



CS5054NT: Advanced Programming and Technologies

50% Group Coursework

2023 spring

Group Name:			
SN	Student Name	College ID	University ID
	Binaya Thapa	NP05CP4A210031	21041073
	Anisha Acharya	NP05CP4A210033	21041068
	Ekeshta Bohora	NP05CP4A210022	21041078
	Selisha Rai	NP05CP4A210013	21041095

Assignment Due Date: Tuesday, March 21, 2023

Assignment Submission Date: Monday, May 8, 2023

Project File Links:

Google Drive Link:	https://drive.google.com/file/d/1j_FSEVS_FLlqYnYqyZG3om9sHr0MNXkN/view?usp=share_link
---------------------------	---

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

Table of Contents

1. Introduction	1
1.1 Aims	2
1.2 Objective	2
2. User Interface Design	2
2.1 Wireframe.....	2
2.2 Actual design.....	9
3. Class Diagram	15
4. Method Description.....	15
5. Test Cases	24
6. Tools and libraries used.....	44
6.1. Tools Used.....	44
6.2. Libraries Used.....	46
7. Development Process.....	47
8. Critical Analysis	49
9. Conclusion.....	51
Bibliography	52

List of Figures

Figure 1: Wireframe for user registration.....	3
Figure 2: Wireframe for user login.....	3
Figure 3: Wireframe for admin login	4
Figure 4:Edit User Profile	4
Figure 5:Wireframe for home page.....	5
Figure 6: Wireframe for order list.....	5
Figure 7: Wireframe to view user profile.....	6
Figure 8: Wireframe for editing user profile	6
Figure 9: Wireframe for add product.....	7
Figure 10: Wireframe for product list	7
Figure 11: Wireframe for add product.....	8
Figure 12: Wireframe for view your cart	8
Figure 13: Wireframe for page not found.....	9
Figure 14: Actual design of registration page	9
Figure 15: Actual design of login page	10
Figure 16: Actual design of home page.....	10
Figure 17: Actual Design of user profile	11
Figure 18: Actual Design to edit user profile.....	11
Figure 19: Actual Design of your cart page	12
Figure 20: Actual Design for admin login.....	12
Figure 21: Actual design of product page.....	13
Figure 22: Actual design to add product.....	13
Figure 23: Actual design for order page	14
Figure 24: Actual design of page not found.....	14

List of Tables

Table 1: Table for Test 1	24
Table 2: Table for Test 2	26
Table 3: Table for Test 3	27
Table 4: Table for Test 4	29
Table 5: Table for Test 5	31
Table 6: Table for Test 6	33
Table 7: Table for Test 7	35
Table 8: Table for Test 8	36
Table 9: Table for Test 9	38
Table 10: Table for Test 10	39
Table 11: Table for Test 11	41
Table 12: Table for Test 12	43

1. Introduction

On moving forward in the race of technology an e-commerce websites platform can be the better option for promoting and advancing the traditional cultural stores (mailchimp.com, 2023). So, advancement of physical store to develop a project of e-commerce website is our project and the name of our project website is “ESBA collection”. This project deals for selling and promoting Clothes product by using Java programming language and have three main tasks.

The project's goal is to establish a user login system as well as an admin panel. Users must register with an image, and both users and administrators can log in using encrypted passwords. Administrators have unique access to the admin panel, which allows them to conduct tasks such as adding, changing, and removing product information, checking the product list, and handling client orders. The site displays detailed product information, such as photos, pricing, and stock levels. Users can add goods to their shopping carts and search for products based on category, price, and rating.

Users must register to utilize the "add to cart" feature. Users that are logged in may modify their accounts, change their passwords, search for products, and view their purchased items and cart. Orders are placed by adding products to the cart, and sessions for logged-in and logged-out users are monitored. For a pleasant user experience, the project closes by implementing validation, exception handling, and correcting small issues.

1.1 Aims

- To develop an e-commerce website for selling different clothes using Model-View-Controller (MVC) pattern.
- To offer wide selection of high-quality products.
- To understand the MVC(Model View Controller) pattern for developing website.
- To understand about the efficient management tool for administrator.

1.2 Objective

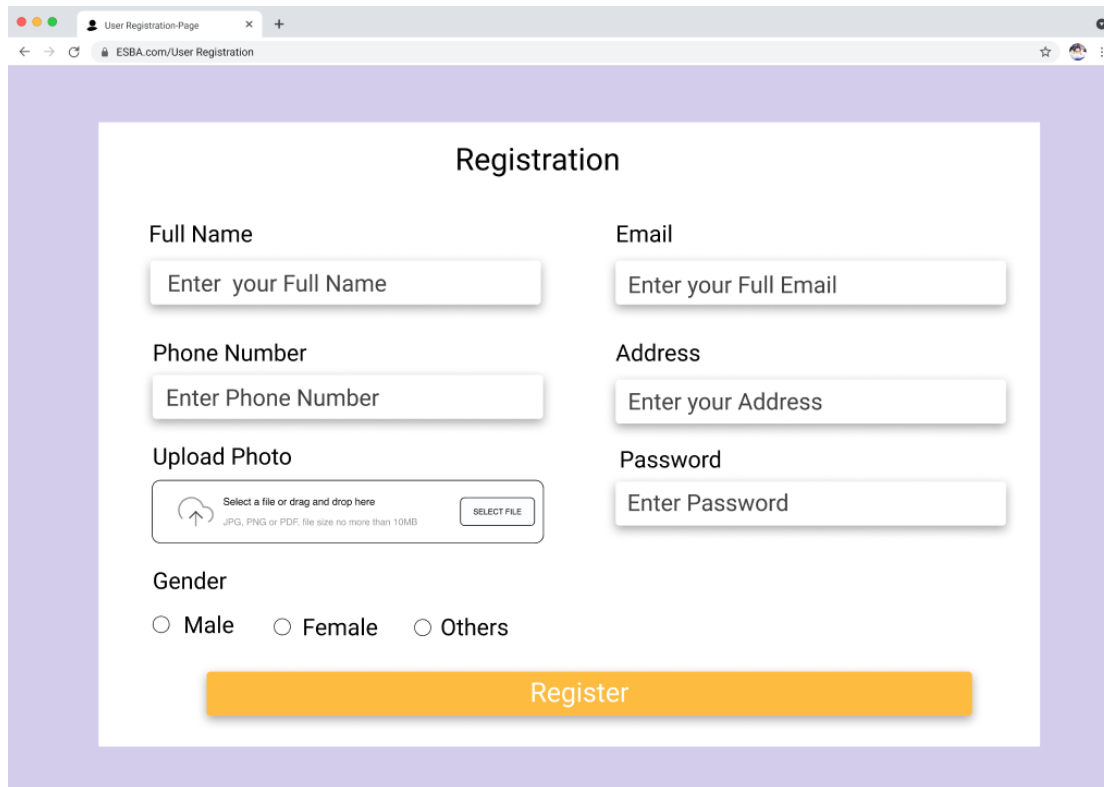
- To develop a user-friendly interface for providing secure payment options and offer easy navigation for customers.
- To improve customer engagement, we intend to actively participate and communicate with customers to provide an improved customer experience.
- To increase the profitability and financial success by increasing your sales and revenue.
- To improve operational efficiency.

2. User Interface Design

User Interface (UI) design is a method of creativity that focuses on designing a digital product's visual appeal, interactive components, and overall user experience. The UI designer systematically develops screens and UI elements, taking into variables such as layout, colour palettes, typography, and user interactions to produce an understandable and visually appealing interface. The design then comes to life through wireframes and prototypes that allow for testing and modifying. (www.uxdesigninstitute.com, 2023).

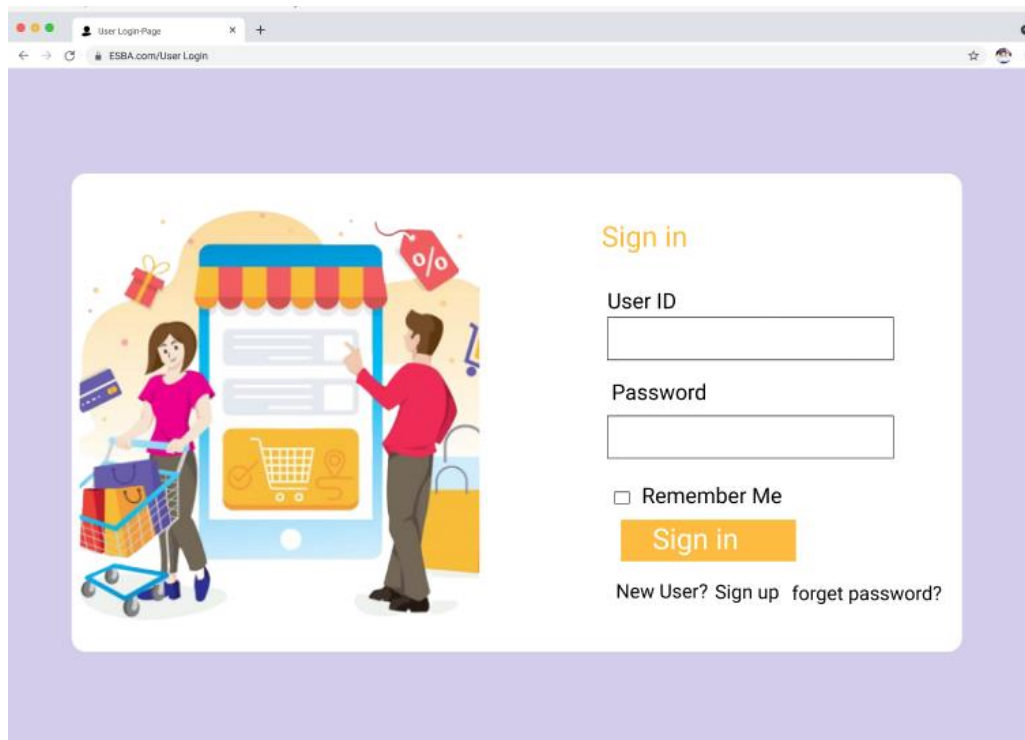
2.1 Wireframe

Wireframe is a rough drawing or blueprint of a web page, mobile app, or software interface. It provides a simplified graphic representation of the final product's layout and structure, focusing on key aspects and functions also specifies the positioning of various components like buttons, menus, and content sections but it excludes detailed design aspects such as colours and graphics (visme.co, 2023).



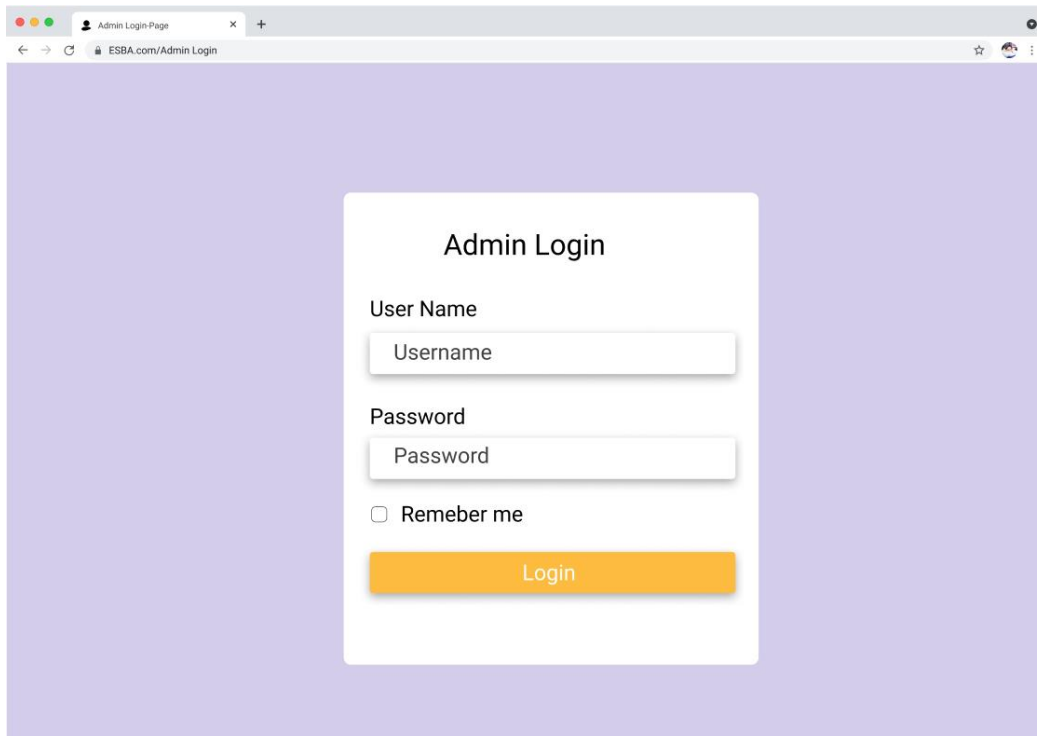
The image shows a web browser window with the title "User Registration-Page" and the URL "ESBA.com/User Registration". The page has a light purple background. In the center, there is a white box titled "Registration". Inside this box, there are several form fields: "Full Name" with a text input field containing the placeholder "Enter your Full Name"; "Email" with a text input field containing the placeholder "Enter your Full Email"; "Phone Number" with a text input field containing the placeholder "Enter Phone Number"; "Address" with a text input field containing the placeholder "Enter your Address"; "Upload Photo" with a file upload area that says "Select a file or drag and drop here" and "JPG, PNG or PDF, file size no more than 10MB", and a "SELECT FILE" button; "Password" with a text input field containing the placeholder "Enter Password"; and "Gender" with three radio buttons labeled "Male", "Female", and "Others". At the bottom of the white box is a large orange button labeled "Register".

Figure 1: Wireframe for user registration



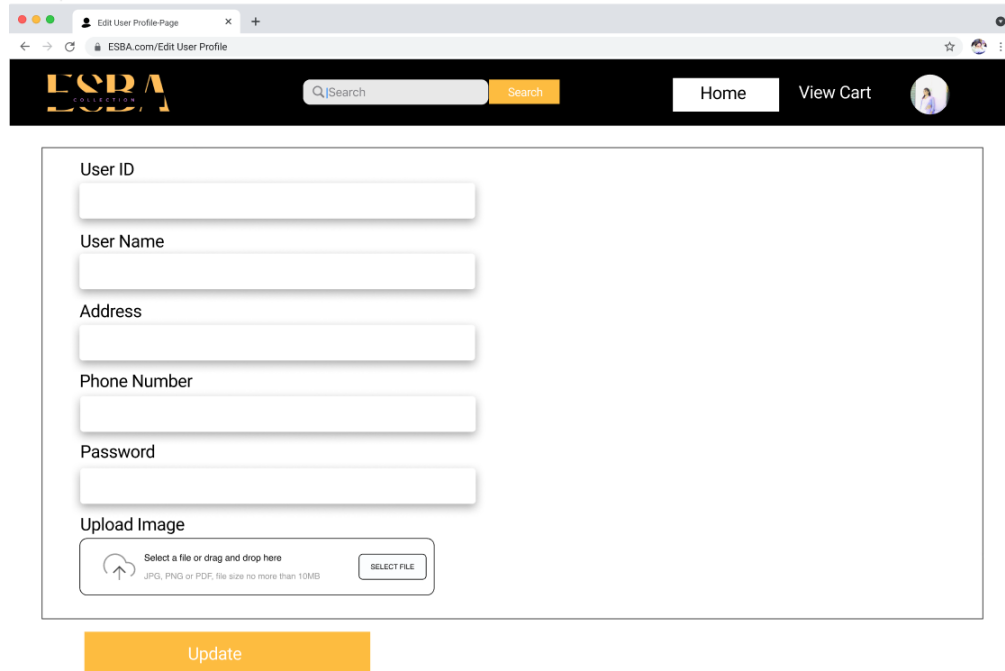
The image shows a web browser window with the title "User Login-Page" and the URL "ESBA.com/User Login". The page has a light purple background. In the center, there is a white box. On the left side of the white box is an illustration of a woman pushing a shopping cart and a man pointing at a large smartphone screen that displays a shopping interface. On the right side of the white box, there is a "Sign in" section. It includes a "Sign in" heading, a "User ID" label with a text input field, a "Password" label with a text input field, a "Remember Me" checkbox, an orange "Sign in" button, and a link "New User? Sign up forget password?".

Figure 2: Wireframe for user login



A wireframe of an admin login page. The page has a light purple background. In the center, there is a white rectangular box with a shadow. Inside this box, the text "Admin Login" is centered at the top. Below it, there are two labels: "User Name" and "Password". Each label is followed by a white input field with a shadow. The "User Name" input field contains the text "Username". Below the "Password" input field, there is a checkbox followed by the text "Remeber me". At the bottom of the white box, there is an orange button with the text "Login".

Figure 3: Wireframe for admin login



A wireframe of an edit user profile page. The page has a black header bar. On the left of the header is the "ESBA" logo. In the center is a search bar with the placeholder text "Search" and an orange "Search" button. On the right are links for "Home" and "View Cart", and a user profile icon. Below the header, there is a white rectangular box with a shadow. Inside this box, there are five labels: "User ID", "User Name", "Address", "Phone Number", and "Password". Each label is followed by a white input field with a shadow. Below the "Password" input field, there is a section for "Upload Image". This section contains a text input field with the placeholder text "Select a file or drag and drop here", a small icon of an upload arrow, and a "SELECT FILE" button. Below the white box, there is an orange button with the text "Update".

Figure 4:Edit User Profile

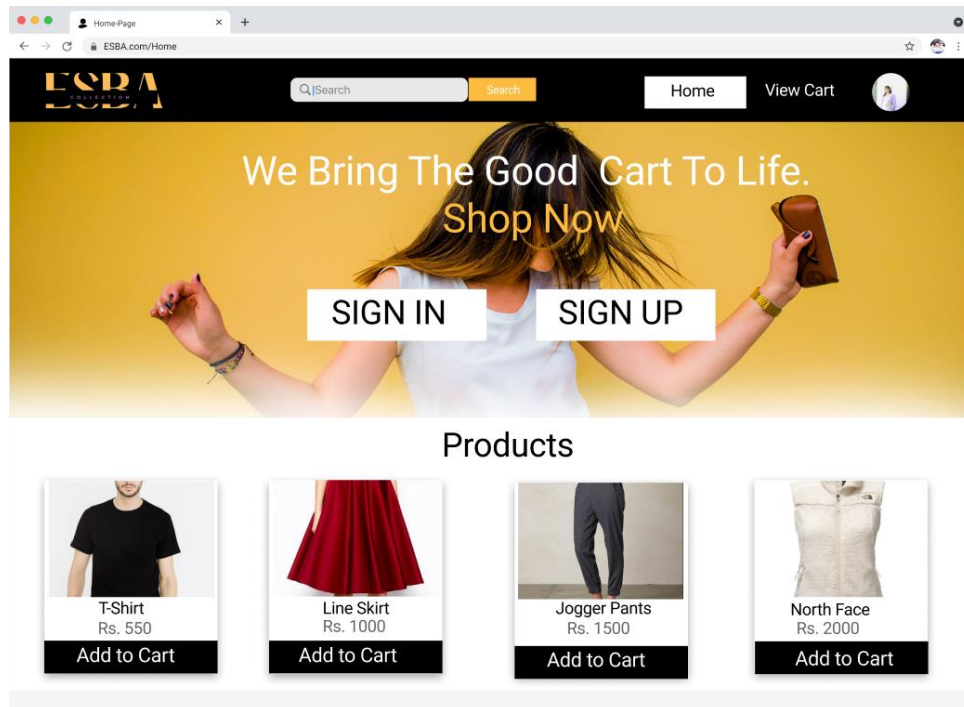


Figure 5: Wireframe for home page

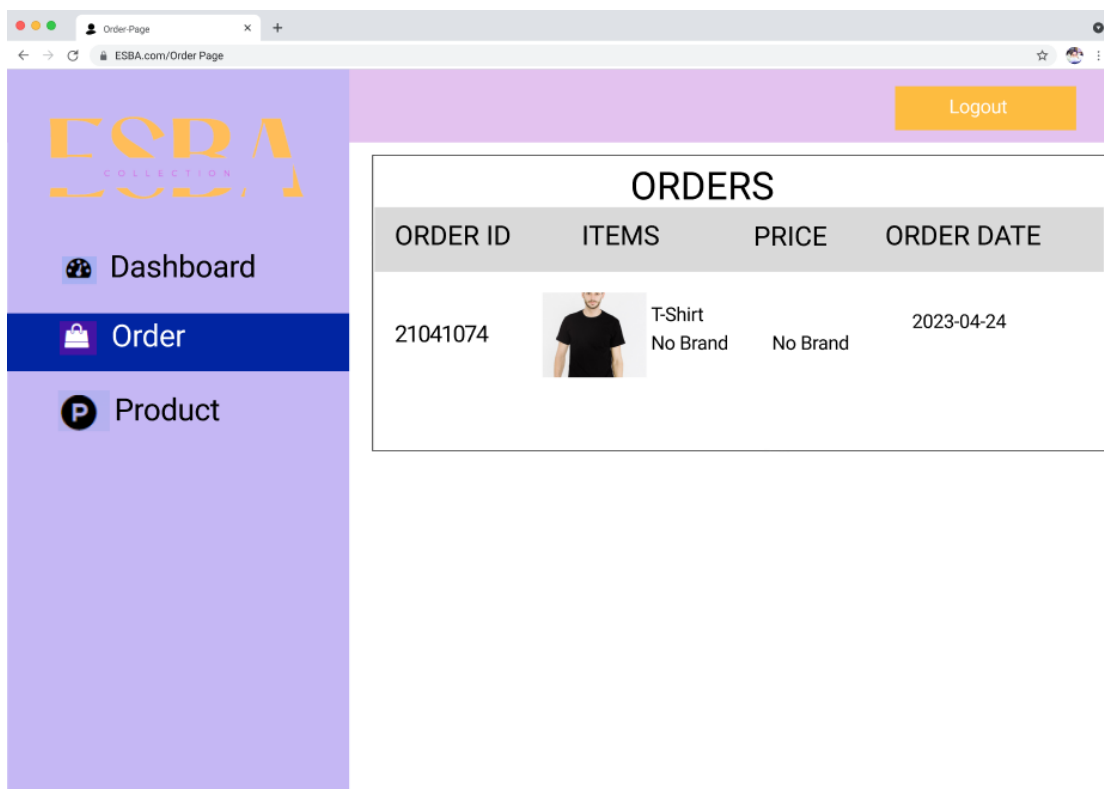


Figure 6: Wireframe for order list

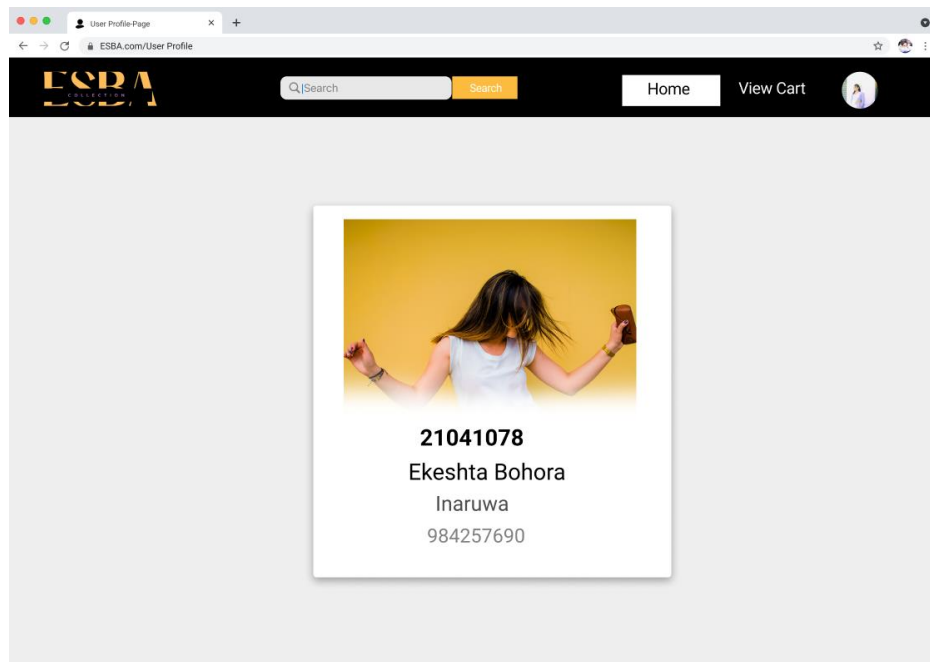


Figure 7: Wireframe to view user profile

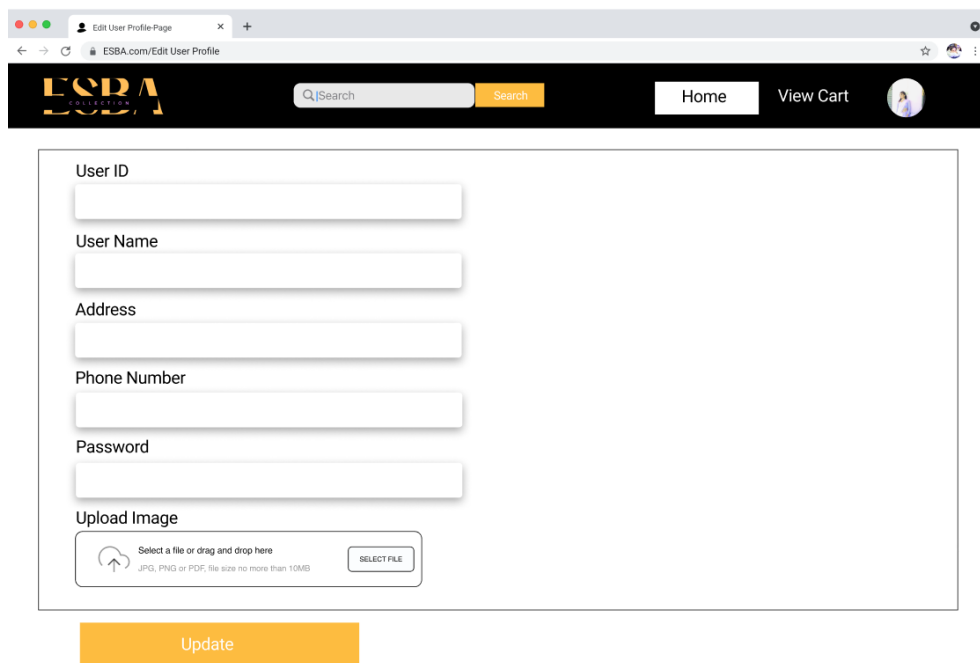


Figure 8: Wireframe for editing user profile

Product ID

Product Name

Price

Brand

Upload Image

Select a file or drag and drop here
JPG, PNG or PDF, file size no more than 10MB

SELECT FILE

Add Product

Figure 9: Wireframe for add product

Logout

Add Product

Products

Product ID	Image	Product Name	Price	Brand	Action
21041074		T-Shirt	Rs. 1500	No Brand	Order Delete
21041074		Jogger Pants	Rs. 1050	No Brand	Order Delete

Figure 10: Wireframe for product list

Product ID

Product Name

Price

Brand

Upload Image

Select a file or drag and drop here
JPG, PNG or PDF, file size no more than 10MB

SELECT FILE

Update

Figure 11: Wireframe for add product

Your Cart

Items	Price	Quantity	Total Price		
T-shirt No brand	Rs.1300	2	Rs.2600	Order Now	Delete
Skirt No brand	Rs.1200	1	Rs.1200	Order Now	Delete

Figure 12: Wireframe for view your cart

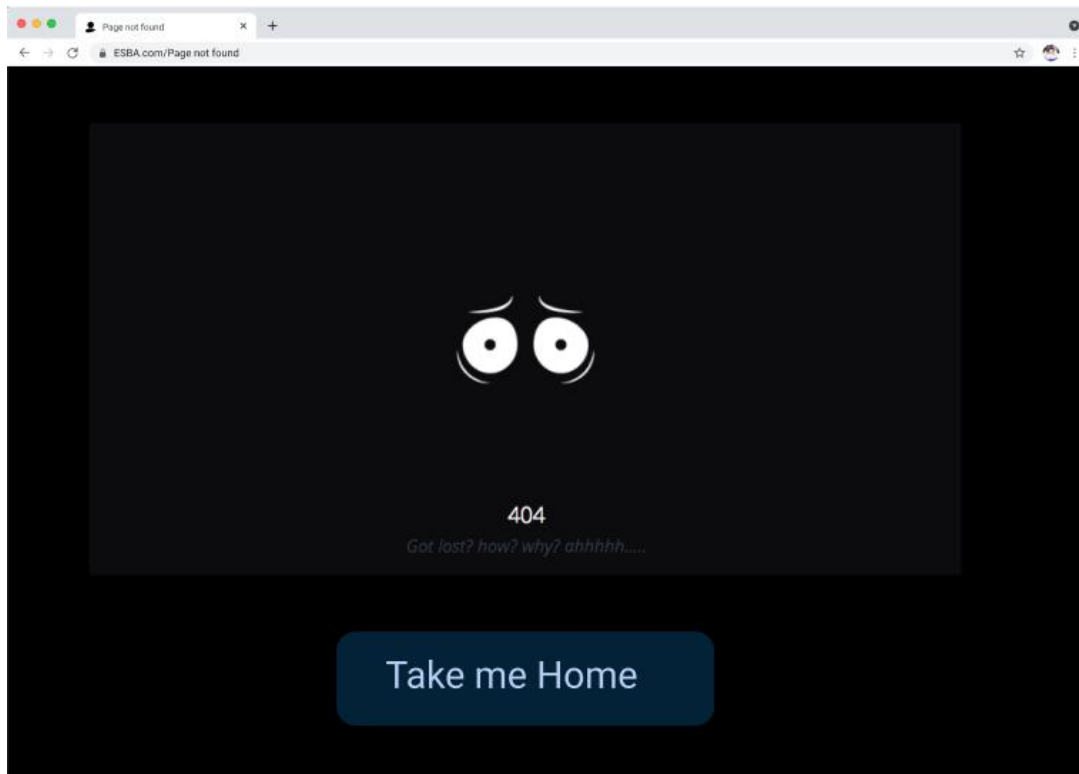


Figure 13: Wireframe for page not found

2.2 Actual design

A screenshot of a registration page. The page has a light purple background. In the center, there is a white rectangular form with rounded corners. The form is titled 'Registration' in bold black text. Below the title, there are six input fields arranged in three rows and two columns. The first row contains 'Full Name' and 'Email'. The second row contains 'Phone Number' and 'Address'. The third row contains 'Upload Photo' and 'Password'. The 'Upload Photo' field has a 'Choose File' button and the text 'No file chosen'. The 'Password' field has the text 'Enter Password'. Below the input fields, there is a 'Gender' section with three radio buttons labeled 'Male', 'Female', and 'Other'. At the bottom of the form, there is a yellow rectangular button with the text 'Register' in black.

Figure 14: Actual design of registration page

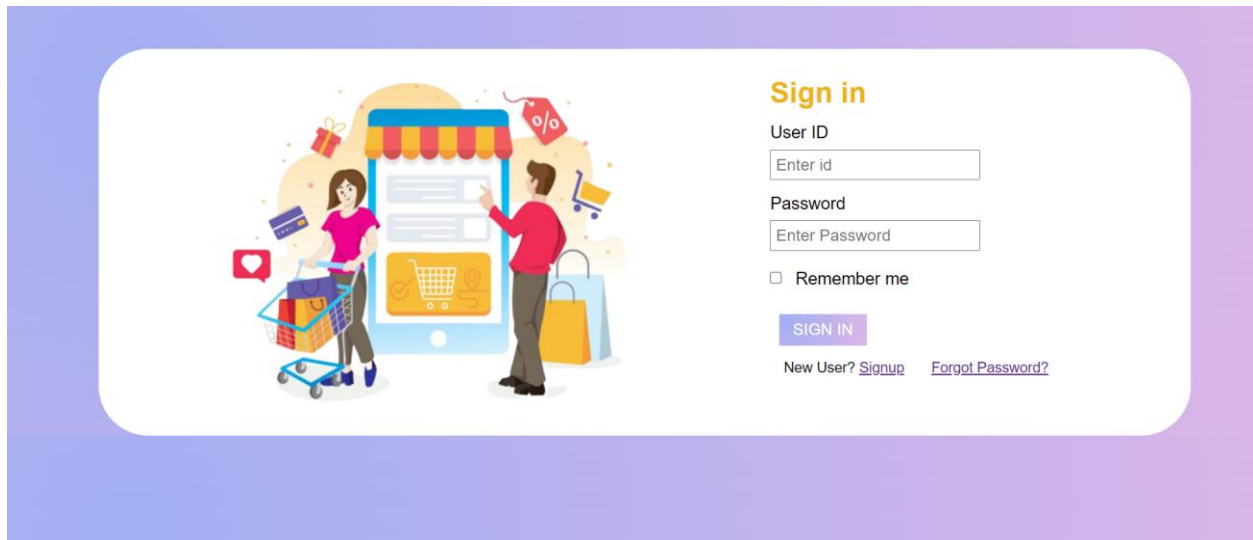


Figure 15: Actual design of login page

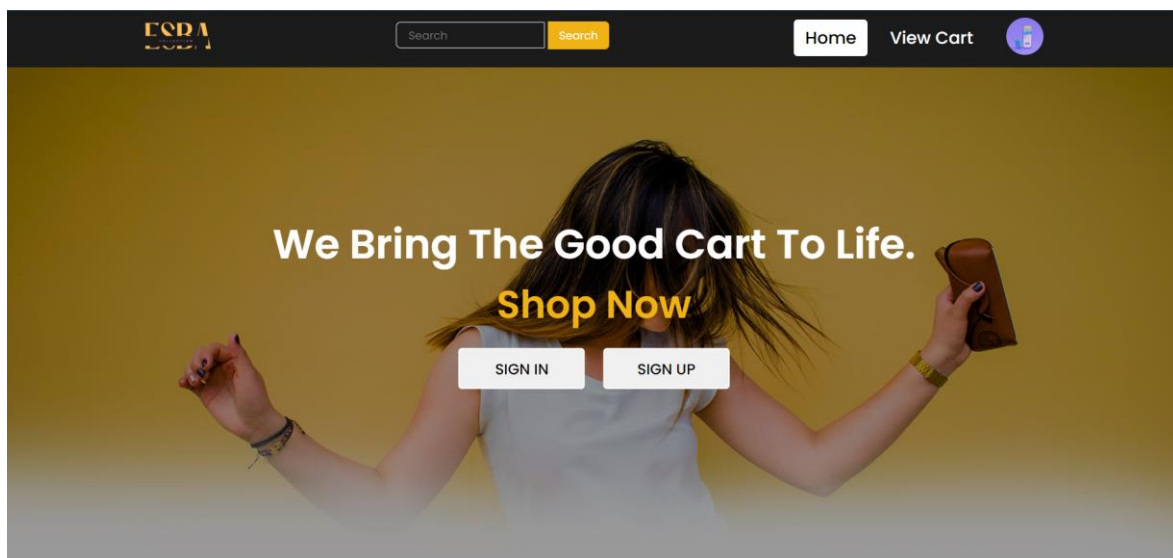


Figure 16: Actual design of home page

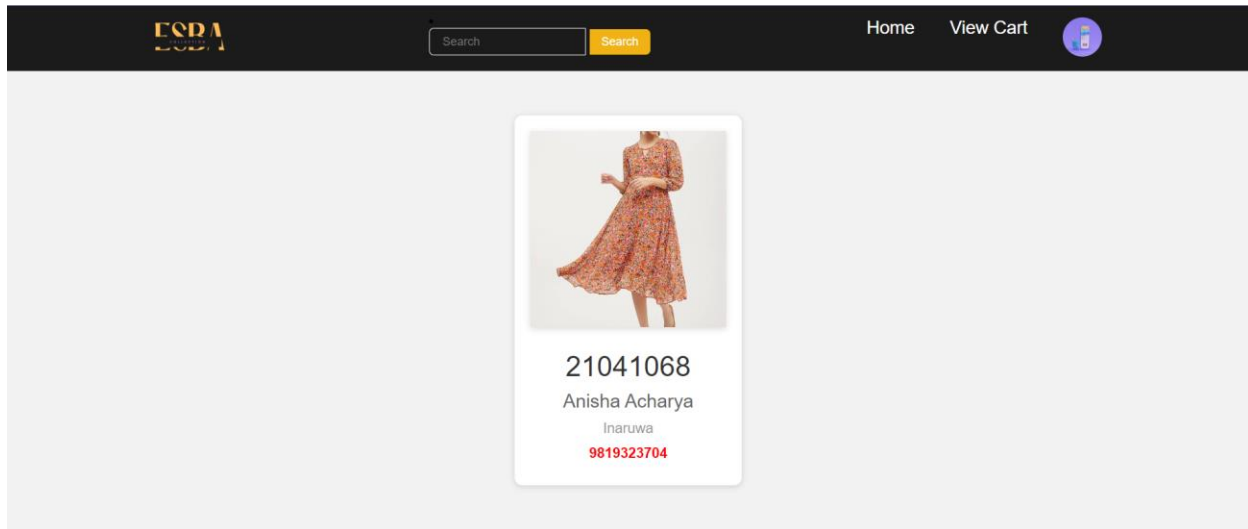


Figure 17: Actual Design of user profile

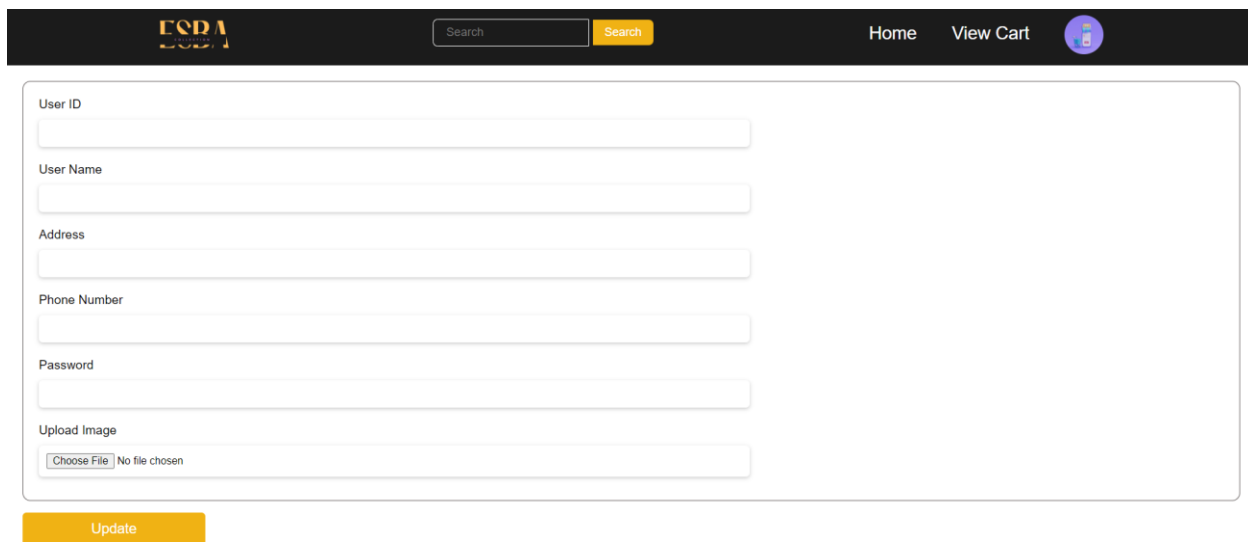
The image shows a form for editing a user profile on a dark-themed website. The form is white and contains several input fields: 'User ID', 'User Name', 'Address', 'Phone Number', and 'Password'. Below these is an 'Upload Image' section with a 'Choose File' button and the text 'No file chosen'. An 'Update' button is located at the bottom of the form. The website header is dark with the 'ESDA' logo, a search bar, and links for 'Home', 'View Cart', and a user icon.

Figure 18: Actual Design to edit user profile

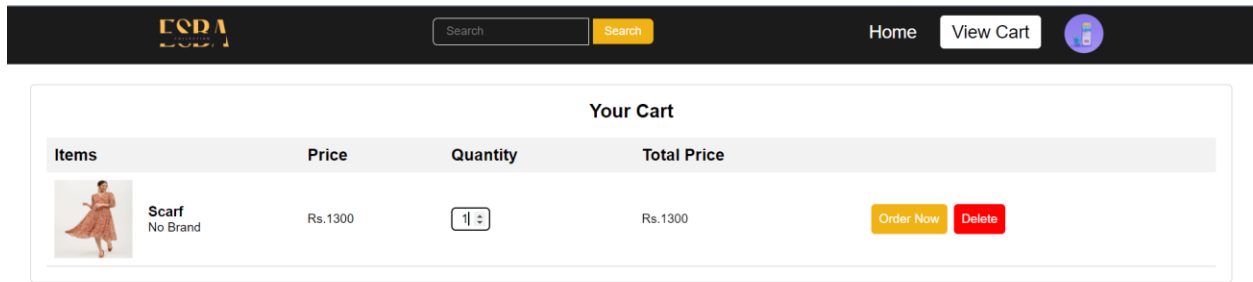


Figure 19: Actual Design of your cart page

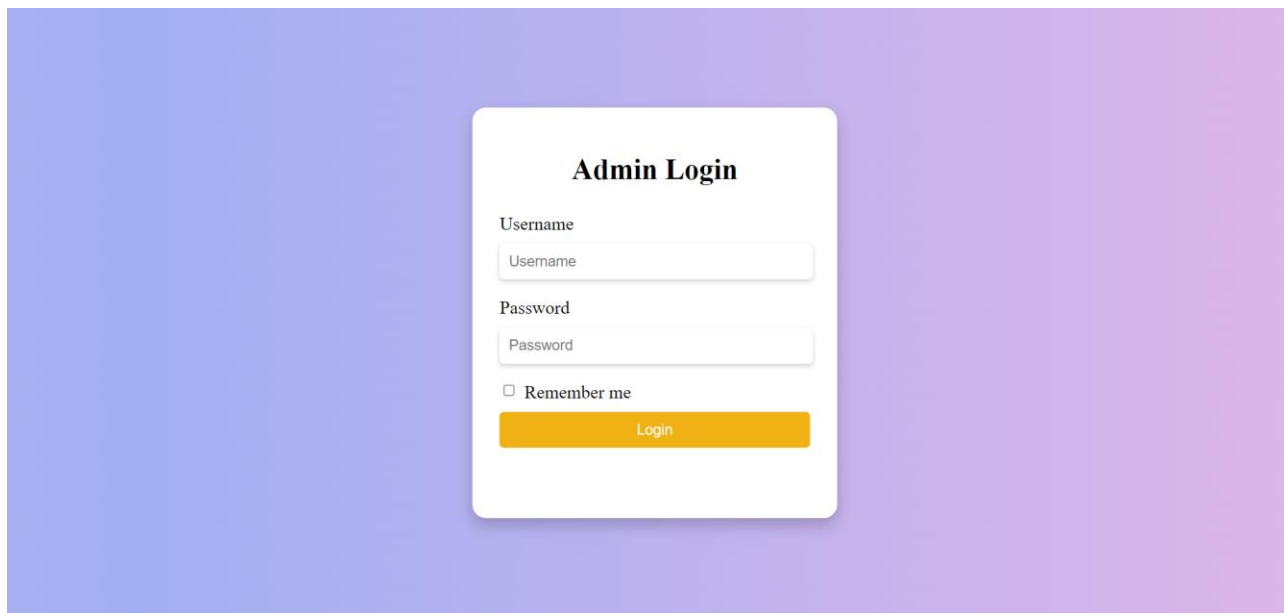
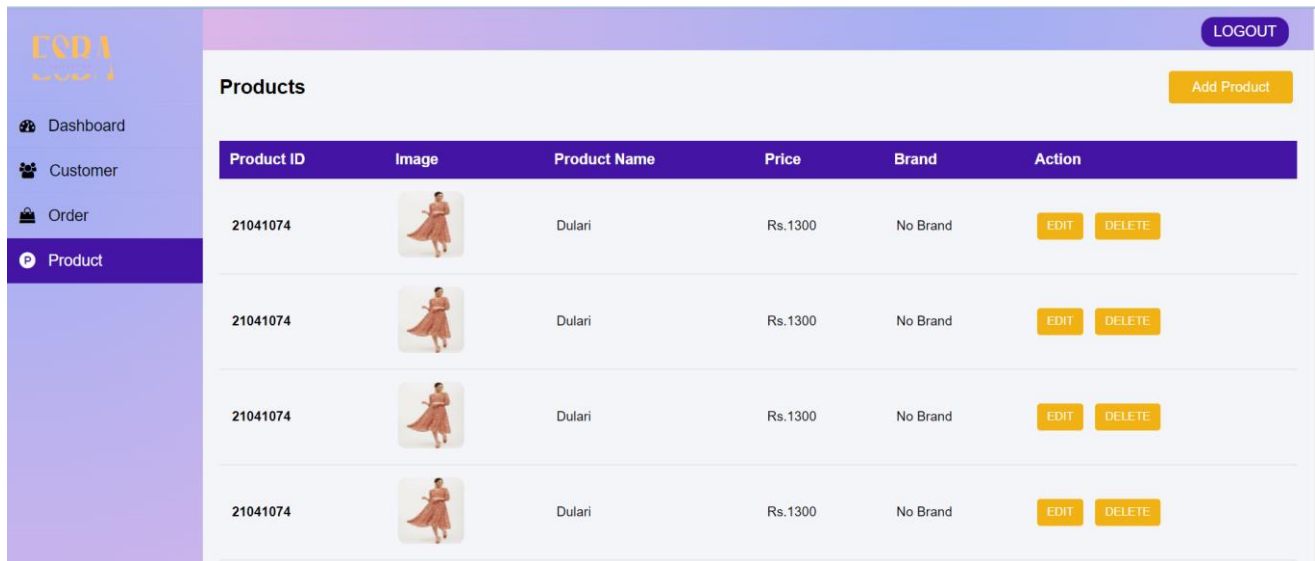


Figure 20: Actual Design for admin login







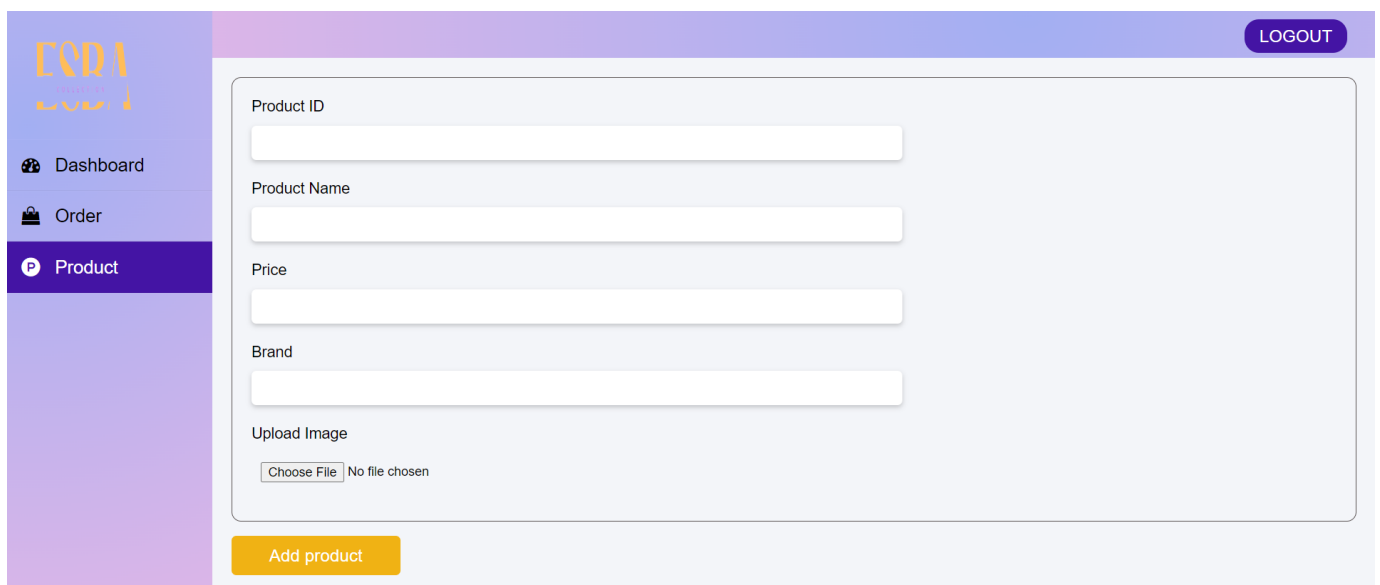
Product ID	Image	Product Name	Price	Brand	Action
21041074		Dulari	Rs.1300	No Brand	EDIT DELETE
21041074		Dulari	Rs.1300	No Brand	EDIT DELETE
21041074		Dulari	Rs.1300	No Brand	EDIT DELETE
21041074		Dulari	Rs.1300	No Brand	EDIT DELETE

Figure 21: Actual design of product page



Product ID

Product Name

Price

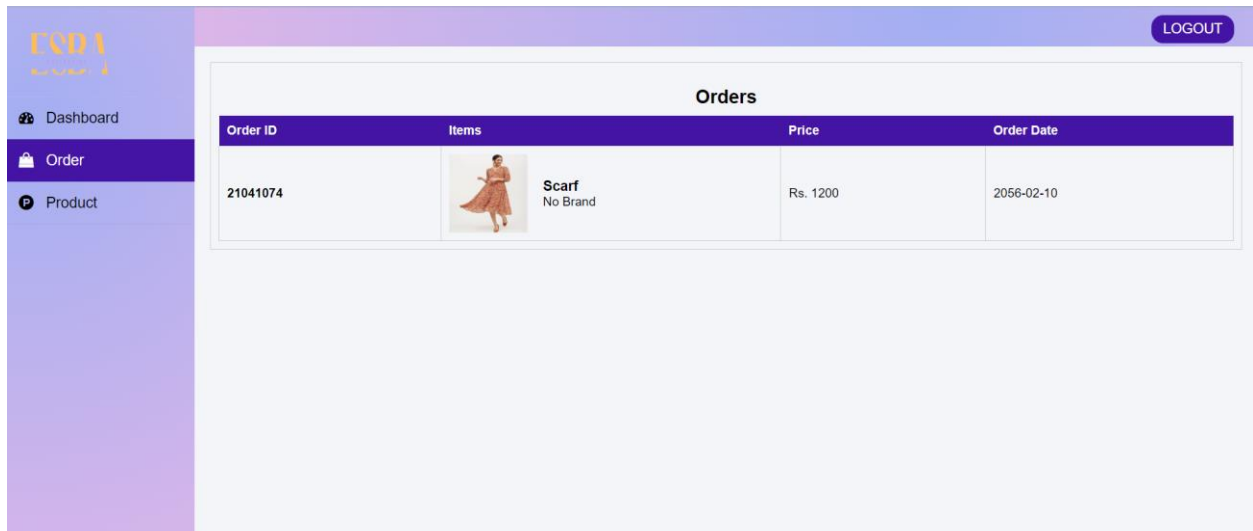
Brand

Upload Image

[Choose File](#) No file chosen

[Add product](#)

Figure 22: Actual design to add product




Orders			
Order ID	Items	Price	Order Date
21041074	 Scarf No Brand	Rs. 1200	2056-02-10

Figure 23: Actual design for order page

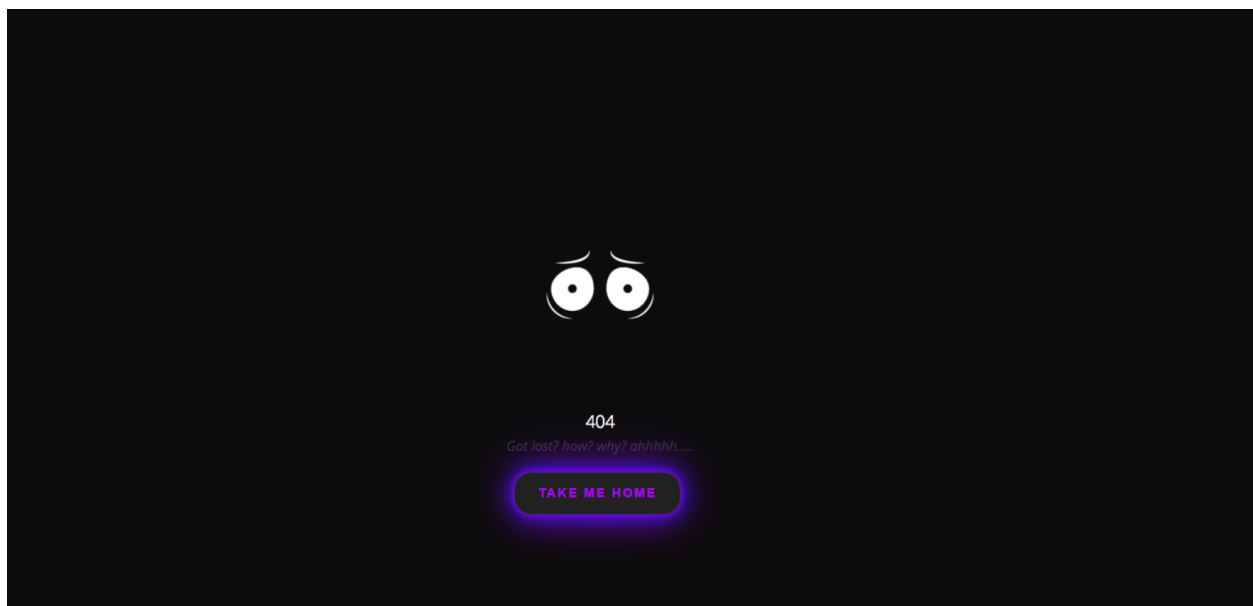


Figure 24: Actual design of page not found

3. Class Diagram

Class diagram is a visual representation that displays the relationships and structure of classes. It is useful in businesses because it shows the links between classes in a business application. It is especially useful for communicating class structures, which indicate how classes are ordered systematically, and class collaborations, which show how classes collaborate to achieve specific functionality (venngage.com, 2023).

4. Method Description

4.1. Method description of AddProduct class

The created method is **service**. It manages HTTP requests and responds as an overridden method from the HttpServlet class. This method receives the parameters from the request object, generates a new Product object, adds the product to the database, sets a message in the session object, and, if the product was added successfully, saves the product image to the file system.

4.2. Method description of AddToCart class

The created method is service. It manages HTTP requests and responses and is an overridden method from the HttpServlet class. This method creates a new Cart object, adds the cart to the database, sets a message in the session object, gets the userId from the session object, retrieves the productId from the request object, retrieves the userId from the request object, and redirects the user to the homepage. Finally, it prints a message to the console.

4.3. Method description of AdminLogin class

The **service()** method handles the HTTP POST request for admin login. It retrieves the email and password parameters, validates them using the UserDAO class, and sets the admin ID as a session attribute if successful. Otherwise, it sets an error message and redirects the user to the login page.

4.4. Method description of DeleteCartItem class

The **service()** method in the DeleteCartItem class manages an HTTP POST request to delete an item from the user's cart. It retrieves the productId parameter, deletes the item, sets a message in the session object, and redirects the user to the viewCart page. Finally, it prints the message to the console.

4.5. Method description of ShowProducts class

The created method is **service** in the ShowProducts class. It handles an HTTP GET request sent by the user to show all available products. It fetches all the products using the fetchProducts() method from the UserDao class, sets the list of products as an attribute in the request object, and forwards the request to the Homepage.jsp page.

4.6. Method description of UserLogin class

The UserLogin class has a **service()** method that handles user login requests. It validates the user's credentials, sets session attributes for a successful login, and redirects to the homepage. If the login fails, it sets an error message and redirects to the user login page. A console message is printed at the end.

4.7. Method description of UserLogout class

This method **service()** in the UserLogin class handles a user login request by validating their email and password, setting a session attribute for the user ID if validation is successful, redirecting to the homepage, and setting an error message and redirecting to the login page if validation fails. It also logs a message to the console.

4.8. Method description of UserRegistration class

The created method is **service** in the UserRegistration class. It handles an HTTP POST request sent by the user to register as a new user. It retrieves user information from the request, encrypts the password, creates a new User object, and calls the registerUser() method from the UserDao class to store the user data in the database. It also uploads a user photo, saves it to a file, and prints a message to the console.

4.9. Method description of ViewCart

The ViewCart class has a **service** method that retrieves the user's ID from the session, fetches the products in their cart using the UserDAO class, and displays them for viewing. The method also redirects the user to the view cart page.

4.10. Method description of LoginFilter class

The **doFilter** method in the LoginFilter class checks if the user is logged in by verifying the "isLoggedIn" session attribute. If not, it redirects the user to the login page and sets an error message as a session attribute. The method also prints a message to the console.

4.11. Method description of AESEncryption class

The created class AESEncryption provides methods for encrypting and decrypting strings using the AES algorithm. The methods use a secret key, defined as a string constant, to encrypt and decrypt the input strings. The encrypt method takes a plaintext string as input and returns its encrypted version as a Base64-encoded string. The decrypt method takes an encrypted string as input and returns its decrypted version as a plaintext string.

4.12. Method description of Cart class

The Cart class has the following methods:

- **Cart(String userId, String productId):** It is a constructor that takes two parameters, userId and productId, and initializes the corresponding instance variables with these values.
- **getUserId():** It is a getter method that returns the value of the userId instance variable.
- **setUserId(String userId):** It is a setter method that sets the value of the userId instance variable to the provided value.
- **getProductId():** It is a getter method that returns the value of the productId instance variable.

- **setProductId(String productId):** It is a setter method that sets the value of the productId instance variable to the provided value.

4.13. Method description of Order class

The Order class has the following methods:

- **Order(String orderId, String productId, String userId, String orderDate, String totalPrice):** It is a constructor that sets the values of the orderId, productId, userId, orderDate, and totalPrice instance variables.
- **getOrderId():** It is a getter method that returns the value of the orderId instance variable.
- **setOrderId(String orderId):** It is a setter method that sets the value of the orderId instance variable.
- **getProductId():** It is a getter method that returns the value of the productId instance variable.
- **setProductId(String productId):** It is a setter method that sets the value of the productId instance variable.
- **getUserId():** It is a getter method that returns the value of the userId instance variable.
- **setUserId(String userId):** It is a setter method that sets the value of the userId instance variable.
- **getOrderDate():** It is a getter method that returns the value of the orderDate instance variable.
- **setOrderDate(String orderDate):** It is a setter method that sets the value of the orderDate instance variable.
- **getTotalPrice():** It is a getter method that returns the value of the totalPrice instance variable.
- **setTotalPrice(String totalPrice):** It is a setter method that sets the value of the totalPrice instance variable.

4.14. Method description of Product class

The Product class has the following methods:

- **Product(String productId, String productName, String productPrice, String productBrand, String productImage)** – It is a constructor method that takes in 5 parameters and initializes the corresponding instance variables.
- **getProductId():** It is a getter method that returns the value of productId instance variable.
- **setProductId(String productId):** It is a setter method that sets the value of productId instance variable.
- **getProductName():** It is a getter method that returns the value of productName instance variable.
- **setProductName(String productName):** It is a setter method that sets the value of productName instance variable.
- **getProductPrice():** It is a getter method that returns the value of productPrice instance variable.
- **setProductPrice(String productPrice):** It is a setter method that sets the value of productPrice instance variable.
- **getProductBrand():** It is a getter method that returns the value of productBrand instance variable.
- **setProductBrand(String productBrand):** It is a setter method that sets the value of productBrand instance variable.
- **getProductImage():** It is a getter method that returns the value of productImage instance variable.
- **setProductImage(String productImage):** It is a setter method that sets the value of productImage instance variable.

4.15. Method description of User class

The User class has the following methods:

- **User(String fullName, String email, String phoneNum, String address, String photoPath, String password, String gender):** This is the constructor of the class. It takes the user's full name, email address, phone number, address, photo path, password, and gender as input parameters and initializes the instance variables of the class with these values.

- **getFullName():** It is a getter method that returns the full name of the user.
- **setFullName(String fullName):** It is a setter method that sets the full name of the user.
- **getEmail():** It is a getter method that returns the email address of the user.
- **setEmail(String email):** It is a setter method that sets the email address of the user.
- **getPhoneNum():** It is a getter method that returns the phone number of the user.
- **setPhoneNum(String phoneNum):** It is a setter method that sets the phone number of the user.
- **getAddress():** It is a getter method that returns the address of the user.
- **setAddress(String address):** It is a setter method that sets the address of the user.
- **getPhotoPath():** It is a getter method that returns the photo path of the user.
- **setPhotoPath(String photoPath):** It is a setter method that sets the photo path of the user.
- **getPassword():** It is a getter method that returns the password of the user.
- **setPassword(String password):** It is a setter method that sets the password of the user.
- **getGender():** It is a getter method that returns the gender of the user.
- **setGender(String gender):** It is a setter method that sets the gender of the user.

4.16. Method description of UserDao class

The UserDao class has the following methods:

- **getConnection():** It is a method that establishes a connection to the MySQL database.
- **registerUser():** It is a method that inserts user data into the database.
- **validateLogin():** It is a method that checks whether the user's login credentials are valid.
- **fetchUserDetails():** It is a method that retrieves the user's details from the database based on the user's email.
- **validateAdminLogin():** It is a method that checks whether the admin's login credentials are valid.

- **addProduct():** It is a method that inserts a new product into the database.
- **fetchProducts():** It is a method that retrieves all product information from the database.
- **addToCart():** It is a method that inserts product data into the cart table in the database.

4.17. Method description of AdminLogout class

The **service()** method is a key method in the HttpServlet class that is responsible for processing an HTTP request and generating an HTTP response. It takes two arguments, a HttpServletRequest object and a HttpServletResponse object, and it is called by the container for each incoming HTTP request.

4.18. Method description of DeleteProduct class

The **service()** method of the DeleteProduct servlet retrieves the productId parameter from the HttpServletRequest object and calls the deleteProduct() method of the UserDao class to delete the product from the database. The deleteStatus attribute of the HttpSession object is set based on the success of the deletion operation. Finally, the user is redirected to the viewProductList page using the HttpServletResponse object.

4.19. Method description of EditProduct class

The service() method of the EditProduct servlet retrieves the productId parameter from the HttpServletRequest object and fetches the corresponding product from the database using an instance of the UserDao class. The Product object is then set as an attribute of the request, and the request is forwarded to the "EditProduct.jsp" page for display and editing.

4.20. Method description of EditProfile class

The service() method of the EditProfile servlet retrieves the loggedId attribute from the HttpSession object and fetches the corresponding user details from the database using an instance of the UserDao class. The User object is then set as an attribute of the session, and the request is forwarded to the "EditUser.jsp" page for display and editing.

4.21. Method description of MakeOrder class

The MakeOrder servlet's service() method retrieves the logged-in user's ID and the selected product ID from the request parameters. It then creates a new order object with the current date and time, and attempts to add the order to the database using the makeOrder() method of the UserDAO class. If successful, the corresponding product is removed from the user's cart, and they are redirected to their cart page. The success or error message is set as a session attribute.

4.22. Method description of ProductList class

The service() method of the ProductList servlet fetches a list of all products using the fetchProducts() method of the UserDAO class and sets it as an attribute of the request. The method then forwards the request to the ProductList.jsp view to display the list of products.

4.23. Method description of ShowOrders class

The service() method of the ShowOrders servlet retrieves the order details from the database using the fetchOrderDetails() method of the UserDAO class, and stores them in an ArrayList of OrderDetails objects. This ArrayList is set as an attribute of the request, and the request is forwarded to the OrderList.jsp view for rendering. This view will display the order details in a table format.

4.24. Method description of UpdateProduct class

The service() method of the UpdateProduct servlet retrieves the product details from the request parameters, creates a new Product object with the updated details, and calls the updateProduct() method of the UserDAO class with the Product object as an argument. The method attempts to update the product in the database and returns a success or error message, which is set as an attribute of the session. If a new image is uploaded, the servlet saves the image to the specified directory using the Part object and the photoPath context parameter. Finally, the servlet redirects the user to the product list page.

4.25. Method description of UpdateUser class

The UpdateUser servlet handles requests to update user details. The **service()** method retrieves and encrypts the updated user details from the request parameters, but does not implement the actual updating of the user details in the database.

4.26. Method description of ViewProfile class

The **service()** method retrieves the user ID from the session attribute, fetches the user details using the UserDao class, and sets the user details as a session attribute. The method then forwards the request and response to the ViewProfile.jsp file to display the user details.

4.27. Method description of AdminLoginFilter

It uses the doFilter() method to intercept the request and check the session object for the loggedAdminId attribute. If the attribute is present, the request is allowed to pass through to the requested URL. If not, the request is redirected to the admin login page and an error message is set in the session attribute. The init() and destroy() methods are also implemented.

4.28. Method description of OrderDetails class

The OrderDetails class has the following methods:

- **OrderDetails(String orderId, String userImage, String fullName, String productImage, String productName, String productBrand, String prodctPrice, String orderDate, String orderTime):** It is a Constructor that initializes all member variables when an object of the class is created.
- **getOrderId():** It returns the order ID.
- **getOrderId():** It returns the order ID.
- **setOrderId(String orderId):** It sets the order ID.
- **getUserImage():** Returns the user image.
- **setUserImage(String userImage):** It sets the user image.
- **getFullName():** It returns the full name.
- **setFullName(String fullName):** It sets the full name.
- **getProductImage():** It returns the product image.
- **setProductImage(String productImage):** It sets the product image.

- **getProductName():** It returns the product name.
- **setProductName(String productName):** It sets the product name.
- **getProductBrand():** It returns the product brand.
- **setProductBrand(String productBrand):** It sets the product brand.
- **getProdctPrice():** It returns the product price.
- **setProdctPrice(String prodctPrice) :** It sets the product price.
- **getOrderDate():** It returns the order date.
- **setOrderDate(String orderDate):** It Sets the order date.
- **getOrderTime():** It returns the order time.
- **setOrderTime(String orderTime):** It sets the order time.

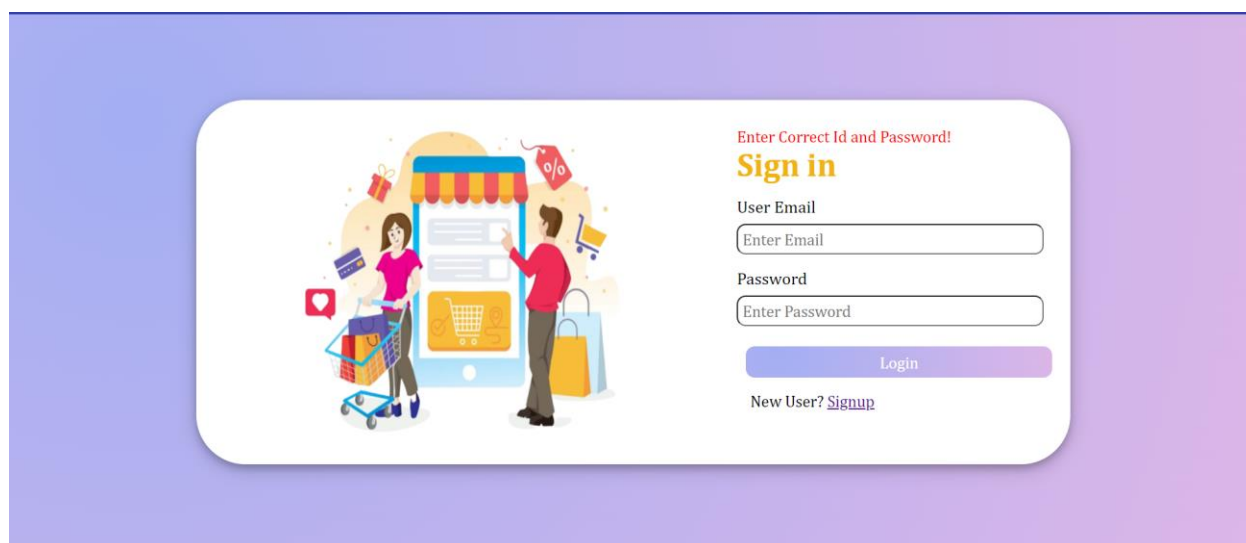
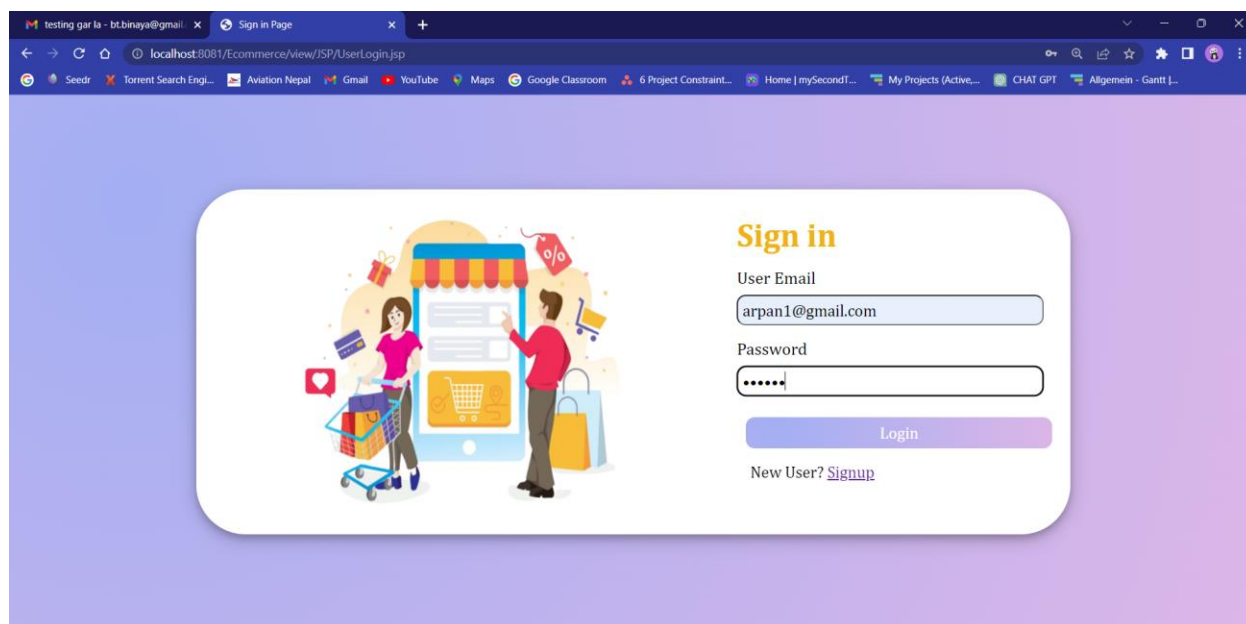
5. Test Cases

Testing is a method for determining if the actual software product complies with expectations and is error-free.

Test-1

Objective	To check whether user can login with incorrect credentials.
Action	Enter incorrect username and password, then click on login button.
Expected Result	The user should not be able to log in with incorrect credentials.
Actual Result	The user was not able to log in with incorrect credentials.
Conclusion	Test Successful.

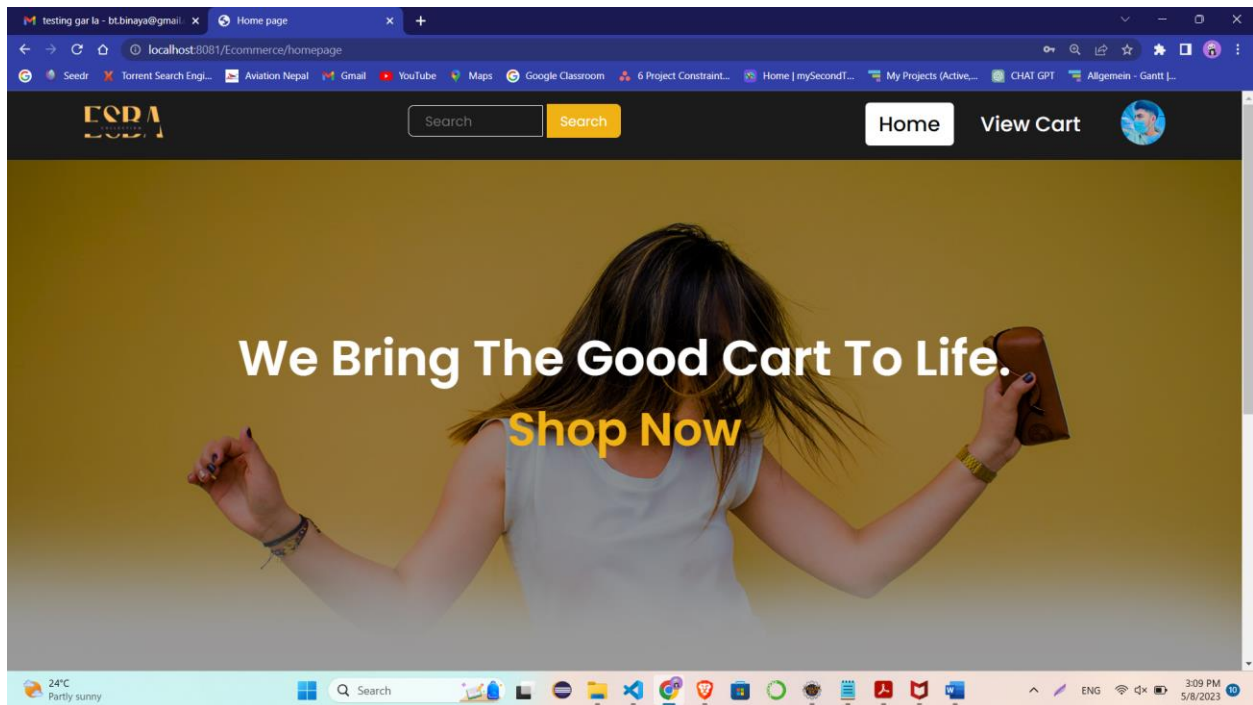
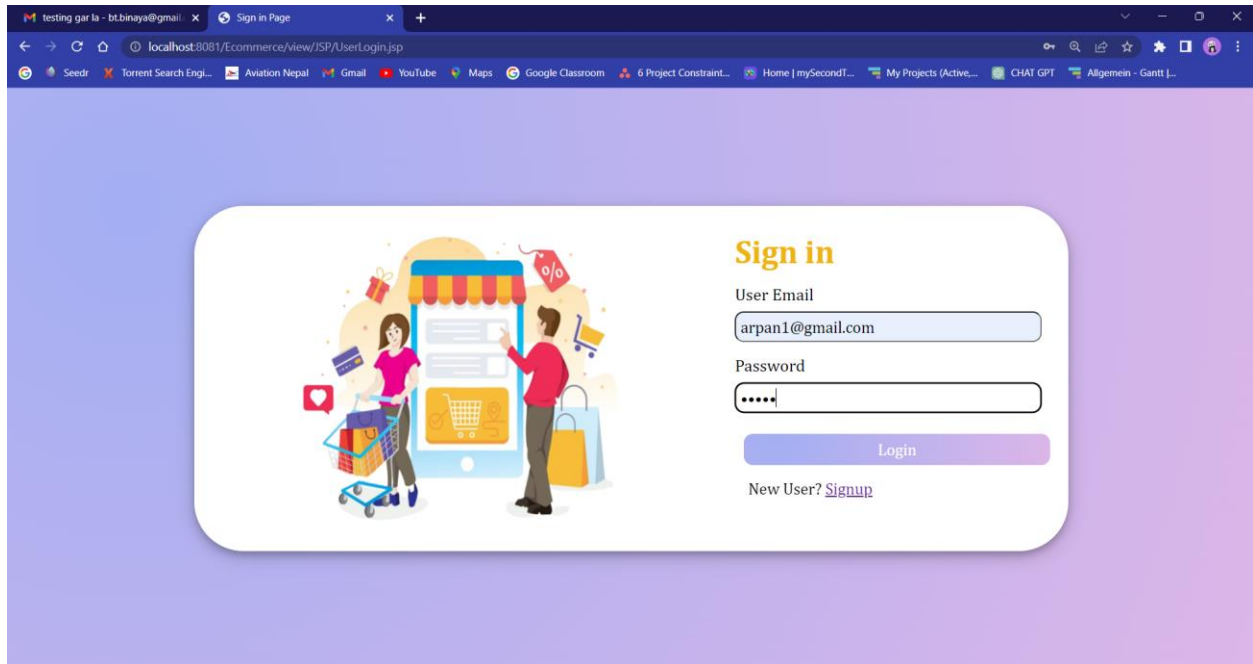
Table 1: Table for Test 1

**Test-2**

Objective	To check whether admin can login with correct credentials
Action	Enter incorrect username and password, then click on login button.
Expected Result	The admin should be able to log in with correct credentials.

Actual Result	The admin was able to log in with correct credentials.
Conclusion	Test Successful.

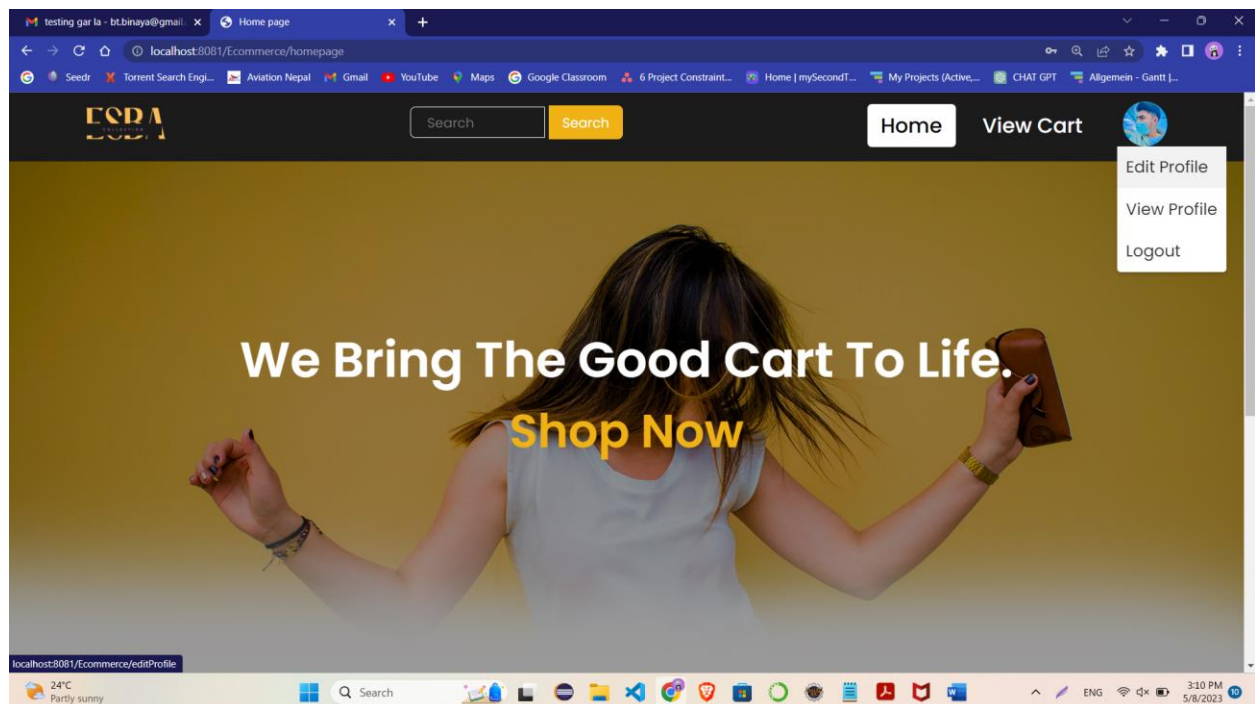
Table 2: Table for Test 2

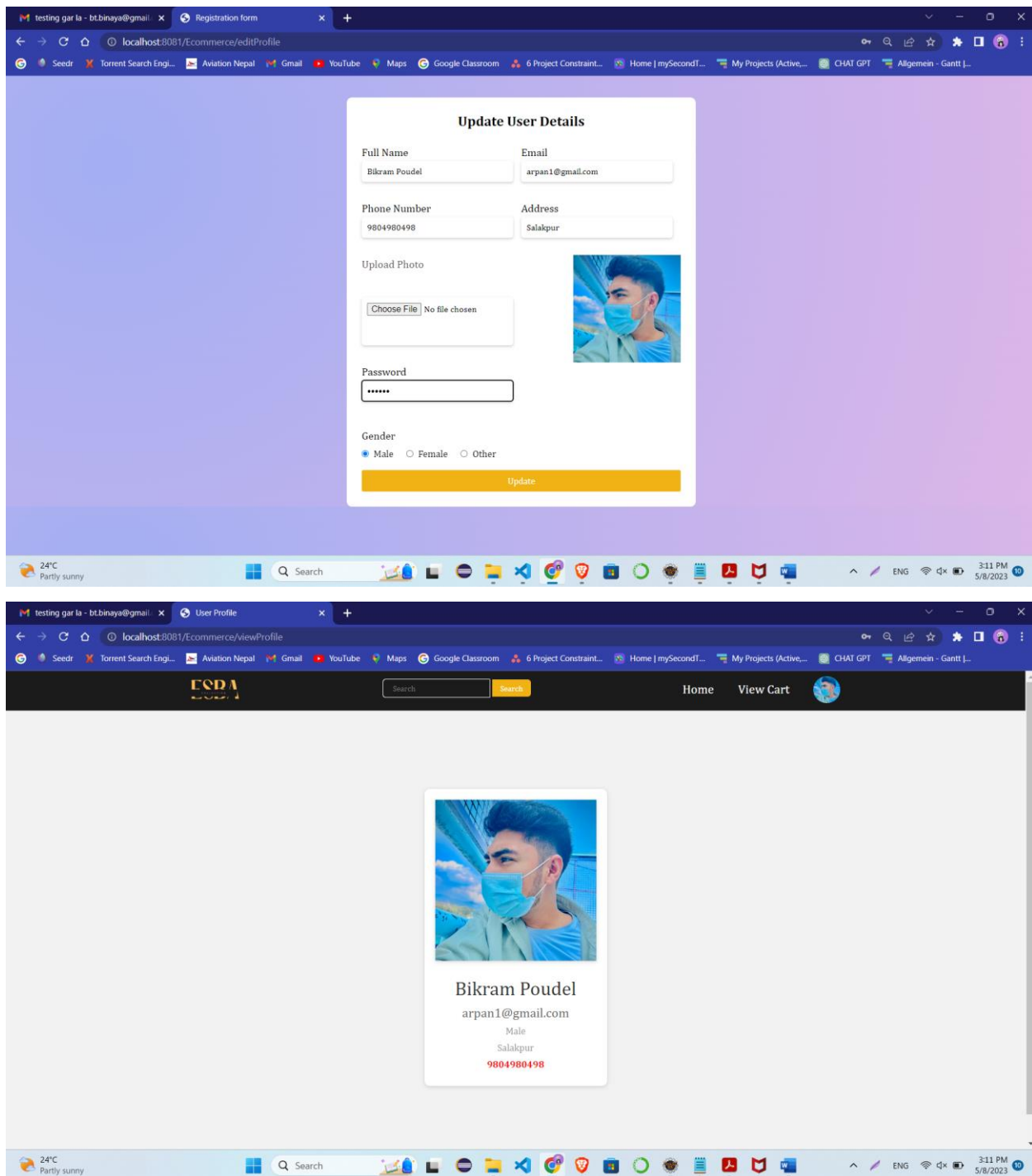


Test-3

Objective	To check whether users can edit their profile.
Action	Login to the user profile and edit the profile information and save changes.
Expected Result	The user should be able to edit their profile information and save the changes successfully.
Actual Result	The user is able to edit their profile information and save the changes successfully.
Conclusion	Test successful.

Table 3: Table for Test 3

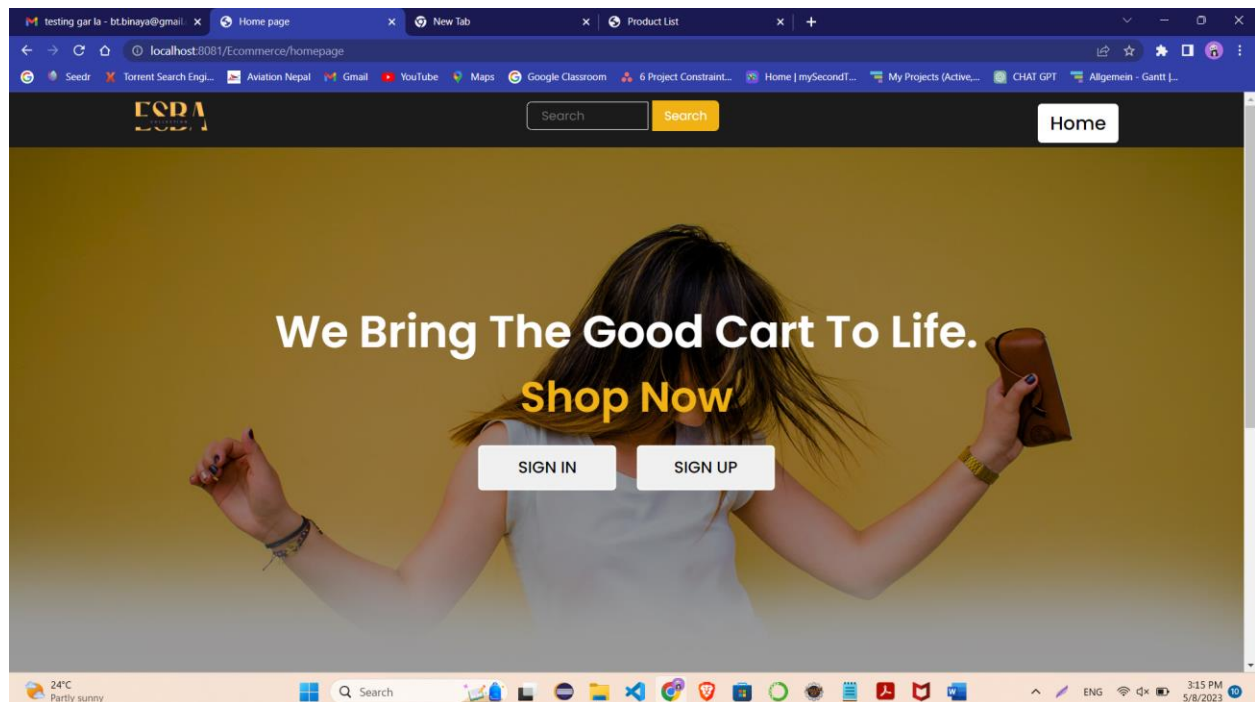


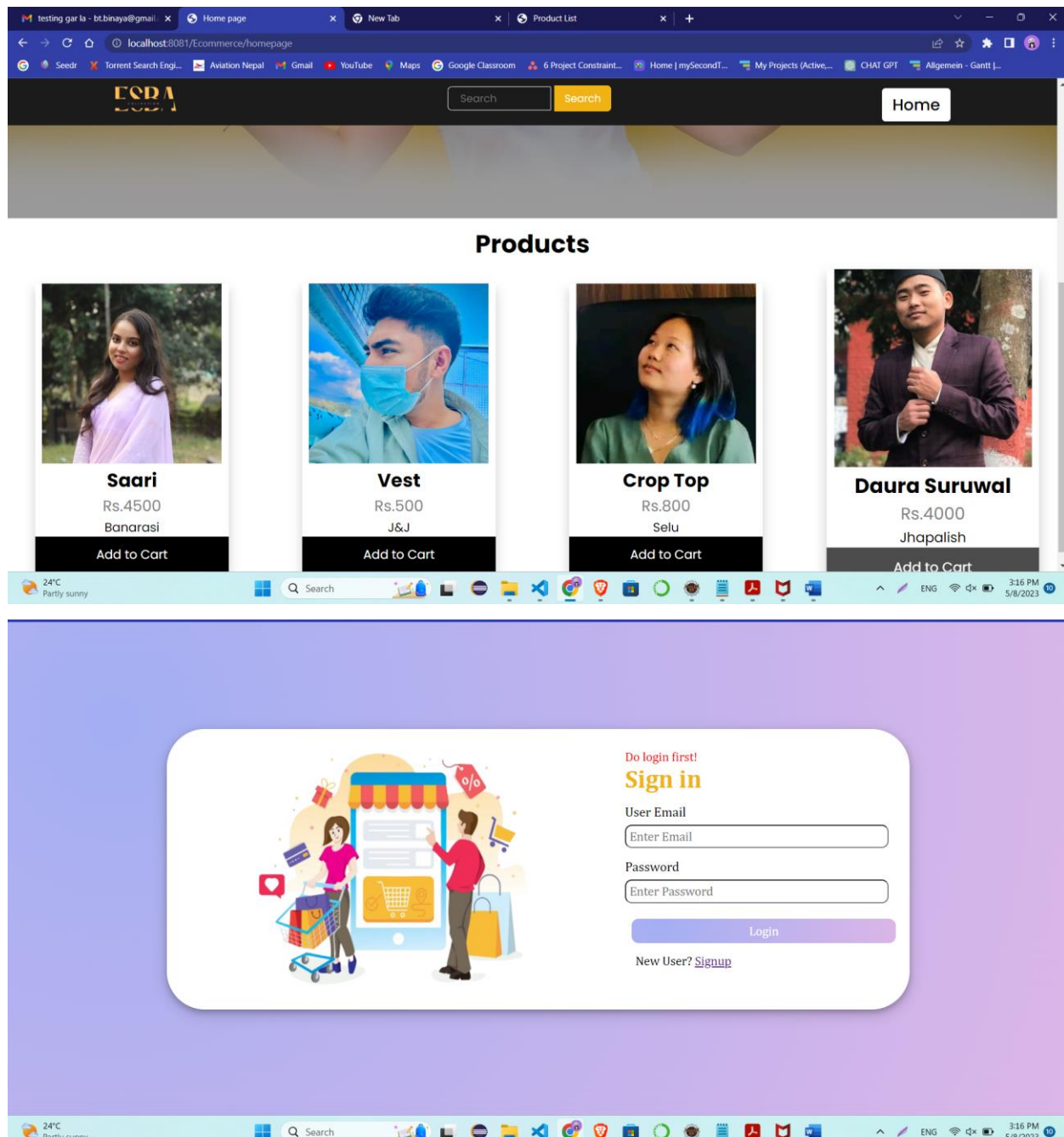
**Test-4**

Objective	To verify that the user is redirected to the login page for adding products to the cart without logging in.
-----------	---

Action	Go to the home page of website without login and try to add a product to the cart.
Expected Result	The user should be redirected to the login page for adding products to the cart without logging in.
Actual Result	The user is redirected to the login page for adding products to the cart without logging in.
Conclusion	Test successful

Table 4: Table for Test 4

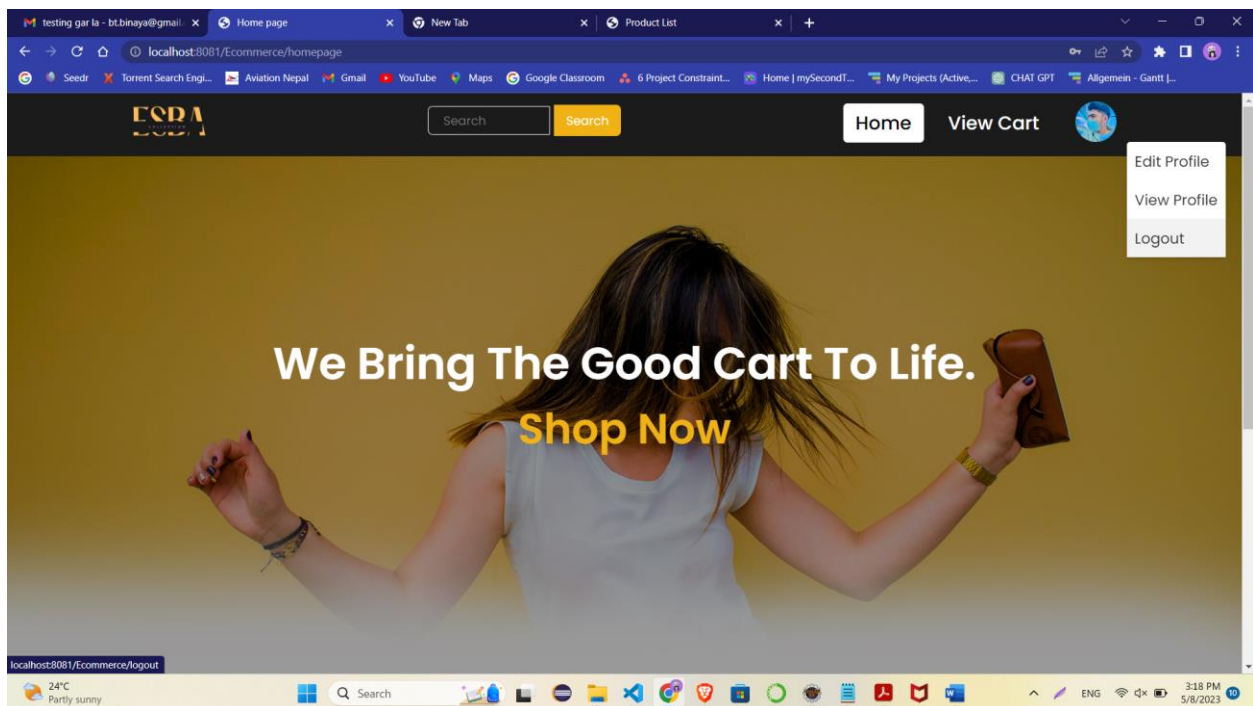


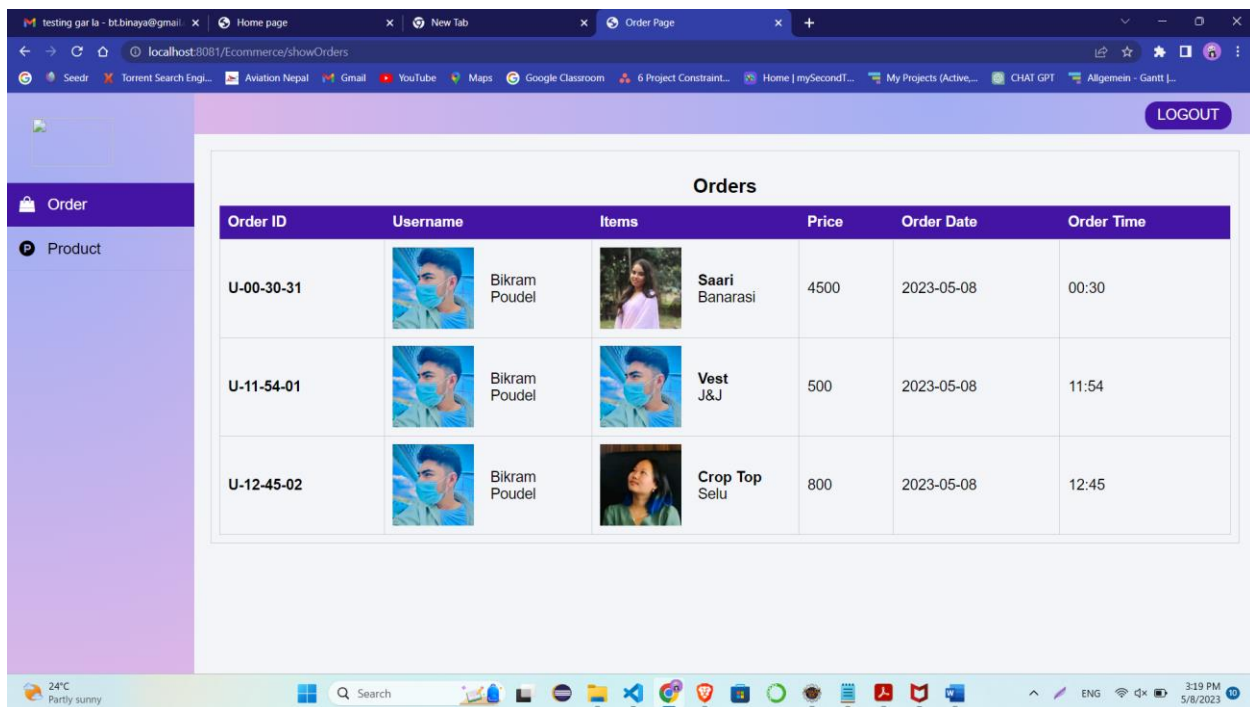
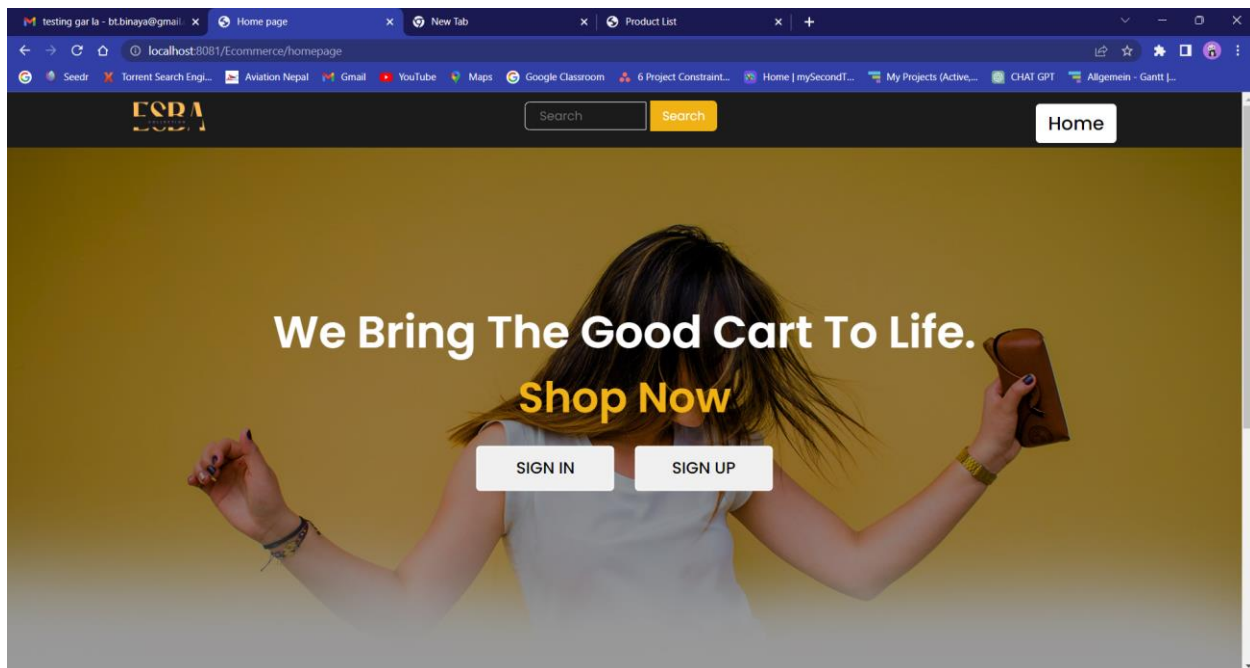
**Test-5**

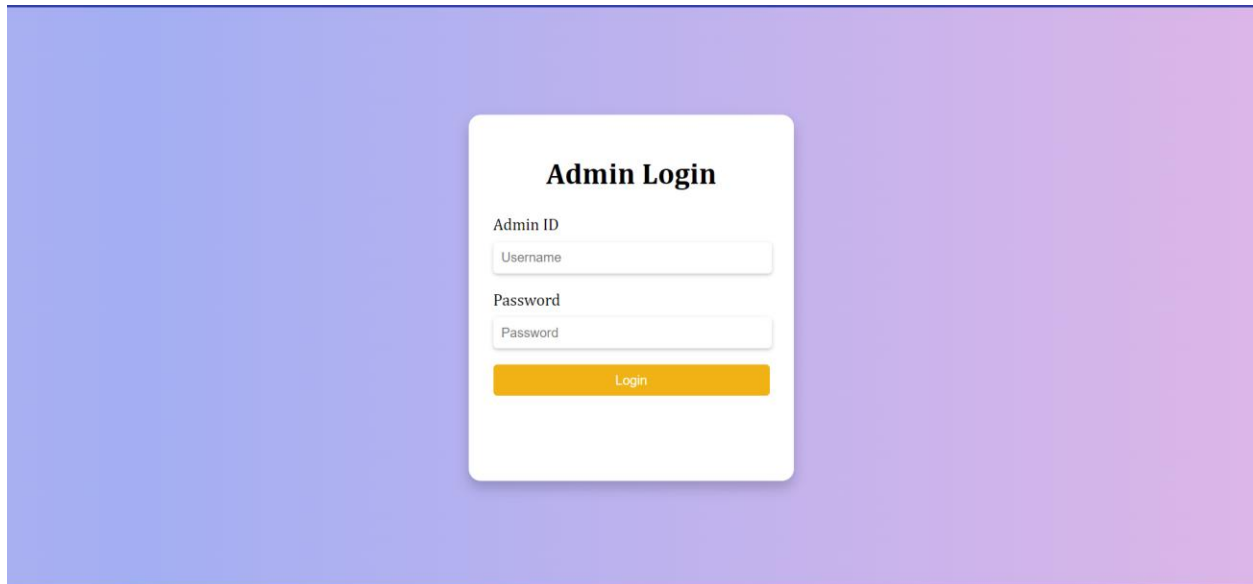
Objective	To verify that both the user and admin can log out of the system.
-----------	---

Action	Click on login button by entering correct credentials and after that again click on logout button.
Expected Result	The user or admin should be able to log out of the system successfully.
Actual Result	The user or admin is able to log out of the system successfully.
Conclusion	Test Successful.

Table 5: Table for Test 5

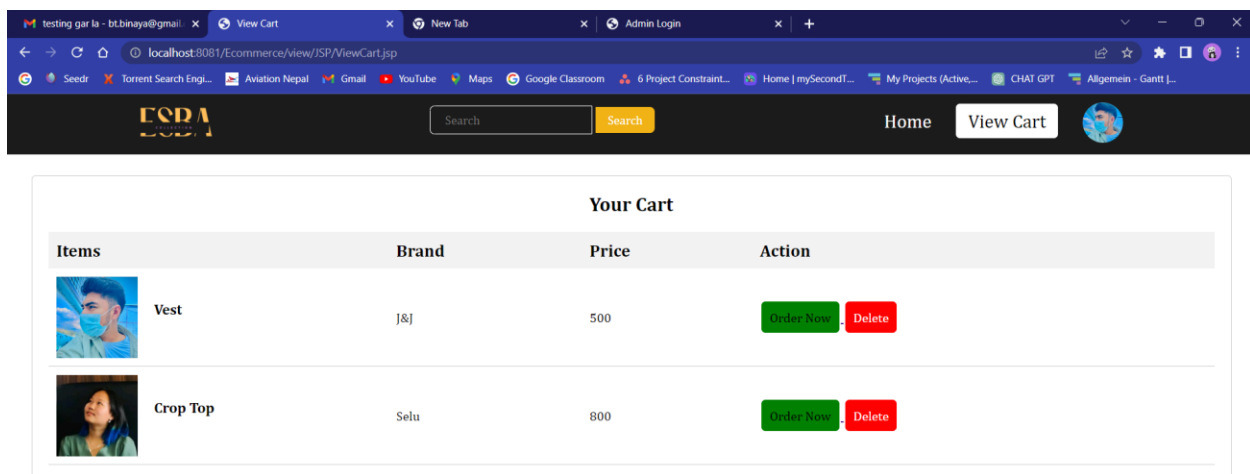
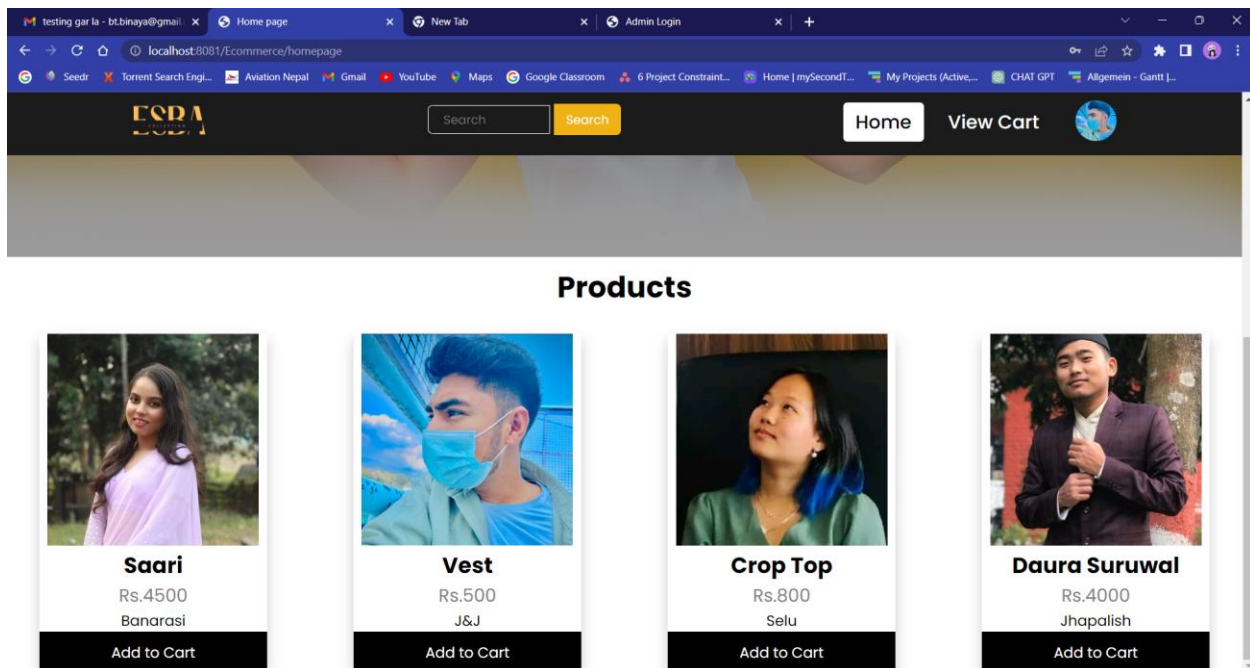




A screenshot of an 'Admin Login' form. The form is white with rounded corners and a subtle shadow, centered on a background with a blue-to-purple gradient. It contains the title 'Admin Login' in bold black text. Below the title are two input fields: 'Admin ID' with a placeholder 'Username' and 'Password' with a placeholder 'Password'. At the bottom of the form is an orange 'Login' button.**Test-6**

Objective	To verify that the user can add products to their cart and view their cart after login.
Action	Login to the website and go to the product page and click on 'Add to Cart' button. After adding cart click on 'view cart' and see your cart.
Expected Result	The user should be able to add products to their cart and view their cart successfully.
Actual Result	The user is able to add products to their cart and view their cart successfully.
Conclusion	Test successful.

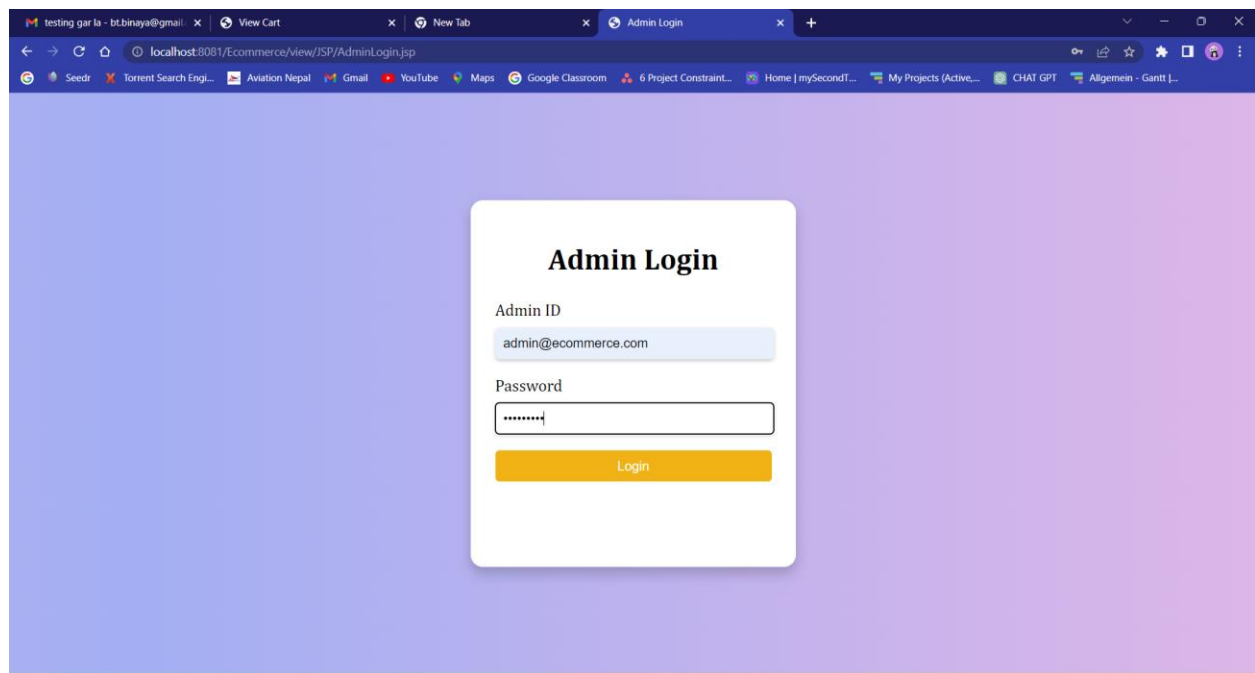
Table 6: Table for Test 6



Test-7

Objective	To verify that the admin can view the order list in the system.
-----------	---

Action	Login to the admin panel and go to order list page.
Expected Result	The admin should be able to access and view the order list in the system.
Actual Result	The admin is able to access and view the order list in the system.
Conclusion	Test Successful.

Table 7: Table for Test 7

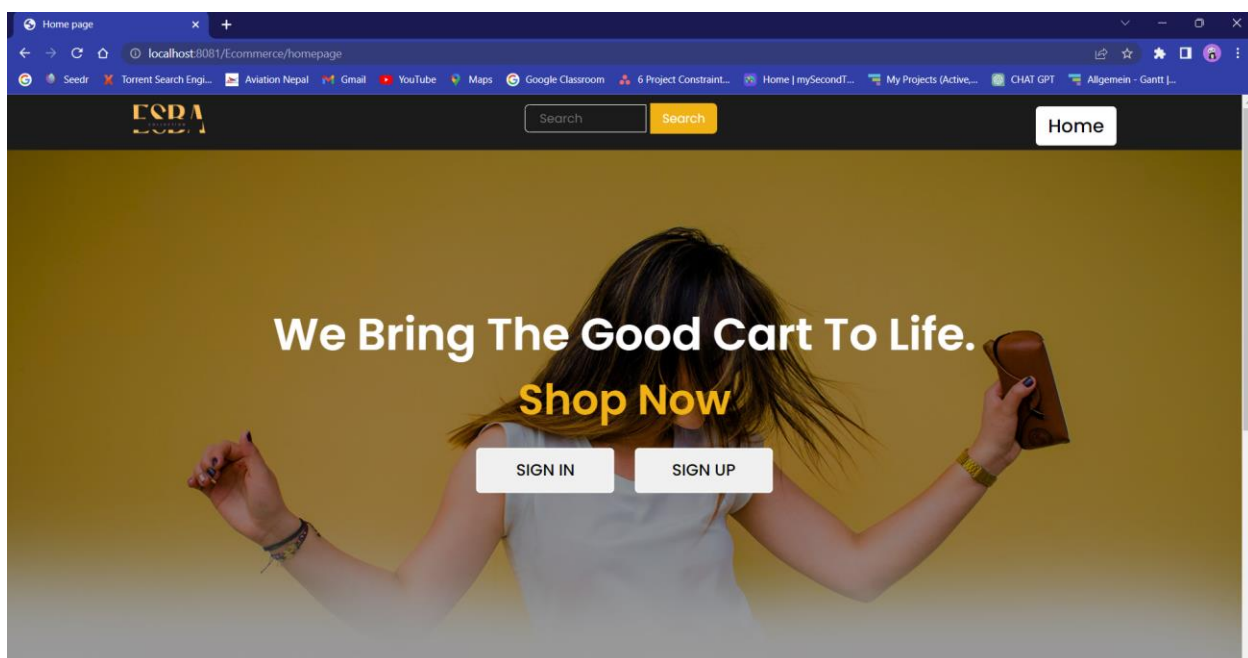
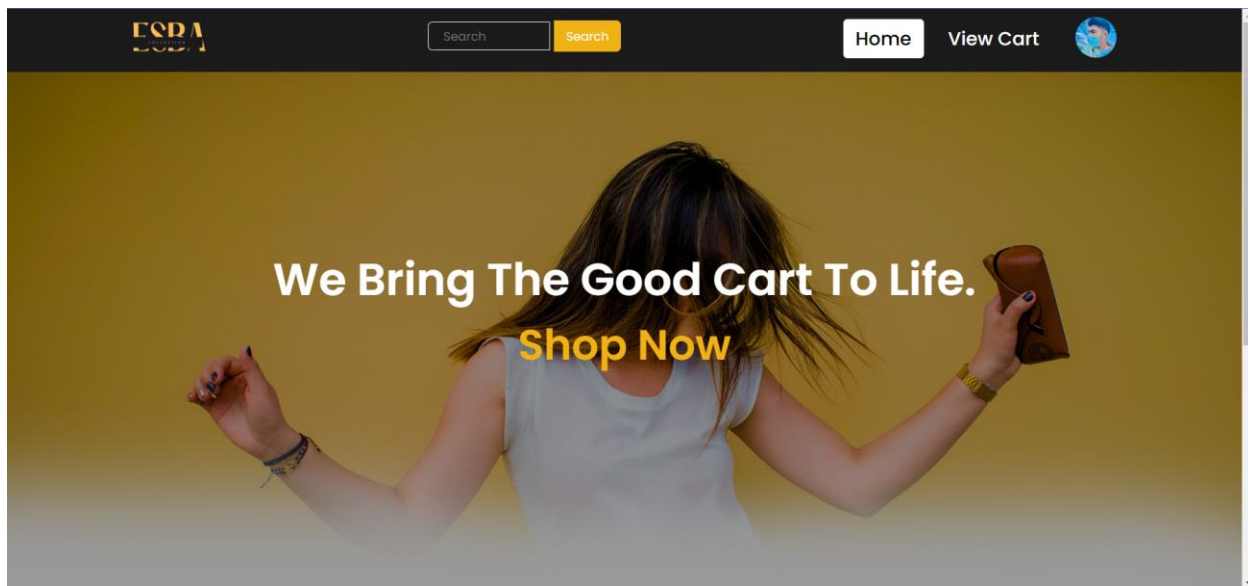
The screenshot shows a web browser window with the address bar displaying `localhost:8081/Ecommerce/view/JSP/AdminLogin.jsp`. The page has a light purple gradient background. In the center, there is a white login form titled "Admin Login". The form contains two input fields: "Admin ID" with the value `admin@ecommerce.com` and "Password" with masked characters. Below the fields is a yellow "Login" button.

Orders					
Order ID	Username	Items	Price	Order Date	Order Time
U-00-30-31	Bikram Poudel	Saari Banarasi	4500	2023-05-08	00:30
U-11-54-01	Bikram Poudel	Vest J&J	500	2023-05-08	11:54
U-12-45-02	Bikram Poudel	Crop Top Selu	800	2023-05-08	12:45

Test-8

Objective	To verify that the login session is being used in the system.
Action	Login to the system as a user and close the browser window. Again, open the browser and go to the website again.
Expected Result	The login session will be used in the system, and the user is still logged out after closing and reopening the browser.
Actual Result	The login session is used in the system, and the user is still logged out after closing and reopening the browser.
Conclusion	Test Successful.

Table 8: Table for Test 8

**Test-9**

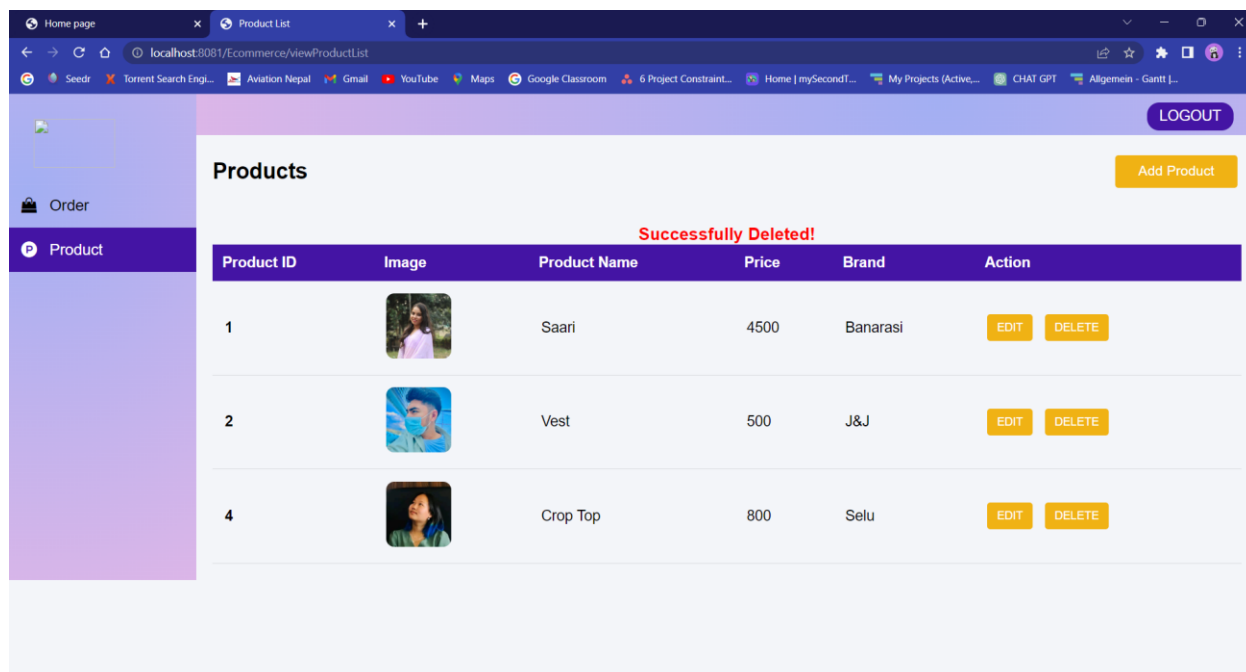
Objective	To Verify that the admin can delete products in the system.
Action	Login to the admin panel and go to product list page and click on 'Delete' button.

Expected Result	The admin should be able to delete the product in the system.
Actual Result	The admin is able to delete the product in the system.
Conclusion	Test Successful.

Table 9: Table for Test 9

The screenshot displays a web application interface for managing products. The browser window shows the URL `localhost:8081/Ecommerce/ViewProductList`. The application has a sidebar with navigation options: 'Order' and 'Product' (the latter is selected). The main content area is titled 'Products' and features a table listing five products. Each product entry includes a product ID, an image, the product name, price, brand, and two action buttons: 'EDIT' and 'DELETE'. A 'LOGOUT' button is located in the top right corner of the application area, and an 'Add Product' button is positioned above the product table.

Product ID	Image	Product Name	Price	Brand	Action
1		Saari	4500	Banarasi	<button>EDIT</button> <button>DELETE</button>
2		Vest	500	J&J	<button>EDIT</button> <button>DELETE</button>
4		Crop Top	800	Selu	<button>EDIT</button> <button>DELETE</button>
5		Daura Suruwal	4000	Jhapaish	<button>EDIT</button> <button>DELETE</button>



Test-10

Objective	To Verify that the admin can edit products in the system.
Action	Login to the admin panel and go to product list page and edit the details of the product.
Expected Result	The admin should be able to edit the details of a product in the system and save the changes made.
Actual Result	The admin is able to edit the details of a product in the system and save the changes made.
Conclusion	Test Successful.

Table 10: Table for Test 10

The image displays two screenshots of a web application interface. The top screenshot shows the 'Products' page, and the bottom screenshot shows the 'Edit Product' page.

Products Page:

- Header: Home page, Product List, +, localhost:8081/Ecommerce/viewProductList, LOGOUT.
- Left Sidebar: Order, Product (selected).
- Right Content: Add Product button, Products table.

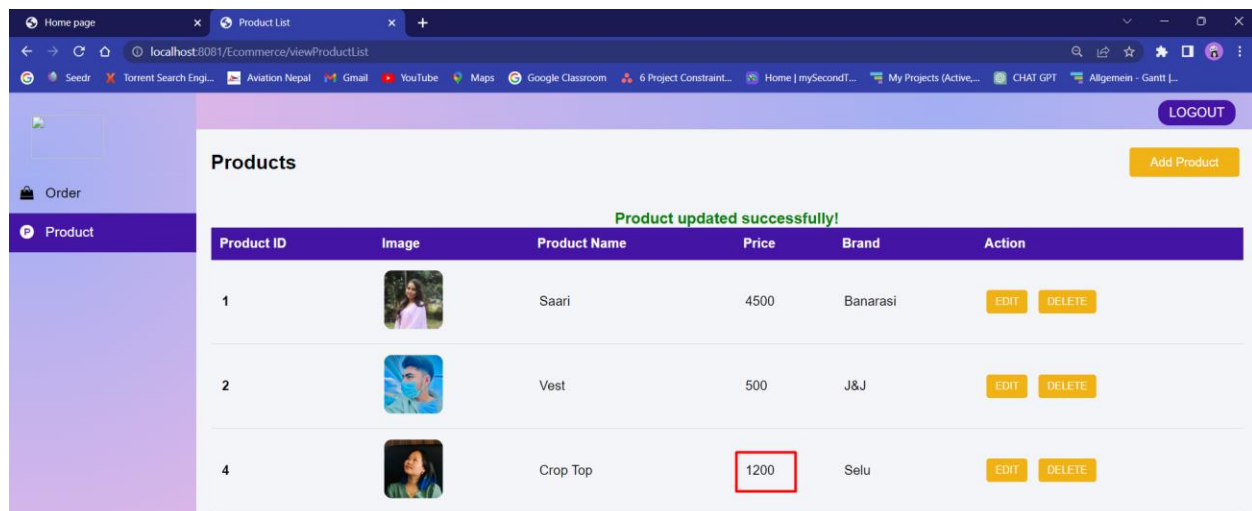
Product ID	Image	Product Name	Price	Brand	Action
1		Saari	4500	Banarasi	EDIT DELETE
2		Vest	500	J&J	EDIT DELETE
4		Crop Top	800	Selu	EDIT DELETE

Edit Product Page:

- Header: Home page, Edit Product, +, localhost:8081/Ecommerce/editProduct?productId=4, LOGOUT.
- Left Sidebar: Order, Product (selected).
- Right Content: Edit Product form.

Edit Product Form:

- Product ID: 4
- Product Name: Crop Top
- Price: 800
- Brand: Selu
- Image: [Choose File](#) No file chosen
-
- [Update](#) button




Test-11

Objective	To check whether that the user can register with image.
Action	Go to the registration page and fill in the registration form with the required information like name, address, image etc and click on 'register' button.
Expected Result	The user should be able to register with an image successfully.
Actual Result	The user is able to register with an image successfully.
Conclusion	Test successful.

Table 11: Table for Test 11

Registration

Full Name	Email
<input type="text" value="Binaya Thapa"/>	<input type="text" value="bt.binaya@email.com"/>
Phone Number	Address
<input type="text" value="9804959804"/>	<input type="text" value="Birtamode-7"/>
Upload Photo	Password
<input type="button" value="Choose File"/> binaya.jpg	<input type="password" value="*****"/>
Gender	
<input checked="" type="radio"/> Male <input type="radio"/> Female <input type="radio"/> Other	
<input type="button" value="Register"/>	

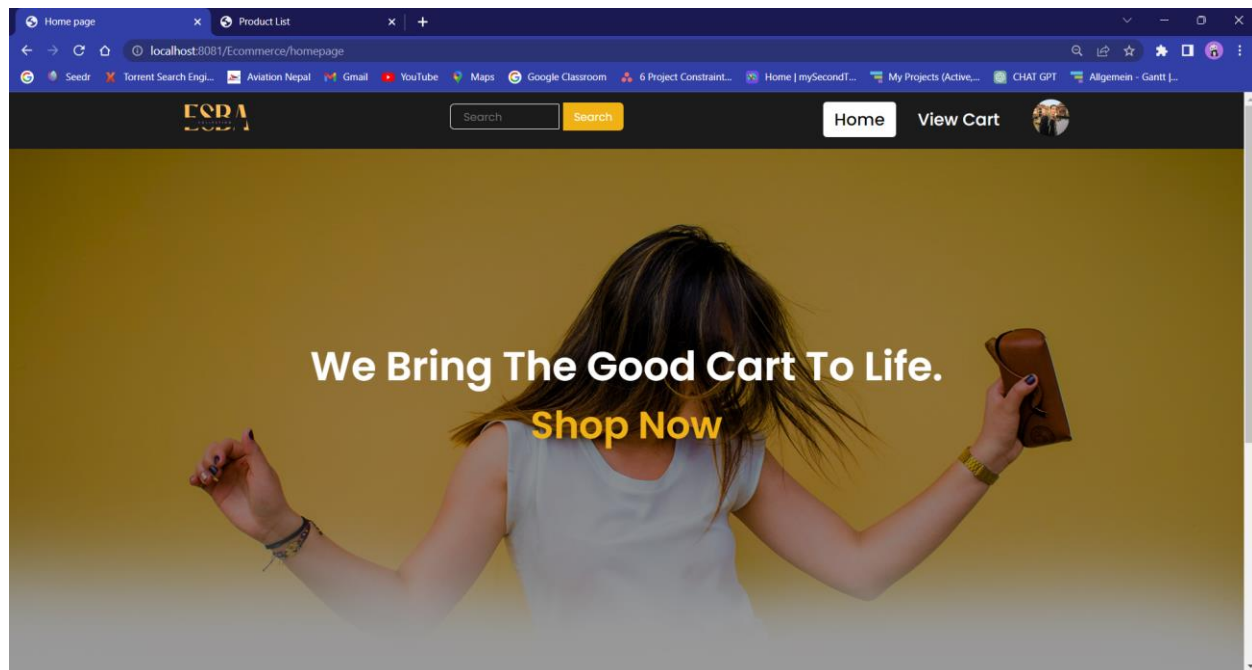


Sign in

User Email

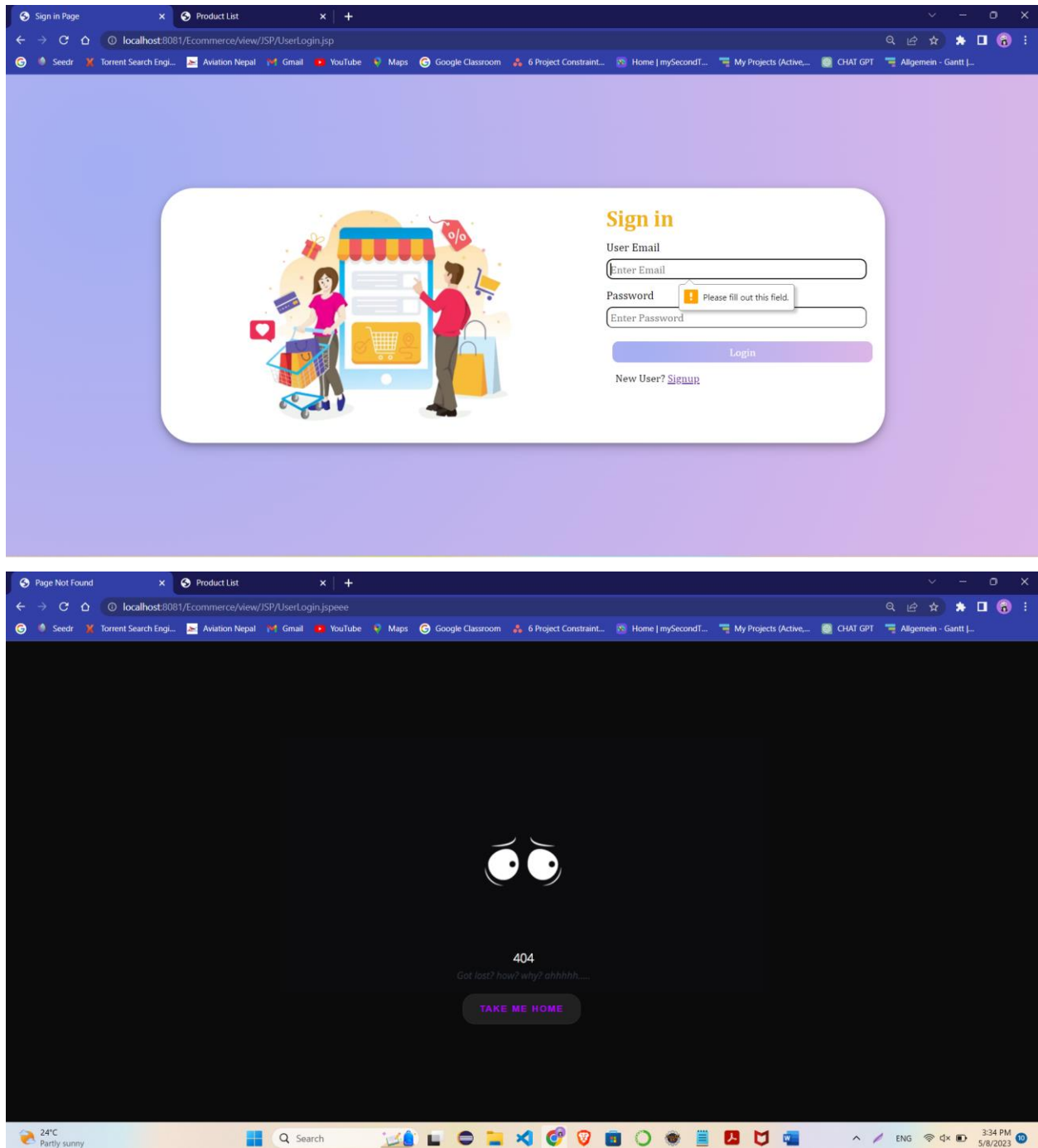
Password

New User? [Signup](#)

**Test-12**

Objective	To check whether the validation and exception handling like 404 or empty text fields
Action	Fill out a form with invalid or empty fields and try to access a URL or page that doesn't exist.
Expected Result	The system should validate all inputs and handle exceptions correctly by displaying appropriate error messages.
Actual Result	The system validates all inputs and handles exceptions correctly by displaying appropriate error messages.
Conclusion	Test successful.

Table 12: Table for Test 12



6. Tools and libraries used.

6.1. Tools Used

- Figma

Figma is a design interference tools for sharing idea about the project with multiple users and allows to get feedback and comments. It helps to integrate our thoughts for designing the project), Designing website pages, creating apps prototype(user interface (UI) and user experience (UX) designers), Project management, Mind map. It is used in our project due to the real-time collaboration feature (www.makeuseof.com, 2023).

- **XAMPP**

XAMPP is an open-source web server solution package that is used for web application for testing on a local host webserver. It is used to serve web pages on internet. It is used for testing and modifying project before releasing (www.educba.com, 2023). To aid for testing and modifying the project and for serving different web pages on internet , XAMPP is used in our project of E-commerce clothe store.

- **Eclipse IDE**

Eclipse is an integrated development environment for developing computer-based applications by using the Java programming language as well as programming languages such as C/C++, Python, PERL, Ruby etc. It has high rich graphics arrangement as well as rich widget set (www.eclipse.org, 2023). It is used in our project because of its functioning framework Such as, Eclipse Communications Framework (ECF), Graphical Editing Framework (GEF), Eclipse Modelling Framework (EMF), Graphical Modelling Framework (GMF).

- **MS word**

Microsoft Word is a word processing program that was created by Microsoft in 1983 and has grown to become one of the most popular tools for creating and editing documents. It additionally provides a range of features and functions that go beyond what a basic text editor can do. In our project the purpose of using Microsoft Word is creating documents for our project. (www.geeksforgeeks.org, 2023)

- **Brave Browser**

Brave is one of the free, fastest, and safest browsers on the market today developed by Brave software, Inc. It is used in our project because it built off the open-source Chromium

Web core that can powers browsers used by billions of people worldwide. Also, it is safe to use, safer than almost any other browser and it has wonderful tools that can attract reader's attention (brave.com, 2023).

- **Tomcat**

Tomcat is an open-source nature Java application server developed by the Apache Software Foundation and it is so designed to deploy Java Servlets and JSPs on your system. It is used in our project because of its open-source nature. It starts quickly and can be redeployed. It can be used freely for any type of commercial project (www.jrebel.com, 2023).

6.2. Libraries Used

- **Java database connectivity (JDBC)**

Java database connectivity is the standard application programming interface (API) that allows Java programs to access database management systems. As JDBC being standard specification, one of the Java programs using the JDBC API can also connect to any database management system (DBMS) by the existence of driver for DBMS (www.ibm.com, 2023).

- **STL (JSP Standard Tag Library)**

JSTL is the standard tag library which provides tags to control the JSP page behaviour. It can be used to write something in JSP page, redirect request to another source, set variable to given scope, catching exception, and wrapping it to object and so on. It is used in our project for iteration purpose and control statements, internationalization, SQL etc (www.digitalocean.com, 2023).

- **MySQL connector:**

MySQL connector is a type of tool which enables us to connect and interact with a different MySQL database. Its main goal is to make it possible for us to perform data

queries and table creations in external MySQL instance. In our project the purpose of using it is specifically to join and integrate data from two different systems. (docs.starburst.io, 2023)

7. Development Process

Our coursework started with the creation of a Gantt chart, which helped us efficiently to manage our tasks. We were able to divide up the work between the group members and decide the order of tasks and their respective deadlines through the Gantt chart. It helps us to determine which tasks should be focused on and how much time was needed to complete each task. Gantt chart gave us an accurate representation of the project's timeline. We managed to effectively organize our project and make sure to use a systematic approach to achieving our goals by using the Gantt chart.

Then, we used Figma to create design for our website. Figma helps us to create wireframes and prototypes for our frontend pages. Every one of our team was able to visually interpret the layout and structure of our website. Using Figma's features and functionalities we could effectively collaborate and proceed on our designs and finished product met needs and expectations. The creation of an understandable and visually stunning website was made due to Figma, which helped us translate our concepts into concrete visual representations.

Thirdly, after completing design we then started development phase. In development phase we followed MVC pattern (Model View Controller). It represents a software architectural design pattern that is frequently used in the creation of desktop and web applications. By encouraging the separation of concerns, the MVC pattern makes the application more modular and simpler to maintain. Better code reuse is made possible because changes to one component don't have an impact on the others. A structured and organized approach to application development is provided by the model, view, and controller, which improves flexibility, scalability, and testability.

Throughout the development phase, we focused testing and verifying the quality of our system. We carefully constructed test cases and scenarios that covered many aspects of the system. We wanted to ensure that all features and capabilities performed as intended and fulfilled our users' needs. We closely observed and documented any errors or mistakes that were discovered during the testing phase. Overall, our emphasis on testing and ensuring system quality was critical during the development process. It enabled us to discover and resolve faults, resulting in a stable and reliable system on which users could trust.

We ran into a few obstacles as we got deeper into this new module. At first, distributing the workload among the team members was a challenge. We had trouble deciding what specific tasks each team member should perform and how to effectively distribute them. Additionally, we ran into issues with both the frontend and backend coding. The coding process presented difficulties as we struggled with new frameworks, libraries, or programming languages while managing unfamiliar territory. The creation of a class diagram for our website was another challenge we faced. It turned out to be a difficult task to define the attributes and relationships between various classes as well as their attributes and methods.

Despite these difficulties, we kept going and looked for answers. To divide tasks more effectively, we had open discussions and worked together to make decisions. When help was required, we sought it out through research, online tools, or by asking more knowledgeable people like teachers, module leaders for advice. We actively sought assistance from online forums, communities, and mentors for coding problems. To better understand the technologies we were using, we made use of the documentation and tutorials. We took the time to examine the website's architecture and the interactions between various components in relation to the class diagram. To visualize and plan the structure of our classes, we made use of the tools and resources that were readily available, such as UML diagrams. Overall, despite initial challenges with workload division, coding, and class diagram creation, we overcame them by approaching them

with tenacity, collaboration, and a proactive mindset, consulting experts and making use of the resources at hand.

To finish the project in our coursework, we worked as a team. Working in a group brought various benefits that helped in the overall success of our coursework. Everyone brought their own viewpoints and abilities to the table, which improved the overall quality of our work. We were able to tackle challenging jobs more effectively and efficiently by pooling our knowledge and abilities. Working as part of a team also promoted excellent communication and open discussions. We communicated our ideas, concerns, and progress on a frequent basis, which helped us keep on track with the project goals and maintained everyone was on the same page. Overall, working as part of a team for our coursework given us with various benefits. It enabled us to take advantage of the team's collective skills and expertise, improve communication and collaboration, develop a sense of shared responsibility, and boost productivity. These benefits ultimately helped in the completion of our project and boosted our overall learning experience.

8. Critical Analysis

This critical analysis describes the project requirements, which include implementing an MVC (Model View Controller) based e-commerce website with a login system, admin panel, home page features, user functionalities, validation and exception handling, and a reflective report documenting the development process.

While developing the e-commerce website we have faced many difficulties and challenges also, we have gotten a chance to improve it. Talking about the difficulties we faced so many challenges as we began to work on the project frontend. The first problem was coming up with a responsive design that functioned well on different screen sizes and devices. Additionally, we gave top priority to website performance optimization to achieve quick page loads and effective execution of codes. Preserving as well as scaling the codebase got harder as the project got more complicated. Version control

management and team collaboration presented additional challenges. Another important aspect was keeping to accessibility standards for users with disabilities.

Finally, maintenance of a unified user interface design required careful attention to detail and efficient teamwork. While doing backend in our project we faced a pile of problems. We had issues ensuring scalability and performance, managing, and organizing data effectively, and putting strong security measures in place. It proved difficult to integrate with third-party services, and careful attention was needed when handling errors and logging them. Debugging took a lot of time, and thorough testing was necessary. There were difficulties in managing deployments and version control.

Despite these difficulties, we persisted and looked for fixes for each problem, eventually succeeding in building a dependable and effective backend system. Wireframe development for our project was a particularly satisfying aspect of our work. Before beginning the actual development process, it gave us the opportunity to examine and visualize the design and organizational structure of our application or website. We were able to unleash our creativity and turn concepts into concrete representations during this stage.

In summary, the e-commerce website design project has various advantages. It followed the MVC pattern as well, resulting in a well-structured and ordered codebase. The use of a login system with encryption and session management added an extra layer of protection. The user experience was simple and straightforward, allowing users to navigate and engage with the website with ease. The development process was methodical, with emphasis on programming styles and good documentation.

However, there have been certain situations where the project might have improved further. Implementing advanced suggestion algorithms, for example, might have improved the user experience by giving individualized product recommendations. Furthermore, giving more extensive comments in the coding would have made it easier for future developers to understand and maintain the code. The reflective report should

have been expanded to incorporate a larger range of development-related experiences and insights.

Despite these problems, the project established a solid platform for future updates and provided great learning opportunities in advanced programming approaches and technologies. It demonstrated the team's ability to create a functional and user-friendly e-commerce website, as well as a platform for testing and deploying new features.

9. Conclusion

This coursework is related to the "E-commerce website" for cloth store. The name of our "E-commerce Website" is ESBA. While doing coursework, we have face many difficulties such as to provide better as well as effective customer service, Website traffic etc. All these issues and challenges were the speed breaker to our project success. With the help of our team, teacher guidance, friends help as well as individual research helps us to catch up the speed for the desired success of our project. This coursework is all about designing the commercial website for cloth store. The designed website can be the key for the commercial business. Thanks to our team who has thoroughly researched the market and user preferences to create an ideal design that can meets the needs and expectations of the user.

To conclude, our team is confident that the design of this E-commerce website for a cloth store will be an asset to the business. This can help to increase their online visibility, reach new customers, and ultimately improve revenues to value your feelings and time, we keep on going maintenance and optimization. We also believe that this website will continue to deliver exceptional results for our user not once but for the long term.

Bibliography

- brave.com. (2023, 05 04). *Secure, Fast & Private Web Browser with Adblocker | Brave Browser*. Retrieved from Secure, Fast & Private Web Browser with Adblocker | Brave
<https://brave.com/#:~:text=It%20blocks%20privacy%2Dinvasive%20ads,possible%20to%20secure%20https%20connections>
- designcode.io. (2023, 05 04). *The Figma Design Tool - Figma Handbook - Design+Code*. Retrieved from The Figma Design Tool - Figma Handbook - Design+Code:
<https://designcode.io/figma-handbook-figma-design-tool>
- mailchimp.com. (2023, 05 04). *How to Build an Ecommerce Website | Mailchimp*. Retrieved from How to Build an Ecommerce Website | Mailchimp:
<https://mailchimp.com/marketing-glossary/ecommerce-website/#:~:text=An%20e%2Dcommerce%20website%20is,brick%2Dand%2Dmortar%20location>
- venngage.com. (2023, 05 04). *How to Make a Class Diagram [+Examples] - Venngage*. Retrieved from How to Make a Class Diagram [+Examples] - Venngage:
<https://venngage.com/blog/class-diagram/#what>
- visme.co. (2023, 05 04). *What is a Wireframe? Guide With Types, Benefits & Tips (2023)*. Retrieved from What is a Wireframe? Guide With Types, Benefits & Tips (2023):
<https://visme.co/blog/what-is-a-wireframe/>
- www.digitalocean.com. (2023, 05 04). *JSTL Tutorial, JSTL Tags Example | DigitalOcean*. Retrieved from JSTL Tutorial, JSTL Tags Example | DigitalOcean:
<https://www.digitalocean.com/community/tutorials/jstl-tutorial-jstl-tags-example>
- www.eclipse.org. (2023, 05 04). *What Is Eclipse? | The Eclipse Foundation*. Retrieved from What Is Eclipse? | The Eclipse Foundation:
<https://www.eclipse.org/home/whatis/#:~:text=Eclipse%20is%20a%20Tools%20Framework,specifically%20related%20to%20software%20development>
- www.educba.com. (2023, 05 04). *What is XAMPP? | Complete Guide to What is XAMPP*. Retrieved from What is XAMPP? | Complete Guide to What is XAMPP:
<https://www.educba.com/what-is-xampp/>

www.ibm.com. (2023, 05 04). *What is JDBC? - IBM Documentation*. Retrieved from What is JDBC? - IBM Documentation: <https://www.ibm.com/docs/en/informix-servers/12.10?topic=started-what-is-jdbc>

www.jrebel.com. (2023, 05 04). *What Is Apache Tomcat? | JRebel & XRebel by Perforce*. Retrieved from What Is Apache Tomcat? | JRebel & XRebel by Perforce: <https://www.jrebel.com/blog/what-is-apache-tomcat>

www.makeuseof.com. (2023, 05 04). *What Is Figma and What Is It Used For?* Retrieved from What Is Figma and What Is It Used For?: <https://www.makeuseof.com/what-is-figma-used-for/>

www.uxdesigninstitute.com. (2023, 05 04). *What is UI design? A complete introductory guide - UX Design Institute*. Retrieved from What is UI design? A complete introductory guide - UX Design Institute: <https://www.uxdesigninstitute.com/blog/what-is-ui-design/>