C-Sample Report of test	88
12	APPENDICES D-Source code	89-106

INTRODUCTION

1. INTRODUCTION

1.1 PROBLEM DEFINITION

Blooming Hearts is a web-based application aimed at providing customers with a convenient platform to browse and purchase bouquets and gift cards. The objective of this project is to develop a system that automates the process of managing products and orders for a florist business. This system will simplify the tasks of browsing flowers, adding items to the cart, and completing purchases for users, while also offering administrative control over products and customer management.

1.2 SCOPE OF THE PROJECT

The Blooming Hearts project is designed to provide a seamless and efficient way for users to browse, select, and purchase bouquets and gift cards online. This system is created to streamline the shopping process, making it convenient for both customers and administrators. The system offers a comprehensive range of features, from user authentication to order management, ensuring a smooth experience for all users involved.

MODULES I THE SOFTWARE:

1. User Authentication:

- **Login:** Users can log in using their email and password.
- **Registration:** New users can create an account by providing necessary details such as name, email, password, and delivery address.

2. Catalog:

- **Bouquet and Gift Card Listings:** Users can browse a catalog of bouquets and gift cards, with detailed descriptions, images, and prices.
- **Product Details:** Each product (bouquet/gift card) will include options for customization (size, color, message) and a detailed description.
- **Add to Cart:** Users can add products to their shopping cart with ease.

3. Search Module:

- **Search Functionality:** Users can search for specific bouquets, gift cards, or categories (e.g.,

birthday, wedding, anniversary).

- **Filters:** Users can filter results based on product type, price range, or occasion to help them find the perfect product.

4. Cart:

- **Add to Cart:** Users can add items (bouquets, gift cards) to their cart with customizable one like quantity and color.
- **View Cart:** Users can review items in their cart, see the total price, and proceed to checkout.
- **Remove/Update Cart:** Users can remove items from the cart or change quantities before finalizing the purchase.

5. Wishlist:

- **Save Items to Wishlist:** Users can add items to their wishlist for future reference or purchase.
- **View Wishlist:** Users can revisit and manage their wishlist items.

6. Payment:

- **Payment Methods:** Users can choose from different payment options such as PayPal, credit/debit card, or direct payment methods.
- **Secure Payment Processing:** Payments are securely processed with encryption for user safety.
- **Order Confirmation:** Users receive an order confirmation once payment is successful, with details of their purchase.

7. Contact Us:

- **Contact Form:** Users can fill out a contact form to ask questions or raise concerns about their order or the website.
- **Customer Support:** Admin can review contact form submissions and respond to customer .

8. Admin Module:

- **User Management:** Admin can view, add, edit, and delete user accounts.

Admin can manage user profiles and reset passwords if necessary.

- **Order Management:** Admin can view, confirm, cancel, or process orders.

Admin can track order statuses (pending, completed, shipped).

Admin can manage customer order history and communication.

- **Feedback:** Admin can view customer feedback submitted via testimonials and reviews.

Admin can approve or delete inappropriate feedback.

Admin can respond to feedback when necessary.

- **Logout:** Admin can log out securely from the Admin Dashboard.

SYSTEM STUDY

2. SYSTEM STUDY

2.1 EXISTING SYSTEM

In the current system, customers have to visit various platforms or agencies to find and purchase bouquets or gift cards. They must manually check the available options, compare prices, and deal with multiple interfaces to complete the process, which often requires a lot of time and effort. This traditional method of operation is tedious for the customers, who have to individually browse catalogs, manage their carts, and deal with complex payment processes. This fragmented approach causes inefficiency and can leave customers frustrated .

Disadvantage of existing system

- Difficulty in operating
- Lack of progress checks of Travel
- Unnecessary navigations

2.2 FEASIBILITY STUDY

A feasibility study is an analysis of how successfully a project can be completed, accounting for factors that affect it such as economic, technological, legal and scheduling factors. Project managers use feasibility studies to determine potential positive and negative outcomes of a project before investing a considerable amount of time and money into it. A feasibility study tests the viability of an idea, a project or even a new business. The goal of a feasibility study is to place emphasis on potential problems that could occur if a project is pursued and determines if, after all significant factors are considered, the project should be pursued. Feasibility studies also allow a business to address where and how it will operate, potential obstacles, competition and the funding needed to get the business up and running.

This project "**Blooming Hearts**" has undergone the following feasibility study:

- Economic Feasibility
- Technical Feasibility
- Behavioural Feasibility
- Schedule Feasibility

Every project is feasible for given unlimited resources and infinite time. A feasibility study is an evaluation of the proposed system regarding its workability, impact on the organization, ability to meet the user needs and effective use of resources. Thus, when a new application is proposed it normally goes through a feasibility study before it is approved for development. Feasibility and risk analysis are related in many ways. The feasibility analysis in this project has been discussed below based on the above-mentioned components of feasibility.

1. Technical Feasibility:

Technical feasibility focuses on the technology used. It means the computerized system is technically feasible i.e., it doesn't have any technical fault and works properly in the given environment. The technologies used in Blooming Hearts include HTML, CSS, JavaScript for the frontend, and PHP with MySQL for the backend. These are reliable, proven technologies that ensure the system is efficient, secure, and compatible with modern web environments. The system is technically feasible if it provides the required output under standard testing and production conditions.

2. Economic Feasibility:

Economic analysis is the most frequently used method for evaluating the effectiveness of a computerized system. We analyze that the Blooming Hearts system is economically more feasible than the traditional manual system, as it significantly reduces the need for manpower, saves operational time, and lowers administrative costs. The system is also highly beneficial in terms of cost-benefit analysis. The investment required to build and maintain this project is reasonable, and it is expected to offer long-term savings by automating many tasks and improving customer satisfaction.

3. Behavioural Feasibility:

Behavioural feasibility is the analysis of how the computerized system is received and utilized by users. In this, we analyze whether the system is operating effectively and communicating properly with its users. Blooming Hearts has a user-friendly interface with soft aesthetics and clearly organized modules such as catalog, cart, wishlist, and payment, which are intuitive and accessible for users. The

Feedback from users during development and testing was positive, indicating acceptance and ease of use. Therefore, the system has proven to be behaviorally feasible.

4. Schedule Feasibility:

Time evaluation is one of the most important considerations in the development of a project. The schedule required for the development of Blooming Hearts was carefully planned and monitored to ensure timely delivery. The entire project development was broken down into well-defined modules, each with its own deadlines and dependencies. Any delay in the timeline could affect cost, resource utilization, and system integration. However, the project progressed smoothly within the estimated schedule, demonstrating strong schedule feasibility.

2.3 PROPOSED SYSTEM

The proposed system, BLOOMING HEARTS, is an advanced and user-friendly bouquet and gift card management platform designed to streamline the entire process of browsing, selecting, customizing, and ordering floral products and gift items. The system aims to resolve the issues present in the existing manual or semi-digital systems by offering an efficient, elegant, and fully digitized solution. The Blooming Hearts platform will provide seamless access for both customers and administrators. Customers can explore a wide catalog of products, add them to their wishlist or cart, and proceed with secure online payments. The system also includes modules for contact support, order tracking, and real-time updates, thereby ensuring a convenient and delightful shopping experience. The administrator module allows authorized personnel to manage users, view orders, respond to feedback, and update catalog content. This eliminates the need for manual recordkeeping and ensures that all operations are streamlined, secure, and transparent

SYSTEM DESIGN

3. SYSTEM DESIGN

In the design phase the architecture is established. This phase starts with the requirement document delivered by the requirement phase and maps the requirements into architecture. The architecture defines the components, their interfaces and behaviours. The deliverable design document is the architecture.

The design document describes a plan to implement the requirements. This phase represents that "how" phase. Details on computer programming languages and environments, machines, packages, application architecture, distributed architecture layering, memory size, platform, algorithms, data structures, global type definitions, interfaces, and many other engineering details are established. The design may include the usage of existing components. Analysing the trade-offs of necessary complexity allows for many things to remain simple which, in turn, will eventually lead to a higher quality product. The architecture team also converts the typical scenarios into a test plan.

In our approach, the team, given a complete requirement document, must also indicate critical priorities for the implementation team. A critical implementation priority leads to a task that has to be done right. If it fails, the product fails. If it succeeds, the product might succeed. At the very least, the confidence level of the team producing a successful product will increase. This will keep the implementation team focused. Exactly how this information is conveyed is a skill based on experience more than a science based on fundamental foundations.

System design is the process of defining the architecture components, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering.

If the broader topic of product development "blends the perspective of marketing, design, and manufacturing into a single approach to product development," then design is the act of taking the marketing information and creating the design of the product to be manufactured. Systems design is therefore the Process of defining and developing systems to satisfy specified requirements of users.

Until the 1990s, systems design had a crucial and respected role in the data processing industry. In the 1990s, standardization of hardware and software resulted in the ability to build modular systems. The increasing importance of software running on generic platforms has enhanced the discipline of software engineering.

Object-oriented analysis and design methods are becoming the most widely used methods for computer systems design. The UML has become the standard language in object-oriented analysis and design. It is widely used for modelling software systems and is increasingly used for high designing non- software systems and organizations.

1.1.1 LOGICAL DESIGN:

The logical design of a system pertains to an abstract representation of the data flows, inputs and outputs of the system. This is often conducted via modelling, using an over-abstract (and sometimes graphical) model of the actual system. In the context of systems, designs are included. Logical design includes entity-relationship diagrams (ER diagrams).

1.1.2 PHYSICAL DESIGN:

The physical design relates to the actual input and output processes of the system. This is explained in terms of how data is input into a system, how it is verified /authenticated, how it is processed, and how it is displayed.

In physical design, the following requirements about the system are decided.

- a. Input requirement,
- b. Output requirements,
- c. Storage requirements,
- d. Processing requirements,
- e. System control and backup or recovery.

Put another way, the physical portion of system design can generally be broken down into three sub-tasks:

1. User Interface Design
2. Data Design
3. Process Design

User Interface Design is concerned with how users add information to the system and with how the system presents information back to them. It is concerned with how the data is represented and stored within the system. Finally, **Process Design** is concerned with how data moves through the system, and with how and where it is validated, secured and/or transformed as it flows into, through and out of the system.

At the end of the system design phase, documentation describing the three sub-tasks is produced and made available for use in the next phase. Physical design, in this context, does not refer to the tangible physical design of an information system.

To use an analogy, a personal computer's physical design involves input via a keyboard, processing within the CPU, and output via a monitor, printer, etc. It would not concern the actual layout of the tangible hardware, which for a PC would be a monitor, CPU, motherboard, hard drive, modems, video/graphics cards, USB slots, etc. It involves a detailed design of a user and a product database structure processor and control processor. The H/S personal specification is developed for the proposed system.

E-R DIAGRAM

3.1 E-R DIAGRAM

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is a component of data. In other words, ER diagrams illustrate the logical structure of databases.

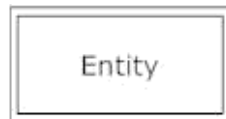
Structure of an Entity Relationship Diagram with Common ERD Notations

An entity relationship diagram is a means of visualizing how the information a system produces is related. There are five main components of an ERD:

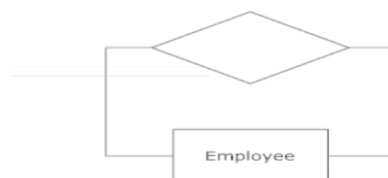
- **Entities** which are represented by rectangles. An entity is an object or concept about which you want to store information.



- **Weak entity** is an entity that must be defined by a foreign key relationship with another entity as it cannot be uniquely identified by its own attributes alone.



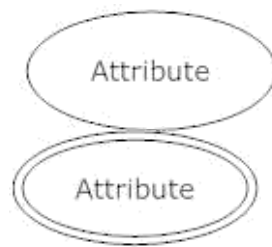
- **Actions**, which are represented by diamond shapes, show how two entities share information in the database. In some cases, entities can be self-linked. For example, employees can supervise other employees.



- **Relationship**: The degree of a relationship is the number of entity types that participate in the relationship.



- **Attributes**, which are represented by ovals. A key attribute is the unique, distinguishing characteristic of the entity. For example, an employee's social security number might be the employee's key attribute.

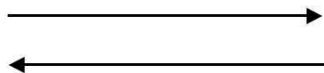


- **Multi-valued attribute** can have more than one value. For example, an employee entity can have multiple skill values.

- **Derived attribute** is based on another attribute. For example, an employee's monthly salary is based on the employee's annual salary.



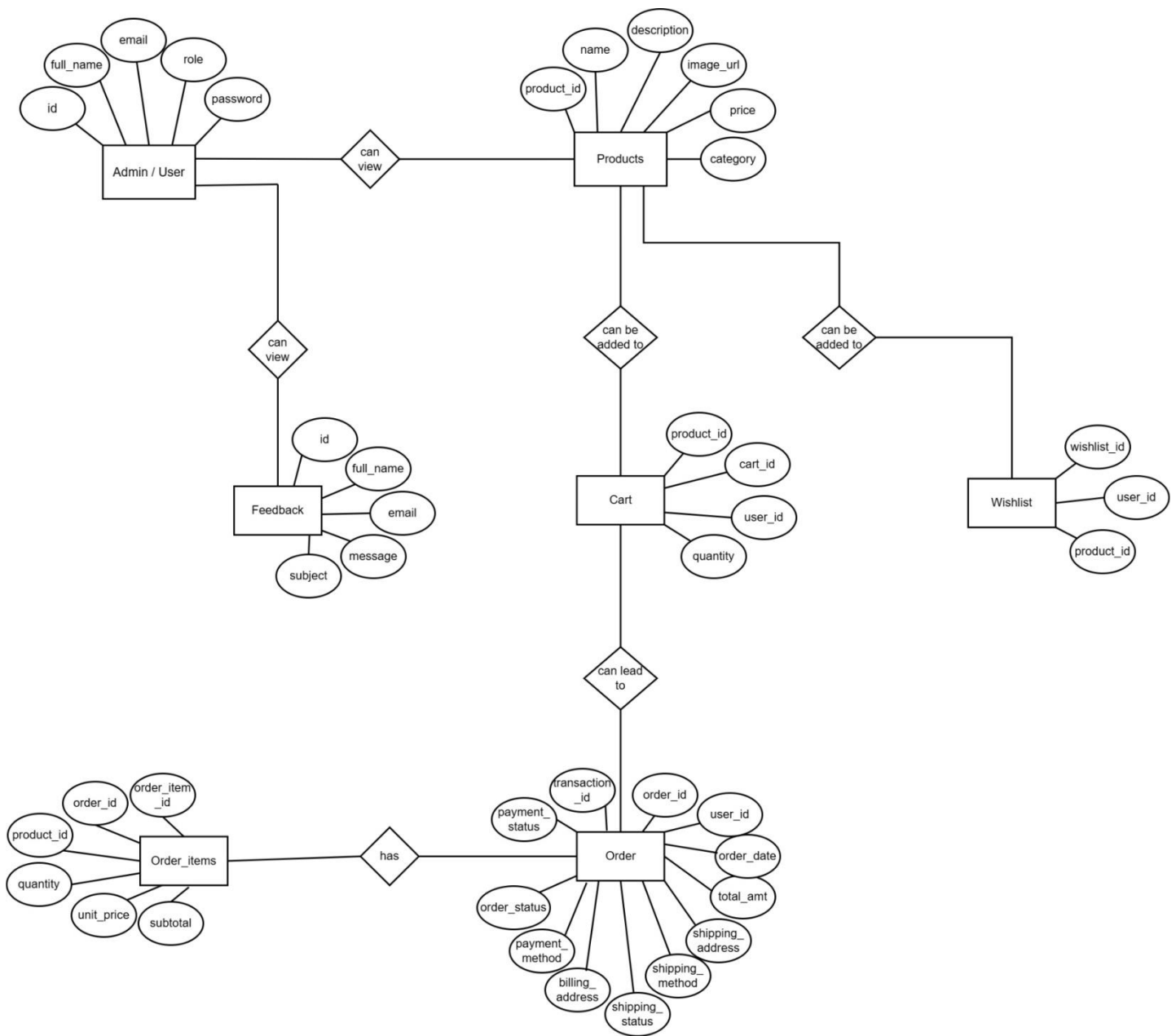
- **Connecting lines**, solid lines that connect attributes to show the relationships of entities in the diagram.



- **Cardinality** specifies how many instances of an entity relate to one instance of another entity. Ordinarily is also closely linked to cardinality. While cardinality specifies the occurrences of a relationship, ordinarily describes the relationship as either mandatory or optional. In other words, cardinality specifies the maximum number of relationships and cordiality specifies the absolute minimum number of relationships.

- One to One
- One to Many
- Many to One
- Many to Many

1.1.3 E-R DIAGRAM FOR BLOOMING HEARTS:



3.2 DATA FLOW DIAGRAM (level 0 and level 1)

The Data Flow Diagrams (DFDs) are used for structure analysis and design. DFDs show the flow of data from external entities into the system. DFDs also show how the data moves and are transformed from one process to another, as well as its logical storage. The following symbols are used within DFDs.

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of process or information about whether processes will operate in sequence or in parallel.

1.1.4 PHYSICAL VS LOGICAL DFD

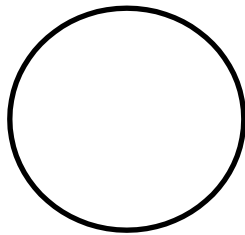
A logical DFD captures the data flows that are necessary for a system to operate. It describes the processes that are undertaken, the data required and produced by each process, and the stores needed to hold the data. On the other hand, a physical DFD shows how the system is actually implemented, either at the moment (Current Physical DFD), or how the designer intends it to be in the future (Required Physical DFD).

Thus, a Physical DFD may be used to describe the set of data items that appear on each piece of paper that move around an office, and the fact that a particular set of pieces of paper are stored together in a filing cabinet. It is quite possible that a Physical DFD will include references to data that are duplicated, or redundant, and that the data stores, if implemented as a set of database tables, would constitute an un-normalized (or de-normalized) relational database. In contrast, a Logical DFD attempts to capture the data flow aspects of a system in a form that has neither redundancy nor duplication.

1.1.5 DATA FLOW SYMBOLS AND THEIR MEANINGS: -

An entity: A source of data or a destination for data.

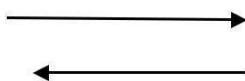
Source/Sink: Represented by rectangles in the diagram. Sources and Sinks are external entities which are sources or destinations of data, respectively.



Process: Represented by circles in the diagram. Processes are responsible for manipulating the data. They take data as input and output an altered version of the data.



Data Store: Represented by a segmented rectangle with an open end on the right. Data Stores are both electronic and physical locations of data. Examples include databases, directories, files, and even filing cabinets and stacks of paper.



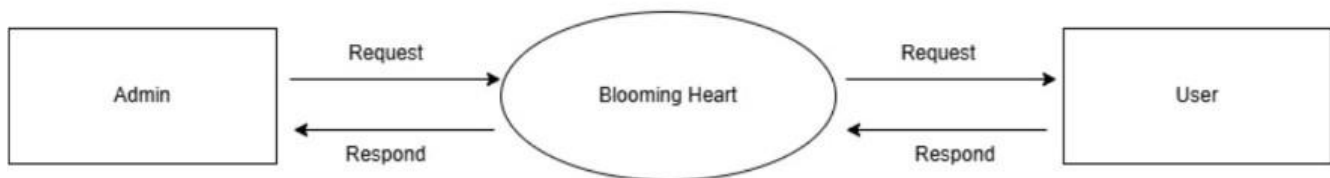
Data Flow: Represented by a unidirectional arrow. Data Flows show how data is moved through the System. Data Flows are labelled with a description of the data that is being passed through it.

A level-0 DFD is the most basic form of DFD. It aims to show how the entire system works at a glance. There is only one process in the system and all the data flows either into or out of this

process. Level-0 DFD's demonstrates the interactions between the process and external entities. They do not contain Data Stores.

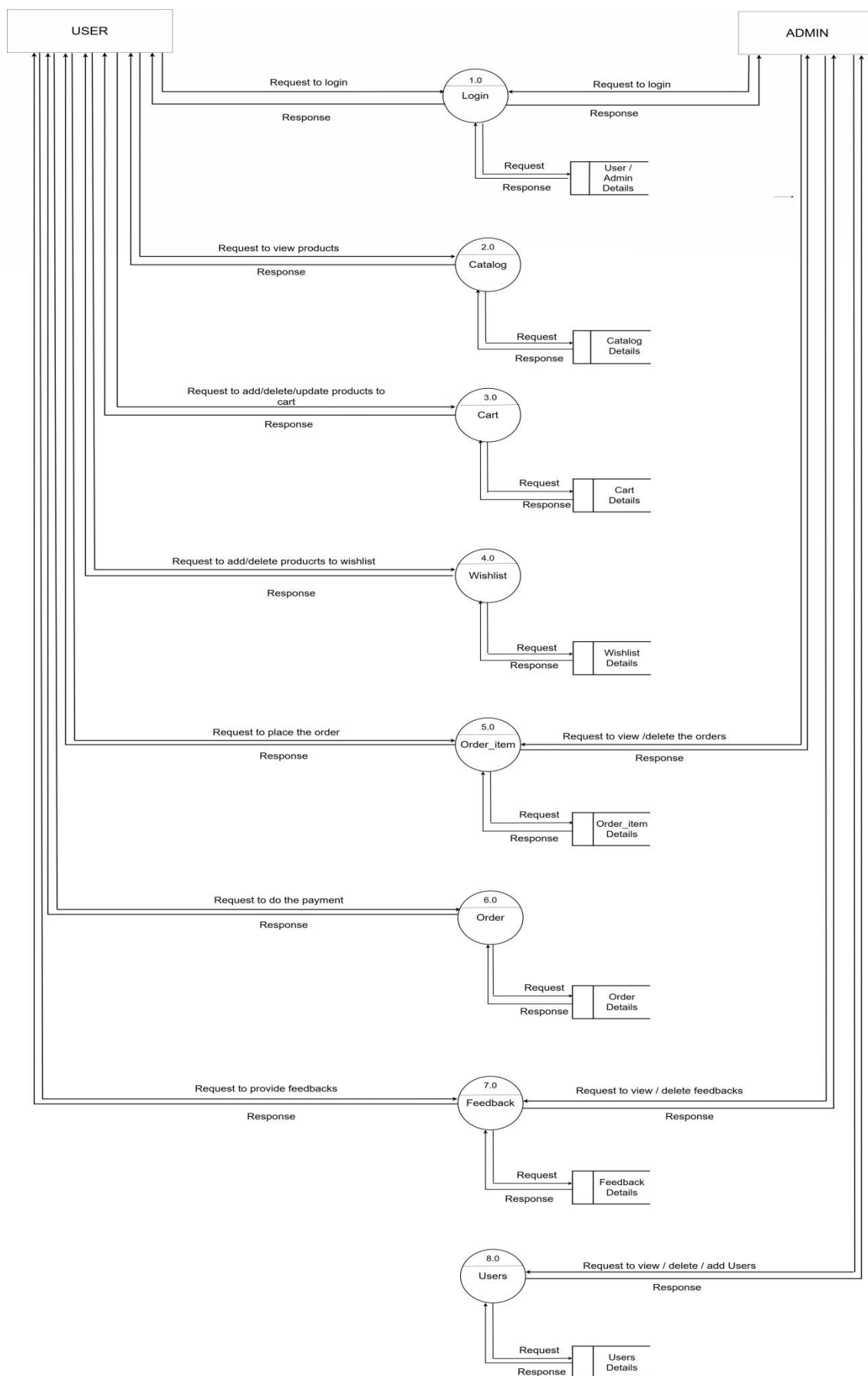
When drawing Level-0 DFD's, we must first identify the process, all the external entities and all the data flows. We must also state any assumptions we make about the system. It is advised that we draw the process in the middle of the page. We then draw our external entities in the corners and finally connect our entities to our process with the data flows.

Level 0 DFD/Context Diagram:



Level 1 DFD:

Level 1 DFD's aim is to give an overview of the full system. They look at the system in more detail. Major processes are broken down into sub-processes. Level 1 DFD's also identifies data stores that are used by the major processes. When constructing a Level 1 DFD we must start by examining the Context Level DFD. We must break up the single process into its subprocesses. We must then pick out the data stores from the text we are given and include them in our DFD. Like the Context Level DFD's, all entities, data stores and processes must be labelled. We must also state any assumptions made from the text.



3.3 ACTIVITY DIAGRAM

An activity diagram visually represents the series of actions or flow of control in a system similar to a flow chart or data flow diagram. Activity diagram are often used in business processing modelling. They can also describe steps in a used case diagram. The activity diagram for Admin module and User module of College Gadget Booking is given below.

GANTT CHART

A Gantt chart is a type of bar chart, devised by Henry Gantt in the 1910s, that illustrates a project schedule. Gantt charts illustrate the start and finish dates of the terminal elements and summary elements of a project. Terminal elements and summary elements comprise the work breakdown structure of the project. Modern Gantt charts also show the dependency (i.e., precedence network) relationships between activities.

1.1.6 HISTORICAL DEVELOPMENT:

The first known tool of this type was developed in 1896 by Karol Adamiecki, who called it a Harmon gram. Adamiecki did not publish his chart until 1931, however, and only in Polish, which limited both its adoption and recognition of his authorship. The chart is named after Henry Gantt (1861–1919), who designed his chart around the years 1910–1915. One of the first major applications of Gantt charts was by the United States during World War I, at the instigation of General William Crozier in the 1980s, personal computers allowed widespread creation of complex and elaborate Gantt charts. The first desktop applications were intended mainly for project managers and project schedulers. With the advent of the Internet and increased collaboration over networks at the end of the 1990s, Gantt charts became a common feature of web-based applications, including collaborative groupware.

1.1.7 GANTT CHART BENEFITS:

Clarity:

One of the biggest benefits of a Gantt chart is the tool's ability to boil down multiple tasks and

timelines into a single document. Stakeholders throughout an organization can easily understand where teams are in a process while grasping the ways in which independent elements come together toward project completion.

Communication:

Teams can use Gantt charts to replace meetings and enhance other status updates. Simply clarifying chart positions offers an easy, visual method to help team members understand task progress.

Motivation:

Some teams or team members become more effective when faced with a form of external motivation. Gantt charts offer teams the ability to focus work at the front of a task timeline, or at the tail end of a chart segment. Both types of team members can find Gantt charts meaningful as they plug their own work habits into the overall project schedule.

Coordination:

For project managers and resource schedulers, the benefits of a Gantt chart include the ability to sequence events and reduce the potential for overburdening team members. Some project managers even use combinations of charts to break down projects into more manageable sets of tasks.

Creativity:

Sometimes, a lack of time or resources forces project managers and teams to find creative solutions. Seeing how individual tasks intertwine on Gantt charts often encourages new partnerships and collaborations that might not have evolved under traditional task assignment systems.

Time Management:

Most managers regard scheduling as one of the major benefits of Gantt charts in a creative environment. Helping teams understand the overall impact of project delays can foster stronger collaboration while encouraging better task organization.

Flexibility:

Whether you use Excel to generate Gantt charts or you load tasks into a more precise chart generator, the ability to issue new charts as your project evolves lets you react to unexpected changes in project scope or timeline. While revising your project schedule too frequently can eliminate some of the other benefits of Gantt charts, offering a realistic view of a project can help team members recover from setbacks or adjust to other changes.

Manageability:

For project managers handling complex assignments, like software publishing or event planning, the benefits of Gantt charts include externalizing assignments. By visualizing all of the pieces of a project puzzle, managers can make more focused, effective decisions about resources and timetables.

Efficiency:

Another one of the benefits of Gantt charts is the ability for teams members to leverage each other's deadlines for maximum efficiency. For instance, while one team member waits on the outcome of three other tasks before starting a crucial piece of the assignment, he or she can perform other project tasks. Visualizing resource usage during projects allows managers to make better use of people, places, and things.

Accountability:

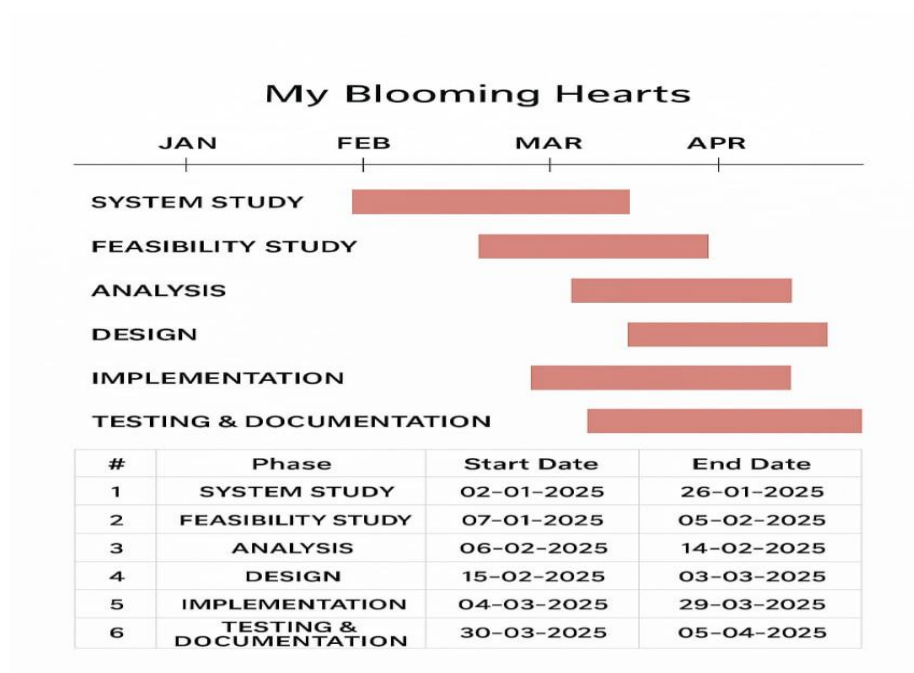
When project teams face major organizational change, documenting effort and outcomes becomes crucial to career success. Using Gantt charts during critical projects allows both project managers and participants to track team progress, highlighting both big wins and major failures during professional review periods; team members who frequently exceed expectations can leverage this documentation into larger raises or bonuses.

Gantt chart Importance:

The project's summary and terminal elements, which combine to form the project's internal structure, are shown on the Gantt chart. Many charts will also depict the precedence rankings and dependencies of various tasks within the project. The charts can illustrate the start and finish project terminal elements in project management. It can also show summary elements and terminal

dependencies. The smallest task tracked as part of the project effort is known as a terminal element. Gantt chart represents the tasks in most modern project scheduling packages. However other management applications use simpler communication tools such as message boards, to-do lists and simple scheduling etc., therefore, they do not use Gantt charts as heavily.

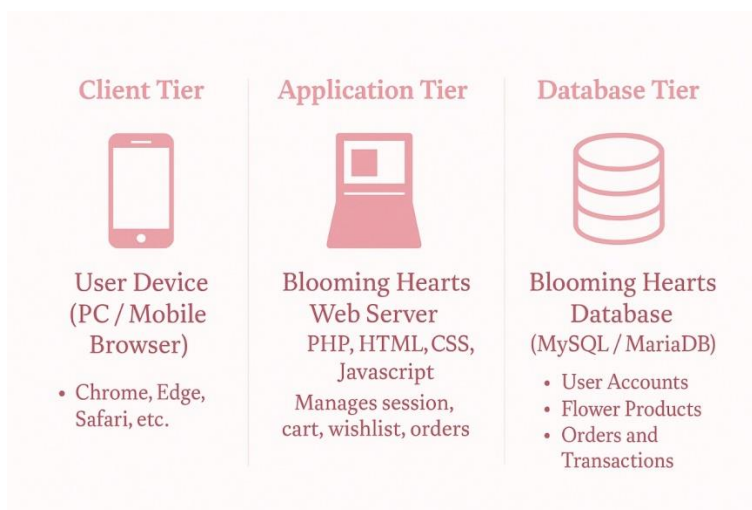
The way to create this chart begins by determining and listing the necessary activities. Next, sketch out how you expect the chart to look. List which items depend on others and what activities take place when. For each activity, list how many man-hours it will require, and who is responsible. Lastly, determine the throughput time.



This technique's primary advantage is its good graphical overview that is easy to understand for nearly all project participants and stakeholders. Its primary disadvantage is its limited applicability for many projects, since projects are often more complex than can be effectively communicated with this chart.

3.4 ARCHITECTURAL DESIGN

An architectural model (in software) is a rich and rigorous diagram, created using available standards, in which the primary concern is to illustrate a specific set of tradeoffs inherent in the structure and design of a system or ecosystem.



INPUT/OUTPUT DESIGN

3.5 INPUT/OUTPUT DESIGN

Input Design INDEX PAGE:

```

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta name="description" content="Blooming Hearts - Luxury Floral Arrangements and Gift Cards">

<title>Blooming Hearts - Luxury Floral Arrangements</title>

<link rel="preconnect" href="https://fonts.googleapis.com">

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<link
href="https://fonts.googleapis.com/css2?family=Playfair+Display:wght@400;700&family=Montserrat:
wght@300;400;500&display=swap" rel="stylesheet">

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css">

<style>

body{ margin:0;padding:0;min-height:100vh;font-family:'Montserrat',sans-serif;background-

.background-wrapper{ position:fixed;top:0;left:0;width:100%;height:100%;z-index:-1 }

.background-image{ position:absolute;width:100%;height:100%;background-
image:url('https://i.pinimg.com/736x/e5/e0/05/e5e00565bc9ab72034843e1374908cb4.jpg');background-
size:cover;background-position:center;filter:brightness(0.8)}

.header{ position:fixed;top:0;left:0;right:0;background:rgba(53,10,24,0.9);padding:1rem 0;z-index:100}

.nav-container{ max-width:1200px;margin:0      auto;display:flex;justify-content:space-between;align-
```

```
items:center;padding:0 2rem }
```

```
.logo{ display:flex;align-items:center;gap:0.5rem;text-decoration:none }
```

```
.logo img{ height:30px }
```

```
.logo span{ color:#fff;font-size:1.5rem }
```

```
.nav-links{ display:flex;gap:2rem }
```

```
.nav-links a{ color:#fff;text-decoration:none;font-size:1rem;transition:opacity 0.3s }
```

```
.nav-links a:hover{ opacity:0.8 }
```

```
.social-links{ position:fixed;right:2rem;top:50%;transform:translateY(-50%);display:flex;flex-direction:column;gap:1.5rem;z-index:90 }
```

```
.social-links a{ color:#fff;font-size:1.2rem;opacity:0.8;transition:all 0.3s }
```

```
.social-links a:hover{ opacity:1;transform:scale(1.1) }
```

```
.hero-content{ position:relative;padding:12rem 5rem 5rem;max-width:600px }
```

```
.valentine-text{ font-family:'Playfair Display',serif;color:#381822;font-size:1.2rem;margin-bottom:1rem;font-style:italic }
```

```
.main-heading{ font-family:'Playfair Display',serif;font-size:3.5rem;margin:0.5rem 0;line-height:1.2;color:#290808 }
```

```
.sub-heading{ font-size:1.1rem;color:#381822;margin:1rem 0 2rem;max-width:400px }
```

```
.explore-btn{ display:inline-block;padding:0.8rem 2rem;background-color:#381822;color:#fff;text-decoration:none;border-radius:4px;font-size:1rem;transition:all 0.3s }
```

```
.explore-btn:hover{ background-color:#290808;transform:translateY(-2px) }
```

```
.slider-dots{ position:absolute;bottom:2rem;left:5rem;display:flex;gap:0.5rem }
```

```
.dot{ width:8px;height:8px;background:rgba(255,255,255,0.5);border-radius:50% }
```

```
.dot.active{ background:#290808 }
```

```
@media(max-width:768px){
```

```
.nav-container{padding:0 1rem}
```

```
.nav-links{display:none}
```

```
.hero-content{padding:8rem 2rem 3rem}
```

```
.main-heading{font-size:2.5rem}
```

```
.social-links{right:1rem}
```

```
.slider-dots{left:2rem}
```

```
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<div class="background-wrapper">
```

```
<div class="background-image"></div>
```

```
</div>
```

```
<div class="glass-container">
```

```
<nav class="main-nav"></nav>
```

```
<main class="hero-content">
```

```
<h1 class="brand-name">Blooming Hearts</h1>
```

```
<p class="main-heading">select flowers you love, packing you want</p>
```

```
<p class="sub-heading">A bloom for beauty, a card for meaning — together, they say it all</p>
```

```
<a href="login.html" class="explore-btn cta-button">Explore more</a>
```

```
</main>
```

```
<script>
```

```
document.addEventListener('DOMContentLoaded',()=>{const ctaButton=document.querySelector('.cta-
```

```
button');ctaButton.addEventListener('click',()=>{ctaButton.style.transform='scale(0.95)';setTimeout(()=>{ctaButton.style.transform='scale(1)'},200);console.log('Let\'s Get Started button clicked!');});});
```

```
</script>
```

```
</body>
```

```
</html>
```

LOGIN PAGE:

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta name="description" content="Blooming Hearts - Login Page">

<title>Blooming Hearts - Login</title>

<link rel="preconnect" href="https://fonts.googleapis.com">

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<link
href="https://fonts.googleapis.com/css2?family=Playfair+Display:wght@400;700&family=Montserrat:wght@300;400;500&display=swap" rel="stylesheet">

<style>

*{margin:0;padding:0;box-sizing:border-box}

body{font-family:'Montserrat',sans-serif;height:100vh;overflow:hidden;position:relative}

.background-image{position:absolute;top:0;left:0;width:100%;height:100%;background-
image:url('https://i.pinimg.com/736x/e5/e0/05/e5e00565bc9ab72034843e1374908cb4.jpg');background-
size:cover;background-position:center;filter:blur(4px);z-index:-1}

.glass-container{position:absolute;top:50%;left:50%;transform:translate(-50%,-50%);width:90%;max-
width:500px;background:rgba(255,255,255,0.2);backdrop-filter:blur(10px);border:1px solid
rgba(255,255,255,0.3);border-radius:20px;padding:2rem;box-shadow:0 8px 32px 0 rgba(31,38,135,0.1)}

.form-group{position:relative;margin-bottom:2rem}

.form-group input{width:100%;padding:1rem;background:rgba(255,255,255,0.1);border:1px solid
rgba(125,87,87,0.3);border-radius:8px;font-family:'Montserrat',sans-serif;font-
size:1rem;color:#111111;transition:all 0.3s ease}
```

```
.form-group label{position:absolute;left:1rem;top:50%;transform:translateY(-50%);color:#674141;pointer-
events:none;transition:all 0.3s ease}
```

```
.form-group input:focus,.form-group input:valid{outline:none;border-color:#e6a4b4}
```

```
.form-group          input:focus+label,.form-group          input:valid+label{top:0;left:1rem;font-
size:0.8rem;background:rgba(255,255,255,0.2);padding:0 0.5rem;color:#e6a4b4}
```

```
.submit-btn{background:#381822;color:#fff;border:none;border-radius:0.5em;cursor:pointer;font-
size:18px;padding:0.7em 1.7em;width:100%;margin-bottom:1rem;transition:all 0.3s}
```

```
.submit-btn: hover{background:#290808}
```

```
.auth-form.active{display:block;opacity:1;transform:translateY(0)}
```

```
.message-container{margin-bottom:1rem;font-size:0.9rem;text-align:center;display:none}
```

```
.message-container.success{color:green}
```

```
.message-container.error{color:red}
```

```
.message-container.info{color:#555}
```

```
@media(max-width:768px){.glass-container{width:95%;padding:1.5rem}}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<div class="background-image"></div>
```

```
<div class="glass-container">
```

```
<h1 class="auth-title">Login to Blooming Hearts</h1>
```

```
<form id="login-form" class="auth-form active">
```

```
<div class="form-group">
```

```
<input type="email" id="login-email" name="email" required>
```

```
<label for="login-email">Email</label>
```

</div>

<div class="form-group">

<input type="password" id="login-password" name="password" required>

<label for="login-password">Password</label>

</div>

<div id="login-message" class="message-container"></div>

<button type="submit" class="submit-btn">Sign In</button>

</form>

</div>

<script>

document.addEventListener('DOMContentLoaded',function(){

const loginForm=document.getElementById('login-form');

const messageContainer=document.getElementById('login-message');

loginForm.addEventListener('submit',async function(e){

e.preventDefault();

const email=document.getElementById('login-email').value.trim();

const password=document.getElementById('login-password').value;

try{

showMessage('Logging in...','info');

const response=await fetch('../php/login.php',{

method:'POST',

headers:{'Content-Type':'application/x-www-form-urlencoded'},

body:email=\${encodeURIComponent(email)}&password=\${encodeURIComponent(password)}

```
});

if(!response.ok){

throw new Error(HTTP error! status: ${response.status});

}

const contentType=response.headers.get('content-type');

if(!contentType||!contentType.includes('application/json')){

throw new Error('Server did not return JSON!');

}

const text=await response.text();

if(!text){

throw new Error('Empty response from server');

}

let data;

try{

data=JSON.parse(text);

}catch(e){

console.error('Failed to parse JSON:',text);

throw new Error('Invalid JSON response from server');

}

if(data.success){

sessionStorage.setItem('userData',JSON.stringify({

full_name:data.full_name,

email:data.email,
```



```
    ));  
  
    showMessage('Login successful! Redirecting...', 'success');  
  
    setTimeout(() => {  
  
        window.location.href = data.redirect;  
  
    }, 1500);  
  
    } else {  
  
        showMessage(data.message || 'Login failed. Please try again.', 'error');  
  
    }  
  
    } catch (error) {  
  
        console.error('Login error:', error);  
  
        showMessage(error.message || 'Network error. Please try again later.', 'error');  
  
    }  
  
    });  
  
    function showMessage(msg, type) {  
  
        messageContainer.textContent = msg;  
  
        messageContainer.className = `message-container ${type}`;  
  
        messageContainer.style.display = 'block';  
  
    }  
  
    });  
  
</script>  
  
</body>  
  
</html>
```

REGISTRATION PAGE:

```

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8"/>

<meta name="viewport" content="width=device-width, initial-scale=1.0"/>

<title>Blooming Hearts - Register</title>

<link rel="preconnect" href="https://fonts.googleapis.com"/>

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin/>

<link href="https://fonts.googleapis.com/css2?family=Playfair+Display:wght@400;700&family=Montserrat:wght@300;400;500&display=swap" rel="stylesheet"/>

<style>

body{ margin:0;padding:0;min-height: 100vh;font-family:'Montserrat',sans-serif;background-color:#1a1a1a;color:#fff;display:flex;align-items:center;justify-content:center}

.background-image{ position:fixed;top:0;left:0;width:100%;height:100%;background-image:url('https://i.pinimg.com/736x/e5/e0/05/e5e00565bc9ab72034843e1374908cb4.jpg');background-size:cover;background-position:center;z-index:-1;filter:brightness(0.6)}

.glass-container{ background:rgba(255,255,255,0.1);backdrop-filter:blur(10px);padding:2rem;border-radius:10px;box-shadow:0 0 20px rgba(0,0,0,0.2);width:100%;max-width:400px}

.auth-title{ font-family:'Playfair Display',serif;text-align:center;font-size:2rem

```

```
;margin-bottom:1.5rem;color:#fff}

.auth-form .form-group{ position:relative;margin-bottom:1.5rem}

.auth-form input{ width:100%;padding:0.75rem;border:none;border-radius:5px

;background-color:rgba(255,255,255,0.15);color:#fff;font-size:1rem}

.auth-form label{ position:absolute;top:50%;left:0.75rem;transform:translateY(-50%);color:

#ccc;font-size:0.9rem;pointer-events:none;transition:0.2s}

.auth-form input:focus+label,.auth-form input:not(:placeholder-shown)+

label{ top:-0.6rem

;font-size:0.75rem;color:#f5f5f5;background:#1a1a1a;padding:0 0.3rem}

.submit-btn{ width:100%;padding:0.75rem;background-color:#381822;border:none;border-

radius:5px;color:#fff;font-size:1rem;cursor:pointer;transition:all 0.3s}

.submit-btn:hover{ background-color:#290808}

.switch-auth{ text-align:center;margin-top:1rem;font-size:0.9rem}

.switch-auth a{ color:#f5c6cb;text-decoration:none}

.message-container{ display:none;margin-bottom:1rem;padding:0.6rem;border-radius:4px;font-

size:0.9rem}

.message-container.success{ background-color:#d4edda;color:#155724}

.message-container.error{ background-color:#f8d7da;color:#721c24

}

</style>

</head>
```

<body>

<div class="background-image"></div>

<div class="glass-container">

<h1 class="auth-title">Create Account</h1>

<form id="register-form" class="auth-form active">

<div class="form-group">

<input type="text" id="register-full-name" name="full_name" required placeholder=" "/>

<label for="register-full-name">Full Name</label>

</div>

<div class="form-group">

<input type="email" id="register-email" name="email" required placeholder=" "/>

<label for="register-email">Email</label>

</div>

<div class="form-group">

<input type="password" id="register-password" name="password" required placeholder=" "/>

<label for="register-password">Password</label>

</div>

<div class="form-group">

<input type="password" id="register-confirm-password" name="confirm_password" required
placeholder=" "/>

<label for="register-confirm-password">Confirm Password</label>

</div>

```
<div id="register-message" class="message-container"></div>
```

```
<button type="submit" class="submit-btn">Create Account</button>
```

```
<p class="switch-auth">Already have an account? <a href="login.html">Login here</a></p>
```

```
</form>
```

```
</div>
```

```
<script>
```

```
document.addEventListener('DOMContentLoaded',function(){const
registerForm=document.getElementById('register-form');const
messageContainer=document.getElementById('register-
message');registerForm.addEventListener('submit',async function(e){e.preventDefault();
const full_name=document.getElementById('register-full-name').
```

```
value.trim();const email=document.getElementById('register-email').value.trim();const
password=document.getElementById('register-password').value.trim();const
confirmPassword=document.getElementById('register-confirm-password').value.trim();const
emailRegex=/^[a-zA-Z0-9._%+-]+@gmail\.com$/;if(!emailRegex.test(email)){showMessage('Please
```

```
enter a valid Gmail address (e.g. yourname@gmail.com)','error');return;}const passwordRegex=
```

```
/^(?=[a-z])(?=[A-
```

```
Z])(?=[!@#$$%^&]).{8,}$$/;if(!passwordRegex.test(password)){showMessage('Password must be at
least 8 characters and include one uppercase letter, one lowercase letter, and one special
```

```
character (!@#$$%^&*'),'error');return;}if(password!==confirmPassword){showMessage('Passwords
```

```
do not match!','error');return;}try{const response
```

```
=await fetch('/rosiee/php/register.php',{method:'POST',headers:{'Content-Type':'application/x
```

```
-www-form-
```

```
urlencoded'},body:full_name=${encodeURIComponent(full_name)}&email=${encodeURIComponent(email)}&
```

```
password=${encodeURIComponent(password)}&confirm_password=${
```

```
{encodeURIComponent(confirmPassword))}
```

```
;const data=await response.json();if(data.success){showMessage('Registration successful! Redirec
```

```
ting
```

```
login...','success');this.reset();setTimeout(()=>{window.location.href='login.html';},1500);}else{s
```

```
howM
```

```
essage(data
```

```
.message||'Registration failed. Please try again.','error');}}catch(error){showMessage('An error
```

```
occurred. Please try again.','error');console.error('Error:',error);});});fun
```

```
ction
```

```
showMessage(message,type){messageContainer.textContent=message;messageContainer.className='
```

```
message-container '+type;messageContainer.style.display='block';});
```

```
</script>
```

```
</body>
```

```
</html>
```

USER DASHBOARD :

```
<!DOCTYPE html>
```

```
<html lang="en">
```

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Blooming Hearts - User Dashboard</title>

<link rel="preconnect" href="https://fonts.googleapis.com">

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<link href="https://fonts.googleapis.com/css2?family=Playfair+Display:wght@400;700&family=

Montserrat

t:wght@300;400;500&family=Dancing+Script:wght@700&display=swap" rel="stylesheet">

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css">

<style>

* {

margin: 0;

padding: 0;

box-sizing: border-box;

}

body {

font-family: 'Montserrat', sans-serif;

height: 100vh;

overflow: hidden;

position: relative;

}

```
.background-image {  
  
    position: absolute;  
  
    top: 0;  
  
    left: 0;  
  
    width: 100%;  
  
    height: 100%;  
  
    background-image:  
url('https://i.pinimg.com/736x/e5/e0/05/e5e00565bc9ab72034843e1374908cb4.jpg');  
  
    background-size: cover;  
  
    background-position: center;  
  
    z-index: -1;  
  
}  
  
.glass-container {  
  
    display: flex;  
  
    flex-direction: column;  
  
    align-items: center;  
  
    justify-content: center;  
  
    min-height: 100vh;  
  
    padding: 2rem;  
  
    text-align: center;  
  
    background: rgba(255, 255, 255, 0.2);  
  
    backdrop-filter: blur(10px);  
  
    border: 1px solid rgba(255, 255, 255, 0.3);
```



```
border-radius: 20px;

box-shadow: 0 8px 32px 0 rgba(31, 38, 135, 0.1);

overflow-y: auto;

}

.dashboard-header {

margin-bottom: 3rem;

}

.welcome-title {

font-size: 2.5rem;

margin-bottom: 1rem;

font-family: 'Dancing Script', cursive;

color: #333;

text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.1);

}

.dashboard-nav {

margin-bottom: 2rem;

}

.nav-grid {

display: grid;

grid-template-columns: repeat(5, 1fr);

gap: 2rem;

margin: 0 auto;

max-width: 1000px;
```

```
}

.nav-item {

    display: flex;

    flex-direction: column;

    align-items: center;

    padding: 1.5rem;

    background: rgba(255, 255, 255, 0.7);

    border-radius: 15px;

    text-decoration: none;

    color: #333;

    transition: all 0.3s ease;

    backdrop-filter: blur(10px);

    position: relative;

}

.nav-item i {

    font-size: 2rem;

    margin-bottom: 0.5rem;

    color: #9b3f56;

}

.nav-item span {

    font-family: 'Montserrat', sans-serif;

    font-size: 1rem;

}
```

```
.cart-count, .wishlist-count {  
  
    position: absolute;  
  
    top: -8px;  
  
    right: -8px;  
  
    background: #b9818c;  
  
    color: rgb(249, 245, 245);  
  
    border-radius: 50%;  
  
    width: 20px;  
  
    height: 20px;  
  
    display: flex;  
  
    align-items: center;  
  
    justify-content: center;  
  
    font-size: 0.8rem;  
  
    font-weight: bold;  
  
}  
  
.nav-item:hover {  
  
    transform: translateY(-5px);  
  
    box-shadow: 0 5px 15px rgba(0, 0, 0, 0.1);  
  
}  
  
@media (max-width: 768px) {  
  
    .glass-container {  
  
        width: 95%;  
  
        height: 95vh;  

```

```
padding: 1.5rem;

}

.welcome-title {

    font-size: 2rem;

}

.nav-grid {

    grid-template-columns: repeat(auto-fit, minmax(120px, 1fr));

}

}

</style>

</head>

<body>

    <div class="background-image"></div>

    <div class="glass-container">

        <header class="dashboard-header">

            <h1 class="welcome-title">Welcome to Your Dashboard, <span id="user-
name">User</span></h1>

        </header>

        <nav class="dashboard-nav">

            <div class="nav-grid">

                <a href="catalog.html" class="nav-item" id="catalog-link">

                    <i class="fas fa-store"></i>

                    <span>Catalog</span>

                </a>

            </div>

        </nav>

    </div>

</body>

</html>
```


<i class="fas fa-shopping-cart"></i>

Cart

0

<i class="fas fa-heart"></i>

Wishlist

0

<i class="fas fa-envelope"></i>

Contact Us

<i class="fas fa-sign-out-alt"></i>

Logout

</div>

</nav>

</div>

<script>

```
document.addEventListener('DOMContentLoaded', () => {

    const userData = JSON.parse(sessionStorage.getItem('userData'));

    if (userData) {

        document.getElementById('user-name').textContent = userData.full_name;

    }

    document.getElementById('cart-link').addEventListener('click', () => {

        window.location.href = 'cart.html';

    });

    document.getElementById('catalog-link').addEventListener('click', () => {

        window.location.href = 'catalog.html';

    });

    document.getElementById('wishlist-link').addEventListener('click', () => {

        window.location.href = 'wishlist.html';

    });

    document.getElementById('contact-link').addEventListener('click', () => {

        window.location.href = 'contact.html';

    });

    document.getElementById('logout-link').addEventListener('click', (e) => {

        e.preventDefault();

        sessionStorage.clear();

        window.location.href = 'login.html';

    });

});
```

```
const navItems = document.querySelectorAll('.nav-item');

navItems.forEach(item => {

  item.addEventListener('mouseenter', () => {

    item.style.transform = 'translateY(-5px)';

  });

  item.addEventListener('mouseleave', () => {

    item.style.transform = 'translateY(0)';

  });

});

loadCartCount();

loadWishlistCount();

});

async function loadCartCount() {

  try {

    const response = await fetch('/rosiee/php/get_cart.php');

    const data = await response.json();

    if (data.success) {

      const cartCount = data.items.reduce((total, item) => total + item.quantity, 0);

      document.querySelector('.cart-count').textContent = cartCount;

    }

  }

}
```

```
    } catch (error) {  
  
        console.error('Error loading cart count:', error);  
  
    }  
  
}  
  
async function loadWishlistCount() {  
  
    try {  
  
        const response = await fetch('/rosiee/php/get_wishlist.php');  
  
        const data = await response.json();  
  
        if (data.success) {  
  
            document.querySelector('.wishlist-count').textContent = data.total_items;  
  
        }  
  
    } catch (error) {  
  
        console.error('Error loading wishlist count:', error);  
  
    }  
  
}  
  
</script>  
  
</body>  
  
</html>
```

ADMIN DASHBOARD :

```
<!DOCTYPE html>  
  
<html lang="en">
```



```
<head>
```

```
<meta charset="UTF-8">
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
<title>Admin Dashboard - Blooming Hearts</title>
```

```
<link rel="preconnect" href="https://fonts.googleapis.com">
```

```
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
```

```
<link href="https://fonts.googleapis.com/css2?family=Playfair+Display:wght@
```

```
400;700&family= Montserrat:wght@300;400;500&family=Dancing+Script:wght@700
```

```
&display=swap" rel="stylesheet">
```

```
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css">
```

```
<style>*{margin:0;padding:0;box-sizing:border-box}body{font-family:'Montserrat',sans-serif;height:100vh;overflow:hidden;position:relative}.background-image{position:absolute;top:0;left:0;width:100%;height:100%;background-image:url('https://i.pinimg.com/736x/e5/e0/05/e5e00565bc9ab72034843e1374908cb4.jpg');background-size:cover;background-position:center;z-index:-1}.glass-container{position:absolute;top:50%;left:50%;transform:translate(-50%,-50%);width:90%;max-width:1200px;height:90vh;background:rgba(255,255,255,0.2);backdrop-filter:blur(10px);border:1px solid rgba(255,255,255,0.3);border-radius:20px;padding:2rem;box-shadow:0 8px 32px 0 rgba(31,38,135,0.1);display:flex;flex-direction:column}.dashboard-header{text-align:center;margin-bottom:2rem}.welcome-title{font-family:'Dancing Script',cursive;font-size:3.5rem;color:#111;text-shadow:2px 2px 4px rgba(0,0,0,0.1);margin-bottom:0.5rem}.admin-subtitle{font-family:'Playfair Display',serif;font-size:1.5rem;color:#34181f}.dashboard-nav{margin:0 auto;width:100%;max-width:800px}.nav-grid{display:grid;grid-template-columns:repeat(3,1fr);gap:2rem;padding:0 2rem}.nav-item{display:flex;flex-direction:column;align-items:center;text-decoration:none;color:#111;padding:1.5rem 1rem;background:rgba(255,255,255,0.
```

```

1);border-radius:15px;transition:all 0.3s ease}.nav-item i{font-size:2rem;

m;margin-bottom:0.5rem;color:#170f11}.nav-item span{font-family:'Playfair Display',se

rif;font-size:1.2rem}.nav-item:hover{background:rgba(255,255,255,0.2);transform:translateY(-
3px);box-shadow:0 4px 15px rgba(230,164,180,0.2)}.nav-
item.active{background:rgba(230,164,180,0.3);box-s

hadow:0 4px 15px rgba(230,164,180,0.3)}.dashboard-content{flex:1;margin-top:2rem;o

verflow-y:auto;padding:0 1rem}.search-bar{display:fl

ex;align-items:center;background:rgba(255,255,255,0.2);border-radius:30px;padding:0.8rem 1.5rem;

margin:0 auto 2rem;width:100%;max-width:500px}.search-bar i{color:#e6a4b4;margin-right:1

rem;font-size:1.2rem}.search-bar input{background:transparent;border:none;outline:none;width:100

%;font-family:'Montserrat',sans-serif;font-size:1rem;color:#333}.search-bar

input::placeholder{color:#777}.content-

section{display:none;opacity:0;transform:translateY(20px);transition:all 0.3s ease}.c

ontent-section.active{display:block;opacity:1;transform:translateY(0)}.section-

header{display:flex;justify-content:space-between;align-items:center;margin-bottom:2rem}.section-

header h2{font-family:

'Playfair Display',serif;font-size:2rem;color:#11

1}.add-btn{background:#e6a4b4;color:white;border:none;padding:0.8rem 1.5rem;border-radius:30px

;font-family:'Montserrat',sans-serif;font-size:1rem;cursor:pointer;display:flex;align-

items:center;gap:0.5rem;transition:all 0.3s ease}.add-btn:hover{background:#e04d72;transform:trans

lateY(-2px);box-shadow:0 4px 15px rgba(230,164,180,0.3)}

.table-container{background:rgba(255,255,255,0.1);border-radius:15px;overflow:hidden;margin-

top:1rem}table{width:100%;border-collapse:collapse}table th,table td{padding:1rem;text-align:left

;border-bottom:1px solid rgba(255,255,255,0.1)}table th{font-weight:500;color:#777;font-size:0.9rem

}table tbody tr:hover{background:rgba(255,255,255,0.1)}.a

ction-btn{background:transparent;border:none;color:#777;cursor:pointer;padding:0.5rem;margin:0

```

0.2rem;border-radius:5px;transition:all 0.3s ease}.

```

action-btn.edit:hover{ color:#2196F3;background:rgba(33,150,243,0.1)}.action-
btn.delete:hover{ color:#F44336;background:rgba(244,67,54,0.1)}.products-grid{ display:grid;grid-
template-columns:repeat(auto-fill,minmax(250px,1fr));gap:2rem;margin-top:1rem}.product-
card{ background:rgba(255,255,255,0.1);border-radius:15px;overflow:hidden;transition:all 0.3s ease}

.product-card:hover{ transform:translateY(-5px);box-shadow:0 8px 15px rgba(230,164,180,0.2)}.p
roduct-image{ width:100%;height:200px;overflow:hidden}.product-image
img{ width:100%;height:100%;object-fit:cover;transition:transform 0.3s ease}.product-card:hover .
product-image img{ transform:scale(1.1)}.product-details{ padding:1.5rem}.product-details h
3{ font-family:'Playfair Display',serif;font-size:1.3rem;color:#111;margin-bottom:0.5rem}.price
{ font-size:1.2rem;color:#e6a4b4;font-weight:500;margin-bottom:0.5rem}.stock{ font-
size:0.9rem;color:#777;margin-bottom:1rem}.product-actions{ display:flex;gap:1rem}.product-actions
.

action-btn{ flex:1;padding:0.8rem;font-size:0.9rem;display:flex;align-items:center;justify-
content:center;gap:0.5rem}.rating{ color:#ffd700;display:inline-flex;gap:2px}.rating
i{ font-size:14px}#feedback-section .table-container td{ max-width:300
px;white-space:nowrap;overflow:hidden;text-overflow:ellipsis}#feedback-section
.filter-buttons{ display:flex;gap:1rem;margin-bottom:1rem}#feedback-section .filter-btn{ padding:0
.5rem 1rem;background:rgba(255,255,255,0.1);border:1px solid rgba(230,164,180,0.3);b
order-radius:20px;color:#111;cursor:pointer;transition:all 0.3s ease}#feedback-section .filter-
btn:hover,

#fee
dback-section .filter-btn.active{ background:#e6a4b4;color:white;border-color:#e6a4b4}

.action-btn.reply:hover{ color:#4CAF50;background:rgba(76,175,80,0.1)
}@media(max-width:992px){.nav-gri

```

```
d{grid-template-columns:repeat(3,1fr);gap:1rem}.products-grid{grid-template-columns:repeat(auto-
fill,minmax(200px,1fr))}}@media(max-width:768px){.glass-
container{width:95%;height:95vh;padding:1.5rem}.welcome-title{font-size:2.5rem}.admin-
subtitle{font-size:1.2rem}.nav-grid{grid-template-columns:repeat(3,1fr);gap:0.8rem}.nav-
item{padding:1rem}.nav-item
i{font-size:1.5rem}.nav-item span{font-size:0.9rem}.section-header{flex-direction:column;gap:1rem;
align-items
:flex-start}.table-container{overflow-x:auto}.products-grid{grid-template-columns:1fr}#feedback-
section
.filter-buttons{flex-wrap:wrap;justify-content:center}#feedback-section .table-container
td{max-width:150px}}</style>
</head>
<body>
<div class="background-image"></div>
<div class="glass-container">
<header class="dashboard-header">
<h1 class="welcome-title">
```

CATALOG PAGE :

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
<meta charset="UTF-8">
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
<title>Admin Dashboard - Blooming Hearts</title>
```

```
<link rel="preconnect" href="https://fonts.googleapis.com">
```

```
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
```

```
<link href="https://fonts.googleapis.com/css2?family=Playfair+Display:wght@400;700&family=
```

```
Montserrat:
```

```
wght@300;400;500&family=Dancing+Script:wght@700&display=swap" rel="stylesheet">
```

```
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css">
```

```
<style>{*{margin:0;padding:0;box-sizing:border-box}body{font-family:'Montserrat',sans-serif;height:100vh;overflow:hidden;position:relative}.background-image{position:absol...
```

[8:14 AM, 4/26/2025] Rose: admin

[8:15 AM, 4/26/2025] Rose: <!DOCTYPE html>

```
<html lang="en">
```

```
<head>
```

```
<meta charset="UTF-8"/>
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0"/>
```

```
<title>Blooming Hearts - Flower Catalog</title>
```

<link

href="https://fonts.googleapis.com/css2?family=Montserrat&family=Dancing+Script&display=swap"
rel="stylesheet"/>

<link rel="stylesheet"

href="https://cdnjs.cloudflare.com/ajax/libs/font-

awesome/6.4.0/css/all.min.css"/>

<style>

* {

margin: 0;

padding: 0;

box-sizing: border-box;

}

body {

font-family: 'Montserrat', sans-serif;

height: 100vh;

overflow: hidden;

position: relative;

}

.background-image {

position: absolute;

top: 0;

left: 0;

width: 100%;

height: 100%;

background-image: url('flower-background.jpg');

```
background-size: cover;
```

```
background-position: center;
```

```
opacity: 0.6;
```

```
z-index: -1;
```

```
}
```

```
.glass-container {
```

```
position: absolute;
```

```
top: 50%;
```

```
left: 50%;
```

```
transform: translate(-50%, -50%);
```

```
width: 90%;
```

```
max-width: 1200px;
```

```
height: 90vh;
```

```
background: rgba(255, 255, 255, 0.2);
```

```
backdrop-filter: blur(10px);
```

```
border: 1px solid rgba(255, 255, 255, 0.3);
```

```
border-radius: 20px;
```

```
padding: 2rem;
```

```
box-shadow: 0 8px 32px rgba(31, 38, 135, 0.1);
```

```
overflow-y: auto;
```

```
}
```

```
.catalog-header {
```

```
display: flex;
```

flex-direction: column;

align-items: center;

margin-bottom: 2rem;

position: relative;

}

.back-link {

position: absolute;

left: 0;

top: 0;

font-size: 1rem;

text-decoration: none;

color: #111;

display: flex;

align-items: center;

gap: 0.5rem;

transition: 0.3s;

}

.back-link:hover {

color: #e6a4b4;

transform: translateX(-5px);

}

.catalog-title {

font-family: 'Dancing Script', cursive;

font-size: 3rem;

margin: 2rem 0;

color: #111;

text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.1);

}

.search-bar {

display: flex;

gap: 0.5rem;

max-width: 500px;

width: 100%;

margin-top: 1rem;

}

.search-bar input {

flex: 1;

padding: 0.8rem;

border-radius: 8px;

background: rgba(255, 255, 255, 0.1);

border: 1px solid rgba(125, 87, 87, 0.3);

color: #111;

}

.search-bar input:focus {

outline: none;

border-color: #e6a4b4;

```
}

.search-btn {

padding: 0.8rem 1rem;

border: none;

background: #e6a4b4;

color: white;

border-radius: 8px;

cursor: pointer;

transition: background 0.3s;

}

.search-btn:hover {

background: #d68ba0;

}

.product-grid {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));

gap: 2rem;

}

.product-card {

position: relative;

overflow: hidden;

border-radius: 12px;

box-shadow: 0 4px 12px rgba(0, 0, 0, 0.1);
```

```
transition: transform 0.3s ease;
```

```
}
```

```
.product-card:hover {
```

```
transform: scale(1.03);
```

```
}
```

```
.product-card img {
```

```
width: 100%;
```

```
height: auto;
```

```
display: block;
```

```
object-fit: cover;
```

```
}
```

```
.overlay {
```

```
position: absolute;
```

```
bottom: 0;
```

```
left: 0;
```

```
right: 0;
```

```
background: linear-gradient(to top, rgba(0, 0, 0, 0.7), transparent);
```

```
color: white;
```

```
padding: 1rem;
```

```
transform: translateY(100%);
```

```
transition: transform 0.3s ease;
```

```
}
```

```
.product-card:hover .overlay {
```

```
transform: translateY(0);

}

</style>

</head>

<body>

<div class="background-image"></div>

<div class="glass-container">

<header class="catalog-header">

<a href="#" class="back-link"><i class="fas fa-arrow-left"></i> Back</a>

<h1 class="catalog-title">Our Collection</h1>

<div class="search-bar">

<input type="text" placeholder="Search..." />

<button class="search-btn"><i class="fas fa-search"></i></button>

</div>

</header>

<section class="product-grid">

<div class="product-card">



<div class="overlay">

<h2>Red Rose</h2>

<p>$9.99</p>

</div>

</div>
```

```
<div class="product-card">



<div class="overlay">

<h2>Sunflower</h2>

<p>$8.50</p>

</div>

</div>

<div class="product-card">



<div class="overlay">

<h2>Tulip</h2>

<p>$7.25</p>

</div>

</div>

</section>

</div>

</body>

</html>
```

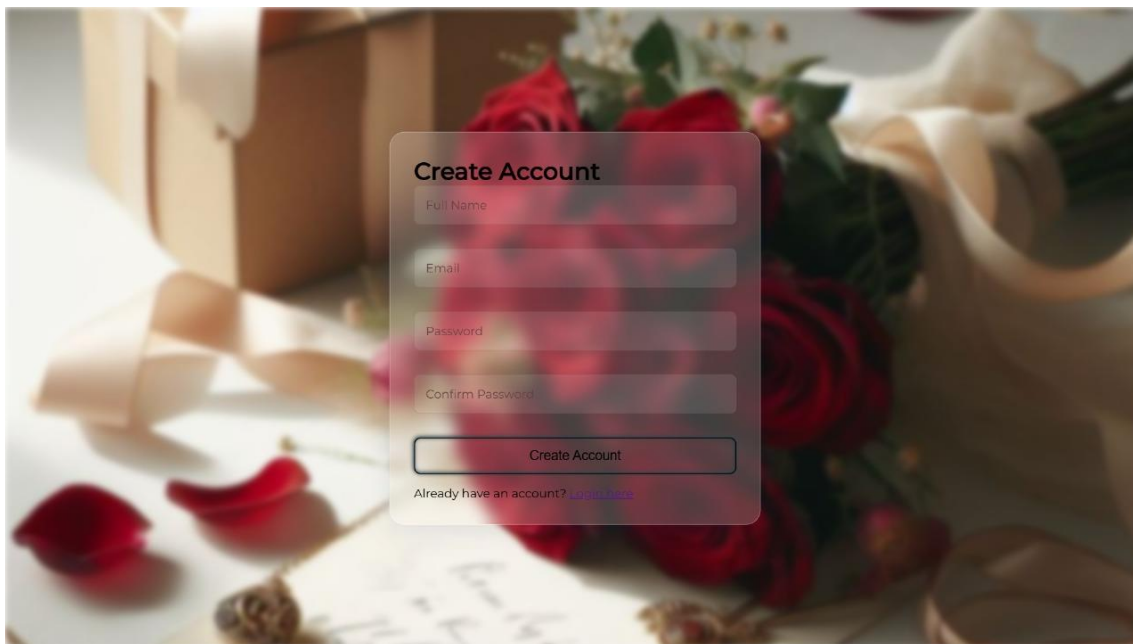
OUTPUT DESIGN

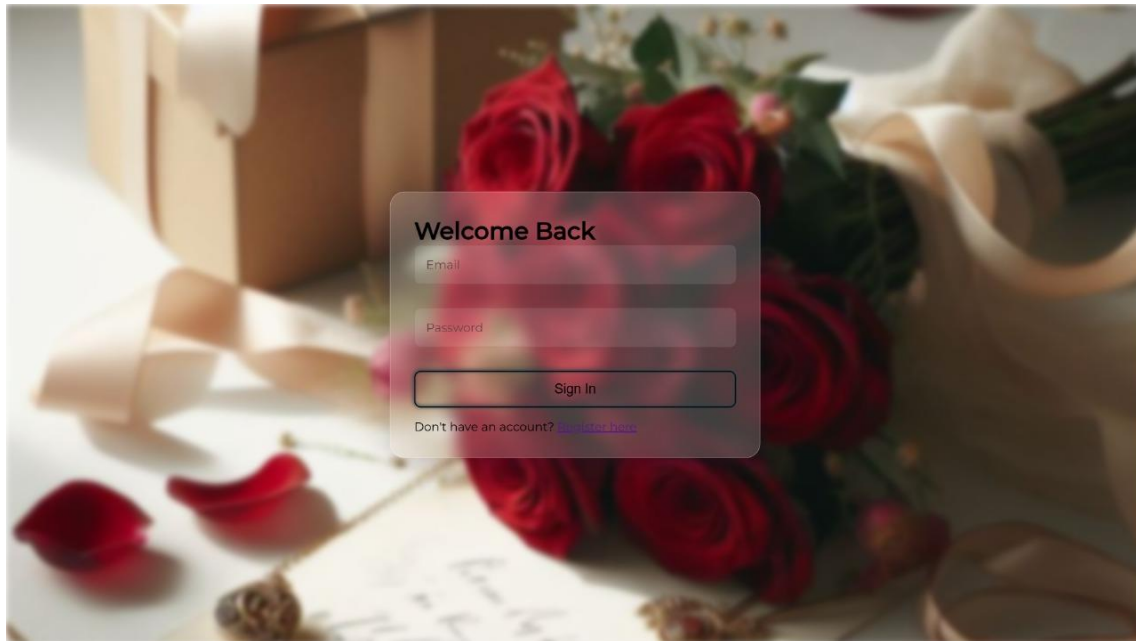
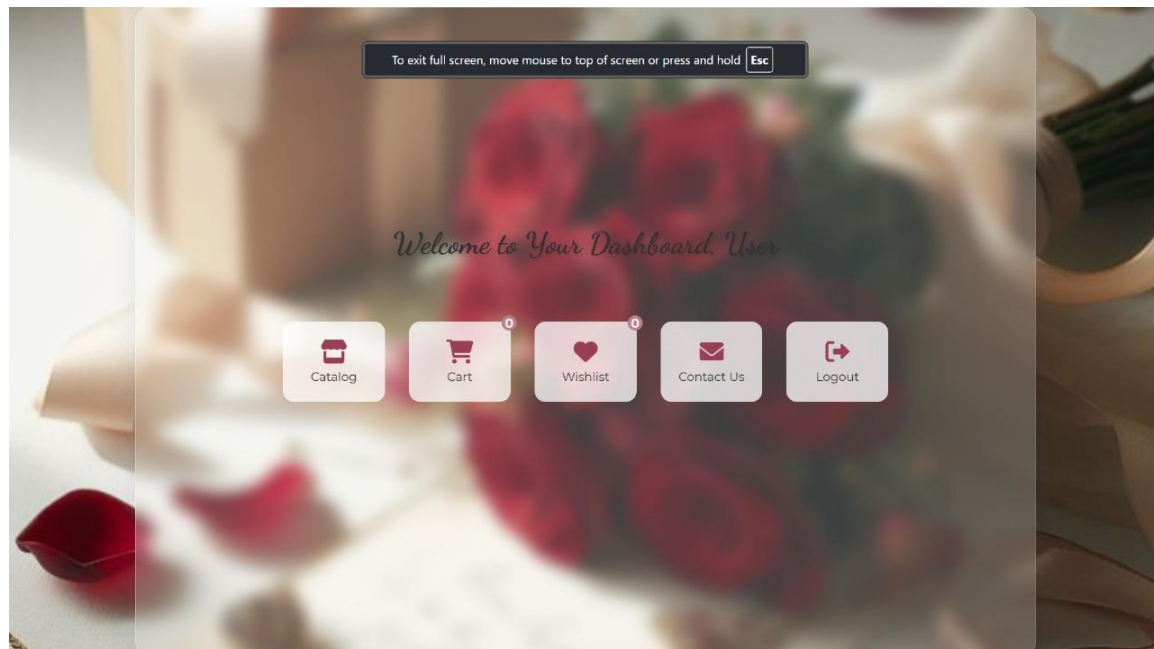
Output Design

INDEX:

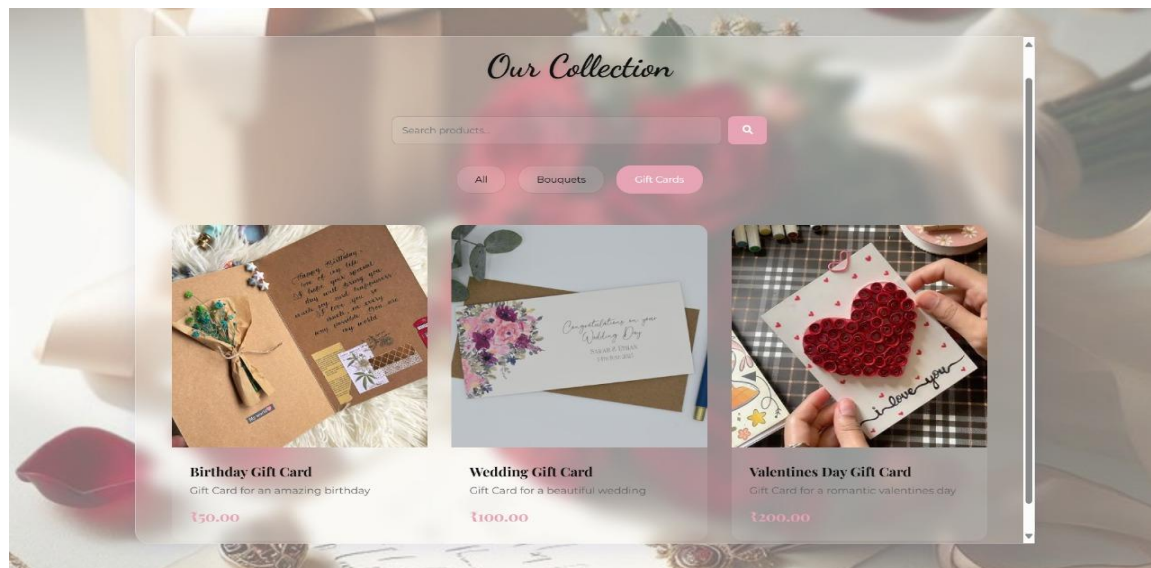
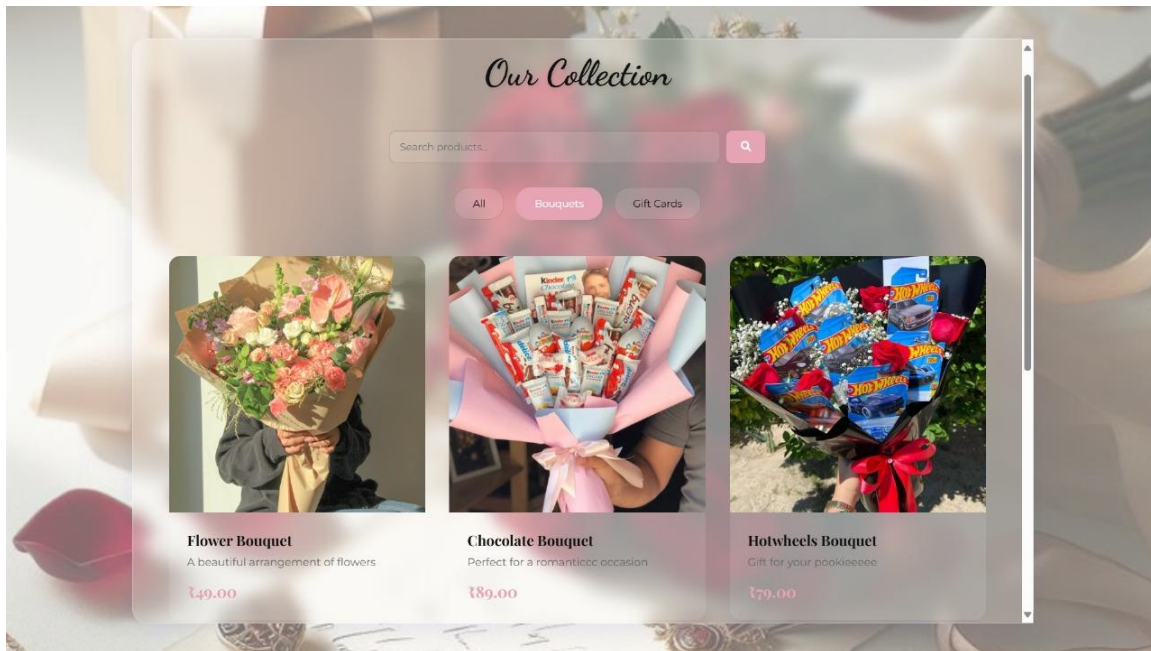


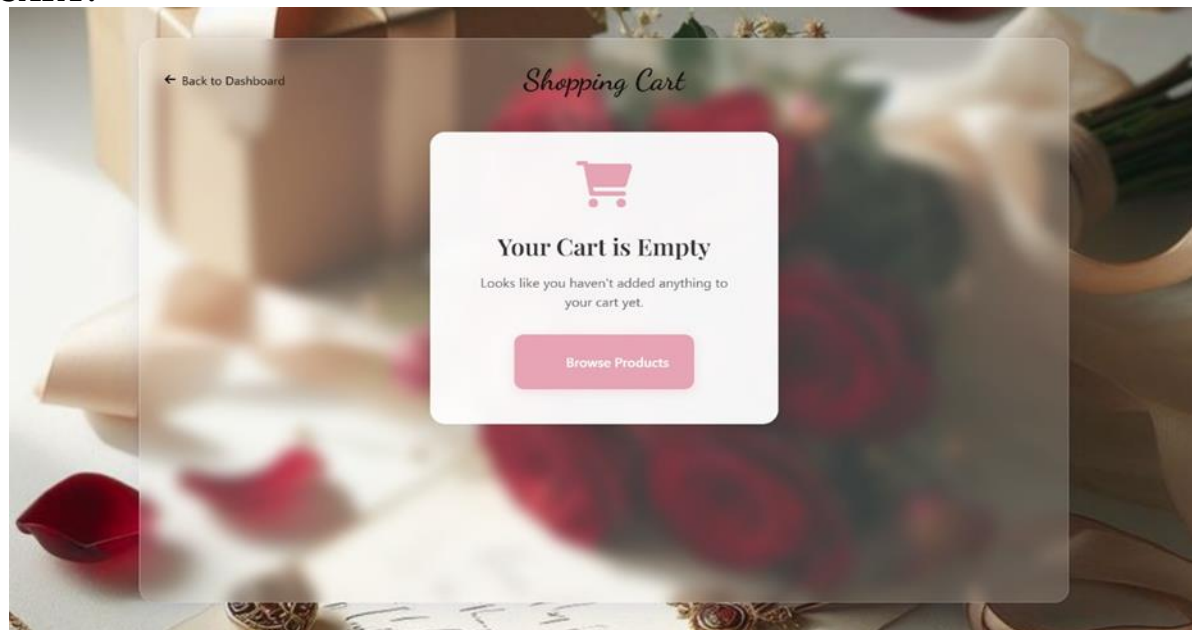
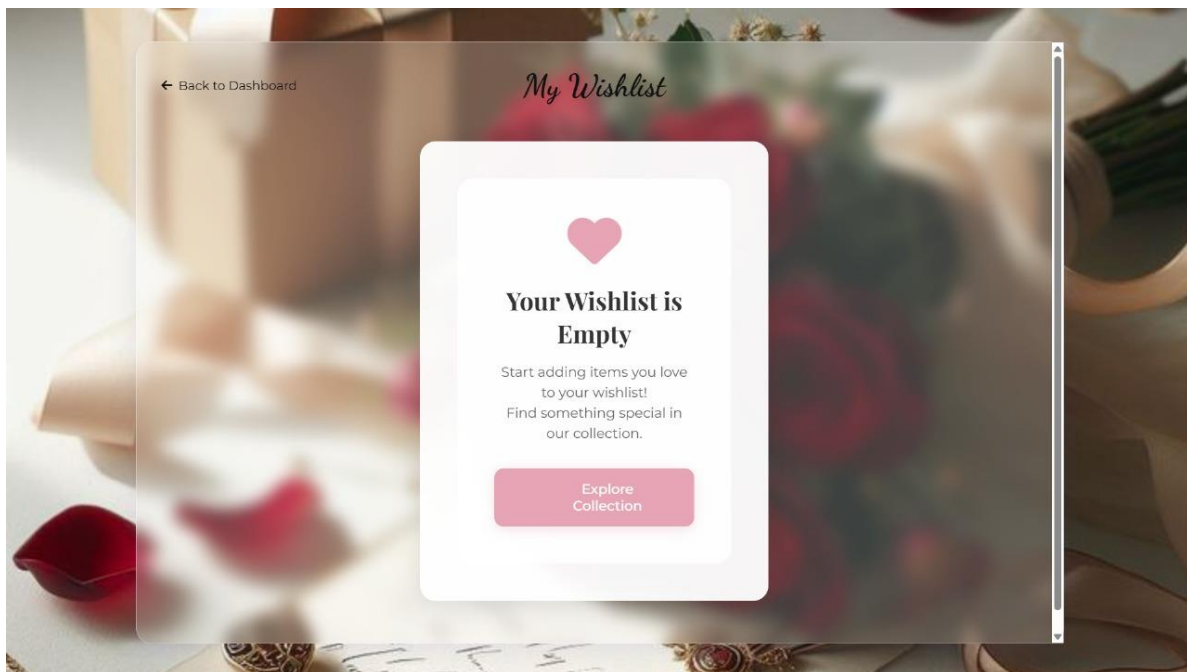
REGISTER:



LOGIN:**USER DASHBOARD:**

CATALOG:



CART:**WISHLIST:**


Shopping Cart - Blooming Hearts x Contact Us - Blooming Hearts x

127.0.0.1:5501/html/contact.html


[← Back to Dashboard](#)

Contact Us

We'd love to hear from you



Visit Us
K.Narayanapura, Kothanur
Bengaluru, Karnataka 560077



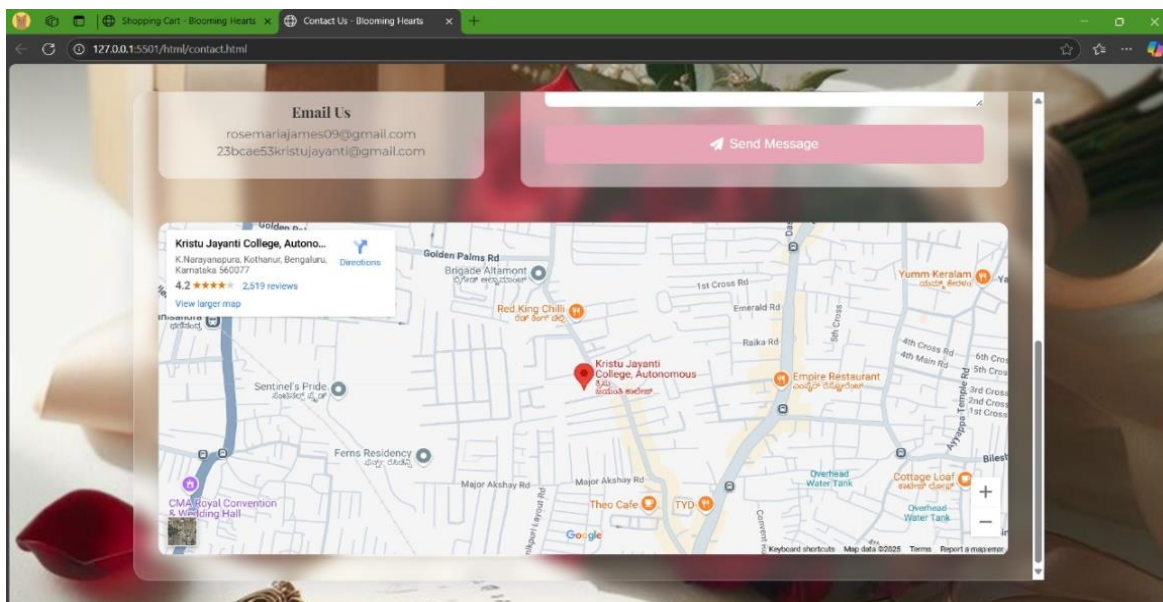
Call Us
+91 8861654623
Mon-Sat: 9:00 AM - 6:00 PM

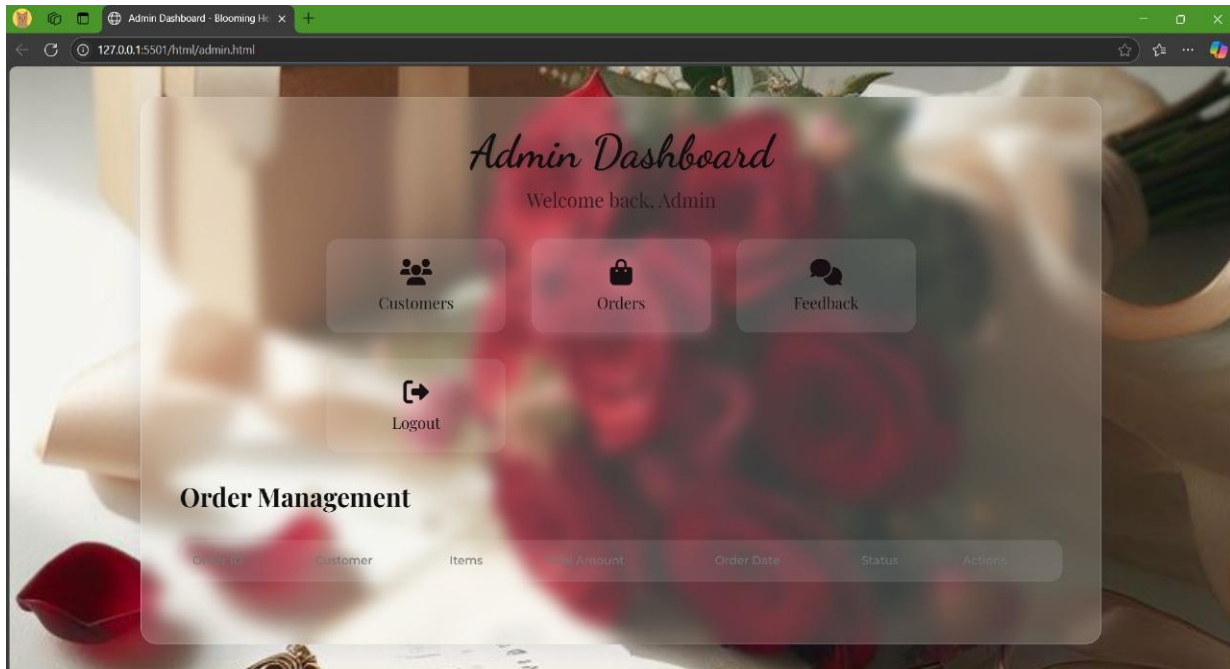
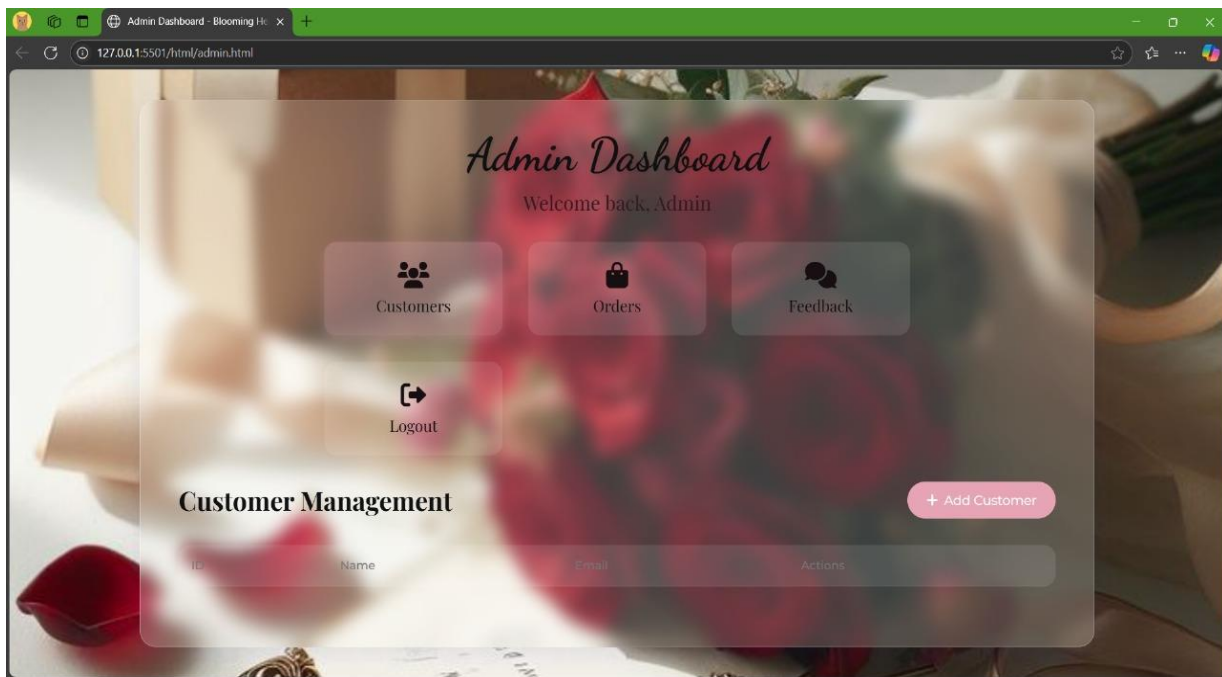
Your Name

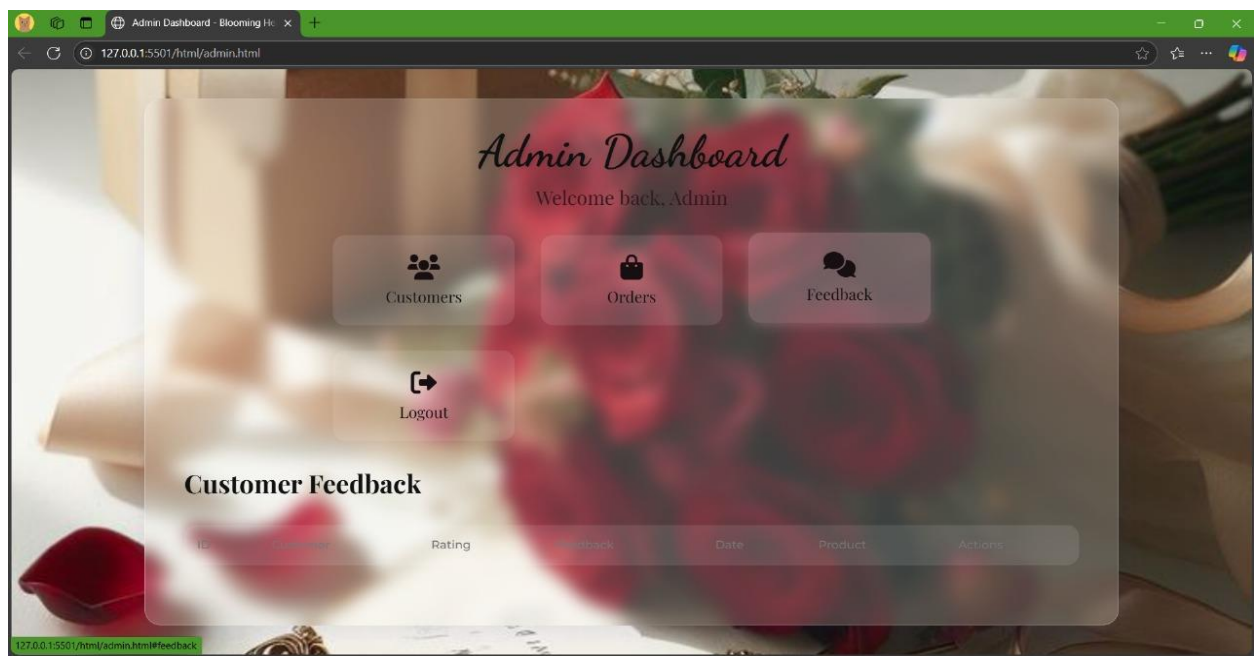
Email Address

Subject

Message



ADMIN DASBOARD:



4. SYSTEM CONFIGURATION

4.1 Hardware requirements

RAM	Recommended 8.00 GB
Hard disk	260 GB
Processor	Intel(R) Core (TM) i3-1005G1
Processing speed	CPU @ 1.20 GHz, 1190 MHz

4.2 Software Requirements:

Front end	HTML ,CSS and JavaScript
Back end	PHP and MySQL
Tools	Visual Studio Code
Operating System	Microsoft Windows 8 or later

5. DETAILS OF SOFTWARE

A development process consists of various phases, each phase ending with a defined output. The phases are performed in an order specified by the process model being followed. The main reason for having a phased process is that it breaks the problem of developing software into successfully performing a set of phases, each handling a different concern of software development.

This ensures that the cost of development is lower than what it would have been if the whole problem were tackled together. A phased development process is central to the software engineering approach for solving the software crisis.

5.1 Overview of Front End

Visual Studio Code (VS Code) is a lightweight and powerful integrated development environment (IDE) that is widely used for front-end development. It supports essential web technologies such as HTML, CSS, and JavaScript and provides a rich set of features to improve productivity. With built-in IntelliSense, VS Code offers smart code completion, helping developers write clean and efficient code quickly. The IDE also includes live preview capabilities, allowing real-time visualization of front-end changes in the browser through extensions like Live Server. Additionally, VS Code supports Git integration for version control, making it easy to manage project files and collaborate with other developers. It's highly customizable, with a vast collection of extensions that enhance functionality, including tools for linting, debugging, and even working with frameworks like React or Angular. This makes Visual Studio Code a versatile and ideal choice for developing the front-end of My Blooming Hearts.

5.2 Overview of Back End

The back-end of My Blooming Hearts is powered by PHP and MySQL. PHP is a widely used open-source scripting language designed for server-side development, enabling dynamic content generation and seamless interaction with databases. MySQL is an open-source relational database management system that provides a robust and efficient platform for storing and managing application data. The combination of PHP and MySQL allows for smooth handling of user data, session management.

5.3 About the Platform

My Blooming Hearts is a web-based platform designed for accessibility across various devices with internet connectivity. The project utilizes HTML, CSS, JavaScript, PHP, and MySQL to create an interactive and dynamic user experience. The platform runs on Microsoft Windows or later versions, with Visual Studio Code as the primary development tool. As a web application, My Blooming Hearts is accessible through any modern web browser, offering seamless interaction and data management through the front-end and back-end technologies. The use of PHP and MySQL ensures reliable server-side processing and database management, making it a robust platform for users to interact with in real-time.

The application is designed to run on Microsoft Windows 8 or later operating systems, ensuring compatibility with the majority of modern computers. The development process leverages Visual Studio Code as the main tool for coding and testing, offering a robust environment for building and deploying the platform. With this combination of front-end and back-end technologies, My Blooming Hearts is positioned to deliver a user-friendly and efficient web experience, with scalability for future updates and features.

The platform is designed to provide an intuitive and responsive user interface, powered by HTML, CSS, and JavaScript for the front-end. These technologies enable the creation of dynamic, interactive web pages that ensure a seamless experience across devices, whether viewed on desktops, tablets, or smartphones. The PHP back-end handles all the server-side logic, ensuring that data is processed efficiently and securely. MySQL serves as the relational database system, providing reliable and fast data storage, retrieval, and management.

6. TESTING

Testing is a vital part of software development, and it is important to start it as early as possible, and to make testing a part of the process of deciding requirements. To get the most useful perspective on your development project, it is worthwhile devoting some thought to the entire lifecycle including how feedback from users will influence the future of the application. The tools and techniques we've discussed in this book should help your team to be more responsive to changes without extra cost, despite the necessarily wide variety of different development processes. Nevertheless, new tools and process improvements should be adopted gradually, assessing the results after each step.

Testing is part of a lifecycle. The software development lifecycle is one in which you hear of a need, you write some code to fulfil it, and then you check to see whether you have pleased the stakeholders—the users, owners, and other people who have an interest in what the software does. Hopefully they like it, but would also like some additions or changes, so you update or augment your code; and so the cycle continues, or every few years.

1.1.4 SOFTWARE DEVELOPMENT LIFE CYCLE

Testing is a proxy for the customer. You could conceivably do your testing by releasing it into the wild and waiting for the complaints and compliments to come back. Some companies have been accused of having such a strategy as their business model even before it became fashionable. But on the whole, the books are better balanced by trying to make sure that the software will satisfy the customer before we hand it over. This platform, My Blooming Hearts, is developed using the Incremental Model and the Waterfall Model.

1.1.5 SOFTWARE TESTING TYPES:

1. FUNCTIONAL TESTING

This type of testing ignores the internal parts and focus on the output is as per requirement or not.

They are:

Black box testing –Internal system design is not considered in this type of testing. Tests are based on system's requirements and functionality.

White box testing – This testing is based on knowledge of the internal logic of an application's code. Also known as Glass box Testing. Internal software and code working should be known for this type of testing. Tests are based on coverage of code statements, branches, paths, conditions.

Unit testing – This involves testing individual software components or modules. It is typically performed by developers, as it requires an understanding of the program's internal design and code. Often, test drivers or harnesses are developed to support this testing.

System testing – Entire system is tested as per the requirements. Black-box type testing that is based on overall requirements specifications, covers all combined parts of a system.

Acceptance testing -. This is conducted to verify that the system meets the customer's specified requirements. It is usually performed by users or clients to decide whether the application should be accepted.

Alpha testing – In house virtual user environment can be created for this type of testing. Testing is done at the end of development. Still minor design changes may be made as a result of such testing.

2. NON-FUNCTIONAL TESTING

Security testing – This type of testing evaluates how well the system is protected against unauthorized access, both internal and external. It ensures that the system and database are secure from potential hacking attempts or vulnerabilities. The goal is to identify any weaknesses that could be exploited and verify that proper security mechanisms are in place.

Usability testing –. This involves checking how user-friendly and intuitive the application is. It tests the overall application flow to ensure that new users can understand and use the platform with ease. It also verifies whether appropriate help documentation or guidance is available if a user encounters any difficulty. Essentially, this testing evaluates system navigation and ease of use.

7. CONCLUSION AND FUTURE ENHANCEMENT

The system has been developed with intuitive and easy navigation, specifically tailored for children as the primary users. It incorporates vibrant backgrounds, engaging sounds, and colorful visuals to capture and retain the attention of young learners. By offering a fun and interactive learning environment, the system ensures that education is both effective and enjoyable. Pronunciations for each word are included to support better understanding and retention, making the platform a one-stop solution for learning. Additionally, the system allows users to resume their learning from where they last left off, making it convenient to track progress.

The system can further be enhanced as follows:

- Centralized Server Access: Enabling users to access their lessons from multiple devices using the same account.
- Learning History Tracker: Keeping a record of the last practice date to help resume learning efficiently after breaks..
- Study Time Reminders: Sending scheduled notifications or alerts based on the user's preferred study time

8. BIBLIOGRAPHY

Book References:

1. Nixon, R. (2022). Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5 (6th ed.). O'Reilly Media, Inc.
2. Duckett, J. (2011). HTML and CSS: Design and Build Websites. Wiley.

Web References:

1. <https://www.w3schools.com>
2. <https://developer.mozilla.org/en-US/>
3. <https://www.geeksforgeeks.org/php/>
4. <https://www.mysql.com>

9. APPENDICES A – Table Structure

Table Name: User

Description: Contains user details along with login details



#	Name	Type	Collation	Attributes	Null	Default
<input type="checkbox"/> 1	id 	int(11)			No	None
<input type="checkbox"/> 2	full_name	varchar(100)	utf8mb4_general_ci		No	None
<input type="checkbox"/> 3	email 	varchar(100)	utf8mb4_general_ci		No	None
<input type="checkbox"/> 4	password	varchar(255)	utf8mb4_general_ci		No	None
<input type="checkbox"/> 5	role	enum('user', 'admin')	utf8mb4_general_ci		Yes	user
<input type="checkbox"/> 6	created_at	timestamp			No	current_timestamp()

Table Name: Products

Description: Contains all the products details


#	Name	Type	Collation
<input type="checkbox"/> 1	id 	int(11)	
<input type="checkbox"/> 2	name	varchar(100)	utf8mb4_general_ci
<input type="checkbox"/> 3	description	text	utf8mb4_general_ci
<input type="checkbox"/> 4	price	decimal(10,2)	
<input type="checkbox"/> 5	category	enum('bouquets', 'gift-cards')	utf8mb4_general_ci
<input type="checkbox"/> 6	image_url	varchar(255)	utf8mb4_general_ci
<input type="checkbox"/> 7	stock	int(11)	
<input type="checkbox"/> 8	created_at	timestamp	

Table Name: Cart

Description: Contains the Cart details




#	Name	Type	Collation	Attributes	Null	Default
<input type="checkbox"/> 1	cart_id 	int(11)			No	None
<input type="checkbox"/> 2	user_id 	int(11)			No	None
<input type="checkbox"/> 3	product_id 	int(11)			No	None
<input type="checkbox"/> 4	quantity	int(11)			Yes	1
<input type="checkbox"/> 5	added_at	timestamp			No	current_timestamp()

Table Name: Order**Description: Contains all the order details**



#	Name	Type	Collation	Attributes	Null	Default
<input type="checkbox"/> 1	order_id 	int(11)			No	None
<input type="checkbox"/> 2	user_id 	int(11)			No	None
<input type="checkbox"/> 3	order_date	timestamp			No	current_timestamp()
<input type="checkbox"/> 4	total_amount	decimal(10,2)			No	None
<input type="checkbox"/> 5	shipping_address	text	utf8mb4_general_ci		Yes	NULL
<input type="checkbox"/> 6	billing_address	text	utf8mb4_general_ci		Yes	NULL
<input type="checkbox"/> 7	payment_method	varchar(50)	utf8mb4_general_ci		Yes	NULL
<input type="checkbox"/> 8	payment_status	varchar(20)	utf8mb4_general_ci		Yes	NULL
<input type="checkbox"/> 9	shipping_method	varchar(50)	utf8mb4_general_ci		Yes	NULL
<input type="checkbox"/> 10	shipping_status	varchar(20)	utf8mb4_general_ci		Yes	NULL
<input type="checkbox"/> 11	order_status	varchar(20)	utf8mb4_general_ci		Yes	Pending
<input type="checkbox"/> 12	transaction_id	varchar(100)	utf8mb4_general_ci		Yes	NULL
<input type="checkbox"/> 13	created_at	timestamp			No	current_timestamp()

Table Name: Order Details**Description: Contains all the order details and information about status**




#	Name	Type	Collation	Attributes	Null	Default
<input type="checkbox"/> 1	order_item_id 	int(11)			No	None
<input type="checkbox"/> 2	order_id 	int(11)			No	None
<input type="checkbox"/> 3	product_id 	int(11)			No	None
<input type="checkbox"/> 4	quantity	int(11)			No	None
<input type="checkbox"/> 5	unit_price	decimal(10,2)			No	None
<input type="checkbox"/> 6	subtotal	decimal(10,2)			No	None

Table Name: Wishlist**Description: Contains all the wishlist details**





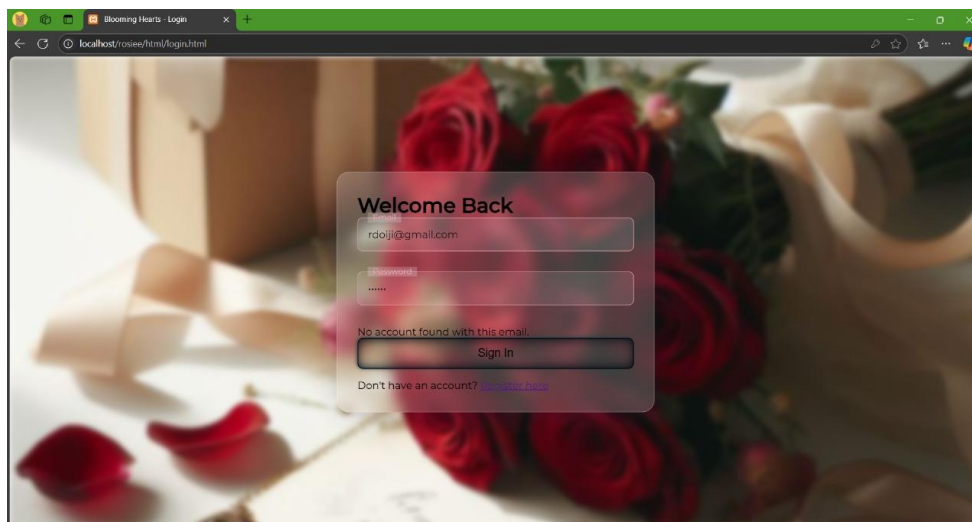
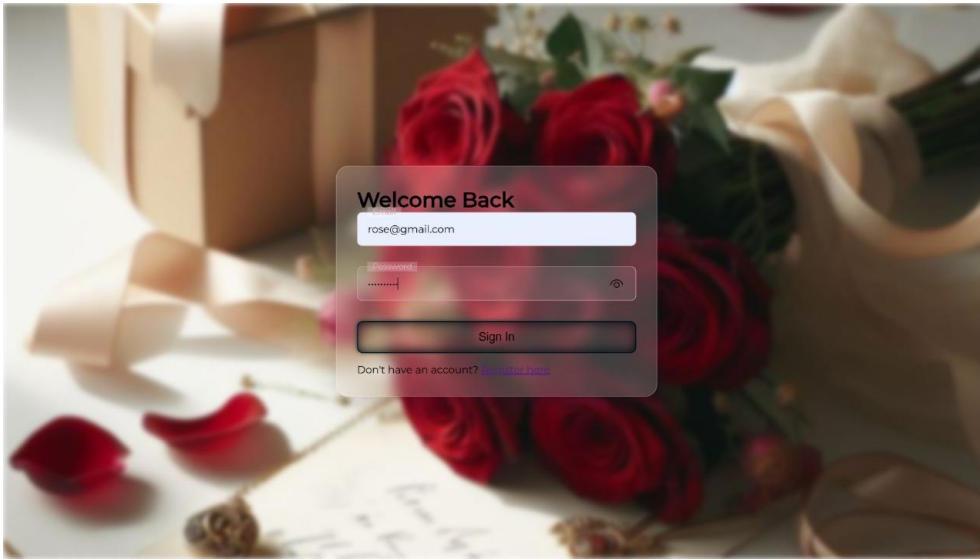
#	Name	Type	Collation	Attributes	Null	Default
<input type="checkbox"/> 1	wishlist_id 	int(11)			No	None
<input type="checkbox"/> 2	user_id 	int(11)			No	None
<input type="checkbox"/> 3	product_id 	int(11)			No	None
<input type="checkbox"/> 4	added_at	timestamp			No	current_timestamp()

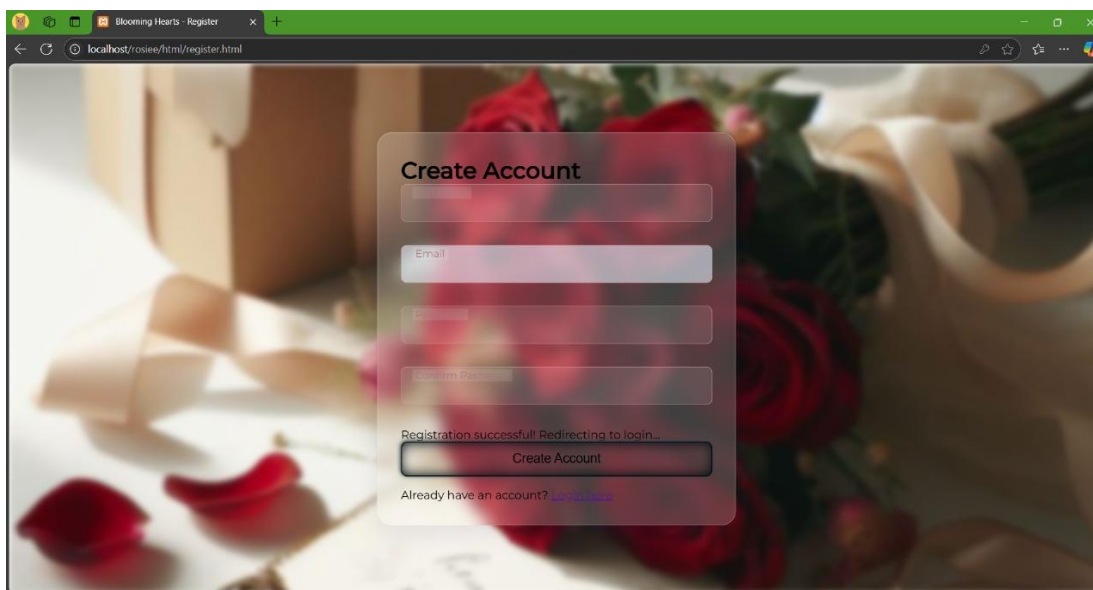
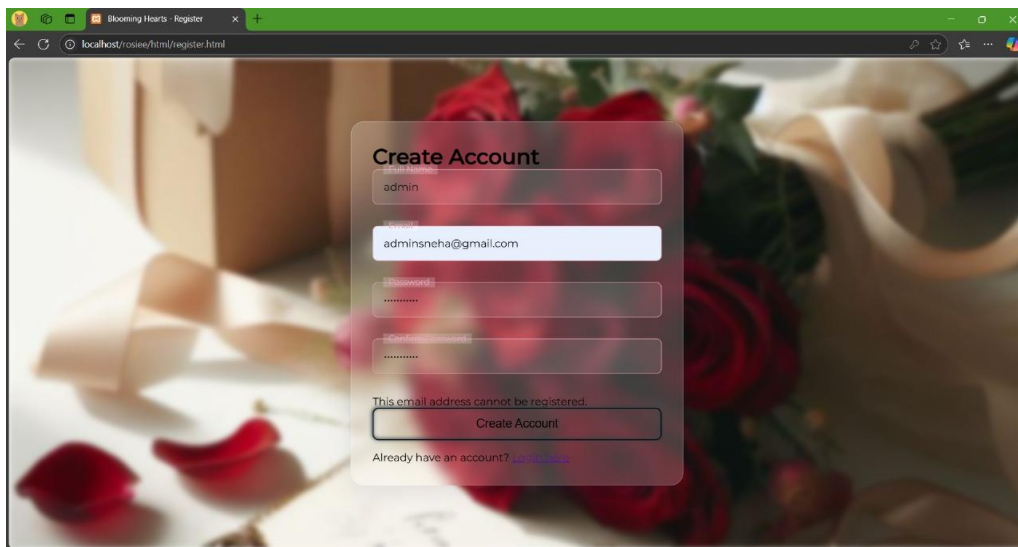
Table Name: Contact Us**Description: Contains all the wishlist details**

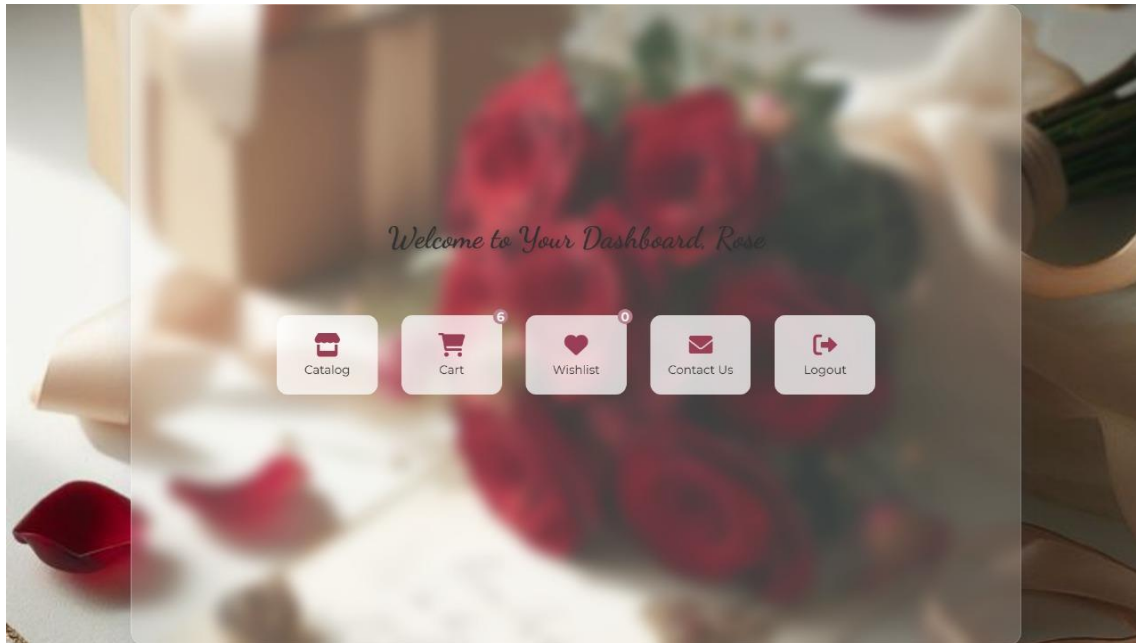
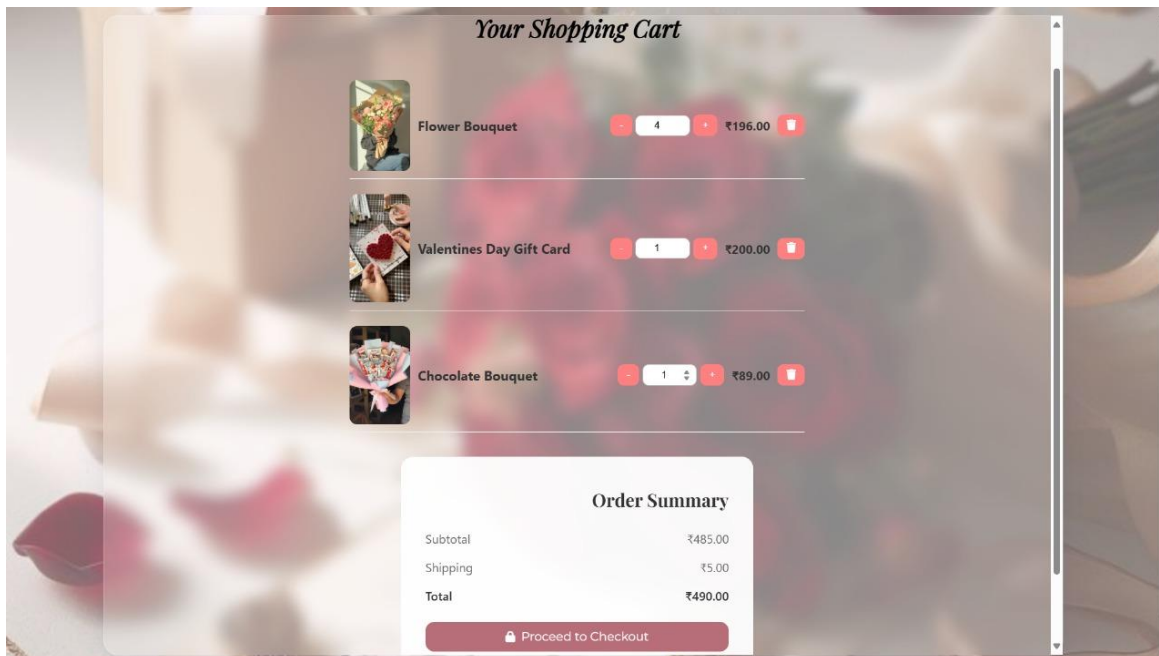
	#	Name	Type	Collation	Attributes	Null	Default
<input type="checkbox"/>	1	id 	int(11)			No	None
<input type="checkbox"/>	2	full_name	varchar(255)	utf8mb4_general_ci		No	None
<input type="checkbox"/>	3	email	varchar(255)	utf8mb4_general_ci		No	None
<input type="checkbox"/>	4	subject	varchar(255)	utf8mb4_general_ci		No	None
<input type="checkbox"/>	5	message	text	utf8mb4_general_ci		No	None
<input type="checkbox"/>	6	submission_date	timestamp			No	current_timestamp()

10.APPENDICES B - SCREENSHOTS

Login:



Registration:

Dashboard:**Cart:**

Payment :

Secure Checkout

Order Summary

Flower Bouquet x4	₹196.00
Valentines Day Gift Card x1	₹200.00
Chocolate Bouquet x1	₹89.00
Subtotal	₹485.00
Shipping	₹5.00
Total	₹490.00

Payment Details

Billing Information

Full Name
Rose

Email
rose@gmail.com

Phone
+916161616161

Address
123 abc road

Card Information

Card Number

Flower Bouquet x4 ₹196.00
Valentines Day Gift Card x1 ₹200.00
Chocolate Bouquet x1 ₹89.00
Subtotal ₹485.00
Shipping ₹5.00
Total ₹490.00

Rose
rose@gmail.com
+916161616161
123 abc road

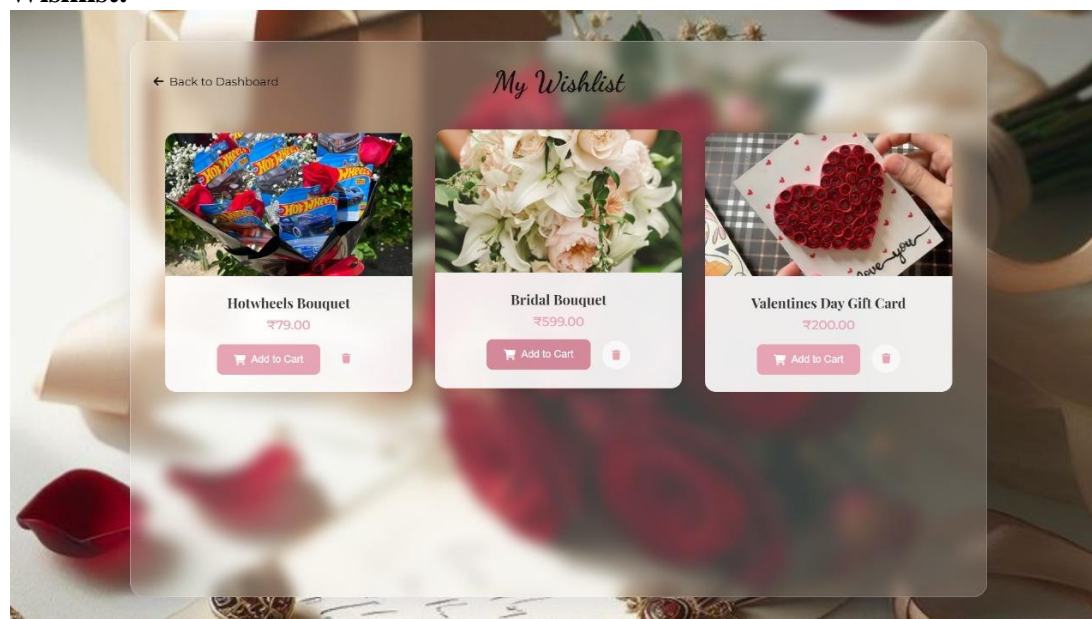
Payment Successful!

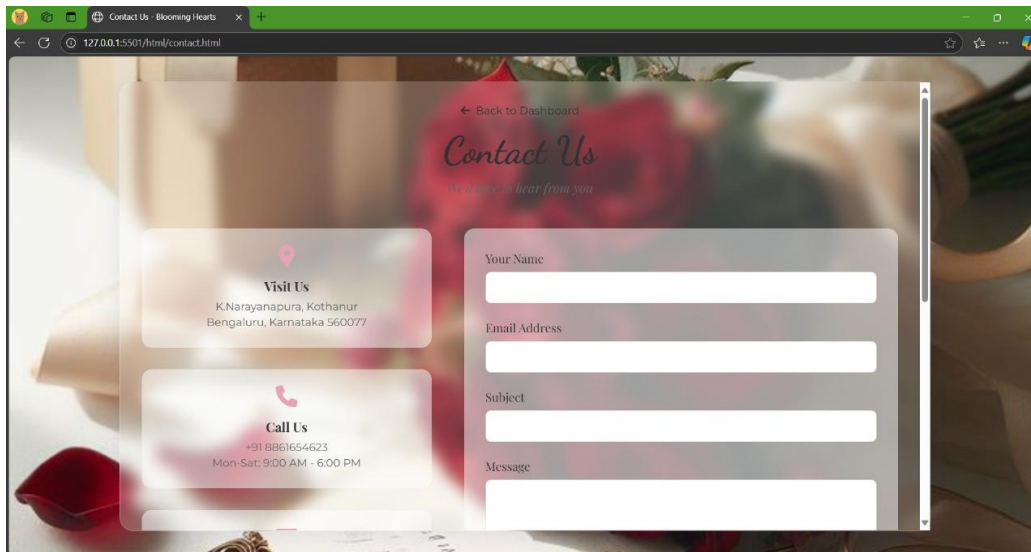
Thank you for your order. Your payment has been processed successfully.

Order Number: #BH78007
Amount Paid: ₹490.00

Done

Pay Securely ₹ 490.00

Wishlist:

Contact US:

← Back to Dashboard

Contact Us

We'd love to hear from you

Visit Us
K.Narayanapura, Kothanur
Bengaluru, Karnataka 560077

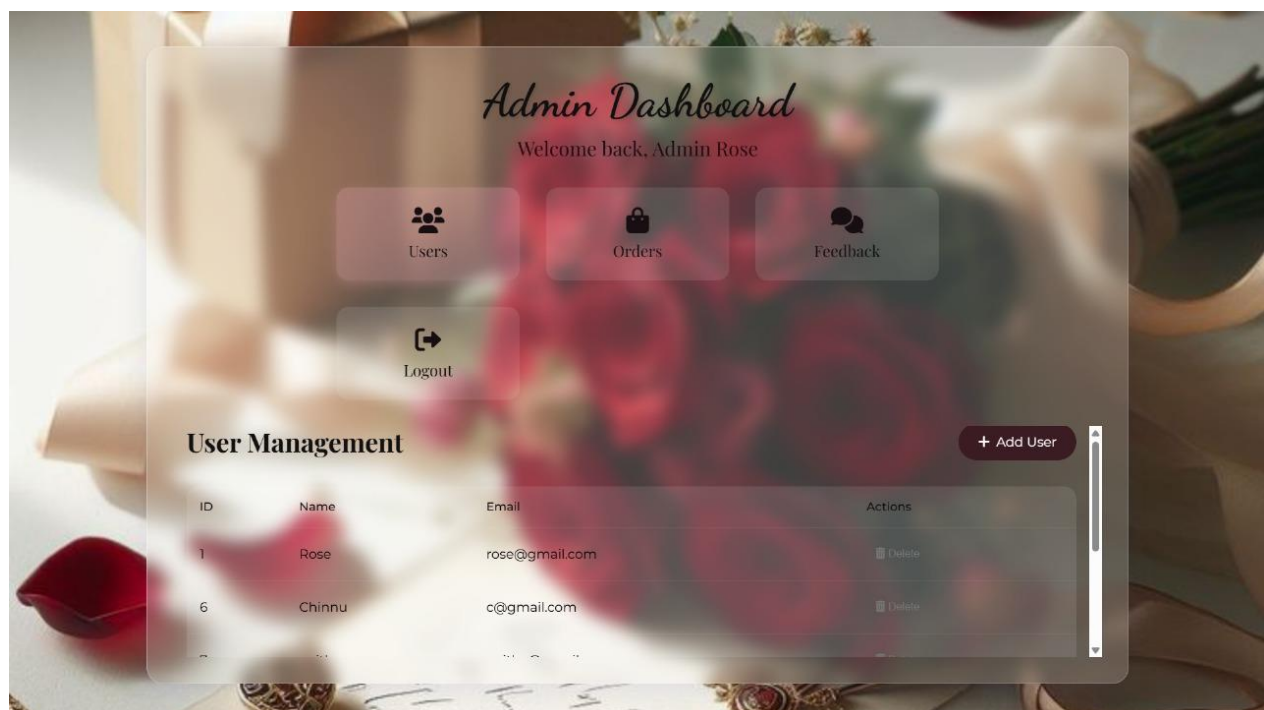
Call Us
+91 8861654623
Mon-Sat: 9:00 AM - 6:00 PM

Your Name

Email Address

Subject

Message

Admin :

Admin Dashboard

Welcome back, Admin Rose

Users

Orders

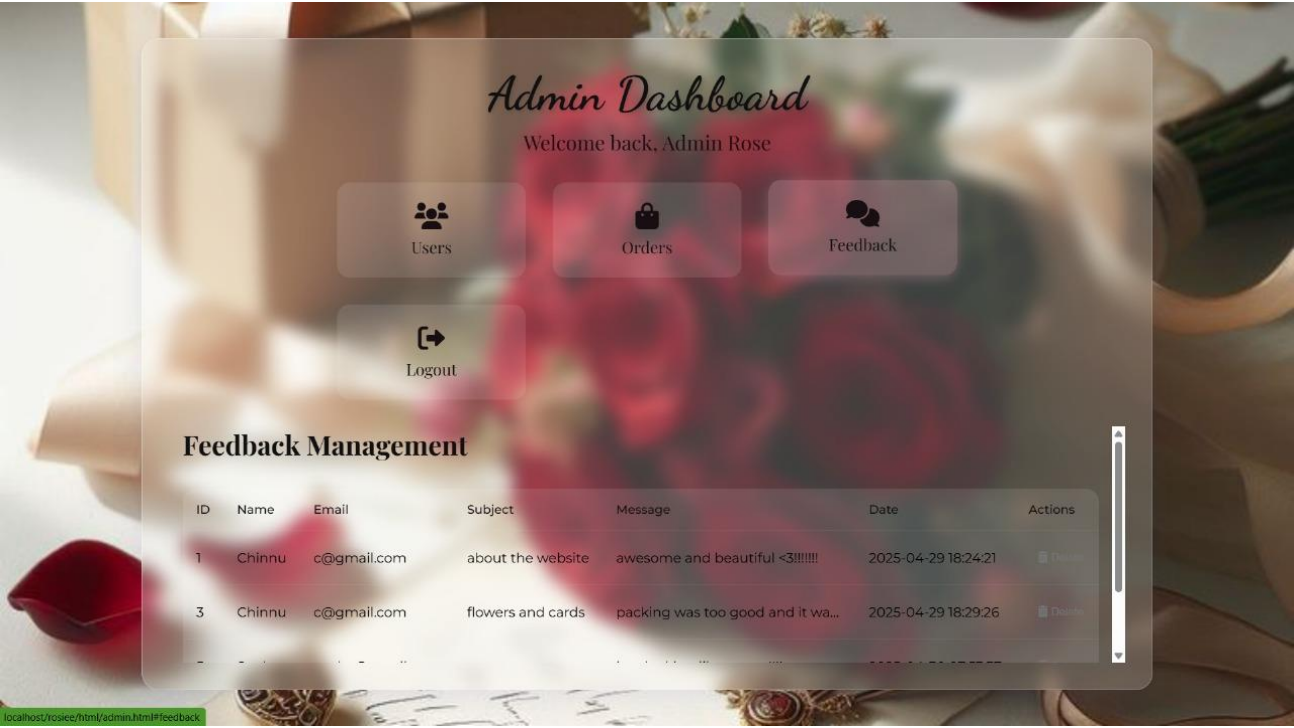
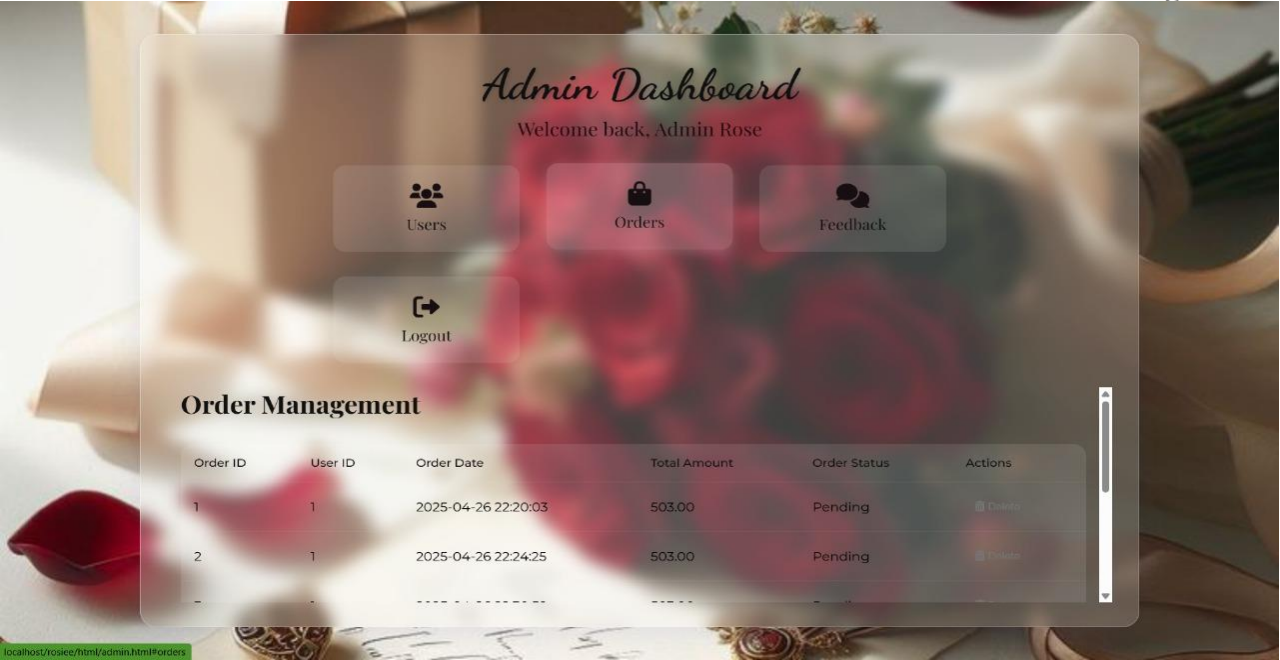
Feedback

Logout

User Management

[+ Add User](#)

ID	Name	Email	Actions
1	Rose	rose@gmail.com	Delete
6	Chinnu	c@gmail.com	Delete



11. APPENDICES C – TEST CASES

APPENDICES – SAMPLE REPORT OF TEST CASES FOR BLOOMING HEARTS

Test Case Id	Test Description	Steps to Execute	Test Data	Expected Result	Status	Status
BH_TC_01	Login with correct credentials	Enter valid.ernal and password. cik	Email: user@mail Password: 129	Relirected to User Dashboard	Pass	Pass
BH_TC_02	Login with incorret password	Enter wrong pass+ password, cik logn	Email: vw unknowh@gm3c)	Show Email not found	Pass	Pass
BH_TC_04	Register with <i>valid inputs</i>	Fill ail registrar tion fields	User/gmail All regist-al.com	Registration successful	Pass	Pass
BH_TC_06	Register with missing inputs	Leave passwort blank, submit	Email. user genail	Show "Please fill all fids	Fail	Fail
BH_TC_07	Add bouquet to cart	Click "Add to cart"	Selected item	Show "Please fill felds	Pass	Pass
BH_TC_08	Add gift card to cart	Selected passviort	Selected item	Item item added to cart	Fail	Fail
BH_TC_09	View cart after adding item	Navigate to cart pap	Navigaten to cart	View cart items with details	Fail	Fail
BH_TC_10	Update quantity in cart	Change quantity of item in cart	Sto 2	Quantity updated price recalculated	Pass	Pass
BH_TC_11	Remove item from cart	Click remove iconn	Item removed from cart	Item removed from cart	Pass	Pass
BH_TC_12	Checkout with filled cart	Click "Checkout" button	Redirect to payment page	Redirects to payment page	Pass	Pass
BH_TC_13	Checkout with empty cart	Click, "Checkout" with no items	Cart is empty	Checkocts to payment page	Fail	Pail

12. APPENDICES D - SOURCE CODE

2 Login:

```
<?php

// Prevent any output before JSON response

ob_start();

// Error handling

error_reporting(E_ALL);

ini_set('display_errors', 0); // Disable error display

ini_set('log_errors', 1); // Enable error logging

ini_set('error_log', 'php_errors.log'); // Set error log file


header('Content-Type: application/json');

session_start();

try {

    // Database connection

    $host = 'localhost';

    $port = 3307; // Using the correct port from db.php

    $username = 'root';

    $password = "";

    $database = 'rose';
```

```
$conn = new mysqli($host, $username, $password, $database, $port);

if ($conn->connect_error) {

    throw new Exception("Connection failed: " . $conn->connect_error);

}

$conn->set_charset("utf8mb4");

// Get and validate input

$email = $_POST['email'] ?? "";

$password = $_POST['password'] ?? "";

if (empty($email) || empty($password)) {

    throw new Exception('Email and password are required.');
```

```
}

// Prepare statement

$stmt = $conn->prepare("SELECT * FROM users WHERE email = ?");

if (!$stmt) {

    throw new Exception('Database error: ' . $conn->error);

}

// Bind parameters and execute

$stmt->bind_param("s", $email);

if (!$stmt->execute()) {

    throw new Exception('Database error: ' . $stmt->error);

}

$result = $stmt->get_result();

if ($result && $result->num_rows > 0) {
```



```
$user = $result->fetch_assoc();

if (password_verify($password, $user['password'])) {

    // Set session variables

    $_SESSION['email'] = $user['email'];

    $_SESSION['full_name'] = $user['full_name'];

    $_SESSION['user_id'] = $user['id'];

    $_SESSION['role'] = $user['role'];

    $redirect = ($user['role'] === 'admin') ? '../html/admin.html' : '../html/dashboard.html';

    $response = [

        'success' => true,

        'message' => 'Login successful!',

        'redirect' => $redirect,

        'full_name' => $user['full_name'],

        'email' => $user['email'],

        'role' => $user['role']

    ];

} else {

    throw new Exception('Incorrect password.');
```

```
}

} else {

    throw new Exception('No account found with this email.');
```

```
}
```

```
$stmt->close();
```

```
$conn->close();

} catch (Exception $e) {

    // Clear any output buffers

    while (ob_get_level()) {

        ob_end_clean();

    }

    $response = [

        'success' => false,

        'message' => $e->getMessage()

    ];

}

// Clear any remaining output buffer

while (ob_get_level()) {

    ob_end_clean();

}

// Ensure we always return a JSON response

echo json_encode($response ?? ['success' => false, 'message' => 'An unknown error occurred'])
```

3 Registration:

```
<?php

error_reporting(E_ALL);

ini_set('display_errors', 1);

KRISTU JAYANTI COLLEGE (Autonomous)
```

```
header('Content-Type: application/json');

session_start();

include 'db.php';

$full_name = trim($_POST['full_name'] ?? '');

$email = trim($_POST['email'] ?? '');

$password = $_POST['password'] ?? '';

$confirm_password = $_POST['confirm_password'] ?? '';

$adminEmails = [

    'adminrose@gmail.com',

    'adminchinnu@gmail.com'

];

if (in_array(strtolower($email), array_map('strtolower', $adminEmails)) ||

    strpos($email, 'admin') !== false) {

    echo json_encode(['success' => false, 'message' => 'This email address cannot be registered.']);

    exit;

}

if (empty($full_name) || empty($email) || empty($password) || empty($confirm_password)) {

    echo json_encode(['success' => false, 'message' => 'All fields are required.']);

    exit;

}

if ($password !== $confirm_password) {

    echo json_encode(['success' => false, 'message' => 'Passwords do not match.']);

    exit;
```

```
}

if (!preg_match('/^[a-zA-Z0-9._%+-]+@gmail\.com$/', $email)) {

    echo json_encode(['success' => false, 'message' => 'Please enter a valid Gmail address (e.g.
yourname@gmail.com)']);

    exit;

}

$stmt = $conn->prepare("SELECT id FROM users WHERE email = ?");

$stmt->bind_param("s", $email);

$stmt->execute();

$result = $stmt->get_result();

if ($result->num_rows > 0) {

    echo json_encode(['success' => false, 'message' => 'Email already registered.']);

    exit;

}

$role = 'user';

$hashed_password = password_hash($password, PASSWORD_DEFAULT);

$stmt = $conn->prepare("INSERT INTO users (full_name, email, password, role) VALUES (?, ?, ?,
?)");

$stmt->bind_param("ssss", $full_name, $email, $hashed_password, $role);

if ($stmt->execute()) {

    echo json_encode([

        'success' => true,

        'message' => 'Registration successful!',
```

```
'full_name' => $full_name
```

```
]);
```

```
} else {
```

```
    echo json_encode(['success' => false, 'message' => 'Registration failed. Please try again.']);
```

```
}
```

```
$stmt->close();
```

```
$conn->close();
```

```
?>
```

4 Add_to_cart:

5

```
<?php
```

```
session_start();
```

```
include('db.php');
```

```
if (!isset($_SESSION['user_id'])) {
```

```
    echo json_encode(['success' => false, 'message' => 'Please log in first.']);
```

```
    exit;
```

```
}
```

```
$user_id = $_SESSION['user_id'];
```

```
$product_id = $_POST['product_id'];
```

```
$quantity = $_POST['quantity'];
```

```
if ($quantity <= 0) {
```

```
    echo json_encode(['success' => false, 'message' => 'Invalid quantity']);
```

```
    exit;
```

```
}
```

```
$query = "SELECT * FROM cart WHERE user_id = ? AND product_id = ?";
```

```
$stmt = $conn->prepare($query);
```

```
$stmt->bind_param("ii", $user_id, $product_id);
```

```
$stmt->execute();
```

```
$result = $stmt->get_result();
```

```
if ($result->num_rows > 0) {
```

```
    $row = $result->fetch_assoc();
```

```
    $new_quantity = $row['quantity'] + $quantity;
```

```
    $update_query = "UPDATE cart SET quantity = ? WHERE cart_id = ?";
```

```
    $update_stmt = $conn->prepare($update_query);
```

```
    $update_stmt->bind_param("ii", $new_quantity, $row['cart_id']);
```

```
    $update_stmt->execute();
```

```
    echo json_encode(['success' => true, 'message' => 'Cart updated successfully.']);
```

```
} else {
```

```
    $added_at = date('Y-m-d H:i:s');
```

```
    $insert_query = "INSERT INTO cart (user_id, product_id, quantity, added_at) VALUES (?, ?, ?, ?)";
```

```
    $insert_stmt = $conn->prepare($insert_query);
```

```
    $insert_stmt->bind_param("iiis", $user_id, $product_id, $quantity, $added_at);
```

```
    $insert_stmt->execute();
```

```
    echo json_encode(['success' => true, 'message' => 'Product added to the cart.']);
```

```
}
```

```
$stmt->close();
```

```
$conn->close();
```

```
?>
```

Update_to_cart:

```
<?php
```

```
session_start();
```

```
include('db.php');
```

```
header('Content-Type: application/json');
```

```
$data = json_decode(file_get_contents('php://input'), true);
```

```
if (!isset($_SESSION['user_id'])) {
```

```
    echo json_encode(['success' => false, 'message' => 'User not logged in']);
```

```
    exit;
```

```
}
```

```
$user_id = $_SESSION['user_id'];
```

```
$product_id = $data['product_id'] ?? null;
```

```
$quantity_change = $data['quantity_change'] ?? null;
```

```
$new_quantity = $data['new_quantity'] ?? null;
```

```
if (!$product_id) {
```

```
    echo json_encode(['success' => false, 'message' => 'Product ID missing']);
```

```
    exit;
```

```
}
```

```
try {
```

```
if ($quantity_change !== null) {
```

```
    $check = $conn->prepare("SELECT quantity FROM cart WHERE user_id = ? AND product_id = ?");
```

```
    $check->bind_param("ii", $user_id, $product_id);
```

```
    $check->execute();
```

```
    $result = $check->get_result();
```

```
    $row = $result->fetch_assoc();
```

```
    $current_quantity = $row['quantity'] ?? 0;
```

```
    $check->close();
```

```
    $updated_quantity = $current_quantity + $quantity_change;
```

```
    if ($updated_quantity < 1) {
```

```
        echo json_encode(['success' => false, 'message' => 'Quantity cannot be less than 1']);
```

```
        exit;
```

```
    }
```

```
    $query = "UPDATE cart SET quantity = ? WHERE user_id = ? AND product_id = ?";
```

```
    $stmt = $conn->prepare($query);
```

```
    $stmt->bind_param("iii", $updated_quantity, $user_id, $product_id);
```

```
} elseif ($new_quantity !== null) {
```

```
    if ($new_quantity < 1) {
```

```
        echo json_encode(['success' => false, 'message' => 'Quantity must be at least 1']);
```

```
        exit;
```

```
    }
```

```
    $query = "UPDATE cart SET quantity = ? WHERE user_id = ? AND product_id = ?";
```



```
$stmt = $conn->prepare($query);

$stmt->bind_param("iii", $new_quantity, $user_id, $product_id);

} else {

    throw new Exception("Invalid quantity update request.");

}

if (!$stmt) {

    throw new Exception("Failed to prepare statement.");

}

$stmt->execute();

echo json_encode(['success' => true, 'message' => 'Cart updated']);

} catch (Exception $e) {

    echo json_encode(['success' => false, 'message' => $e->getMessage()]);

} finally {

    if (isset($stmt)) $stmt->close();

    $conn->close();

}

?>
```

6 Add_to_wishlist:

7

```
<?php

session_start();

header('Content-Type: application/json');


// Check if user is logged in

if (!isset($_SESSION['user_id'])) {

    echo json_encode(['success' => false, 'message' => 'User not logged in']);

    exit;

}


require_once 'db.php';


// Get product ID from POST request

$product_id = isset($_POST['product_id']) ? (int)$_POST['product_id'] : 0;


if ($product_id <= 0) {

    echo json_encode(['success' => false, 'message' => 'Invalid product ID']);

    exit;

}
```

```
try {  
  
    // Check if product exists  
  
    $stmt = $conn->prepare("SELECT id FROM products WHERE id = ?");  
  
    $stmt->bind_param("i", $product_id);  
  
    $stmt->execute();  
  
    $result = $stmt->get_result();  
  
  
    if ($result->num_rows === 0) {  
  
        echo json_encode(['success' => false, 'message' => 'Product not found']);  
  
        exit;  
    }  
  
  
    // Check if item is already in wishlist  
  
    $stmt = $conn->prepare("SELECT id FROM wishlist WHERE user_id = ? AND product_id = ?");  
  
    $stmt->bind_param("ii", $_SESSION['user_id'], $product_id);  
  
    $stmt->execute();  
  
    $result = $stmt->get_result();  
  
  
    if ($result->num_rows > 0) {  
  
        echo json_encode(['success' => false, 'message' => 'Item already in wishlist']);  
  
        exit;  
    }  
}
```

```
// Add item to wishlist

$stmt = $conn->prepare("INSERT INTO wishlist (user_id, product_id) VALUES (?, ?)");

$stmt->bind_param("ii", $_SESSION['user_id'], $product_id);

if ($stmt->execute()) {

    echo json_encode(['success' => true, 'message' => 'Item added to wishlist']);

} else {

    echo json_encode(['success' => false, 'message' => 'Error adding item to wishlist']);

}

} catch (Exception $e) {

    echo json_encode([

        'success' => false,

        'message' => 'Error adding to wishlist: ' . $e->getMessage()

    ]);

}

$conn->close()
```

DB.php:

```
<?php

error_reporting(E_ALL);

ini_set('display_errors', 0);

ini_set('log_errors', 1);

ini_set('error_log', 'php_errors.log');

function getConnection() {

    try {

        $host = 'localhost';

        $port = 3307;

        $username = 'root';

        $password = "";

        $database = 'rose';

        $conn = new mysqli($host, $username, $password, $database, $port);

        if ($conn->connect_error) {

            throw new Exception('Database connection failed: ' . $conn->connect_error);

        }

        $conn->set_charset("utf8mb4");

        return $conn;

    } catch (Exception $e) {

        throw new Exception('Database connection error: ' . $e->getMessage());

    }

}
```

```
try {  
    $conn = getConnection();  
} catch (Exception $e) {  
    if (count(debug_backtrace()) === 0) {  
        header('Content-Type: application/json');  
        echo json_encode([  
            'success' => false,  
            'message' => $e->getMessage()  
        ]);  
        exit;  
    }  
    throw $e;  
}
```

?>

Get_Cart:

```
<?php  
session_start();  
require_once 'db.php';  
  
if (!isset($_SESSION['user_id'])) {  
    echo json_encode(['success' => false, 'message' => 'User not logged in']);  
}
```

```
        exit;
    }

$user_id = $_SESSION['user_id'];

$query = "
    SELECT
        c.product_id,
        c.quantity,
        p.name,
        p.price,
        p.image_url
    FROM cart c
    JOIN products p ON c.product_id = p.product_id
    WHERE c.user_id = ?
";

$stmt = $conn->prepare($query);
$stmt->bind_param("i", $user_id);
$stmt->execute();
$result = $stmt->get_result();
```

```
$cart_items = [];  
  
while ($row = $result->fetch_assoc()) {  
    $cart_items[] = [  
        'id' => $row['product_id'],  
        'name' => $row['name'],  
        'price' => $row['price'],  
        'image' => $row['image_url'],  
        'quantity' => $row['quantity']  
    ];  
}  
  
echo json_encode([  
    'success' => true,  
    'items' => $cart_items  
]);  
?>
```