



## Module Code & Module Title CS5054NI Advanced Programming & Technologies

**Assessment Type**  
**50% Group Coursework**

**Semester**  
**2024 Spring**

### Group Members

London Met ID	Student Name
22085542	KSHITIZ GIRI
22085588	RISHI NEUPANE
22085608	SANGITA BELBASE
22085505	ASHIM POUDEL
22068169	SHERAYS BHANDARI

**Project Title: CameraHub**

**Assignment Due Date:** Tuesday, May 14, 2024

**Assignment Submission Date:** Click or tap to enter a date.

**Submitted to:** Mr. Prithivi Maharjan

**Word Count:** Enter the total word count

*I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.*

## Contents

1. INTRODUCTION.....	4
----------------------	---

1.2 Aim .....	4
1.3 Objective.....	4
2. UI Design .....	5
2.2 WireFrame .....	5
2.3 Actual Design .....	10
2.3.1 User Screen .....	11
2.3.2 Admin Screen .....	15
3. Class Diagram.....	17
4. Method Description .....	18
4.1 Authentication filter.java .....	18
4.2 EncDyc.java.....	20
4.3 editProfile.java .....	20
4.4 productQuery.java.....	21
4.5 Query.java.....	21
5. Test Cases .....	23
5.1 Test Case 1 .....	23
5.2 Test Case 2.....	25
5.3 Test Case 3.....	28
5.4 Test Case 4.....	28
5.5 Test Case 5.....	30
5.6 Test Case 6.....	32
5.7 Test Case 7 .....	33
5.8 Test Case 8.....	35
5.9 Test Case 9.....	36
5.10 Test case 10.....	38
5.11 Test case 11 .....	40
6. Tools and Libraries Used .....	42
6.1 Tools .....	42
6.2 Libraries .....	43
7. Development Process .....	44
8. Critical Analysis .....	46
9. Conclusion: .....	48
10. References .....	49

## Table of Figure

Figure 1 Create Account Page.....	8
Figure 2 Login page of the system. ....	9
Figure 3 Navigation Bar of the system. ....	9
Figure 4 Add product .....	10
Figure 5 Product in Cart/Bag for checkout. ....	10
Figure 6 User Home page. ....	11
Figure 7 About us page. ....	12
Figure 8 Actual design of home page. ....	14
Figure 9 Actual design of Cart Page when no items are added. ....	14
Figure 10 Actual design of Cart Page when item is Added. ....	15
Figure 11: About us in actual design. ....	15
Figure 12 Before editing user's details. ....	16
Figure 13 Editing the details. ....	17
Figure 14 Actual design of purchase History page .....	18
Figure 15: Login Page. ....	18
Figure 16 Actual design of Admin Dashboard 1.....	19
Figure 17 Actual design of admin dashboard.....	19
Figure 18 Actual design of Editing Product Page. ....	20
Figure 19 Actual Design of Manage Order Page. ....	20
Figure 20: Redirecting .....	21
Figure 21 Class Diagram. ....	22
Figure 22 Valid Login 1 .....	28
Figure 23: Valid Login 2. ....	29
Figure 24: Evidence for Valid Login. ....	30
Figure 25 Valid Registration 1 .....	31
Figure 26 Valid registration 2 .....	32
Figure 27 Encrypted Password 1. ....	33
Figure 28 Logging in before adding products.....	35
Figure 29 Add to cart feature .....	36
Figure 30: Evidence for cart. ....	37
Figure 31 Add to cart feature .....	38
Figure 32 Editing user Profile .....	39
Figure 33 Editing user Profile2 .....	40
Figure 34: Evidence for User Profile. ....	41
Figure 35: Check Purchased History. ....	42
Figure 36 Add Product .....	43
Figure 37 Product display .....	44
Figure 38 Product Description .....	45
Figure 39 Product Added. ....	46
Figure 40 Delete Product .....	47
Figure 41 To Delete Product .....	48
Figure 42 Eclipse (Linux Adictos, 2024) .....	49
Figure 43 Xampp (Javatpoint).....	50
Figure 44 Database .....	56

## Table of Table

Table 1 Method Description of Authentication Server. ....	24
Table 2 Method Description of EncDyc. ....	25
Table 3 Method Description of editProfile. ....	25
Table 4 Method Description of ProductQuery. ....	26
Table 5 Method Description of Query. ....	27
Table 6 Valid Login ....	28
Table 7 Valid Registration ....	31
Table 8 Encrypted Password ....	33
Table 9 Logging in before adding products ....	34
Table 10 Add to cart feature ....	35
Table 11:Remove product from cart. ....	38
Table 12 Editing user Profile ....	39
Table 13 Checking User Purchased History ....	42
Table 14 Adding Product ....	43
Table 15 Editing Product ....	45
Table 16 Deleting Product ....	47
Table 17 Libraries and Function ....	51
Table 18 Roles of Member ....	54

## 1. INTRODUCTION

This group project focuses on creating an e-commerce website about cameras and equipment. During the entire development process, this project will practically implement the Model-View-Controller (MVC) design. This coursework demonstrates the thorough implementation of essential features for creating a visually appealing, reliable, and fully usable e-commerce website, such as login, register, admin panel, password encryption, html, CSS, etc.

### 1.2 Aim

- The primary goal of this coursework is to create an e-commerce platform that is completely operational, effective, and easy to use so that users may browse and buy camera and equipment items.

### 1.3 Objective

- We are developing a login system that will allow administrators and users to safely access their accounts using encrypted passwords.
- Our goal is to provide an admin panel that is simple to use so that administrators may add, update, or remove products with ease as well as view product lists.

- Our goal is to provide an intuitive platform where users can quickly search for cameras and camera equipment products by category, price, brand, and other parameters. They can also easily update their passwords and customize their profiles.
- Users will find it easy to add products to their cart and examine their list of ordered or purchased products thanks to the helpful features.
- We're putting in place a logout system to give users additional protection and peace of mind while protecting their private and personal data.

## **2. UI Design**

UI design is the term used to describe a digital product's interface design. It's the process of creating user interfaces with a focus on look, design, and interactivity for websites and applications. The visual components and their interactive features that facilitate user engagement, as well as the way displays switch between them, are created by a user interface (UI) designer. (Alana, 2024). The purpose of the user interface is to design a product's entire feel, concept, behavior, usability, and visual look.

### **2.2 WireFrame**

A wireframe is a graphic representation of a user interface design, frequently taken as a static image of a web page or app screen. A typical wireframe is quick and straightforward to put together, consisting of simple lines, boxes, and dummy text in black and white. Wireframes can be used throughout development to illustrate and test various layout ideas because of their simplicity. (White, 2024)

[Home](#)[About us](#)[Cart](#)[Profile](#)[Login](#)

## CREATE AN ACCOUNT

First Name

Last Name

Contact Number

Email Address

Create Password

Upload Profile Picture

No file chosen

**REGISTER NOW**

Already have an account? [Login](#)

1 Create Account Page

Figure

[Home](#)[About us](#)[Cart](#)[profile](#)[Login](#)

## LOGIN

Email Address

Password

[LOGIN](#)

Dont have an account? [Create Account](#)

Figure 2 Login page of the system.

[Home](#)[About us](#)[Cart](#)[Admin](#)[Logout](#)

Customer Name	Customer Email	Product Name	Products Ordered
---------------	----------------	--------------	------------------

Figure 3 Navigation Bar of the system.

[Home](#) [About us](#) [Cart](#) [Admin](#) [Logout](#)

Add Product

No file chosen

Add Product

Figure 4 Add product

[Home](#) [About us](#) [Cart](#) [Profile](#) [Login](#)

Checkout Now

Your Bag

Order Summary

Subtotal:\$

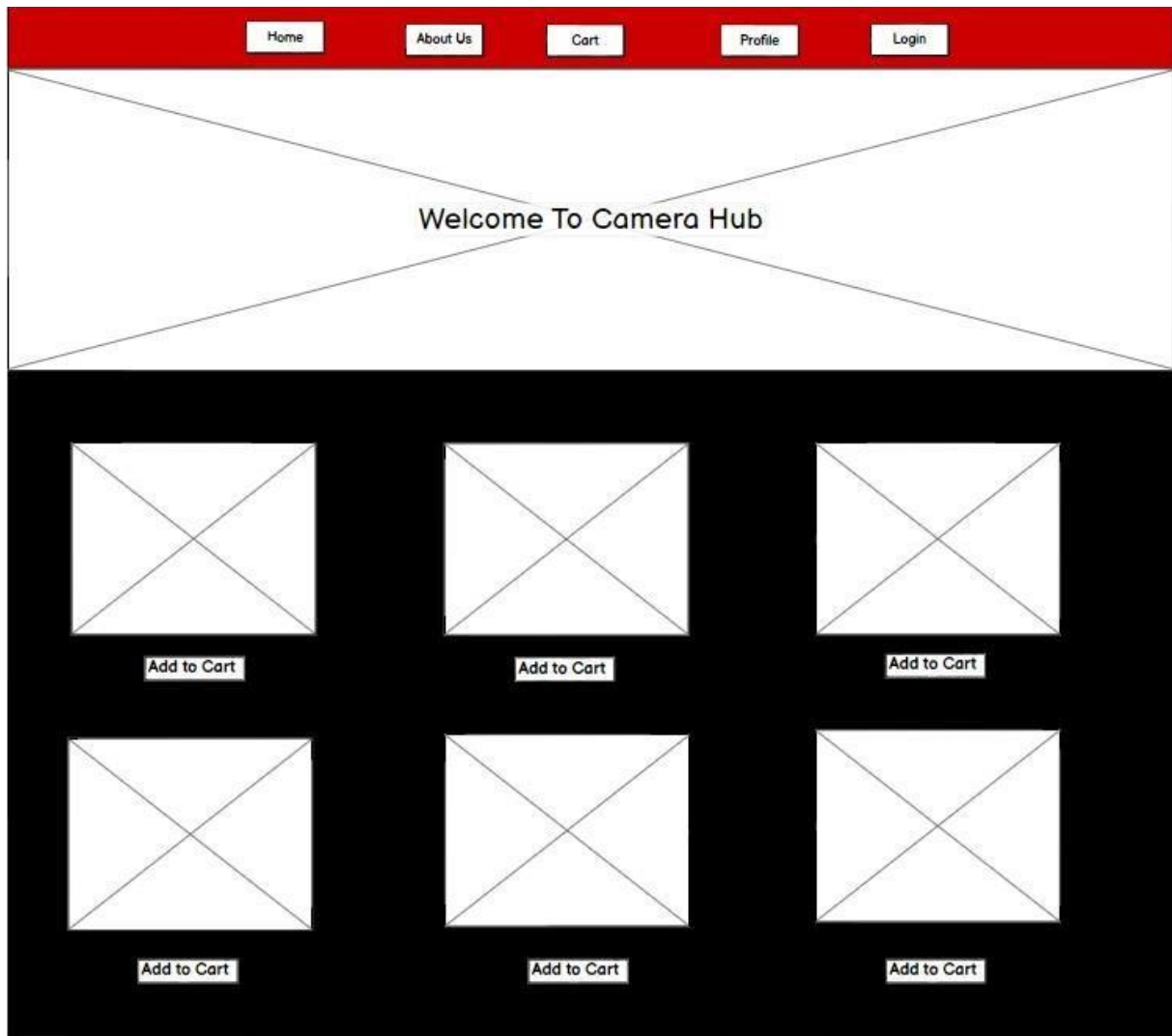
Estimated Total:\$

Proceed to Checkout

☐ e-Sewa ☐ PayPal

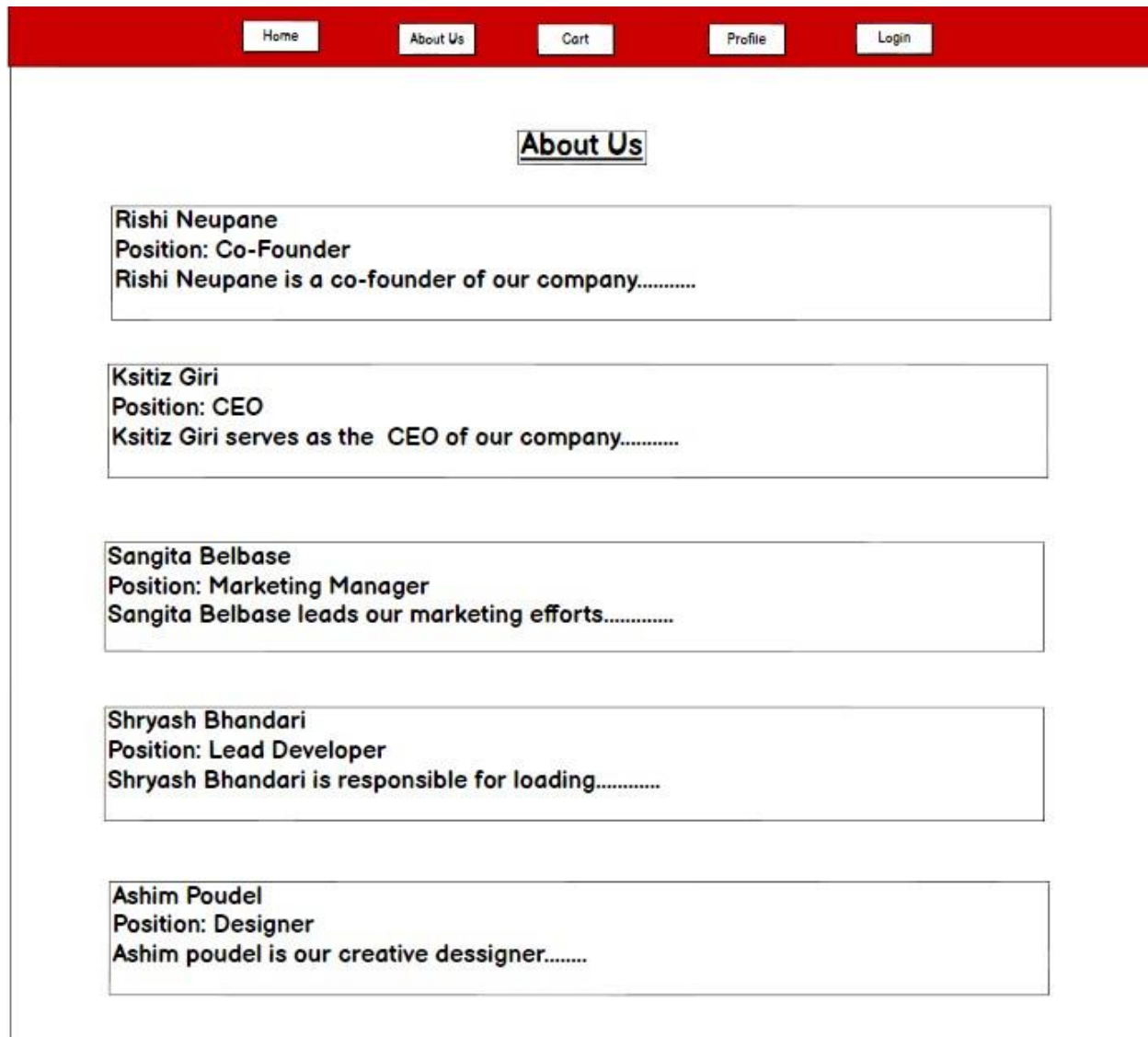
Figure 5 Product in Cart/Bag for checkout.





6 User Home page.

Figure



*Figure 7 About us page.*

## 2.3 Actual Design

Our goal as we develop our e-commerce website is to create an attractive interface while giving usability and functionality top priority. Our goal is to provide a user experience that is accessible, responsive, and intuitive to meet the wide range of needs of our users. We aim to produce a design that improves the user experience overall in addition to looking fantastic through collaboration, iteration, and a user-centered approach. (Lewis)

### 2.3.1 User Screen

Greetings from our user-friendly online store! The main goals of our design are usability, clarity, and simplicity. We're here to make your purchasing experience easy and pleasurable, from simple navigation to a frictionless checkout. Prepare to browse our most recent camera and equipment offers easily and confidently.

#### Home Page:

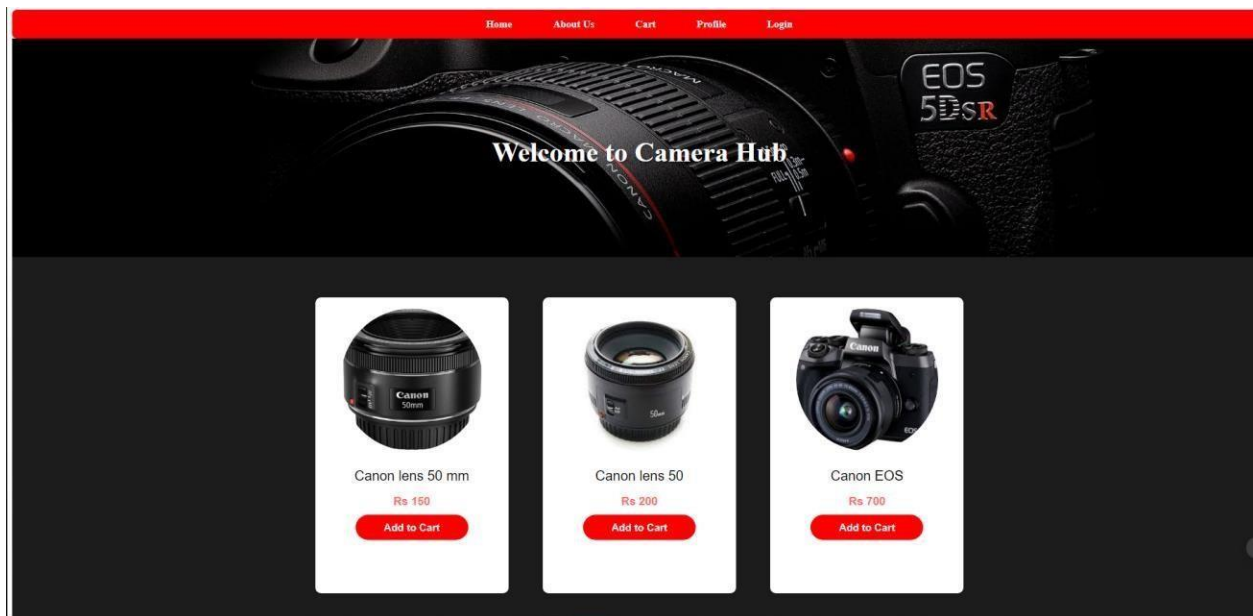


Figure 8 Actual design of home page.

#### Cart Page:

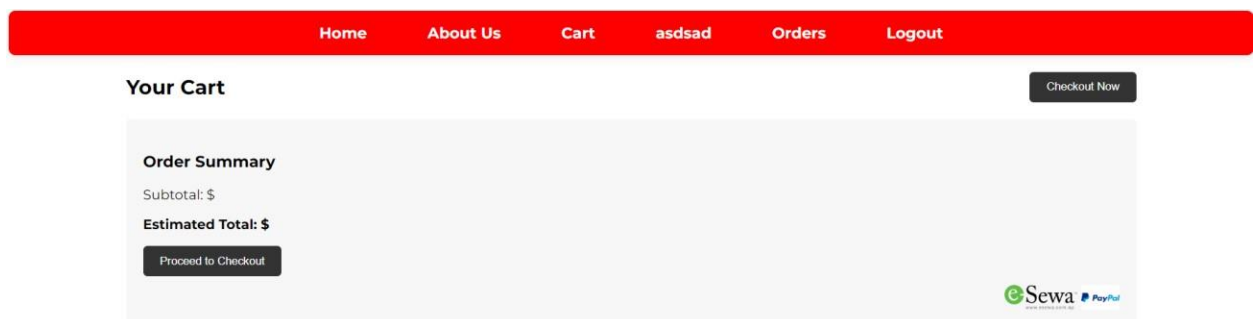


Figure 9 Actual design of Cart Page when no items are added.

When items are added in the Cart:

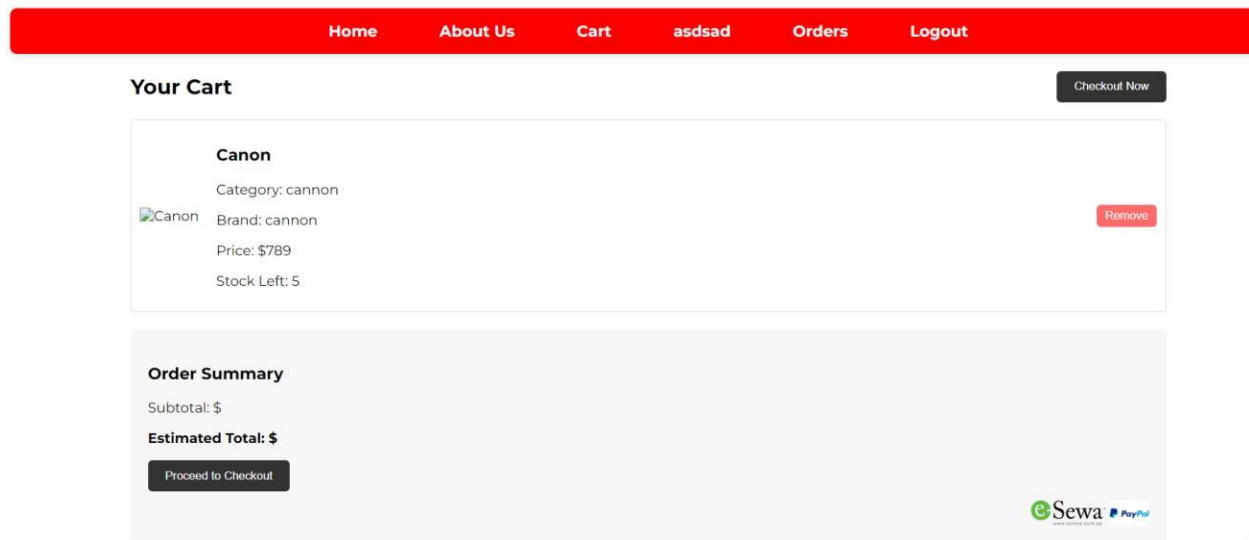


Figure 10 Actual design of Cart Page when item is Added.

## About us:

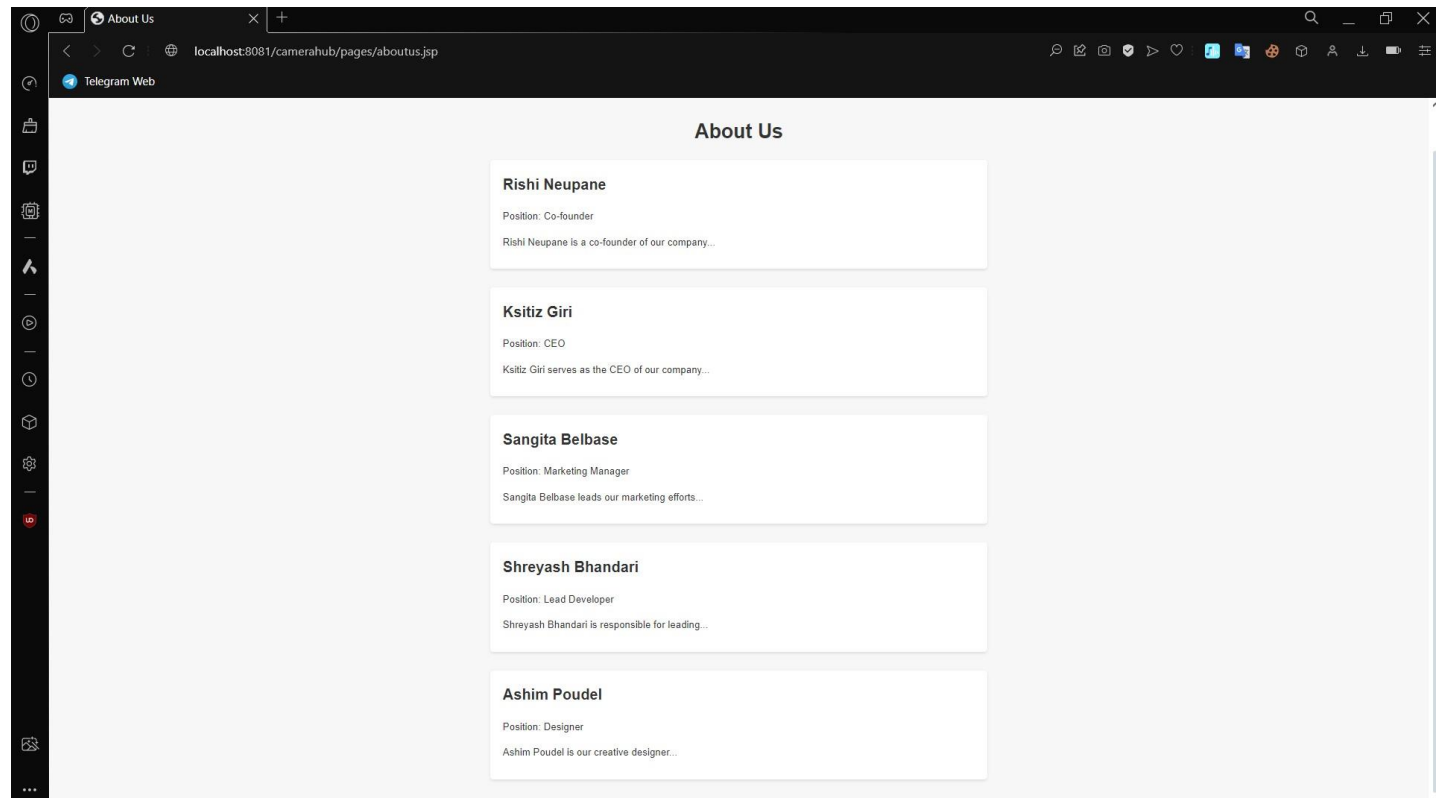



Figure 11: About us in actual design.

## User Profile:

## Edit Profile Details



**Name:** Kishitiz

**Contact:** Rishi@Gmail.Com

### Update Details

### Update Profile Picture


Choose File

No file chosen

Save Changes

Figure 12 Before editing user's details.

### Edit Profile Details



**Name:** Kishitiz

**Contact:** Rishi@Gmail.Com

#### Update Details

#### Update Profile Picture

Choose File

No file chosen

Save Changes

Figure 13 Editing the details.

Home	About Us	Cart	Rishi	Orders	Logout
------	----------	------	-------	--------	--------

Order ID: 4	2024-05-08
User Name: Rishi	Product ID: 9
Order ID: 5	2024-05-08
User Name: Rishi	Product ID: 9
Order ID: 7	2024-05-09
User Name: Rishi	Product ID: 11

Figure 14 Actual design of purchase History page

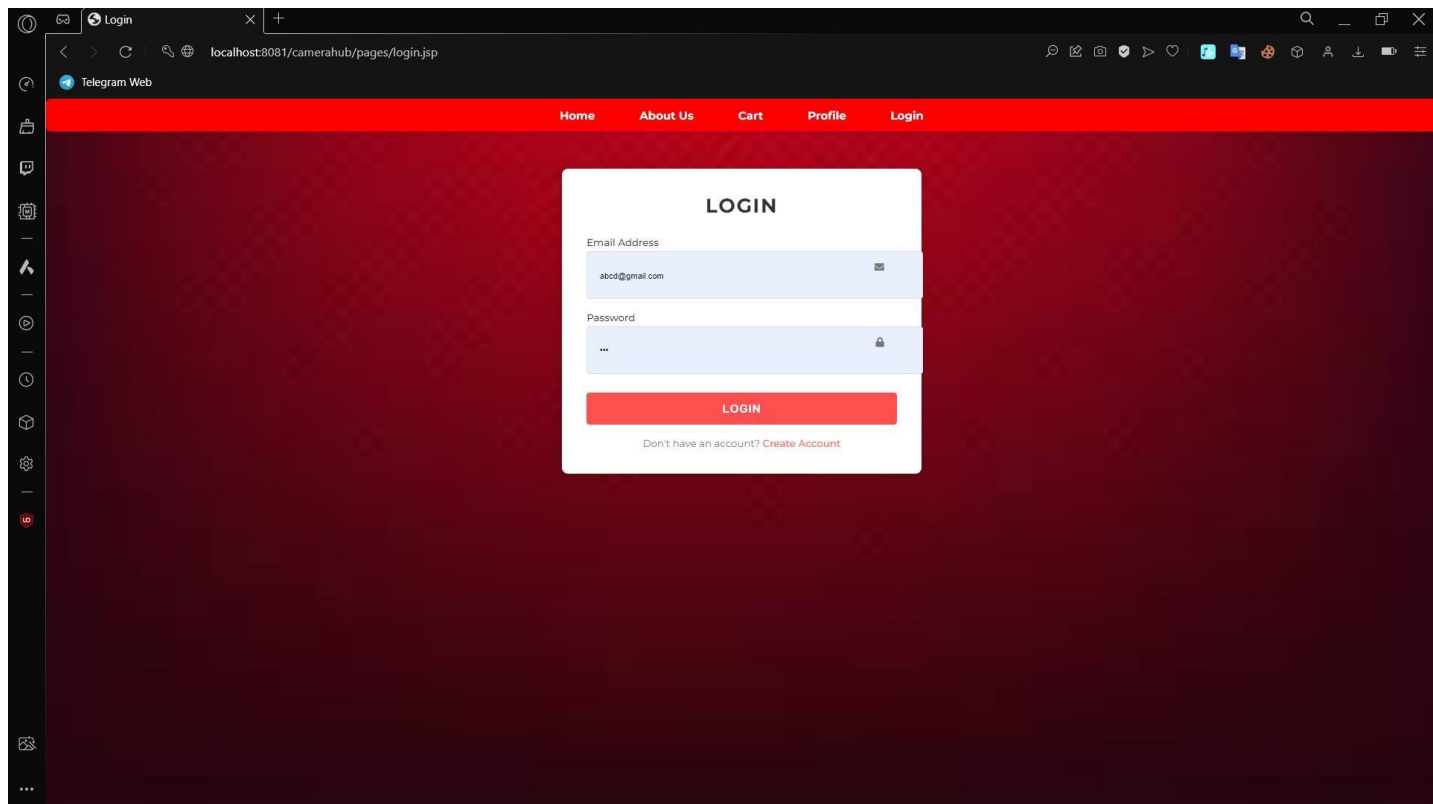


Figure 15: Login Page.

### 2.3.2 Admin Screen

Admin may easily manage products, track orders, and maintain the functionality of the website from this location. Our admin panel design prioritizes efficiency and simplicity, offering user-friendly tools for adding, changing, and removing products as well as quick access to important data. Admin may remain on top of responsibilities and guarantee outstanding user experience with streamlined workflows and simple navigation.

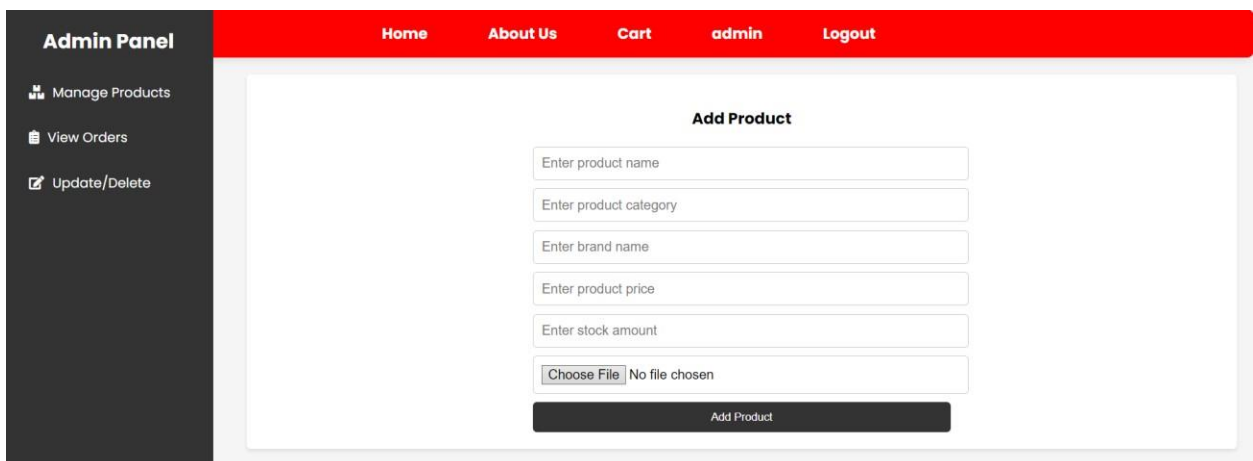


Figure 16 Actual design of Admin Dashboard 1

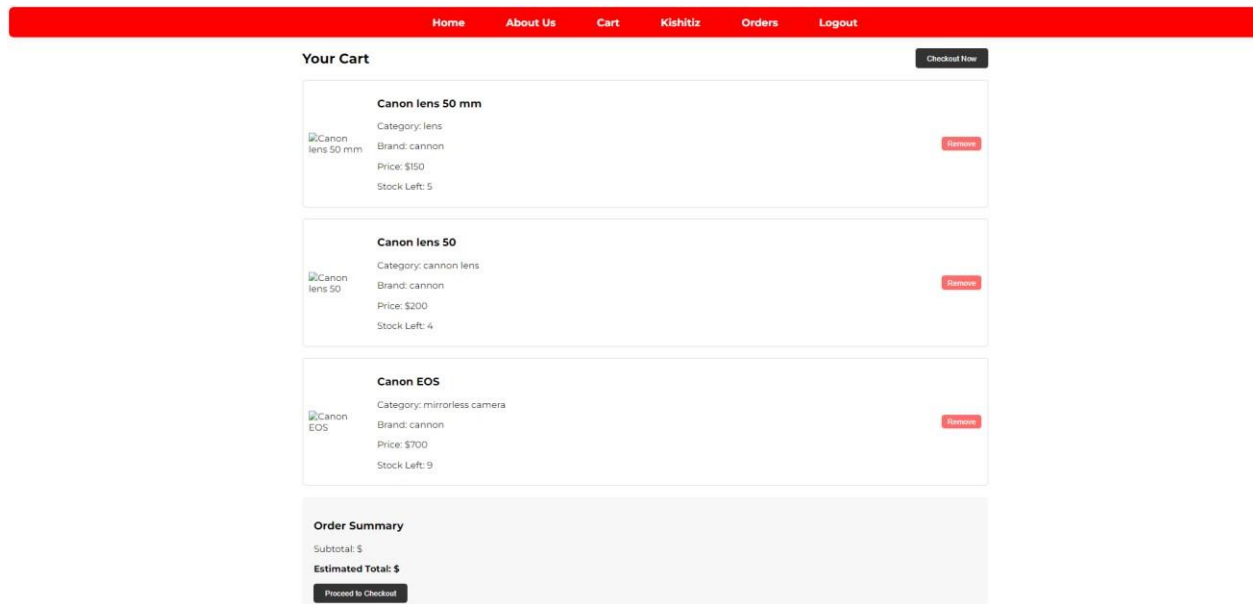
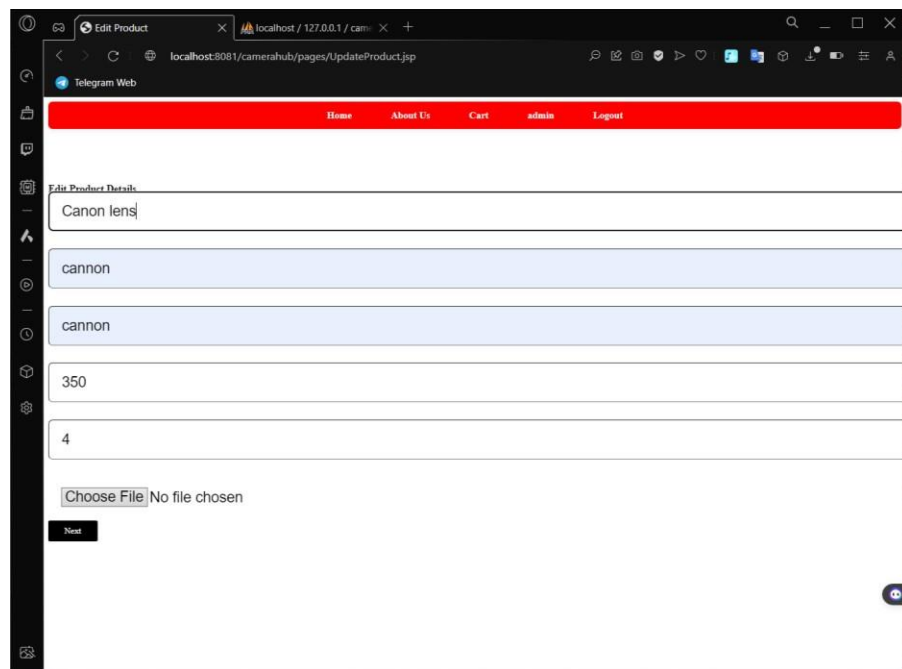


Figure 17 Actual design of admin dashboard.



18 Actual design of Editing Product Page.

Figure



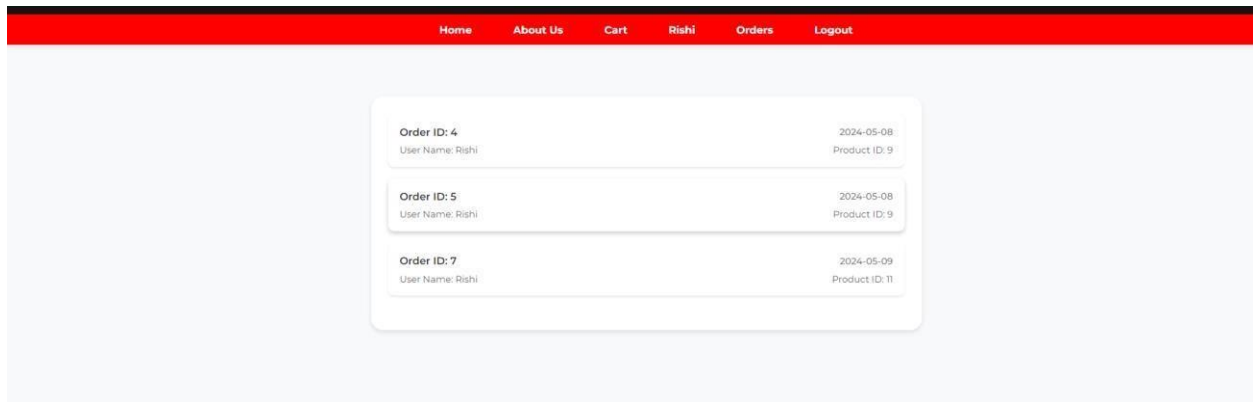


Figure 19 Actual Design of Manage Order Page.

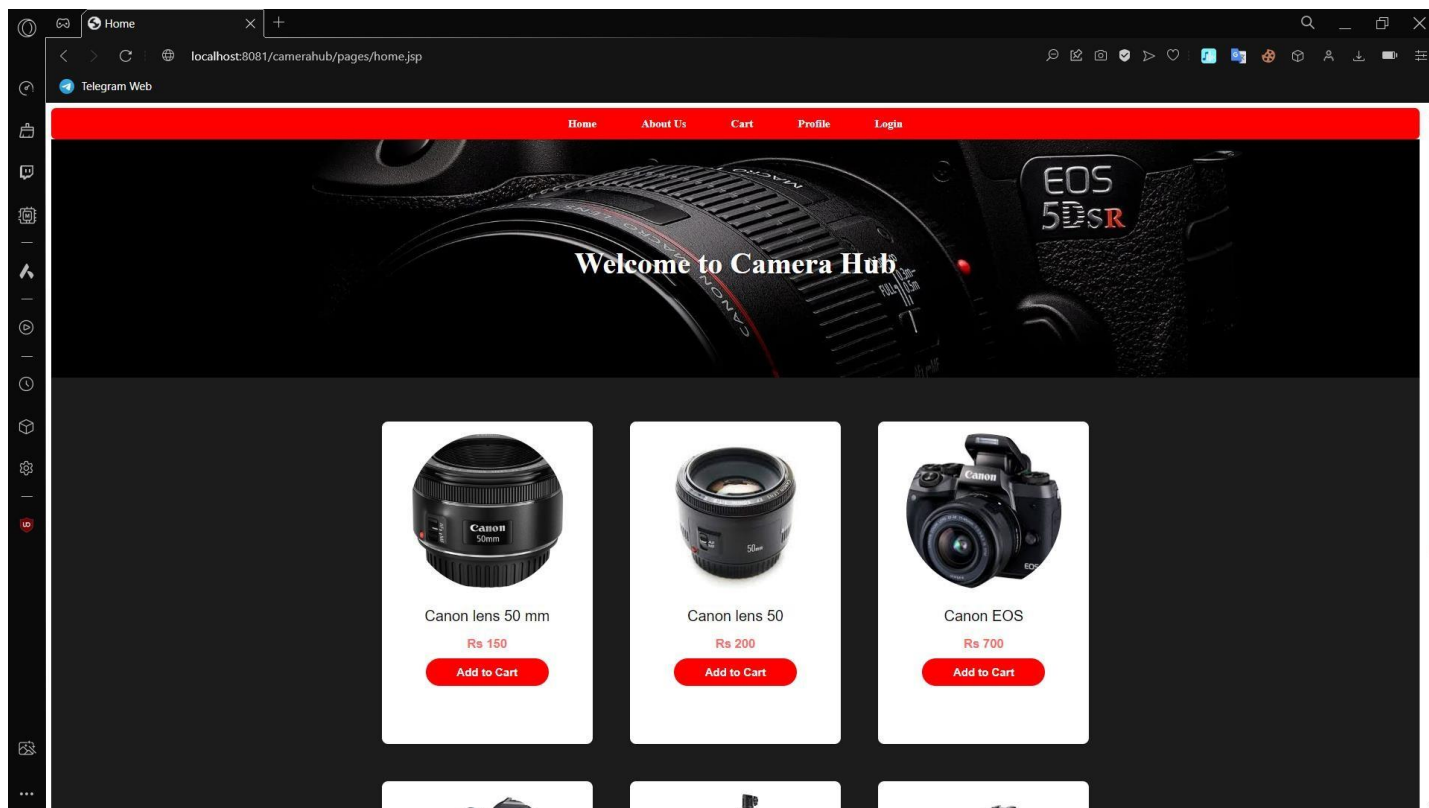


Figure 20: Redirecting

### 3. Class Diagram

The structure and relationships between classes in a system are visually represented using class diagrams, a type of UML (Unified Modeling Language) diagram used in software engineering. Software system design and documentation are helped by the standard modeling language known as UML. They contribute significantly to the design and documentation stages of the software development process. (Bhumika, 2024)

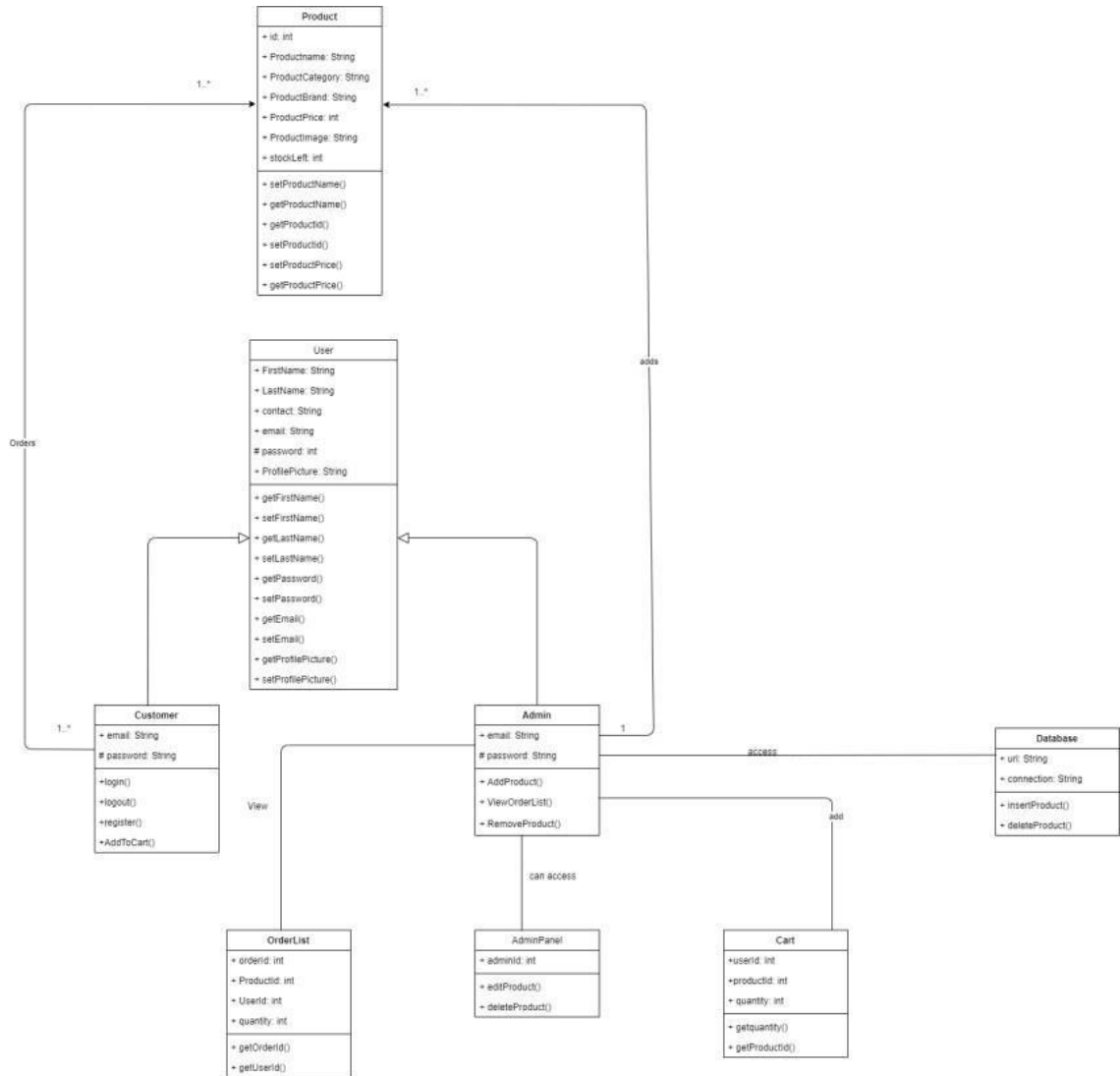


Figure 21 Class Diagram.

## 4. Method Description

### 4.1 Authentication filter.java

Method Name	Description

<b>delete()</b>	This function is called by the web container to notify the filter that it is about to be disabled. With this filter, there is no need for a specific cleanup method because the delete () function is empty. The delete () function can be empty and perform nothing if the filter does not consume any resources during initialization that must be released after deletion.
<b>doFilter()</b>	The primary filter method that handles authentication is called doFilter. As arguments, it takes in ServletRequest, ServletResponse, and FilterChain objects. It retrieves the user object from the session to determine whether the user is logged in or not. If the person is signed in and has admin@gmail.com as their associated email address, the filter chain can proceed. If the user is not logged in or does not have admin@gmail.com as their email address, they are sent to the home.jsp page.
<b>Init()</b>	This function is called by the web container to notify the filter that it is about to go into service. This filter does not require any initialization tasks because it is empty in this case. Since this technique is a part of the Filter Interface, it still needs to be applied. If this method is not
	implemented when the filter is initialized, the web container produces a ServletException.
<b>doPost()</b>	HTTP POST requests, which frequently involve transmitting data to the server, are handled by the doPost() function. When a user submits the login form, the doPost() function, for example, may be used to process the input and confirm the user's credentials.

<b>doGet()</b>	When handling HTTP GET requests, the doGet() method is frequently used to retrieve data from the server. For example, the home page of our website may be shown using the doGet() function.
<b>DataBaseConnection()</b>	Importantly, the function DataBaseConnection() is used to establish a connection to the database. The connection object that is typically provided by this function can be used to submit queries, retrieve data from the database, and insert, edit, and remove data.
<b>encryption()</b>	In this project, the user password is encrypted using the encryption() method before being stored in the database. Because it enables safe password storage and authentication, this technique is vital.

*Table 1 Method Description of Authentication Server.*

## 4.2 EncDyc.java

<b>Method Name</b>	<b>Method Description</b>
<b>encryption()</b>	In this project, the user password is encrypted using the encryption() method ahead of being stored in the database. Because it enables safe password storage and authentication, this technique is essential.
<b>decrypt()</b>	Using AES decryption, this method takes an encrypted string and its secret key string, "secretkey," and decrypts it.
<b>generatekey()</b>	This technique uses UTF-8 encoding to transform a secret key into a byte array.

*Table 2 Method Description of EncDyc.*

## 4.3 editProfile.java

<b>Method Name</b>	<b>Description</b>

<b>change()</b>	This function, which is part of the 'Query' class, updates the user record by accepting several parameters relating to user profile data.
-----------------	---

*Table 3 Method Description of editProfile.*

#### 4.4 productQuery.java

Method Name	Description
<b>productQuer ()</b>	A connection object is provided into this constructor method, which then assigns it to the class variable "con."
<b>addProduct()</b>	This function adds the properties of the product object into the database's product table using the product object as input.
<b>deleteProduc:()</b>	This method deletes the record from the database's "product" table that has the corresponding integer "id," accepting it as an input.
<b>editProduct()</b>	This method removes the record associated with the given integer "id" after receiving it as input.
<b>addOrder()</b>	This method inserts order details into the database's "order" table using the order details as input.

*Table 4 Method Description of ProductQuery.*

#### 4.5 Query.java

Method Name	Method Description
<b>register()</b>	A new user is added to the database using this technique.
<b>check()</b>	This technique determines if a specific email is already stored in the database.
<b>login()</b>	With this method, a user's email address and password are used to log in.
<b>change()</b>	The user's profile is updated via this method.

*Table 5 Method Description of Query.*



## 5. Test Cases

### 5.1 Test Case 1

○ Valid Login:

<b>Objective</b>	<b>To access the website using either the admin or user login.</b>
<b>Action</b>	Entering their email address and password allows the user or admin to access the website.
<b>Expected Result</b>	The user is directed to the home page and the admin panel page, respectively, by the system after identifying them.
<b>Actual Result</b>	The user and admin roles were recognized by the system, which then sent them to the appropriate pages the admin page and the home page, respectively.
<b>Conclusion</b>	The test was successfully.

Table 6 Valid Login

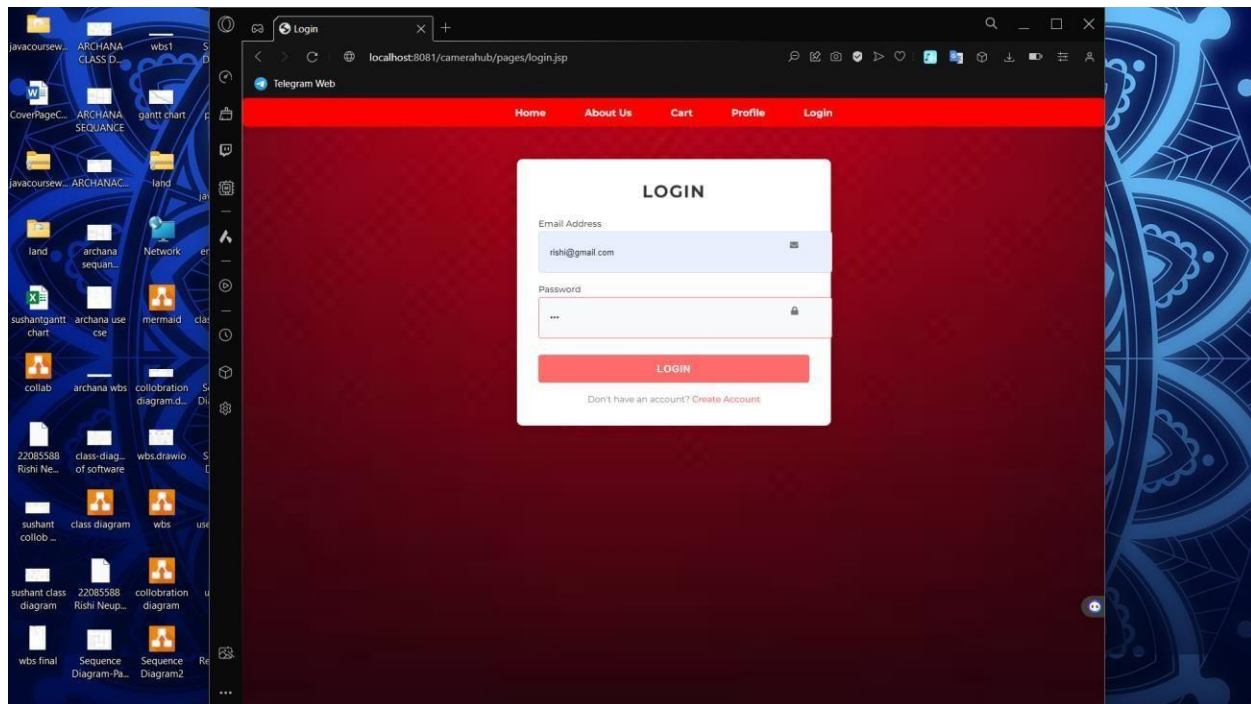


Figure 22 Valid Login 1

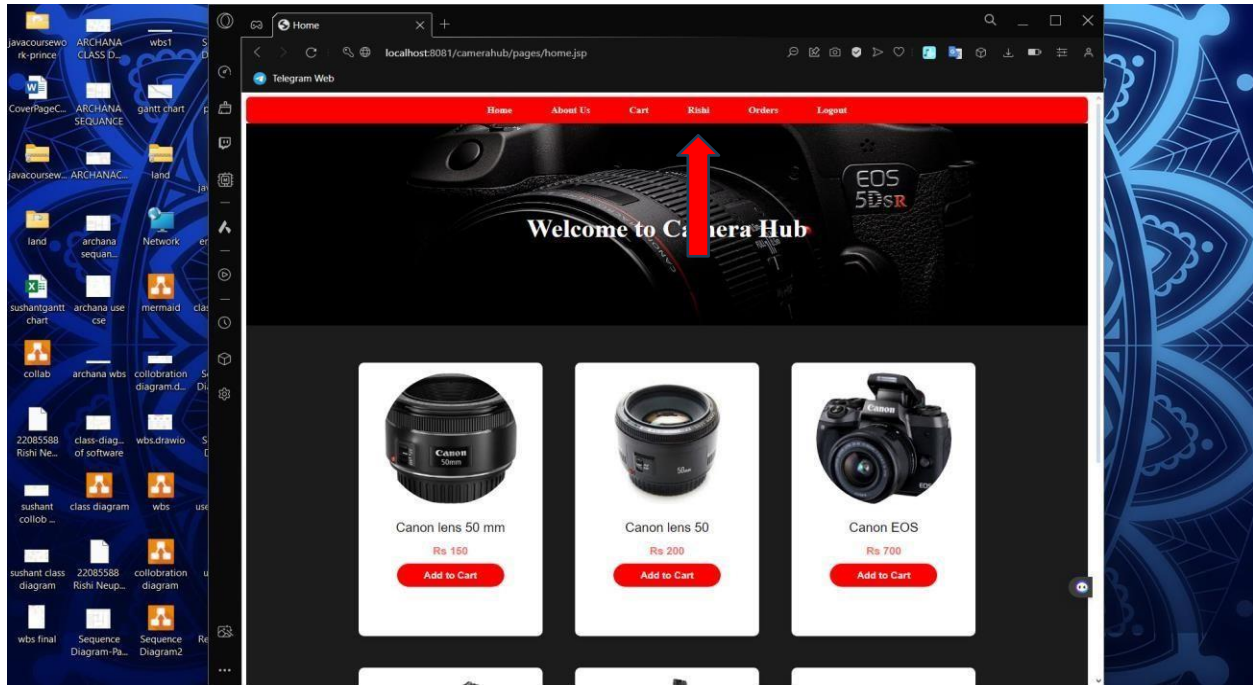


Figure 23:Valid Login 2.

**Evidence for valid Login:**



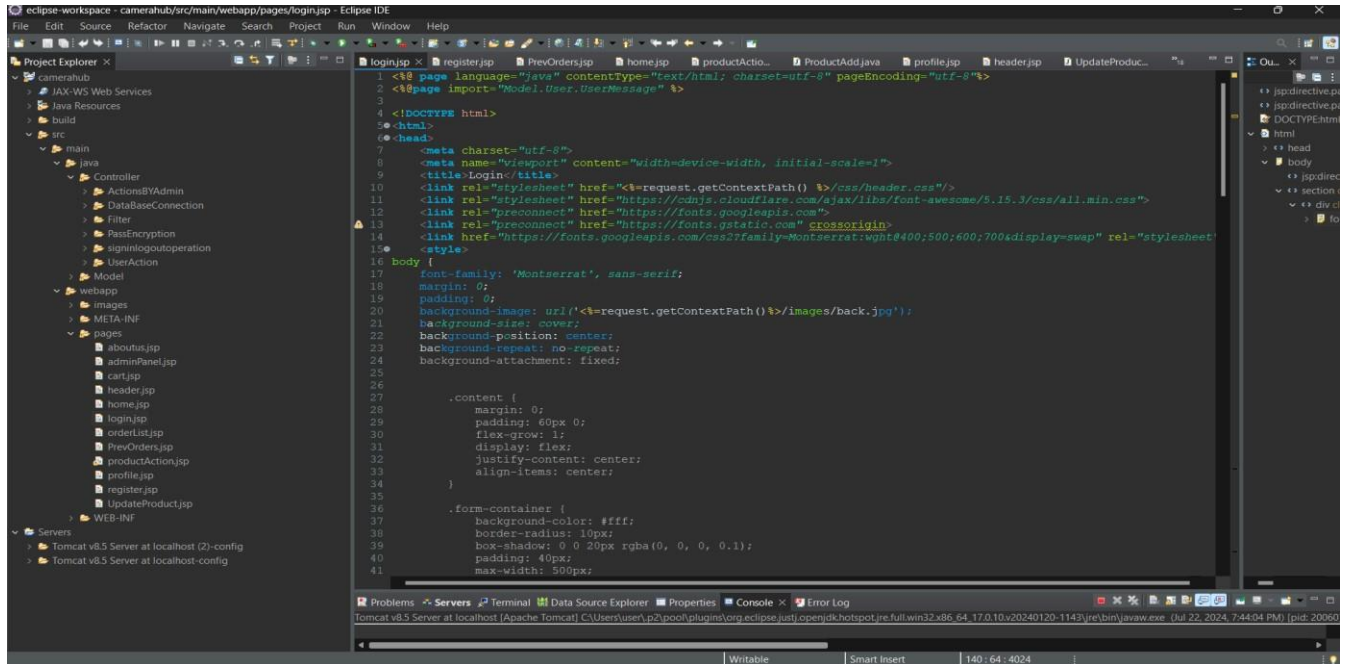


Figure 24: Evidence for Valid Login.

The JSP code combines HTML, CSS, and Java to produce a responsive login page. It imports the required files, such as Font Awesome for styling and external CSS. In addition to providing links to stylesheets and custom styles for the page layout, the head section also sets up the meta tags. The body has a user login form with email and password fields that are indicated by icons. An alert shows the content of a UserMessage object if one is in the session. The message is deleted from the session after it has been displayed. The LoginServlet receives the user credentials from the form and processes them. The page also has a link to the page where new users can register. This configuration guarantees an intuitive login experience with relevant feedback and options for navigation.

## 5.2 Test Case 2

### Valid Registration:

Objective	To register new user.
Action	Click the Register Now button after entering the necessary information on the registration page, which consists of your First and Last Name, Contact Number, Email Address, Password, and Image.
Expected Result	Users should be redirected to the login page and their data should be kept in the database.
Actual Result	The user was redirected to the login page after their

	credentials were successfully saved in the database.
<b>Conclusion</b>	The test was successfully.

Table 7 Valid Registration

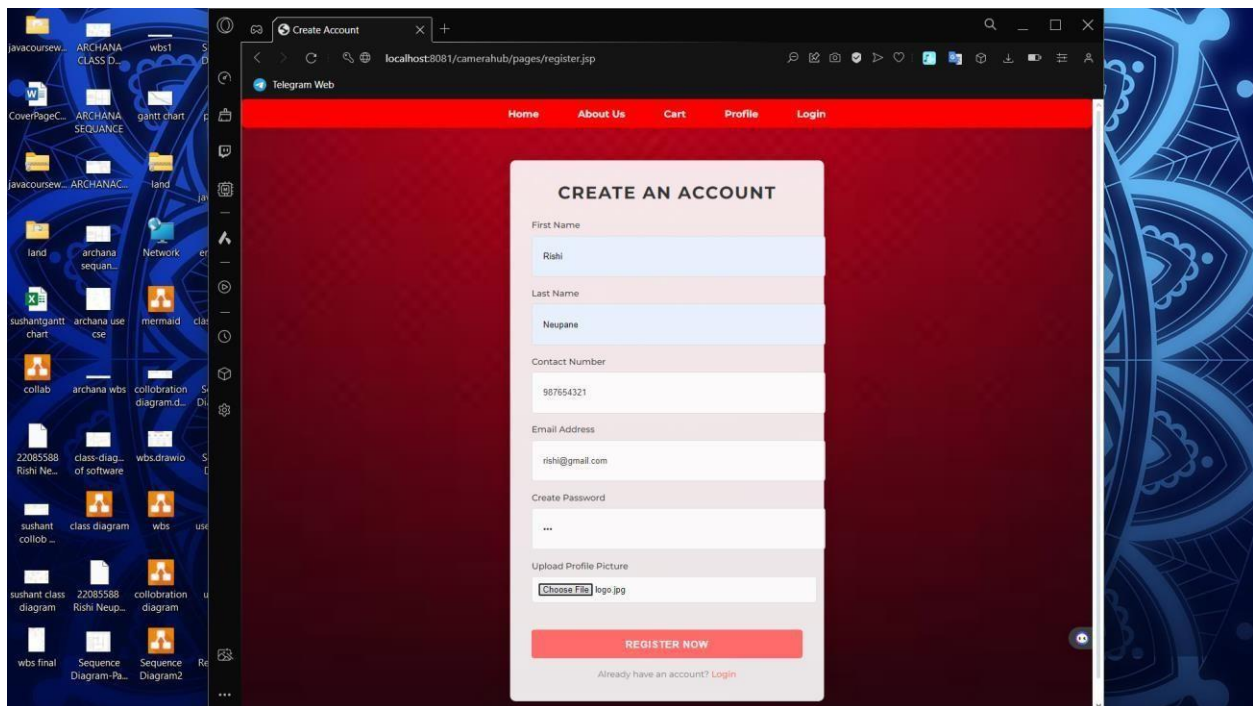


Figure 25 Valid Registration 1

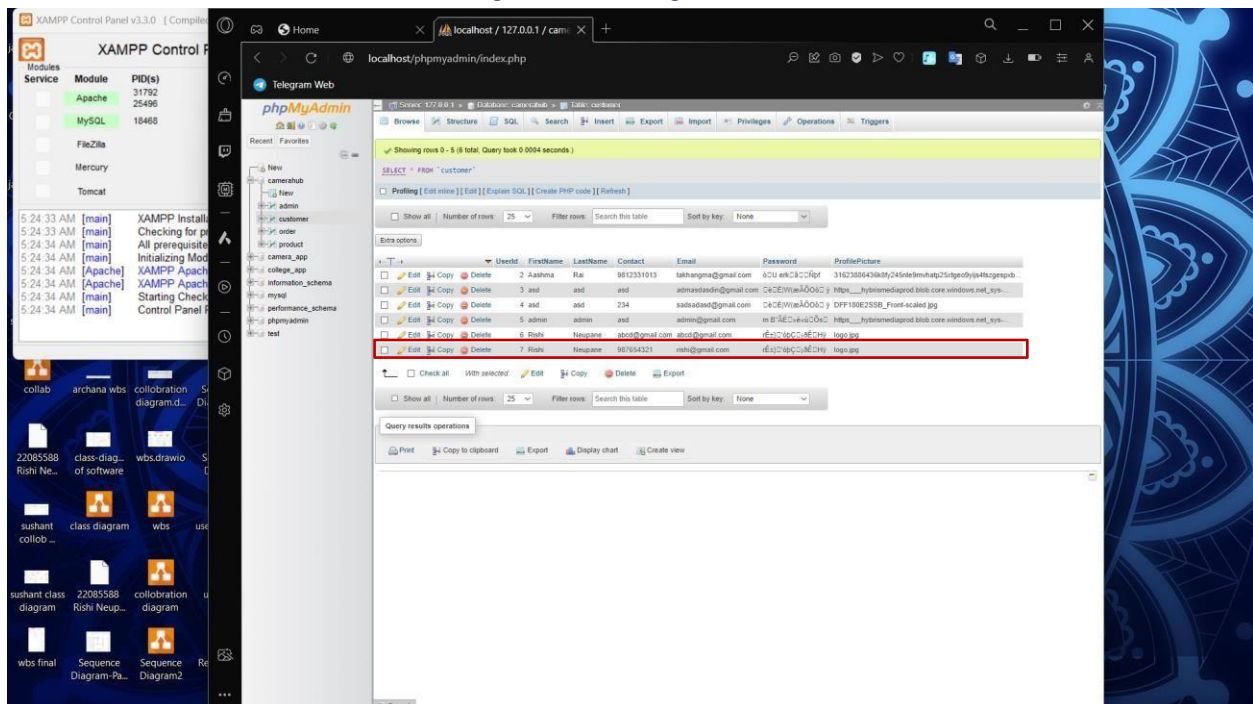


Figure 26 Valid registration 2



### 5.3 Test Case 3

- Encrypted Password:

<b>Objective</b>	<b>To Encrypt the user's password while registering.</b>
<b>Action</b>	Users create an encrypted password during registration, which is kept in the database.
<b>Expected Result</b>	After a successful registration, the password entered by the user should be encrypted and kept in the database.
<b>Actual Result</b>	The user successfully registered, and their password was encrypted and stored in the database.
<b>Conclusion</b>	The test was successfully.

Table 8 Encrypted Password

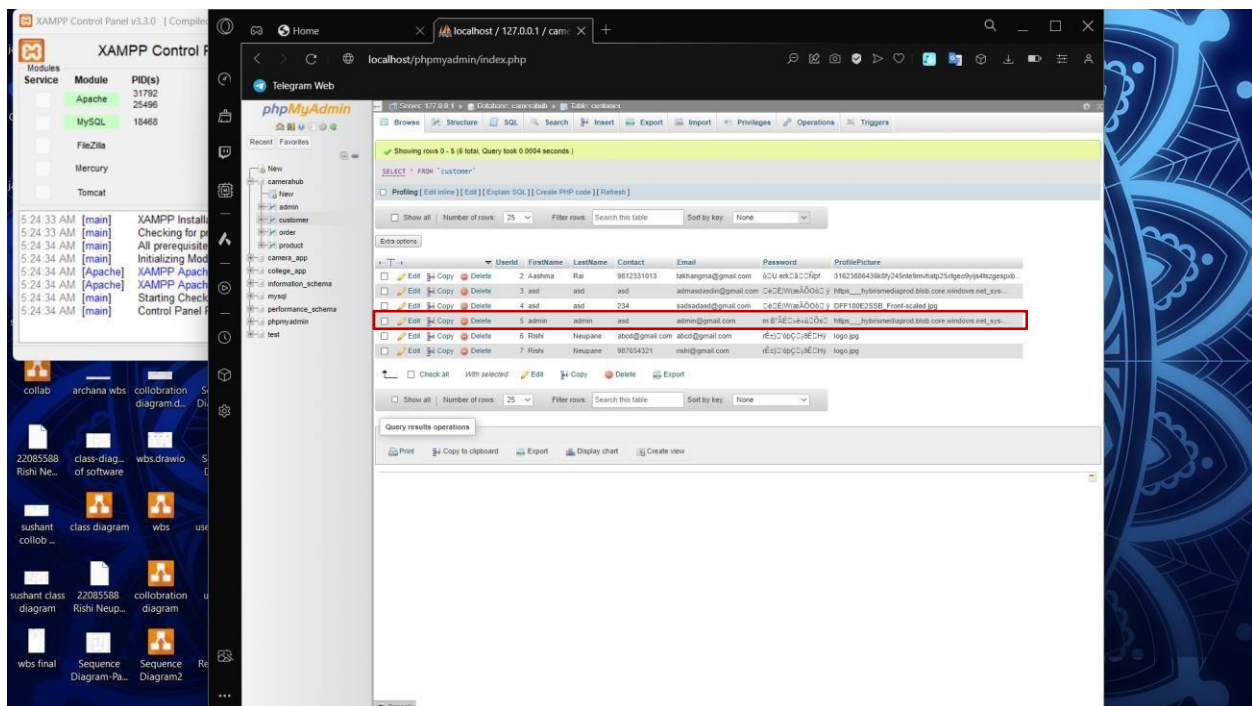


Figure 27 Encrypted Password 1.

### 5.4 Test Case 4

- Logging in before adding products:

<b>Objective</b>	<b>When placing an order, direct the customer to the login</b>
------------------	--

	page if they haven't already logged in and register them if they haven't already.
<b>Action</b>	To place an order for the desired product, the user selects the Add to Cart option.
<b>Expected Result</b>	In case the user is not logged in, they should be redirected to the website's login page. If they haven't registered yet, they can create an account by clicking on the "create account" button.
<b>Actual Result</b>	The login page is displayed to the anonymous user, who is then prompted to register and log in before making any purchases from the website.
<b>Conclusion</b>	The test was successfully.

*Table 9 Logging in before adding products*

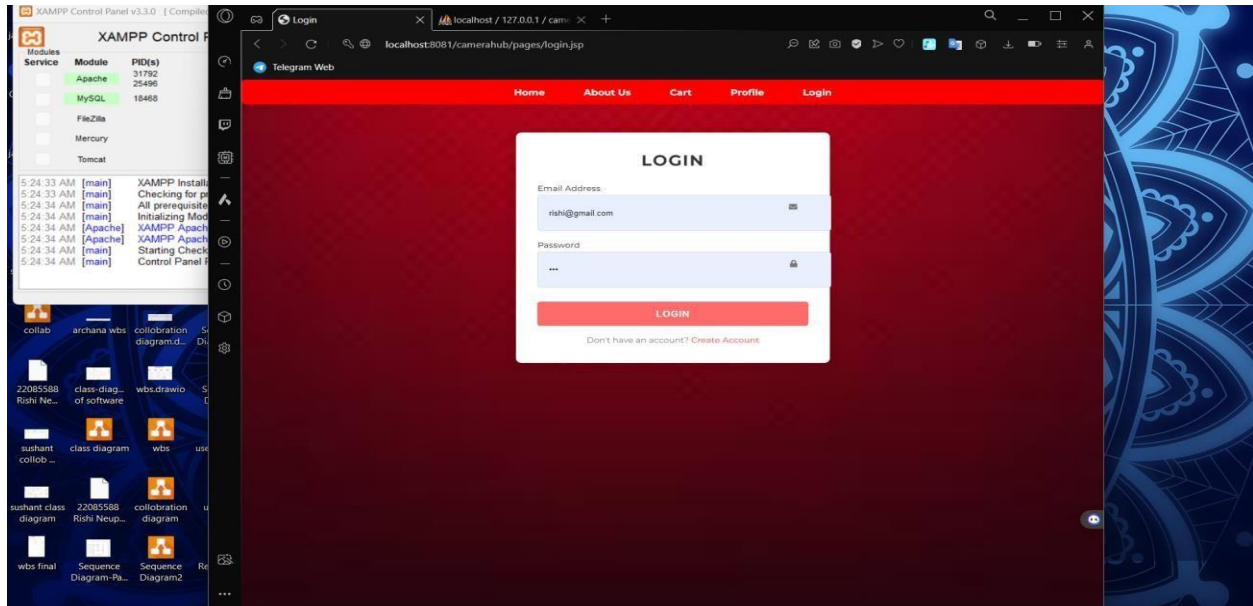


Figure 28 Logging in before adding products.

## 5.5 Test Case 5

- Add to cart feature:

<b>Objective</b>	To enable users to add items to their shopping carts.
<b>Action</b>	On the product's home page, the user should click the "add product" button.
<b>Expected Result</b>	The product needs to be added to the user's cart.
<b>Actual Result</b>	The product has been added to the user's cart.
<b>Conclusion</b>	The test was successfully.

Table 10 Add to cart feature

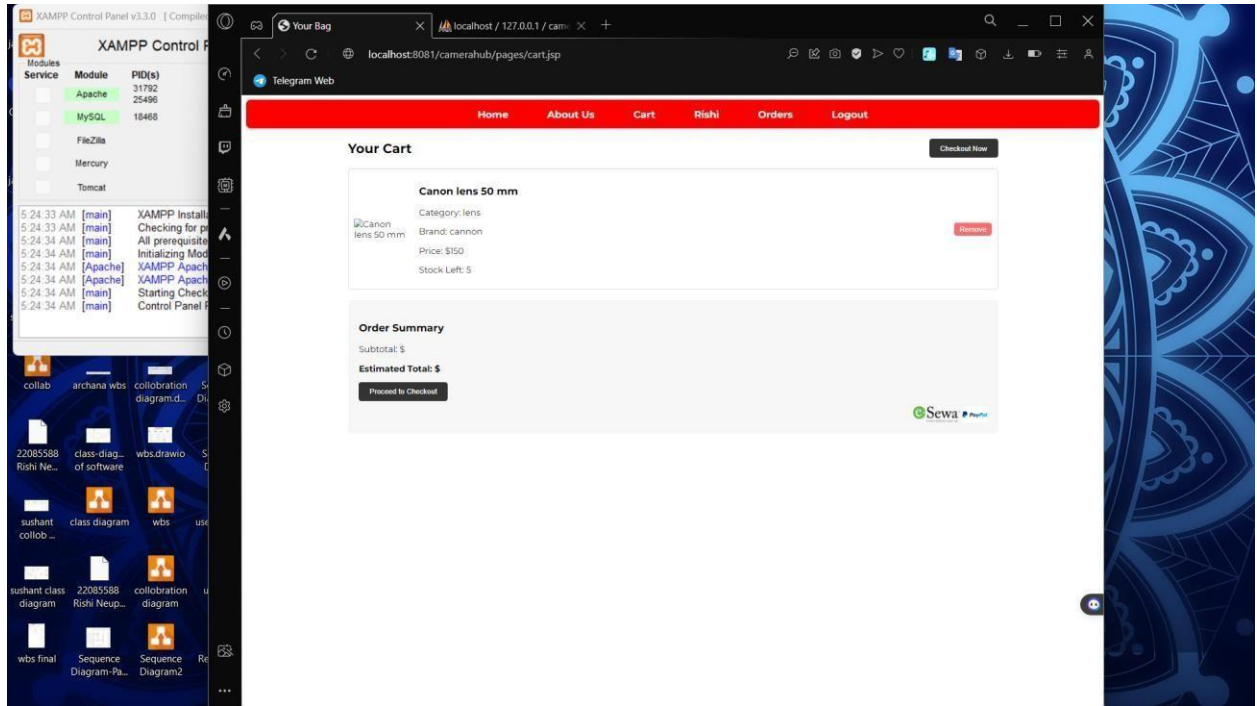


Figure 29 Add to cart feature



**Evidence for cart:**

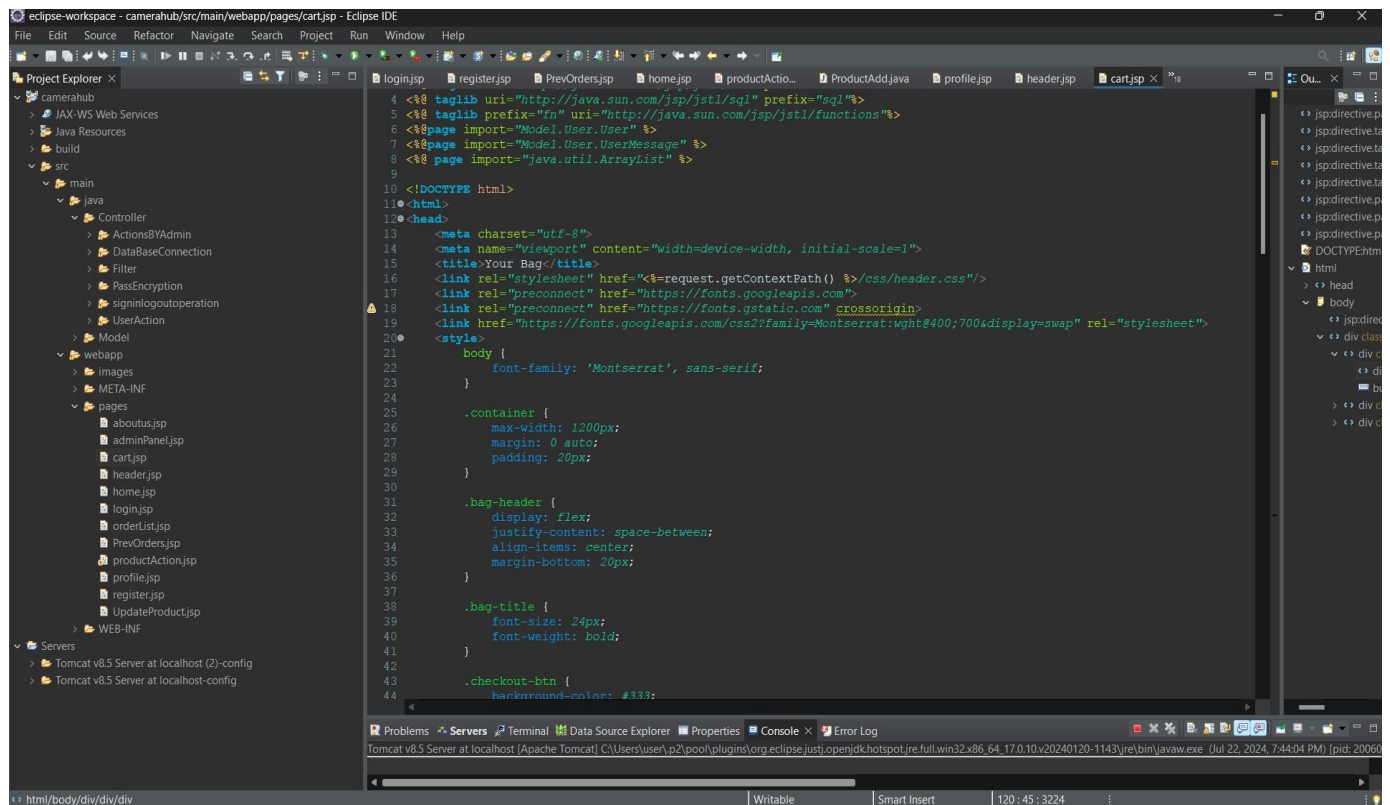


Figure 30: Evidence for cart.

For an e-commerce website, the supplied JSP code creates a shopping cart page. It imports classes and contains the required JSTL tag libraries for working with users and message objects. The head section of the HTML framework defines bespoke CSS and links to external stylesheets for a contemporary, responsive look. A header and a main container displaying cart items and an order summary are included in the body. It uses JSTL SQL tags to retrieve product details for every item in the cart from a MySQL database. The product details are then displayed in a stylized layout along with an image and a delete button. A checkout button containing concealed form data for order processing is included in the order summary area, which also computes and shows the total cost. It also displays the accepted payment options along with the relevant logos. This configuration guarantees an extensive and intuitive shopping cart user interface.

## 5.6 Test Case 6

- Remove product from cart:

<b>Objective</b>	To enable users to remove products from the cart.
<b>Action</b>	The product's cart page should be the user's place to click the remove button.
<b>Expected Result</b>	The product needs to be taken out of the user's cart.



<b>Actual Result</b>	The product has been removed from the cart of the user.
<b>Conclusion</b>	The test was successfully.

Table 11:Remove product from cart.

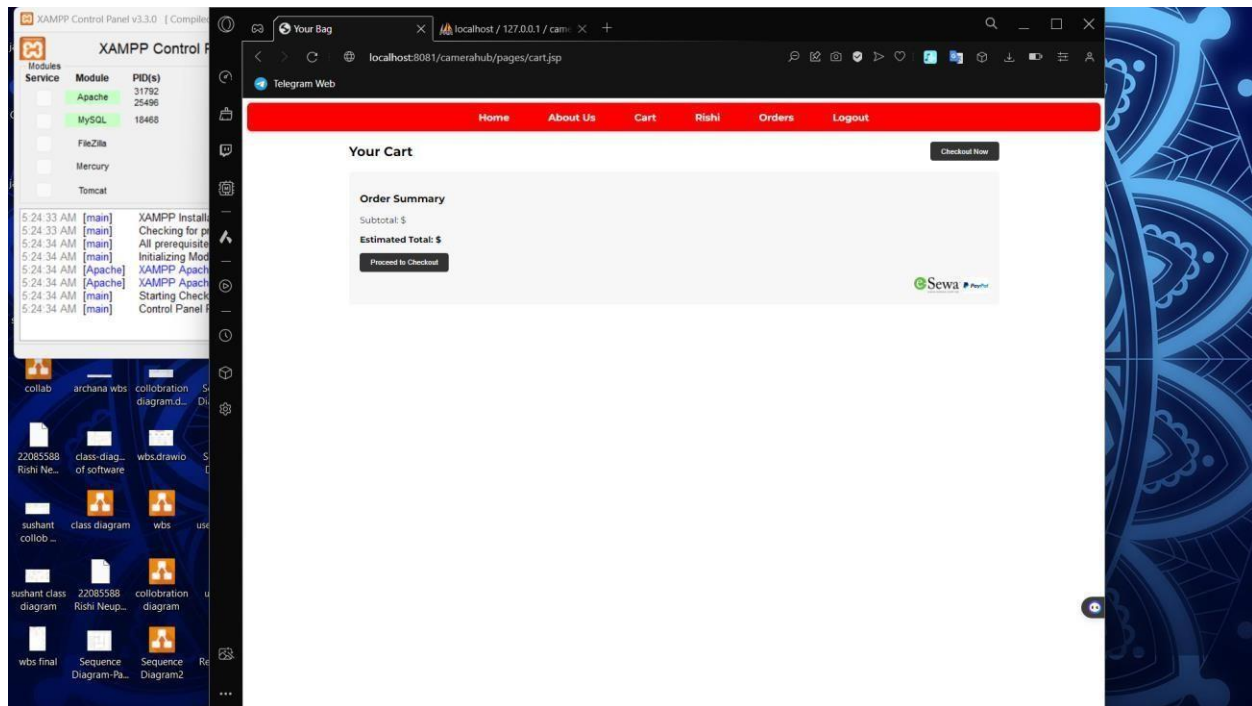


Figure 31 Add to cart feature

## 5.7 Test Case 7

- Editing user Profile:

<b>Objective</b>	To enable users, change their profile information.
<b>Action</b>	The user can edit their information by going to their profile page.
<b>Expected Result</b>	Both the webpage and the database should have the user details updated.
<b>Actual Result</b>	Both the webpage and the database have updated user details.
<b>Conclusion</b>	The test was successfully.

Table 12 Editing user Profile

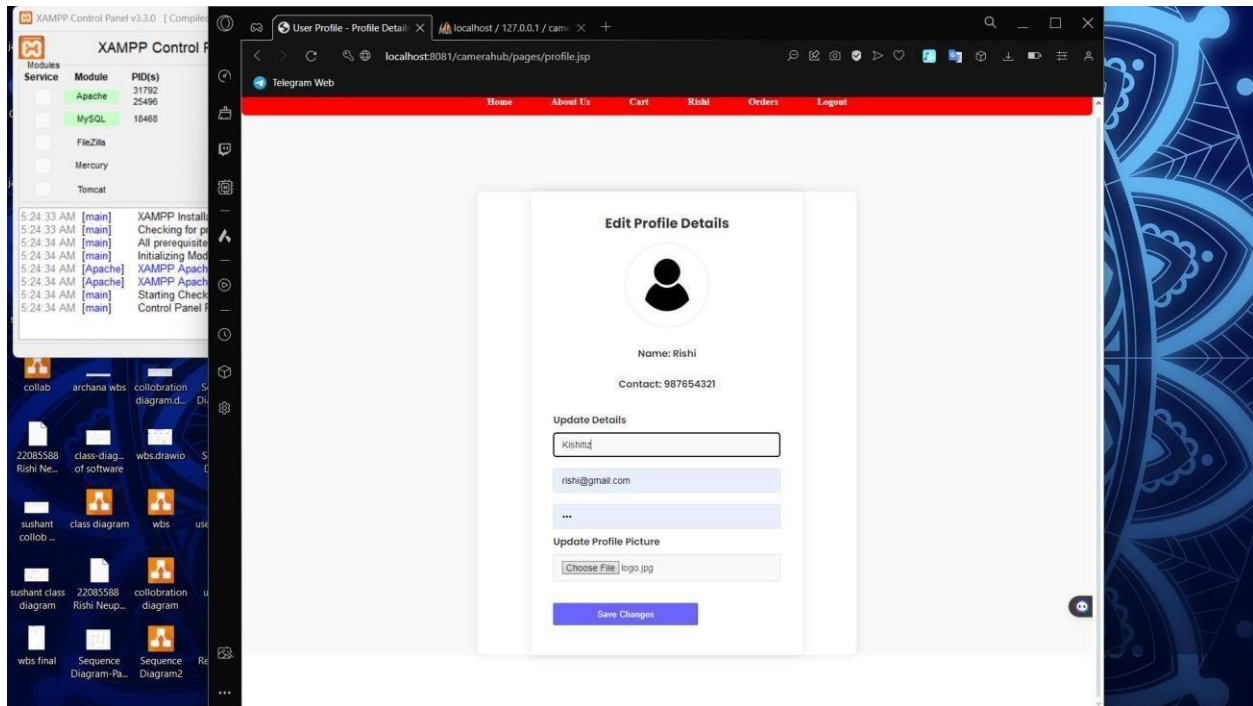


Figure 32 Editing user Profile

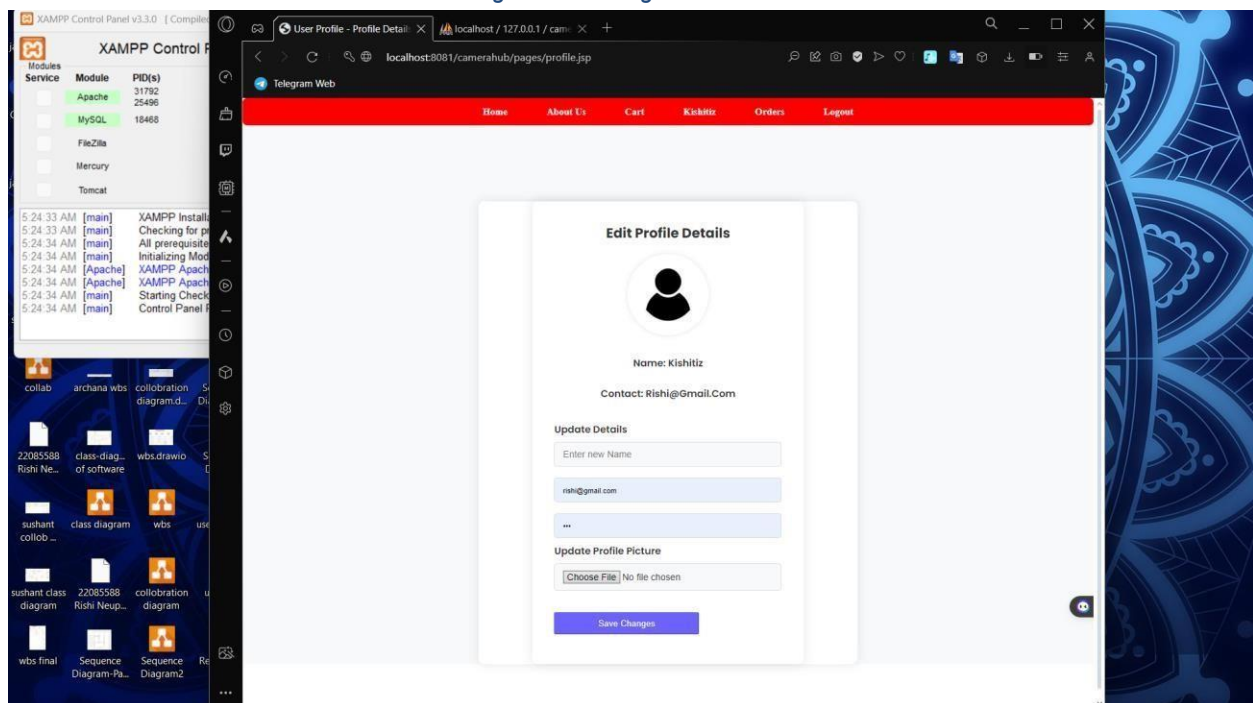
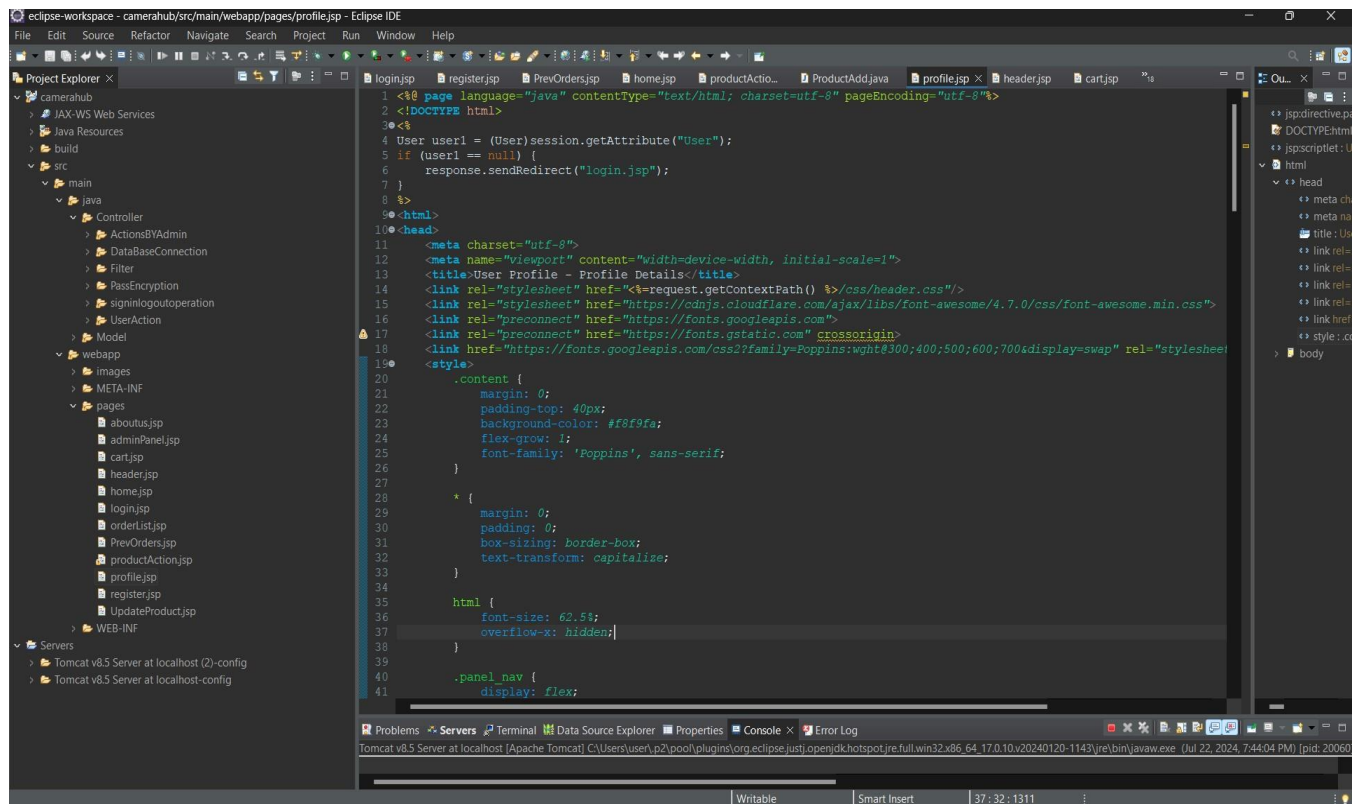


Figure 33 Editing user Profile2

**Evidence For User Profile:**



34: Evidence for User Profile. Figure

Authenticated users can read and change their profile details on the user profile page generated by the JSP code. The first thing it does is see if there is a `User` object in the session. If not, it will reroute to the login page. Links to external stylesheets and the required meta tags for styling are included in the HTML structure. A form that shows the user's name, contact details, and current profile image is included in the page's body. Using input fields and a file upload option, users can alter their name, contact information, password, and profile image. The `editProfile` servlet receives the modified data from the form and processes it. With the help of CSS styling, the page has a contemporary, responsive design that makes managing profiles easy for users.

## 5.8 Test Case 8

- Check User Purchased History:

<b>Objective</b>	To review users' previous purchases.
<b>Action</b>	The user can view their purchase history by going to their profile page.

<b>Expected Result</b>	It should be possible for the user to see their purchase history.
<b>Actual Result</b>	The user's purchase history is visible.
<b>Conclusion</b>	The test was successfully.

Table 13 Checking User Purchased History

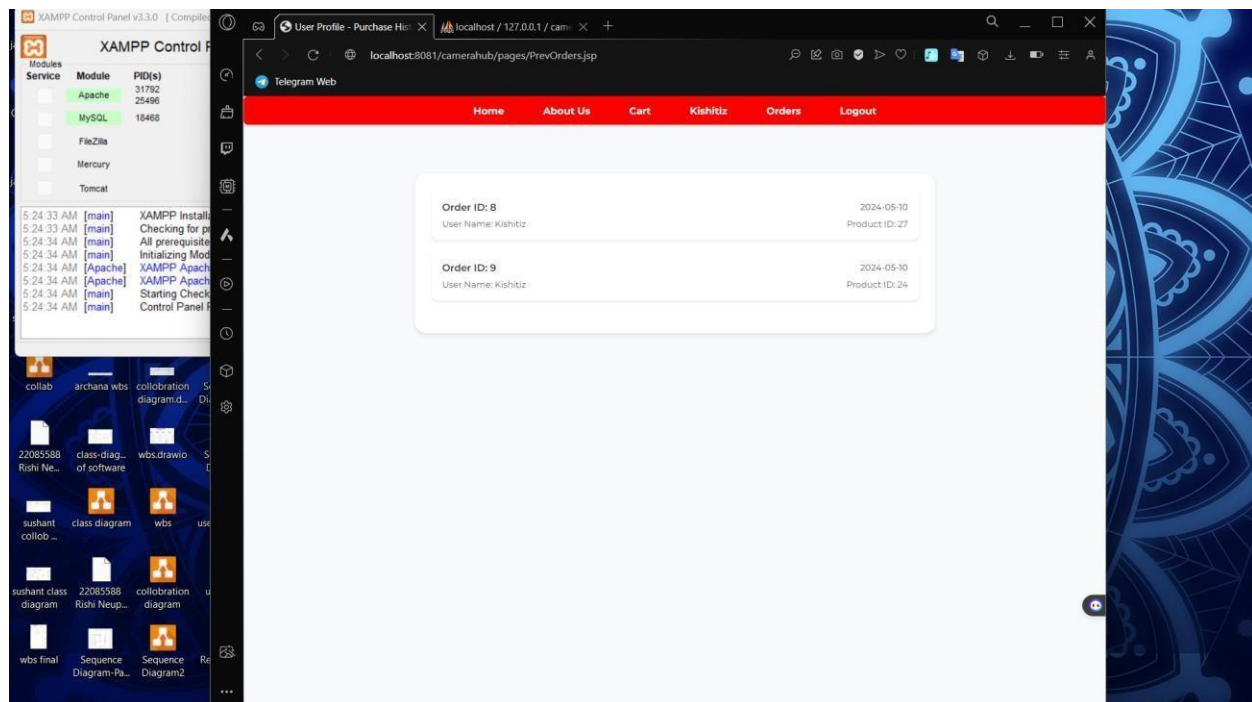


Figure 35: Check Purchased History.

## 5.9 Test Case 9

### 5.9.1 Add product:

<b>Objective</b>	To enable admin, add products to the website.
<b>Action</b>	Admin needs to correctly fill out the required fields.
<b>Expected Result</b>	Both the database and the website should have the product added.
<b>Actual Result</b>	The user's purchase history is visible.
<b>Conclusion</b>	The test was successfully.

Table 14 Adding Product



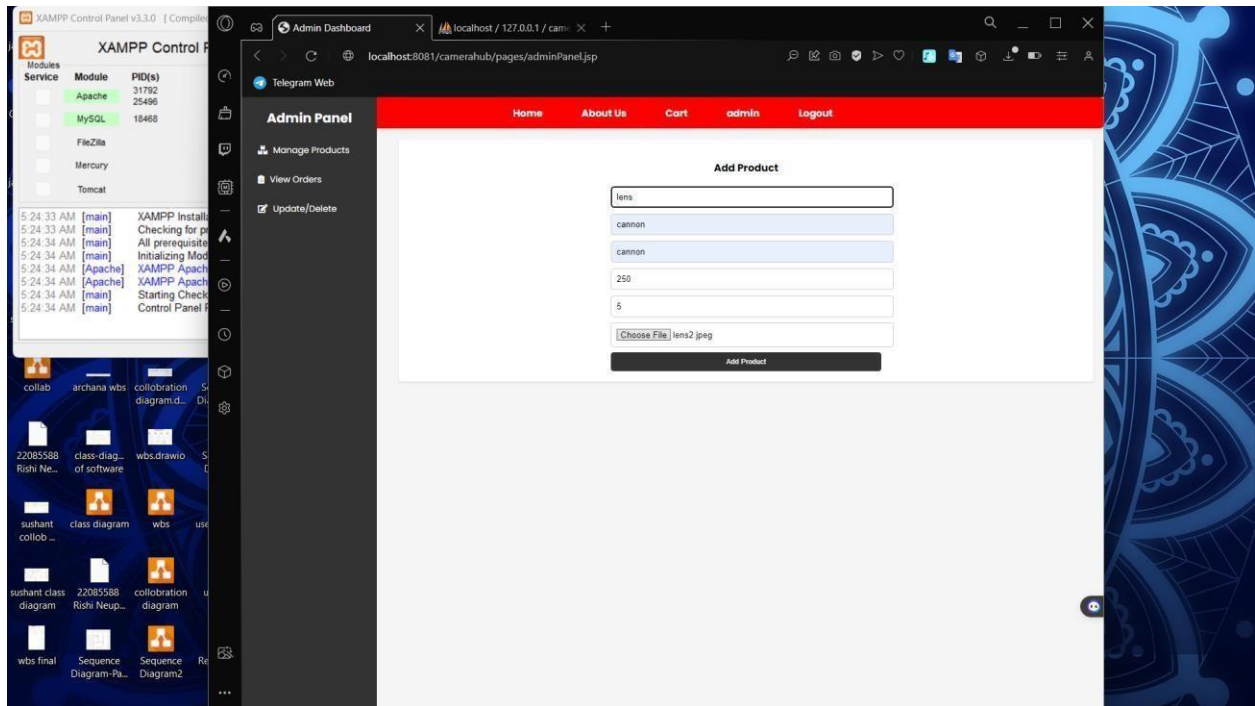


Figure 36 Add Product

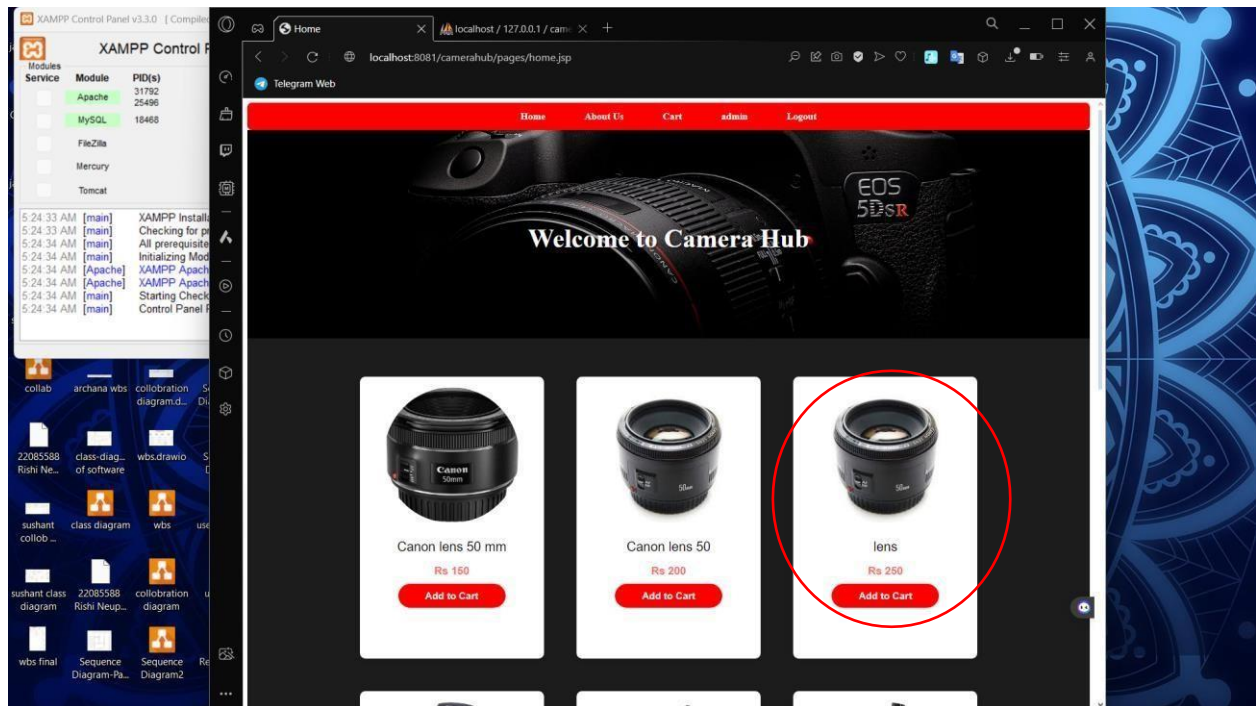


Figure 37 Product display

## 5.10 Test case 10

### 5.10.1 To edit the product:

<b>Objective</b>	To enable admin, edit products to the website.
<b>Action</b>	The product that needs to be edited should have the edit button pressed by the admin.
<b>Expected Result</b>	Both the product's database entry and its webpage should be updated.
<b>Actual Result</b>	Each product is edited on the website as well as database.
<b>Conclusion</b>	The test was successfully.

Table 15 Editing Product

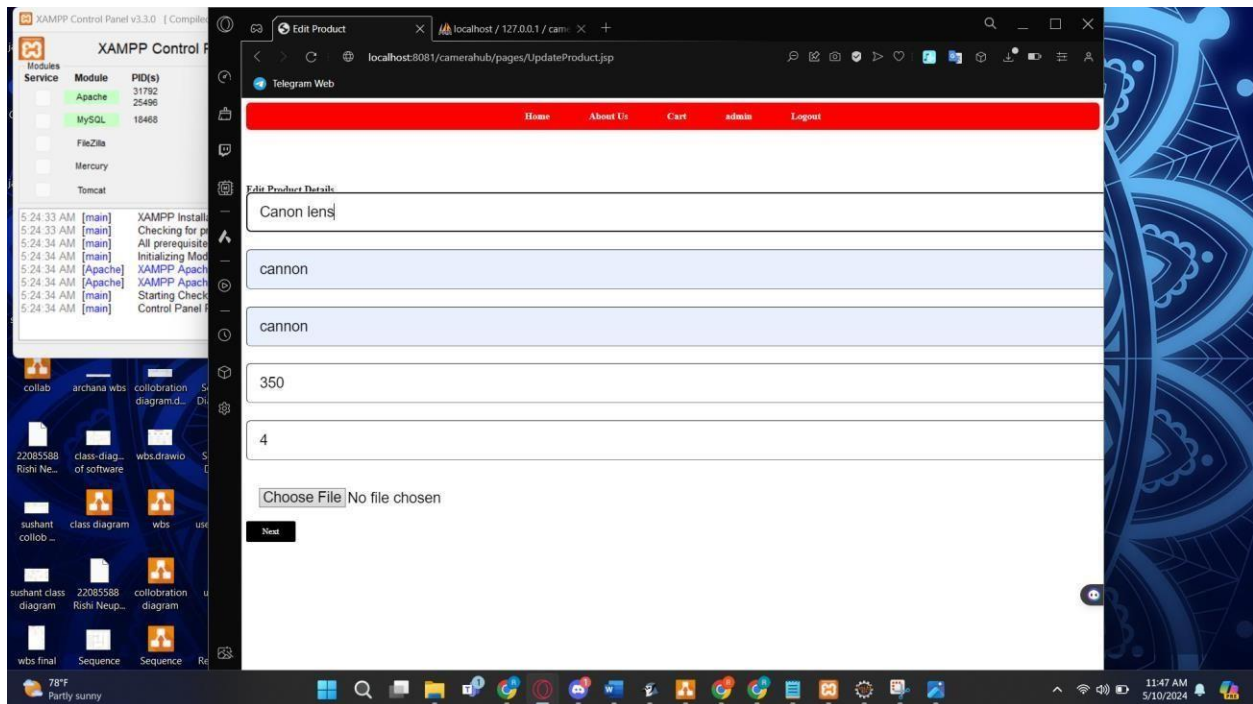


Figure 38 Product Description

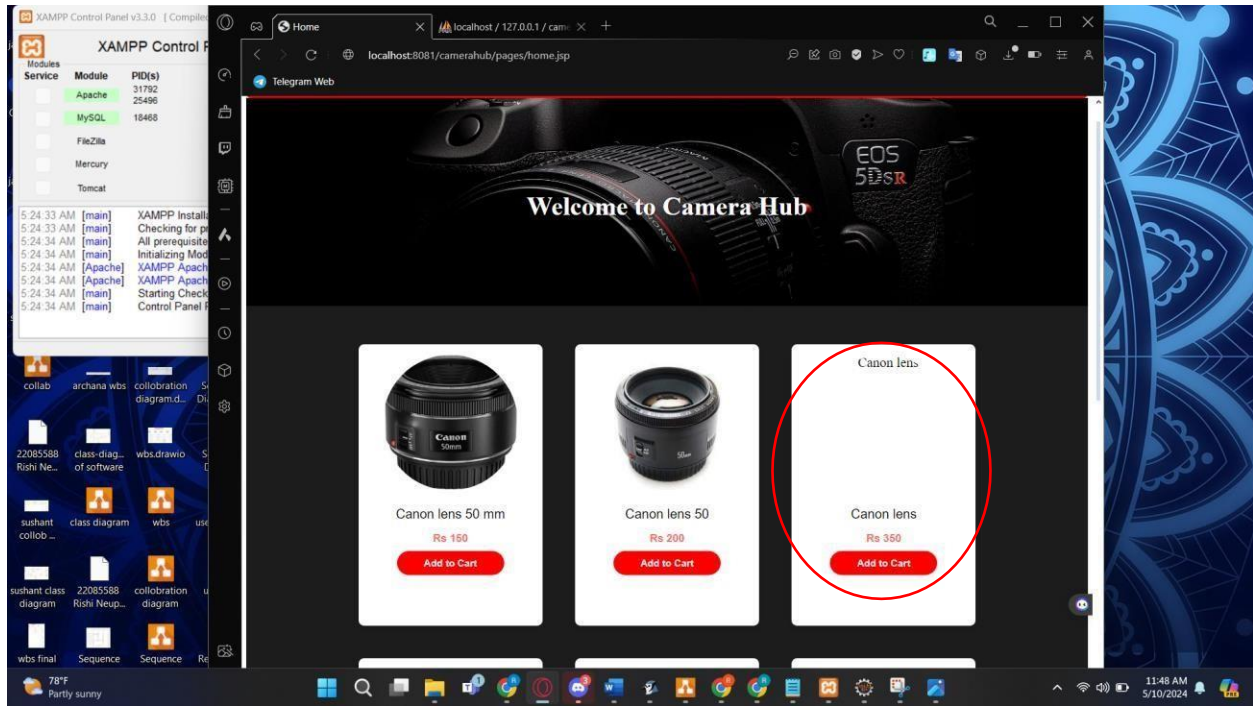


Figure 39 Product Added.

## 5.11 Test case 11

### 5.11.1 To delete product:

<b>Objective</b>	To enable admin, delete products to the website.
<b>Action</b>	The product's admin should click the delete button. to be removed.
<b>Expected Result</b>	The product should be deleted on the database as well as in the database.
<b>Actual Result</b>	Each product is deleted on the website as well as database.
<b>Conclusion</b>	The test was successfully.

Table 16 Deleting Product

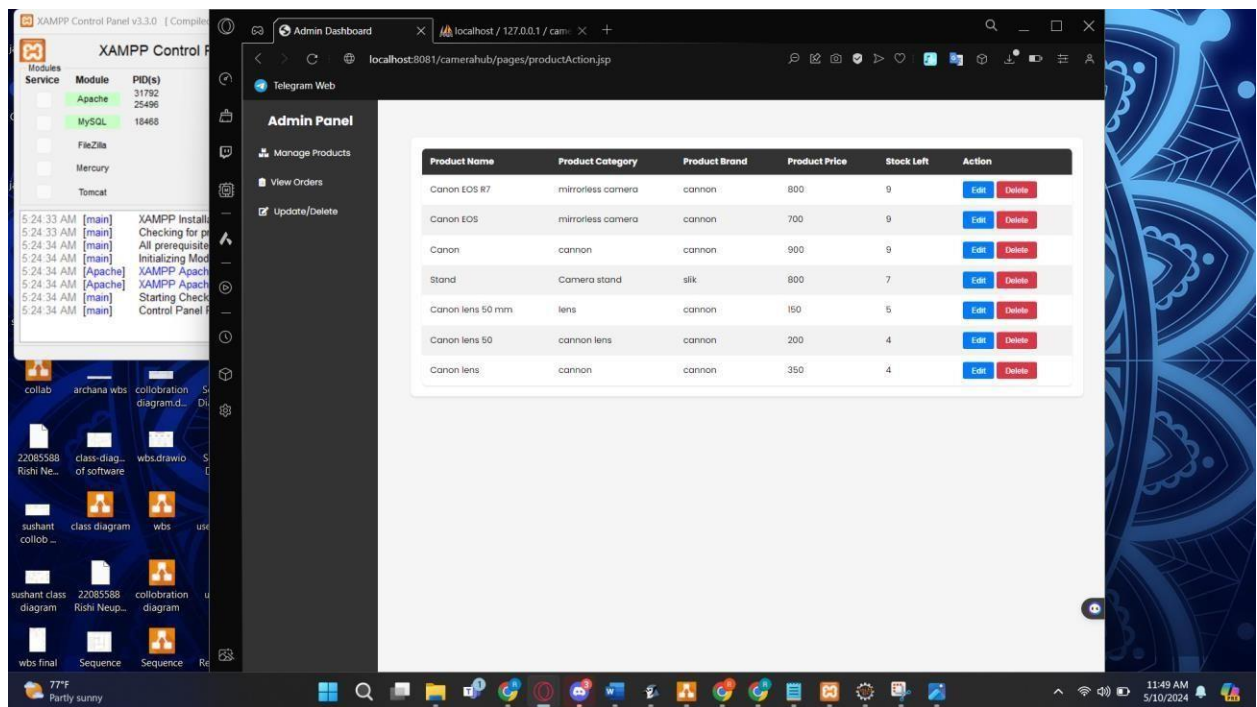


Figure 40 Delete Product



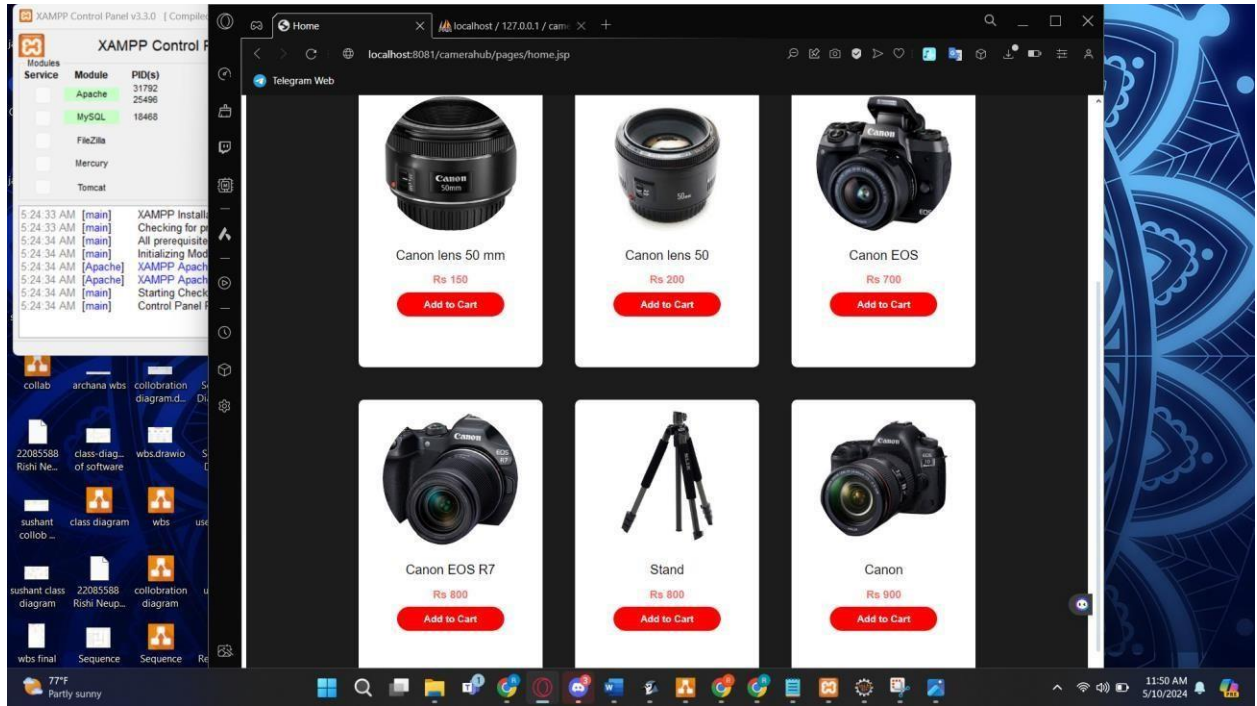


Figure 41 To Delete Product

## 6. Tools and Libraries Used

### 6.1 Tools

- o **Eclipse:** Eclipse is a programming language integrated development environment (IDE) that supports Java as well as other languages including C, C++, PHP, Ruby, and so forth. The Eclipse Java development tools (JDT) for Java, Eclipse CDT for C/C++, and Eclipse PDT for PHP are just a few of the development environments that Eclipse offers. (tutorialspoint, 2024)



*Figure 42 Eclipse (Linux Adictos, 2024)*

The reason Eclipse IDE is used is that it makes a variety of different parts referred to as plug-ins. Many sophisticated features for developing Java-based applications are included in the Eclipse IDE. In contrast to alternative IDEs, the eclipse IDE's Because the pre-packaged bundle is available for download, users can install Eclipse for programming purposes straight away, making the installation process rather straightforward.

The Eclipse Integrated Development Environment (IDE) is utilized in this project due to its various functionalities that facilitate the development process, such as code editing, debugging, and project management. The web application is run on the Apache Tomcat server, which is accessed through Eclipse (IDE) for both coding and managing Java code.

o **Xampp:** One popular cross-platform web server that aids developers in developing and testing their applications on a local web server is XAMPP. It was created by Apache Friends, and viewers are free to edit or modify the native source code. It is made up of the MariaDB database, the Apache HTTP Server, and interpreters for various programming languages, including PHP and Perl. It is compatible with several operating systems, including Linux and macOS x64 and Windows IA-32 packages, and is offered in eleven different languages. (Javatpoint, 2024)



Figure 43 Xampp (Javatpoint)

For this project, the developer builds a local server on his PC using XAMPP to host and test the e-commerce website. The website can be viewed by going to localhost in a web browser and accessing the files stored in XAMPP. The database for the website is similarly built with the required tables and relationships and is set up and configured with XAMPP. XAMPP makes it easy and convenient to set up a development environment for an e-commerce website.

## 6.2 Libraries

There are various types of libraries used for this project some of them are described below:

Libraries	Functions
java.sql	This package is a part of the Java Standard Library, which is needed to connect to databases and manipulate data.
java.util	In addition to utilities, it offers date and time management, algorithms, and commonly used data structures.
Java.io	This includes classes and interfaces that allow for memory mapped I/O, which is used to handle I/O tasks more efficiently, and nonblocking I/O.
java.crypt	Many classes and interfaces are available in this package for use in different cryptographic techniques, including message authentication, encryption, decryption, and key generation.
javax.servlet	This package includes classes and interfaces that describe the parts and features required to build Java web applications.
JSTL	This library provides typical sets of identifies to perform tasks, which helps to speed up creating JSP pages.

java.security	The various classes and interfaces required for adding security features and services to applications are provided by this package.
---------------	---

*Table 17 Libraries and Function*

## 7. Development Process

We began this project by figuring out the technologies and frameworks we would use and by carefully analyzing its needs. With a clear image in hand, we used wireframes to build the user interface and lay out the interactions between the various sections. We added the Tomcat Server and required external jar files to our Eclipse IDE environment before we started coding.

Five tables (Admin, Customer, Product, Order, and New User) were established in XAMPP to help us handle our data efficiently. For every element of our website, these tables would hold pertinent data. Our code was organized into model, view, and controller packages in accordance with the MVC paradigm, which allowed for a clear division of responsibilities.

Following the MVC architecture, we integrated essential features like login and registration while considering user roles like admins and customers. Encrypting passwords improved security and safeguarded user privacy. We used login sessions to safely store authentication information after a successful login.

Every page, including the admin panel and homepage, was carefully developed using the MVC design principles. We made sure that consumers had to log in or register before they could make purchases from the homepage. We carried out extensive testing during the development phase to find and fix any problems that came up.

After building the front-end pages with HTML and CSS, we concentrated on integrating backend features to go along with each feature. Every page was organized and designed with its own purpose in mind, resulting in a unified and easily navigable interface. Finally, using Java for backend operations and HTML and CSS for the interface, we were able to create a completely functional e-commerce website.

Name	Role
------	------

Kisitiz Giri	<p>Report Preparation: Conclusion, User Interface Design, Libraries.</p> <p>System Development:</p> <ul style="list-style-type: none"> <li>• Responsible for the creation of the user interface design, ensuring end users a smooth and simple experience.</li> <li>• Managing the database connection to ensure smooth data retrieval and storage, as well as integrating necessary libraries into the system to improve speed.</li> <li>• Handled the report's conclusion section, which summarized the most important conclusions and revelations discovered during the development process.</li> </ul>
Rishi Neupane	<p>Report Preparation: Test Cases, Class Diagram, Method Description, Development Process, Actual Design.</p> <p>System Development:</p> <ul style="list-style-type: none"> <li>• Detailed test cases that include a range of situations and edge cases were developed to verify the system's durability and reliability.</li> <li>• A comprehensive class diagram was made, giving a graphical representation of the relationships and system structure, including database entities and their interactions.</li> <li>• Methods, such as database query methods and data manipulation techniques, are completely described, with comprehensive descriptions of their purpose and functionality within the system.</li> </ul>
Sangita Belbase	<p>Report Preparation: Wireframe, Tools and aims.</p> <p>System Development:</p> <ul style="list-style-type: none"> <li>• Designed detailed wireframes that describe the system's visual design and flow and help team members communicate more effectively.</li> <li>• Investigated a range of technologies and techniques to determine which ones would best meet the needs and goals of the project.</li> <li>• Contributed to the report's aims section, which outlined the project's objectives and goals and made sure they were in line with stakeholders' expectations.</li> </ul>
Shreyas Bhandari	<p>Report Preparation: Introduction, Aims and Critical Analysis. System Development:</p>

	<ul style="list-style-type: none"> <li>• Created the report's opening, which gave a brief overview of the project's goals, objectives, and methodology.</li> <li>• Examined the project's advantages, disadvantages, possibilities, and threats to make an informed conclusion.</li> <li>• Helped define the project's goals, making sure they were accurate and in line with the broader goals.</li> </ul>
Ashim Poudel	<p>Report Preparation: Wireframe, Objective and User Interface.</p> <p>System Development:</p> <ul style="list-style-type: none"> <li>• Working together to create wireframes, which helped to speed up the development process by providing a visual representation of the system's functionality and layout.</li> <li>• Made sure stakeholders understood the project's goals by providing a detailed explanation of them in the report.</li> <li>• Had an important impact on the user interface's design, with a focus on improving accessibility and usability for a range of user groups.</li> </ul>

*Table 18 Roles of Member*

## 8. Critical Analysis

When building this website, we followed all instructions and specifications to the letter.

Modular and well-organized programming was made possible by the Model-View-Controller (MVC) architectural pattern. We used a variety of Java Servlet APIs and technologies, including Tag Libraries, JDBC, and JSTL, which are common web development tools, throughout the project.

We encountered several difficulties when putting our selected strategies into practice, some of which are outlined below:

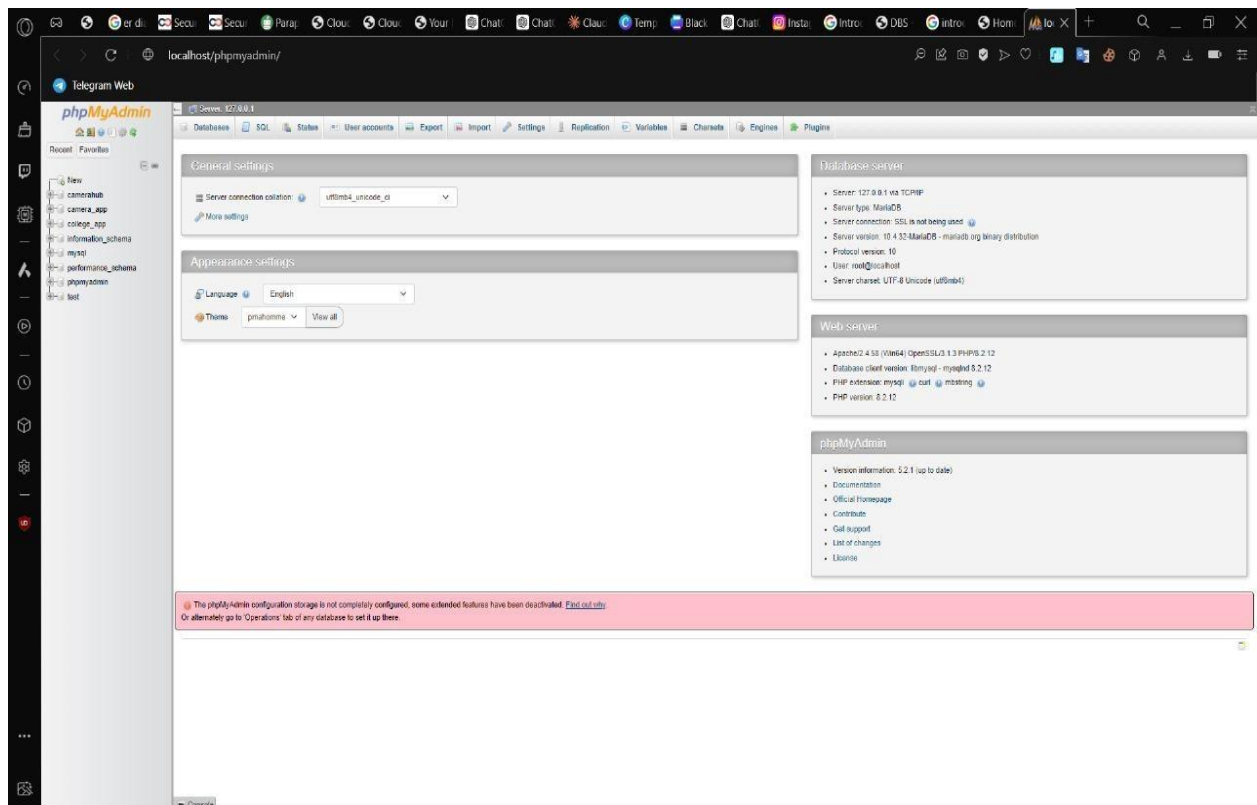
Implementing the MVC architecture: Using the MVC pattern was one of our project's main goals. This required a major problem in successfully separating the controller, view, and model components. It took careful attention to test within the MVC pattern to make sure that every separate part worked as intended and blended in perfectly with the others.

Designing a user-friendly interface: It was difficult to build a website that satisfied our users' requirements and preferences while adding necessary features. Our goal was to

create a platform that would be easy to navigate and have an easy-to-understand intuitive user experience. One of the biggest challenges we faced was finding a balance between functional efficiency and aesthetic appeal.

**Database Management:** There were challenges in managing the database, especially in creating and keeping connections. We had trouble making sure that our code and the database communicated with each other smoothly. Nevertheless, we were able to successfully fix these problems by carefully comparing the table and column names in our code with those in the database.

## Evidence of Database



## **9. Conclusion:**

We have successfully created CameraHub, an e-commerce platform that specializes in cameras and camera gear, for our class. We carefully developed a strong website that reflects our dedication to web development excellence using the Eclipse IDE and several libraries. We easily integrated managing databases with necessary features like user and admin management, login, registration, product management, and order processing by adhering to the MVC architecture. With a clean UI and simple navigation, CameraHub puts an emphasis on a smooth user experience while guaranteeing security, effectiveness, and user-friendliness. In overall, CameraHub is evidence of our dedication to provide a state-of-the-art e-commerce platform that caters to the demands of those who love photography while providing a smooth and fulfilling purchasing experience.



## 10. References

Alana, 2024. *CareerFoundry*. [Online]

Available at: <https://careerfoundry.com/en/tutorials/ui-design-for-beginners/what-is-user-interface-design>

Bhumika, 2024. *GeeksforGeeks*. [Online]

Available at: <https://www.geeksforgeeks.org/unified-modeling-language-uml-class-diagrams/>

Javatpoint, 2024. *Javatpoint*. [Online]

Available at: <https://www.javatpoint.com/xampp>

Lewis, J. R., n.d. *ResearchGate*. [Online] Available

at:

[https://www.researchgate.net/publication/373487143\\_USABILITY\\_AND\\_USER\\_EXPERIENCE\\_DESIGN\\_AND\\_EVALUATION](https://www.researchgate.net/publication/373487143_USABILITY_AND_USER_EXPERIENCE_DESIGN_AND_EVALUATION)

[Accessed 2024].

tutorialspoint, 2024. *tutorialspoint*. [Online]

Available at: <https://www.tutorialspoint.com/eclipse/index.htm>

White, M. J., 2024. *Springboard*. [Online]

Available at: <https://www.springboard.com/blog/design/what-is-wireframe>