

## PROJECT DOCUMENTATION

### SUBURBAN OUTFITTERS - ONLINE SHOPPING PORTAL

#### INTRODUCTION

Suburban Outfitters has decided to venture into the online market. It's evident that people are now doing most of their shopping online. To meet this demand, we aim to build a user-friendly platform where customers can feel the essence of our physical store while enjoying the convenience of online shopping.

To make this online platform come to life, we're using various tools and techniques. HTML and CSS are our foundation, ensuring the site is visually appealing. With JavaScript, we're adding functionalities that make the website more interactive and engaging. On the backend, PHP and MySQL are being used. They help in storing product details, user accounts, order histories, and more. We're also using XAMPP to host on server. It's a simple and lightweight solution that allows us to develop and test the site locally before we launch it for everyone.

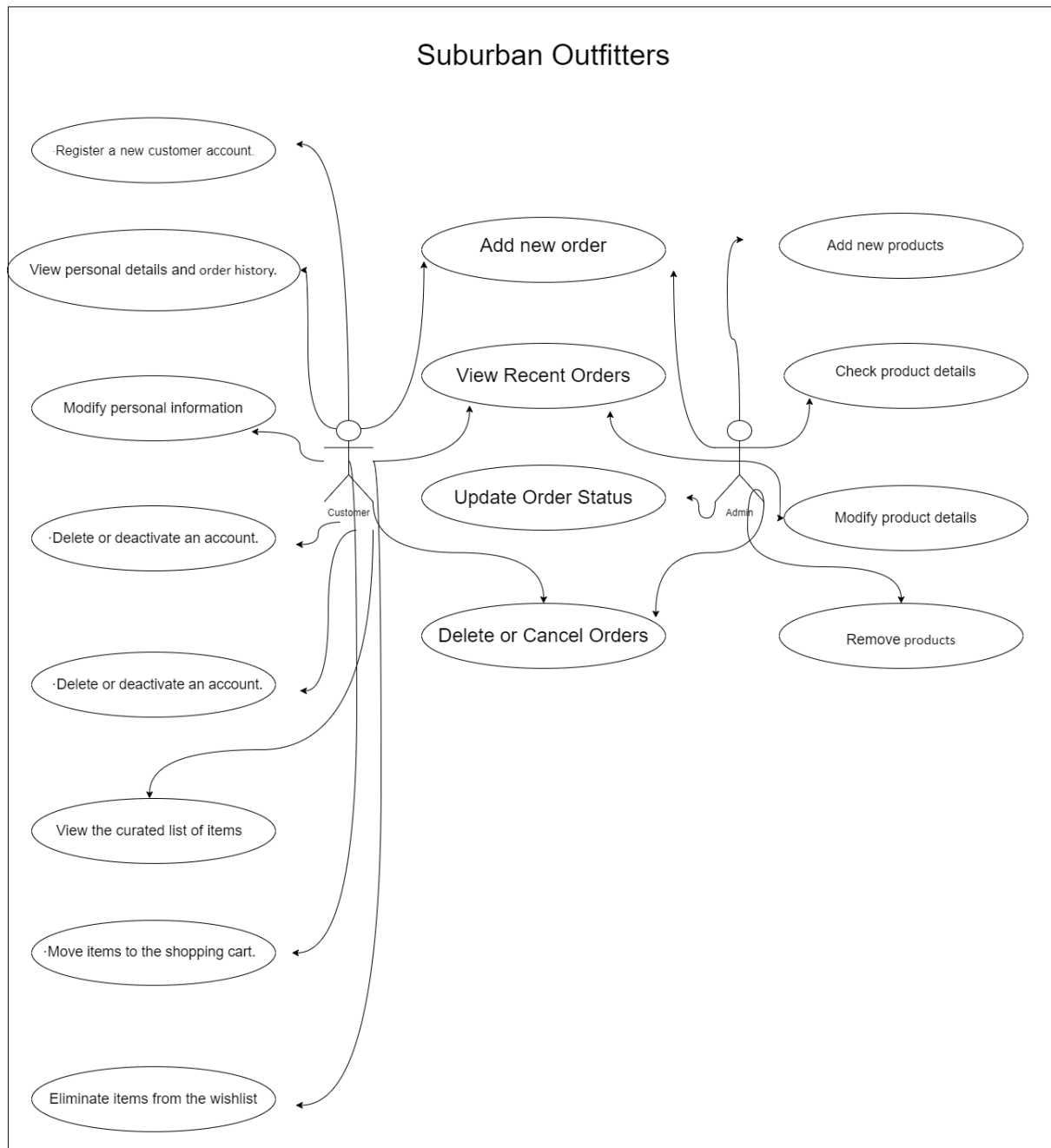
Planning and executing such a project requires a systematic approach. That's why we've adopted the Agile System Development method. This method isn't just about coding; it's about understanding what the customer wants, setting clear goals, and working towards them in manageable steps or "sprints." Throughout this process, we'll be using tools like flow diagrams to map out how users will navigate the site and ERDs (Entity Relationship Diagrams) to figure out how the data interacts.

Additionally, we will continuously gather feedback, making sure that the website serves our customers well and offers them a seamless shopping experience. As we proceed, we'll be designing various use cases. These will help us understand the different ways users might interact with the site. For instance, how they'll add items to their cart, how they'll check out, and how they'll track their orders.

All these efforts are in service of our primary objective: providing an efficient, enjoyable, and comprehensive online shopping platform for our loyal customers at Suburban Outfitters. We believe that with the right combination of technology and method, our online platform will not only meet but exceed expectations.

Github Link: <https://github.com/abho11/suburban-outfitters>

## 1. Use Case Model



---

## 2. Use Case Document

### 2.1 Use Case: Register a new customer account (Create)

Attribute	Description
Use Case	Register a new customer account (Create)
Objective	Allow new customers to create an account for a personalized shopping experience.
Business Event	Customer wants to register for a new account.
Primary Actor(s)	Customer
Secondary Actor(s)	None
Pre-condition	Customer is on the site homepage.
Post-condition	New customer account is created, and customer is logged in.
Failure outcomes	Error message 'Error creating account'.

#### Flow of Events:

Step	Actor Action	System Response	Associated File
1	Customer clicks 'Sign Up'	Display registration form	register.php
2	Customer fills in personal details and submits	Validate input and create new account in database	-
3	-	Log the customer in and forward to profile.php	profile.php

## 2.2 Use Case: View personal details and order history (Read)

Attribute	Description
Use Case	View personal details and order history (Read)
Objective	Allow customers to view their profile details and order history.
Business Event	Customer wants to view their account details and purchase history.
Primary Actor(s)	Customer
Secondary Actor(s)	None
Pre-condition	Customer is logged into their account.
Post-condition	Customer views their profile and order history.
Failure outcomes	Error message 'Unable to fetch details'.

### Flow of Events:

Step	Actor Action	System Response	Associated File
1	Customer clicks 'My Profile'	Display personal details	<b>profile.php</b>
2	Customer clicks 'View Recent Orders'	Fetch and display list of past orders	<b>order-history.php</b>

### 2.3 Use Case: Modify personal information (Update)

Attribute	Description
Use Case	Modify personal information (Update)
Objective	Allow customers to change their account details.
Business Event	Customer wants to update their profile.
Primary Actor(s)	Customer
Secondary Actor(s)	None
Pre-condition	Customer is logged into their account.
Post-condition	Customer's details are updated in the system.
Failure outcomes	Error message 'Failed to update details'.

#### Flow of Events:

Step	Actor Action	System Response	Associated File
1	Customer clicks 'Edit Profile'	Display editable form with current details	<b>edit-profile.php</b>
2	Customer modifies details and submits	Validate input and update details in the database	-
3	-	Display success message and forward to <b>profile.php</b>	<b>profile.php</b>

## 2.4 Use Case: Delete or deactivate an account (Delete)

Attribute	Description
Use Case	Delete or deactivate an account (Delete)
Objective	Allow customers to remove their account from the system.
Business Event	Customer decides to delete their account.
Primary Actor(s)	Customer
Secondary Actor(s)	None
Pre-condition	Customer is logged into their account.
Post-condition	Customer's account is deleted or deactivated.
Failure outcomes	Error message 'Failed to delete account'.

### Flow of Events:

Step	Actor Action	System Response	Associated File
1	Customer clicks 'Delete Account'	Ask for confirmation	<b>delete-account.php</b>
2	Customer confirms deletion	Remove or deactivate account from the database	-
3	-	Log the customer out and display <b>login.php</b> with a message	<b>login.php</b>

## 2.5 Use Case: Add new products or restock existing items (Create)

Attribute	Description
Use Case	Add new products or restock existing items (Create)
Objective	Efficiently introduce new items or replenish stock.
Business Event	Admin wants to add a new product or restock.
Primary Actor(s)	Admin
Secondary Actor(s)	None
Pre-condition	Admin is logged into the system.
Post-condition	New product is added, or existing stock is updated.
Failure outcomes	Error message 'Unable to add/update product'.

### Flow of Events:

Step	Actor Action	System Response	Associated File
1	Admin selects 'Add Product' in <b>inventory-management.php</b>	Display product addition form	<b>add-product.php</b>
2	Admin fills out product details & submits	Validate input and add product to database	-
3	-	Display success message and forward to <b>inventory-management.php</b>	<b>inventory-management.php</b>

## 2.6 Use Case: Check product details, availability status, and inventory levels (Read)

Attribute	Description
Use Case	Check product details, availability status, and inventory levels (Read)
Objective	Provide comprehensive product information.
Business Event	Admin checks product details.
Primary Actor(s)	Admin
Secondary Actor(s)	None
Pre-condition	Admin is logged into the system.
Post-condition	Admin views product details.
Failure outcomes	Error message 'Unable to fetch product details'.

### Flow of Events:

Step	Actor Action	System Response	Associated File
1	Admin clicks on a product from list from <b>inventory-management.php</b>	Display detailed product info	<b>inventory-management.php</b>



## 2.7 Use Case: Modify product details, pricing, or discontinue items (Update)

Attribute	Description
Use Case	Modify product details, pricing, or discontinue items (Update)
Objective	Keep product data current and accurate.
Business Event	Admin wants to edit product details.
Primary Actor(s)	Admin
Secondary Actor(s)	None
Pre-condition	Admin is logged into the system.
Post-condition	Product details are updated in the system.
Failure outcomes	Error message 'Failed to update product'.

### Flow of Events:

Step	Actor Action	System Response	Associated File
1	Admin selects 'Update' for a item from 'View Inventory'	Display products edit form with current details	<b>update-inventory.php</b>
2	Admin modifies details & submits	Validate input and update product details in database	-
3	-	Display success message and redirect to <b>inventory-management.php</b>	<b>update-inventory.php</b>

## 2.8 Use Case: Remove products that are no longer available or relevant (Delete)

Attribute	Description
Use Case	Remove products that are no longer available or relevant (Delete)
Objective	Maintain an accurate product catalog.
Business Event	Admin decides to delete a product.
Primary Actor(s)	Admin
Secondary Actor(s)	None
Pre-condition	Admin is logged into the system.
Post-condition	Product is removed from the system.
Failure outcomes	Error message 'Failed to delete product'.

### Flow of Events:

Step	Actor Action	System Response	Associated File
1	Admin selects 'Delete Product'	Ask for confirmation	<b>delete-product.php</b> and <b>delete-product-from-inventory.php</b>
2	Admin confirms deletion	Remove product from the database	-
3	-	Display success message on <b>inventory-management.php</b>	<b>inventory-management.php</b>

## 2.9 Use Case: Add new Order (Create)

Attribute	Description
Use Case	Add or Create new Order (Create)
Objective	Create order with products from inventory.
Business Event	Customer wants to buy a product.
Primary Actor(s)	Customer
Secondary Actor(s)	Admin
Pre-condition	Customer is logged into the system and has purchased the product.
Post-condition	Submitted the payment for order confirmation
Failure outcomes	Error message 'Failed to create order.'

### Flow of Events:

Step	Actor Action	System Response	Associated File
1	Customer selects 'Place Order' for a order	Display checkout form	<b>checkout.php</b>
2	Customer enters review details & submits	Validate input and save review to database	-
3	-	Display success message and redirect to <b>order-receipt.php</b>	<b>order-receipt.php</b>

## 2.10 Use Case: View Recent Orders Placed. (Read)

Attribute	Description
Use Case	View Recent Orders Placed (Read)
Objective	Allow customers to view orders created.
Business Event	Customer wants to view order details.
Primary Actor(s)	Customer
Secondary Actor(s)	None
Pre-condition	Customer placed the order.
Post-condition	Customer views recent orders.
Failure outcomes	Error message 'Order not found.'

### Flow of Events:

Step	Actor Action	System Response	Associated File
1	Customer clicks on a <b>order-history.php</b>	Display product details along with all reviews	<b>order-history.php</b>

### 2.11 Use Case: Update status of order to Processed. (Update)

Attribute	Description
Use Case	Update status of order to Processed. (Update)
Objective	Allow admin to modify order status.
Business Event	Admin wants to process orders.
Primary Actor(s)	Admin
Secondary Actor(s)	None
Pre-condition	Customer placed an order.
Post-condition	Admin processed the order.
Failure outcomes	Error message 'Order not found.'

#### Flow of Events:

Step	Actor Action	System Response	Associated File
1	Admin selects 'Order Processing'	Display review edit form with previous details	<b>order-process.php</b>
2	Customer modifies review & submits	Validate input and update review in database	-
3	-	Display success message and redirect to <b>profile.php</b>	<b>profile.php</b>

## 2.12 Use Case: Customer wants to cancel an order. (Delete)

Attribute	Description
Use Case	Customer wants to cancel an order. (Delete)
Objective	Allow customers to remove the order.
Business Event	Customer wants to delete their order.
Primary Actor(s)	Customer
Secondary Actor(s)	None
Pre-condition	Customer is logged in and has previously placed an order.
Post-condition	Order is cancelled.
Failure outcomes	Error message 'Order not found.'

### Flow of Events:

Step	Actor Action	System Response	Associated File
1	Customer selects 'Cancel Order'	Ask for confirmation	cancel-order.php
2	Customer confirms deletion	Remove review from the database	-
3	-	Display success message on <b>profile.php</b>	<b>profile.php</b>

### 2.13 Use Case: Add a product to wishlist/favorites (Create)

Attribute	Description
Use Case	Add a product to wishlist/favorites (Create)
Objective	Allow customers to bookmark their favorite products for future reference or purchase.
Business Event	Customer wants to save a product to their wishlist.
Primary Actor(s)	Customer
Secondary Actor(s)	None
Pre-condition	Customer is logged into the system.
Post-condition	Product is added to customer's wishlist.
Failure outcomes	Error message 'Failed to add to wishlist'.

#### Flow of Events:

Step	Actor Action	System Response	Associated File
1	Customer clicks on 'Add to Wishlist' for a product	Validate and add product to customer's wishlist in the database	update-wishlist.php
2	-	Display success message or wishlist icon update	product-detail.php

## 2.14 Use Case: View all products in the wishlist/favorites (Read)

Attribute	Description
Use Case	View all products in the wishlist/favorites (Read)
Objective	Allow customers to see all their bookmarked products.
Business Event	Customer wants to view their wishlist.
Primary Actor(s)	Customer
Secondary Actor(s)	None
Pre-condition	Customer is logged into the system.
Post-condition	Customer views all products in their wishlist.
Failure outcomes	Error message 'Failed to fetch wishlist'.

### Flow of Events:

Step	Actor Action	System Response	Associated File
1	Customer clicks on 'My Wishlist'	Display all products saved in the customer's wishlist	wishlist.php



## 2.15 Use Case: Move a product from wishlist to cart (Update)

Attribute	Description
Use Case	Move a product from wishlist to cart (Update)
Objective	Allow customers to move products they wish to purchase from their wishlist to their shopping cart.
Business Event	Customer decides to purchase a product from their wishlist.
Primary Actor(s)	Customer
Secondary Actor(s)	None
Pre-condition	Customer is logged into the system and the product is in their wishlist.
Post-condition	Product is added to the shopping cart and removed from the wishlist.
Failure outcomes	Error message 'Failed to move product to cart'.

### Flow of Events:

Step	Actor Action	System Response	Associated File
1	Customer clicks on 'Move to Cart' for a product in wishlist	Validate, remove product from wishlist and add to cart in the database	<b>add-to-cart.php</b>
2	-	Display success message and update cart icon	<b>product-detail.php</b>

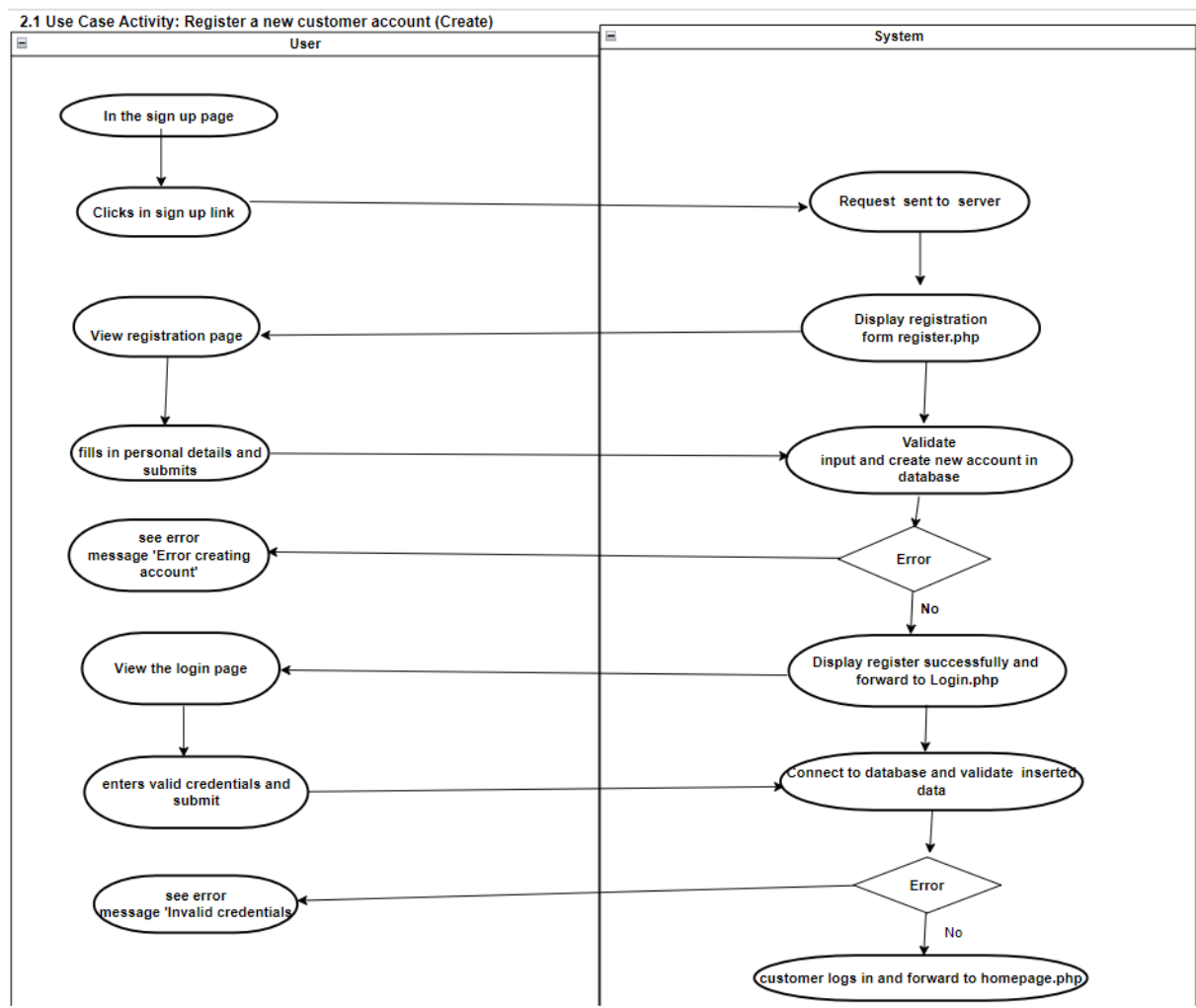
## 2.16 Use Case: Remove a product from the wishlist/favorites (Delete)

Attribute	Description
Use Case	Remove a product from the wishlist/favorites (Delete)
Objective	Allow customers to manage their bookmarked products.
Business Event	Customer wants to remove a product from their wishlist.
Primary Actor(s)	Customer
Secondary Actor(s)	None
Pre-condition	Customer is logged into the system and the product is in their wishlist.
Post-condition	Product is removed from the customer's wishlist.
Failure outcomes	Error message 'Failed to remove from wishlist'.

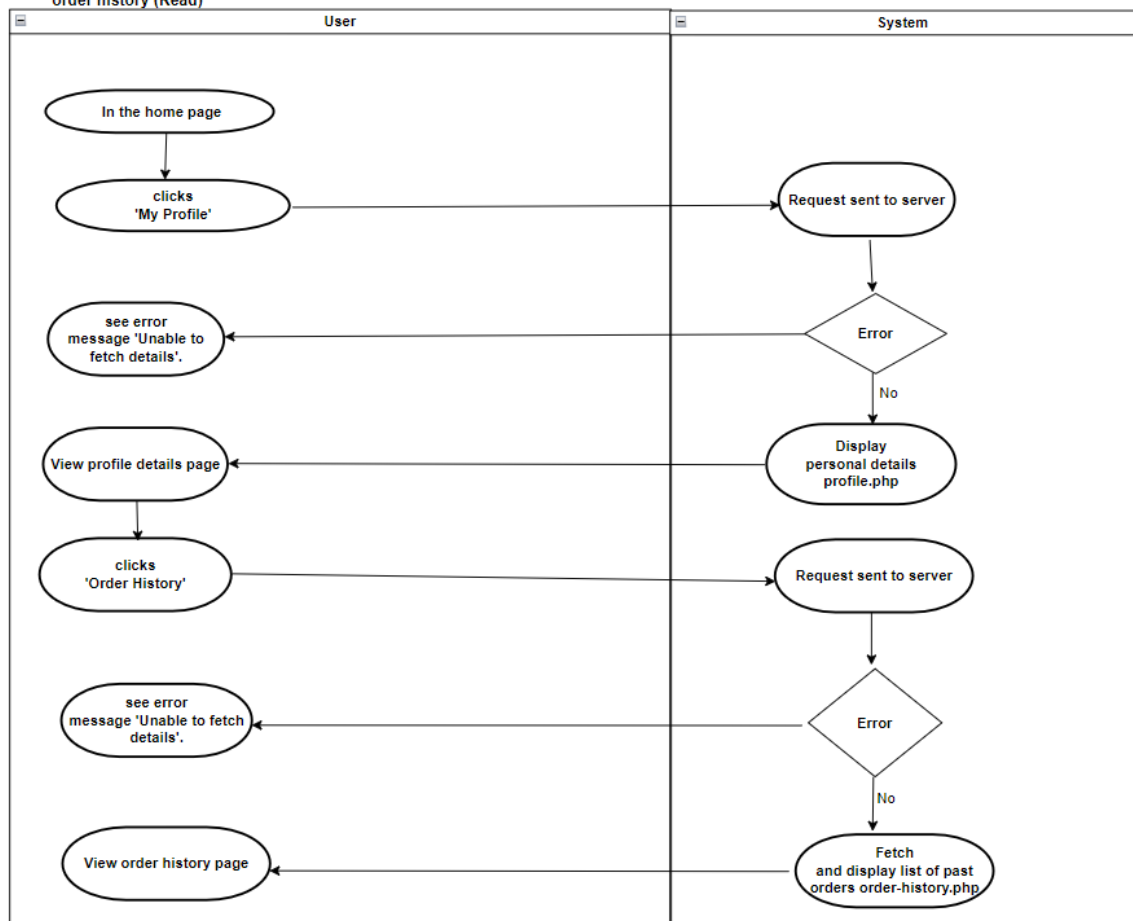
### Flow of Events:

Step	Actor Action	System Response	Associated File
1	Customer clicks on 'Remove from Wishlist' for a product	Validate and remove product from the customer's wishlist in the database	<b>update-wishlist.php</b>
2	-	Display success message or wishlist icon update	<b>product-detail.php</b>

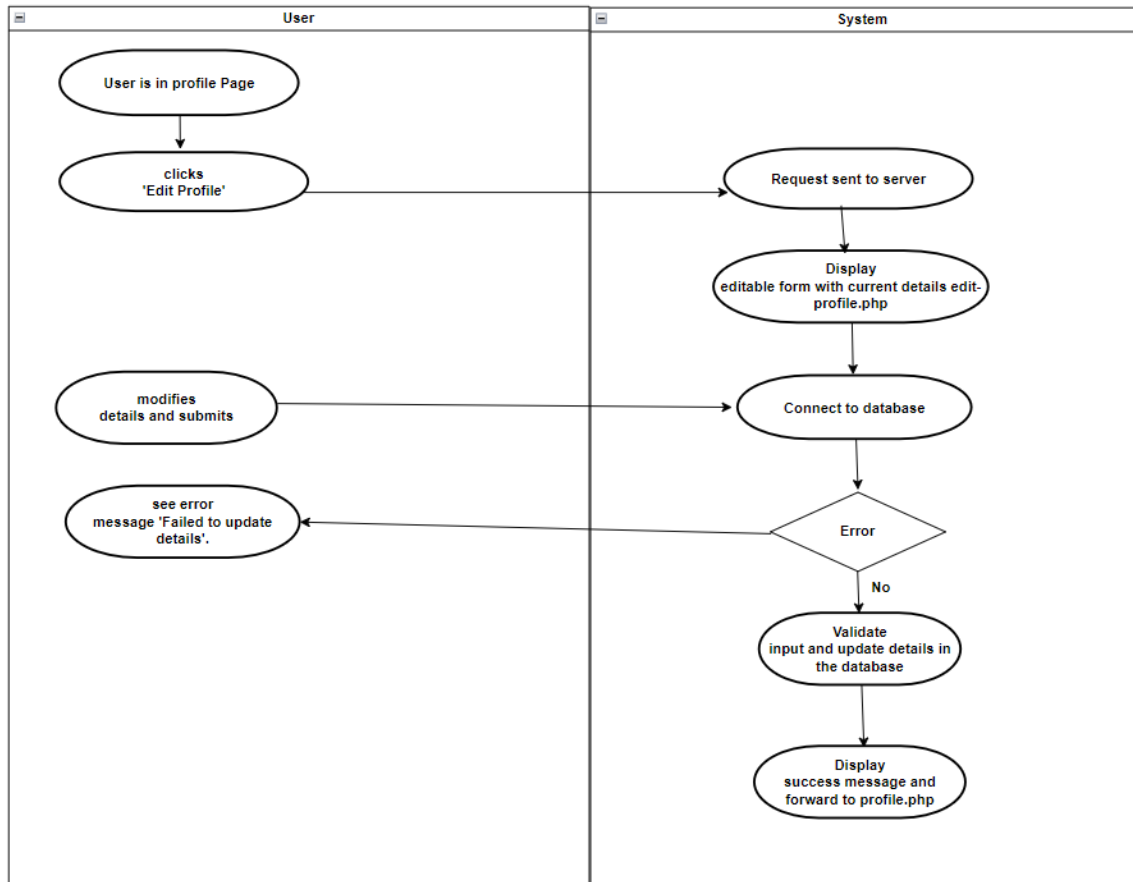
### 3. Use Case Activity Diagrams



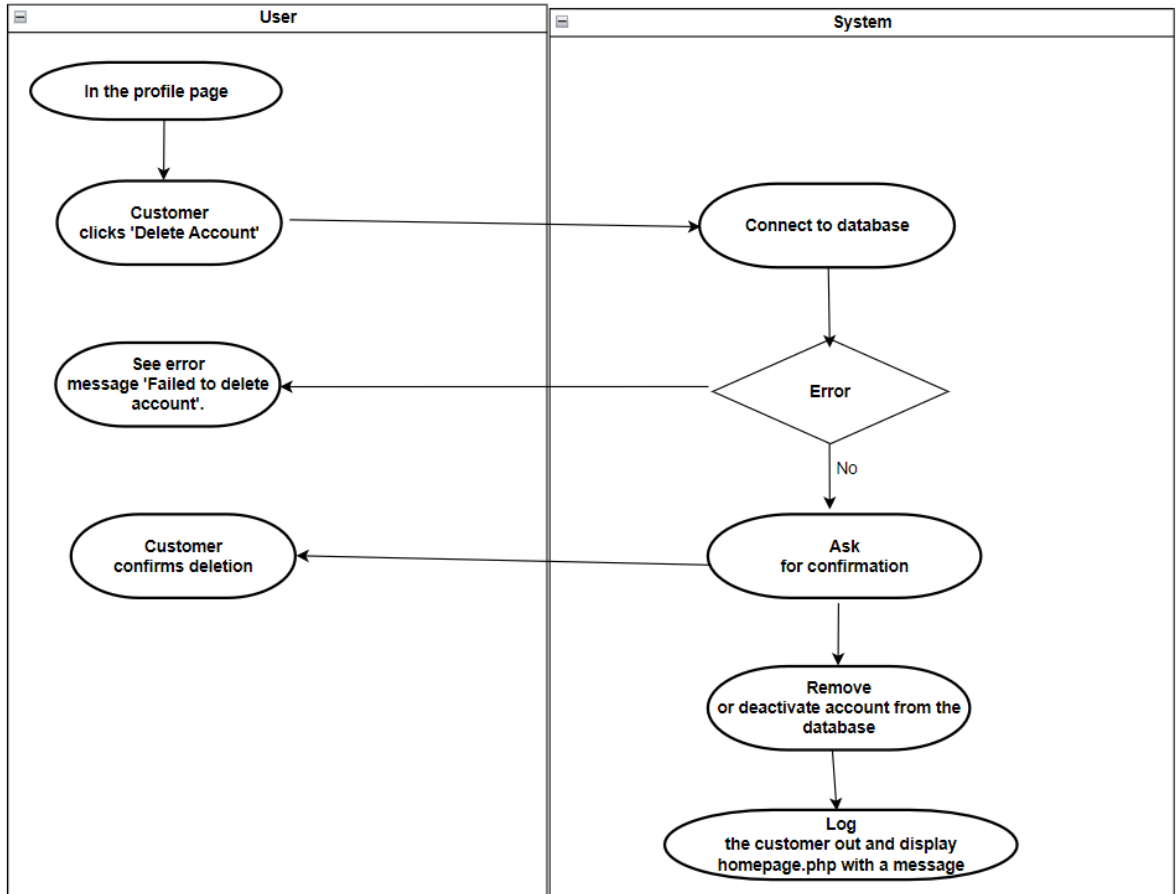
2.2 Use Case: View personal details and order history (Read)



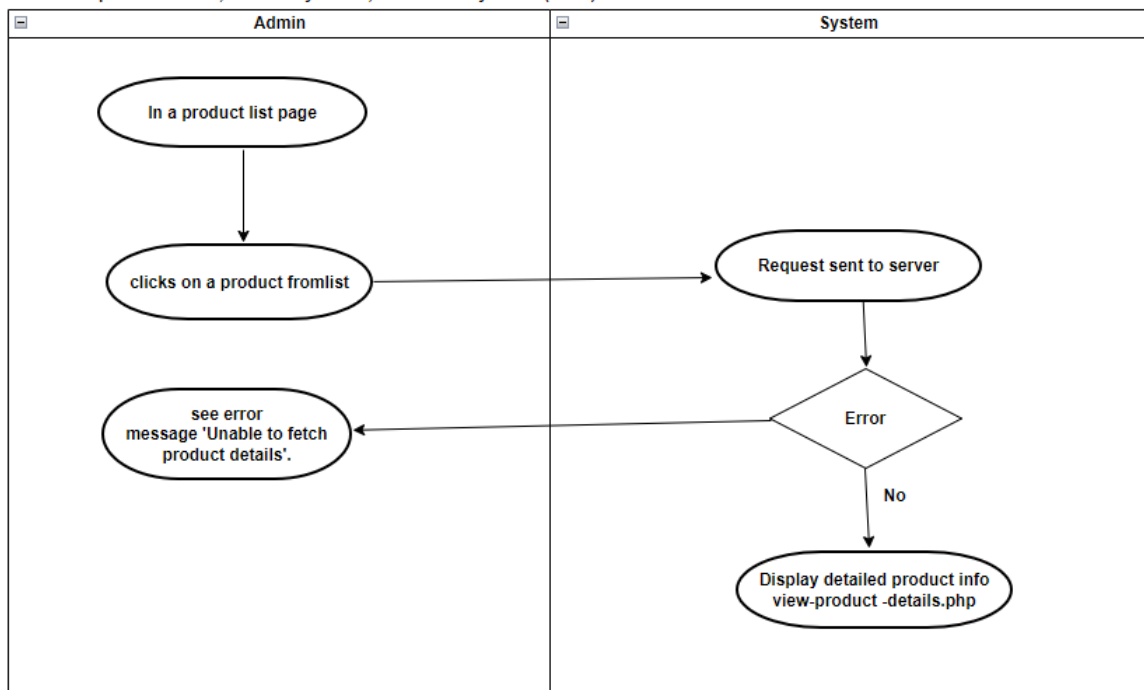
2.3 Use Case Activity: Modify personal information (Update)



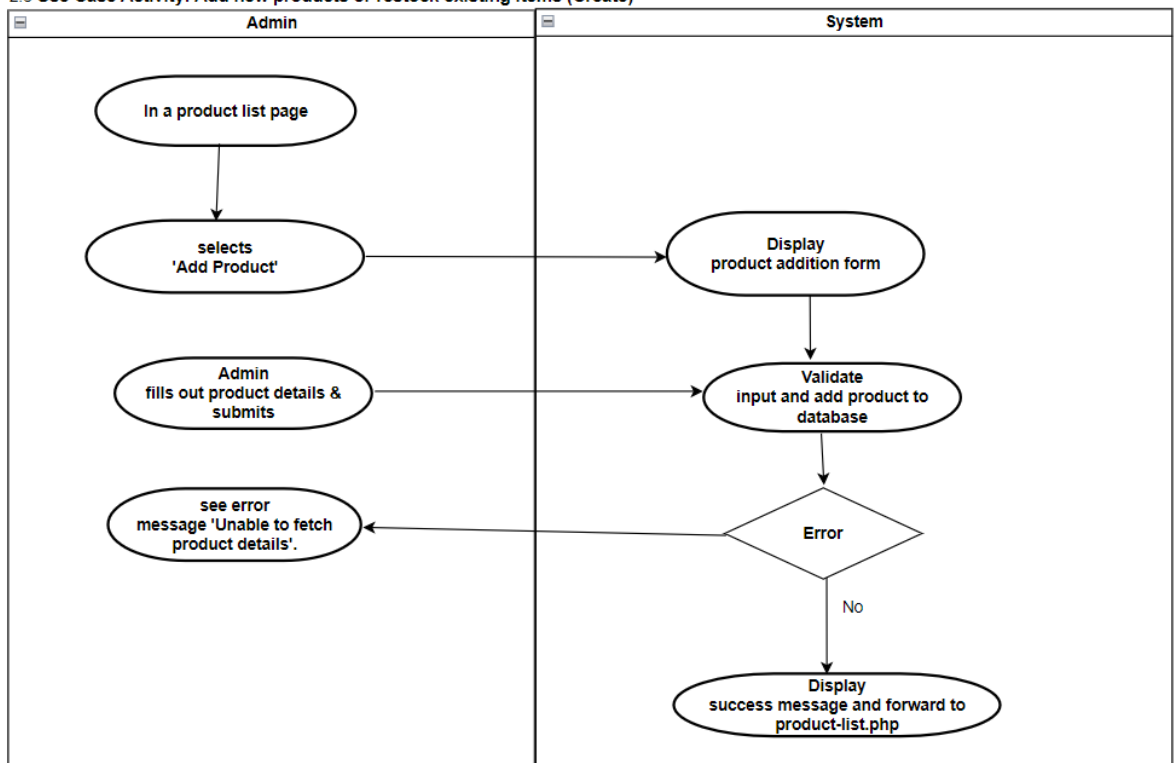
#### 2.4 Use Case Activity: Delete or deactivate an account(Delete)



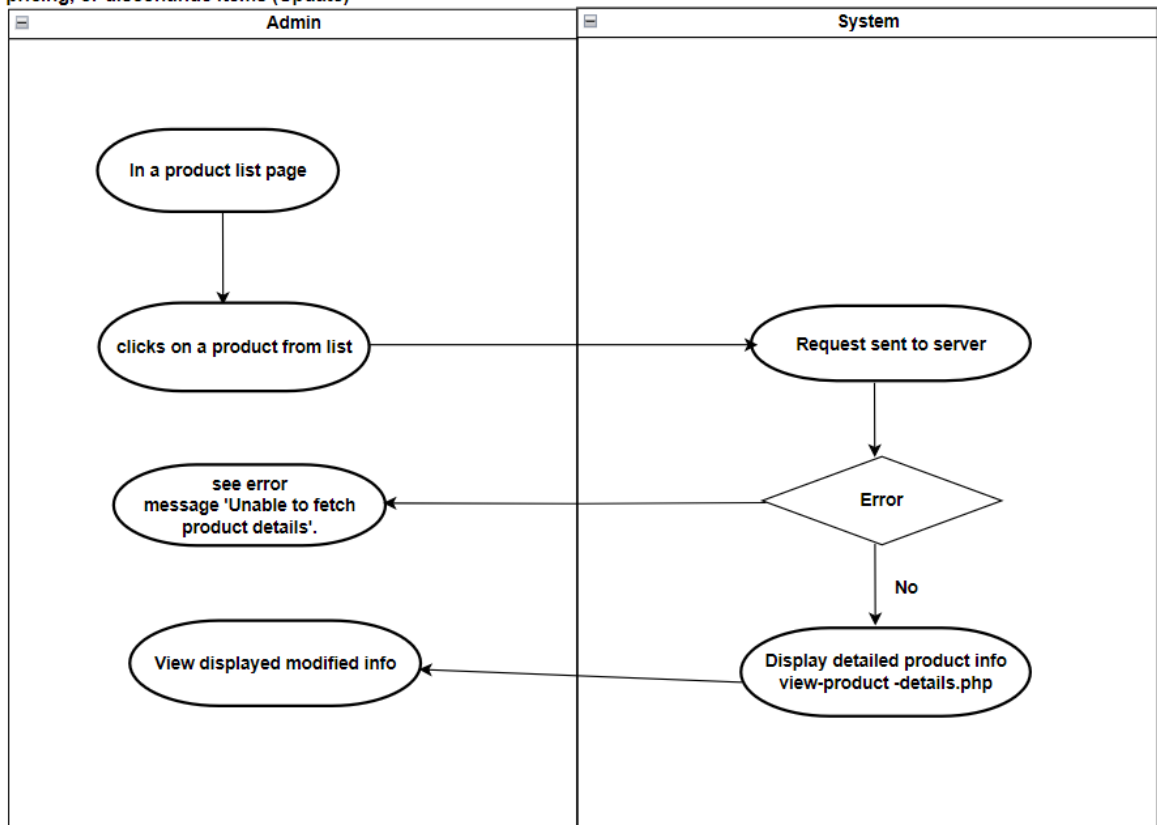
#### 2.5 Check product details, availability status, and inventory levels (Read)



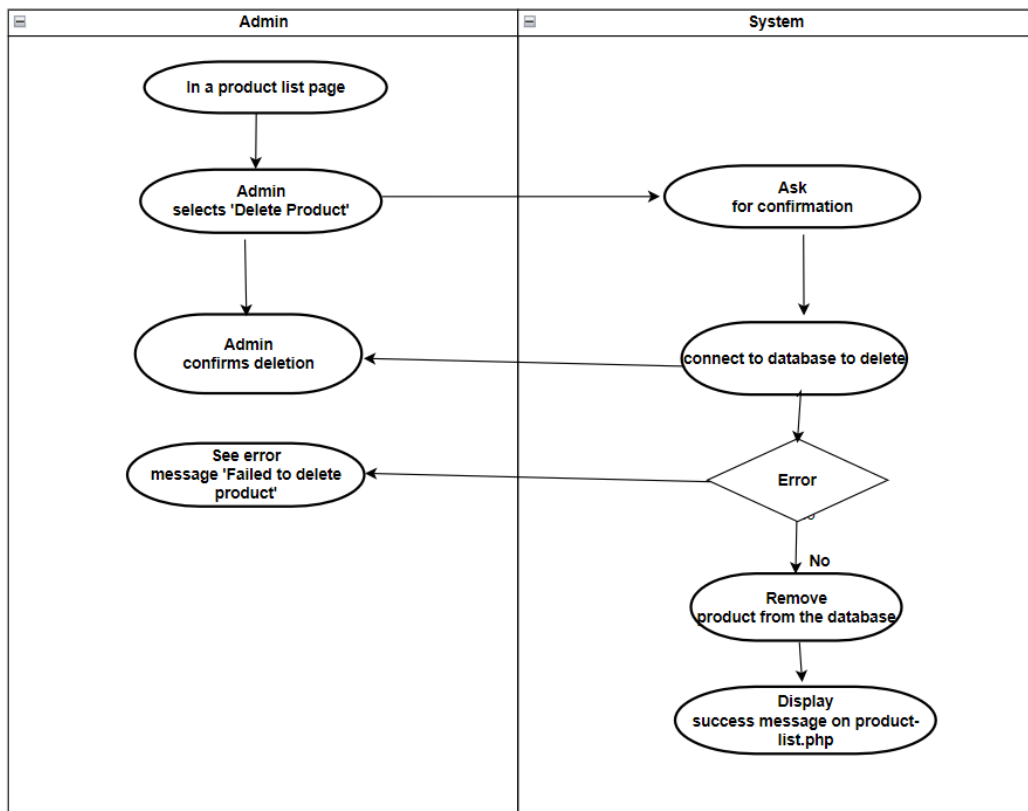
## 2.6 Use Case Activity: Add new products or restock existing items (Create)



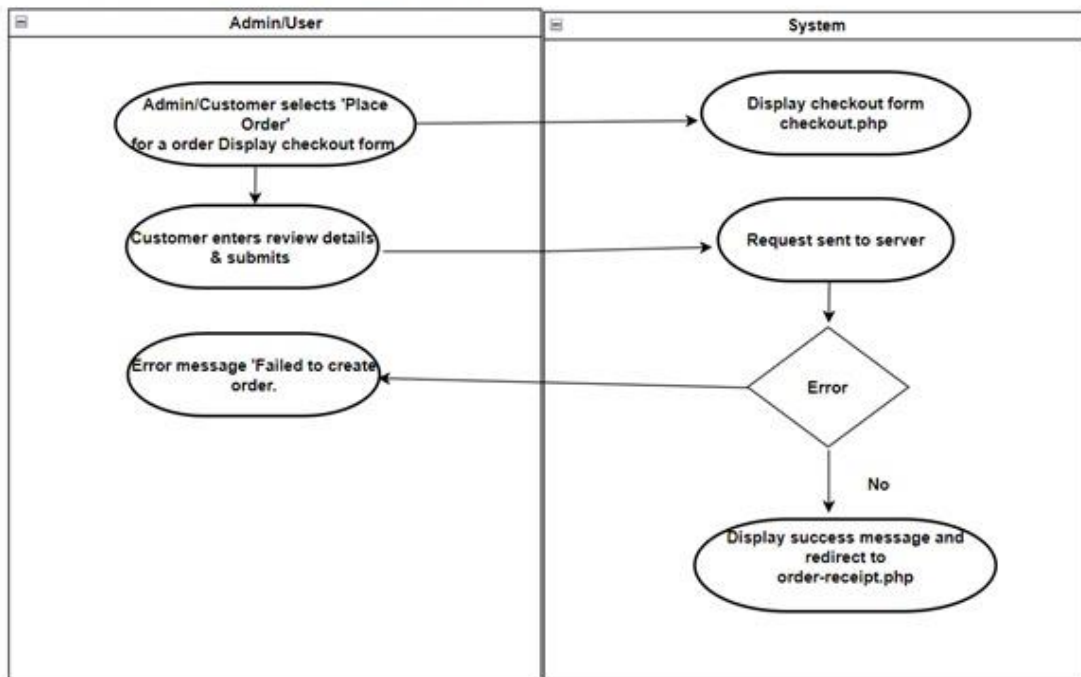
## 2.7 Use Case Activity: Modify product details, pricing, or discontinue items (Update)



## 2.8 Use Case Activity: Remove products that are no longer available or relevant (Delete)

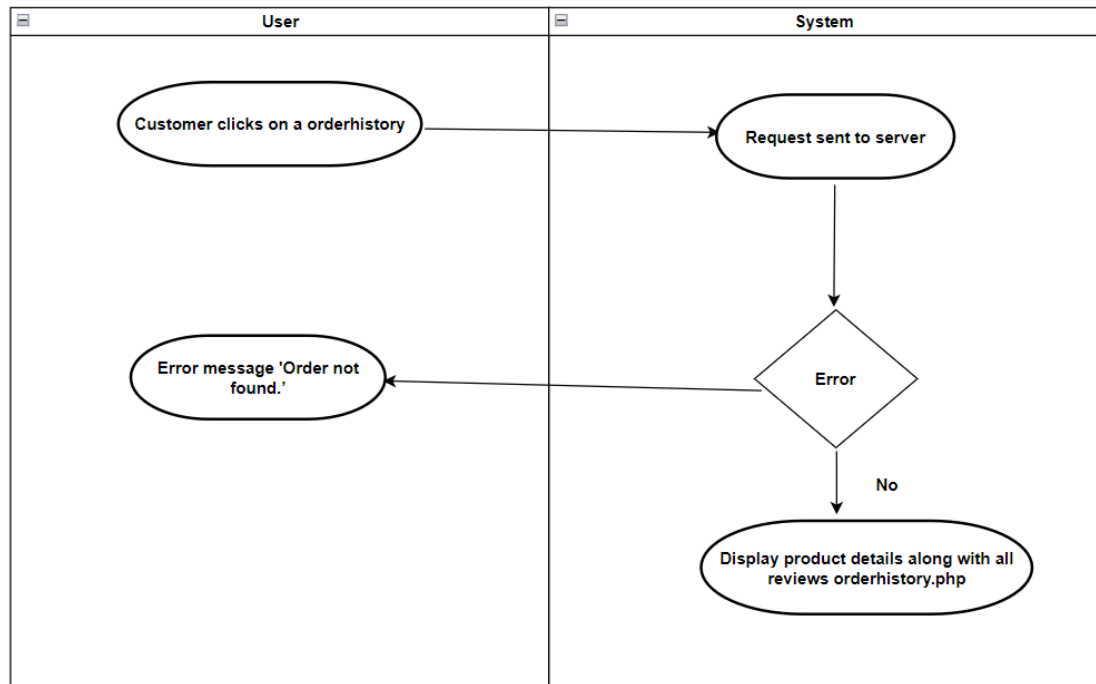


## 2.9 Use Case Activity: Add new order (Create)

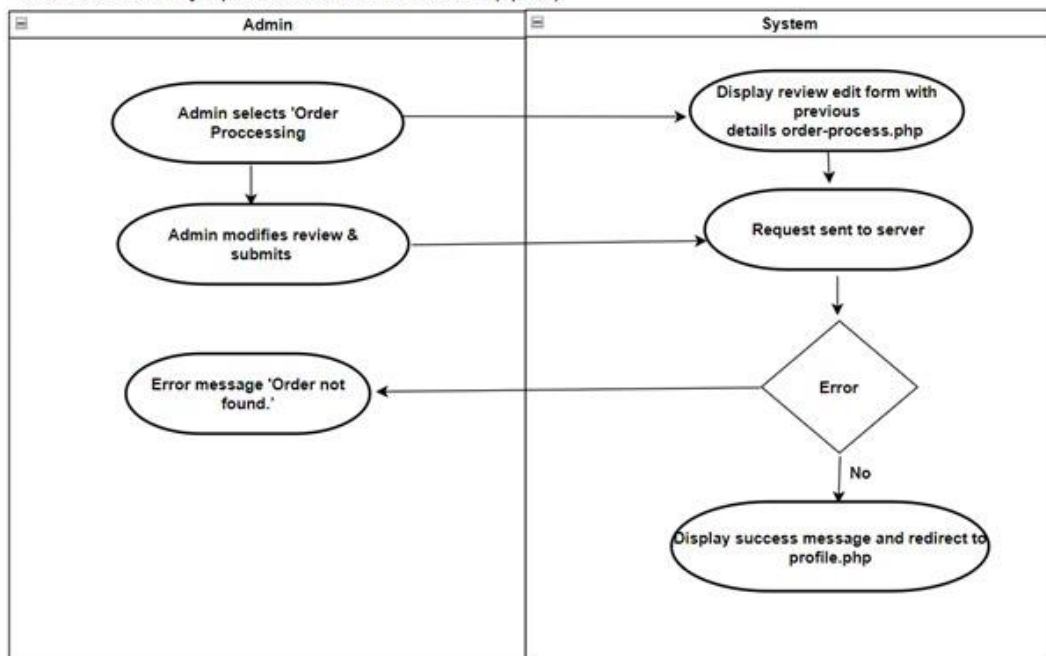




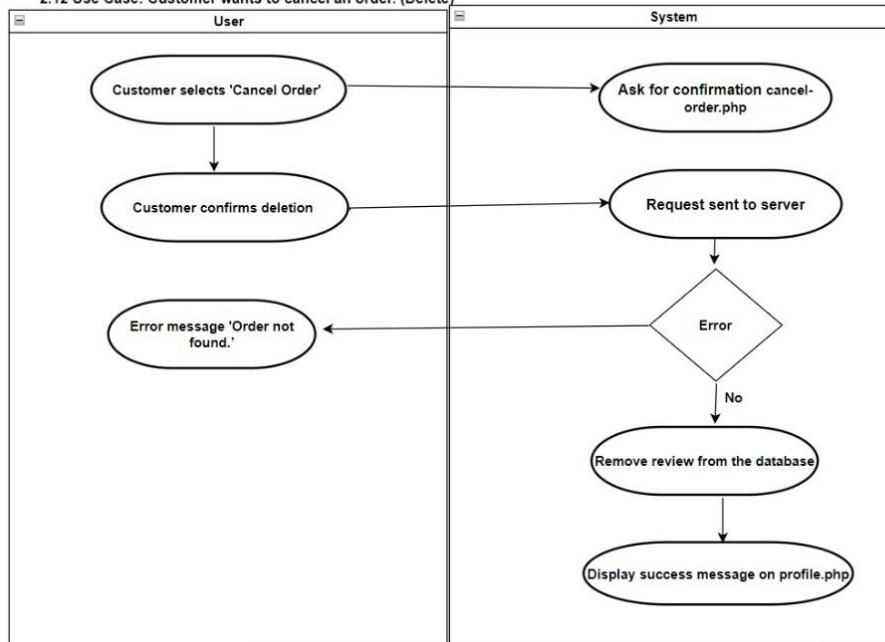
#### 2.10 Use Case Activity: View Recent Orders Placed. (Read)



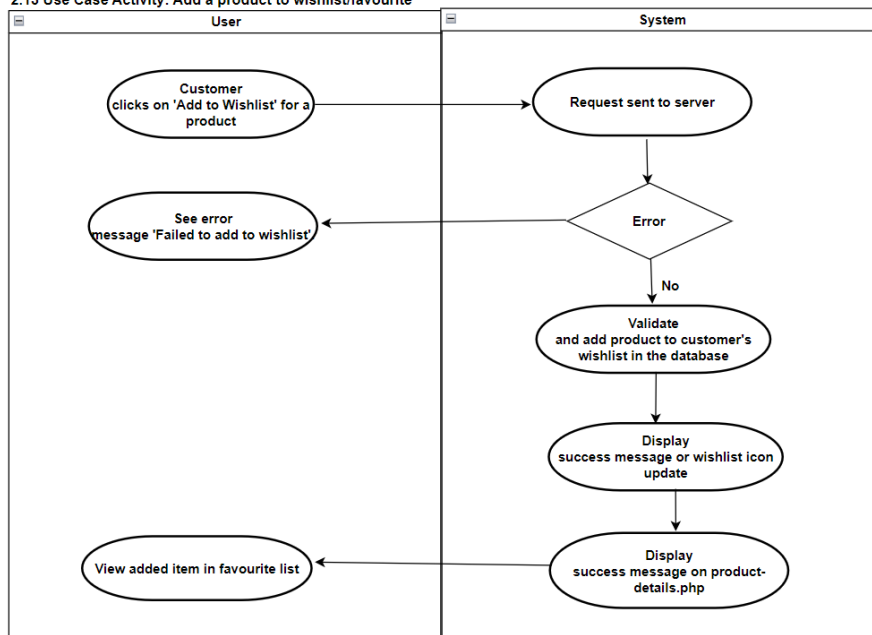
#### 2.11 Use Case Activity: Update status of order to Processed. (Update)



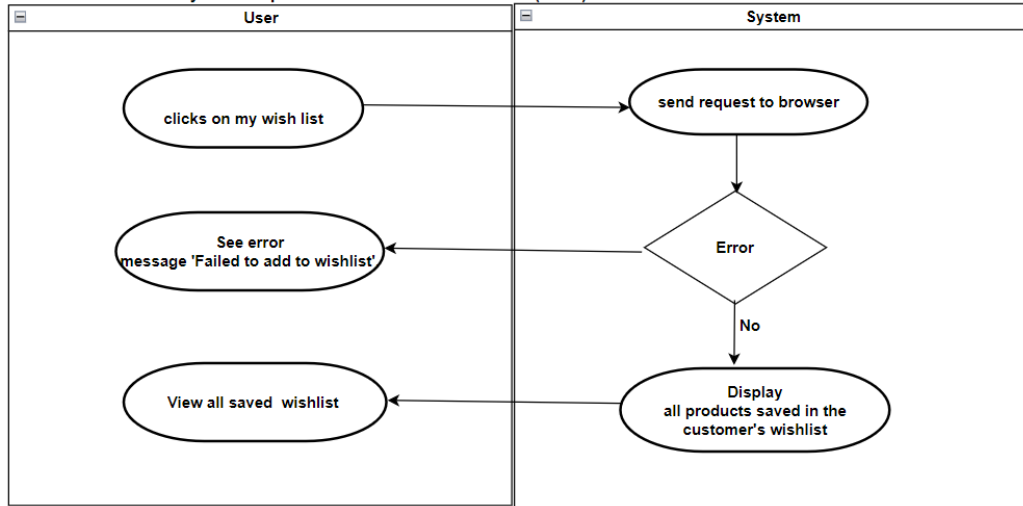
2.12 Use Case: Customer wants to cancel an order. (Delete)



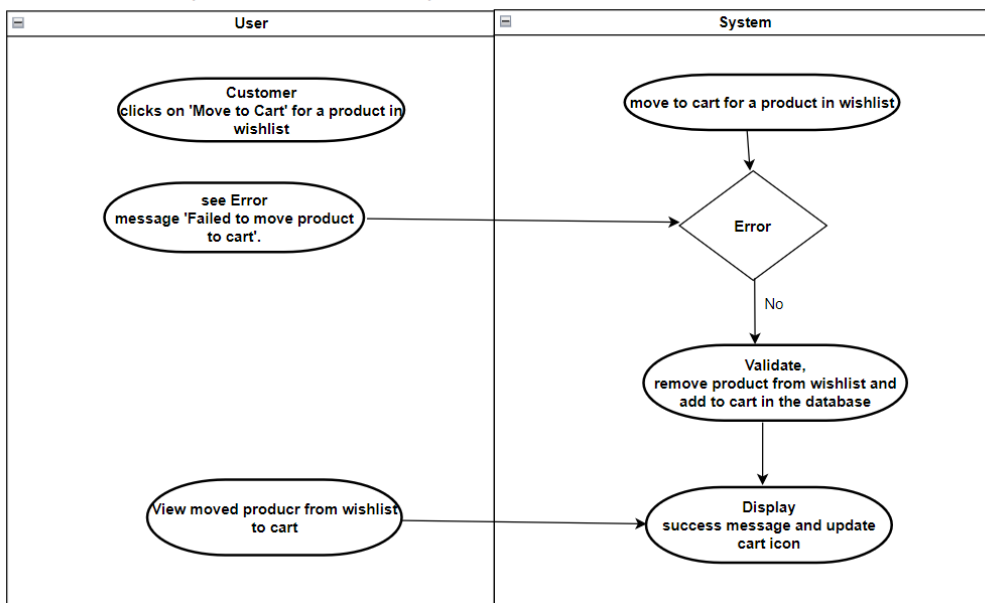
2.13 Use Case Activity: Add a product to wishlist/favourite



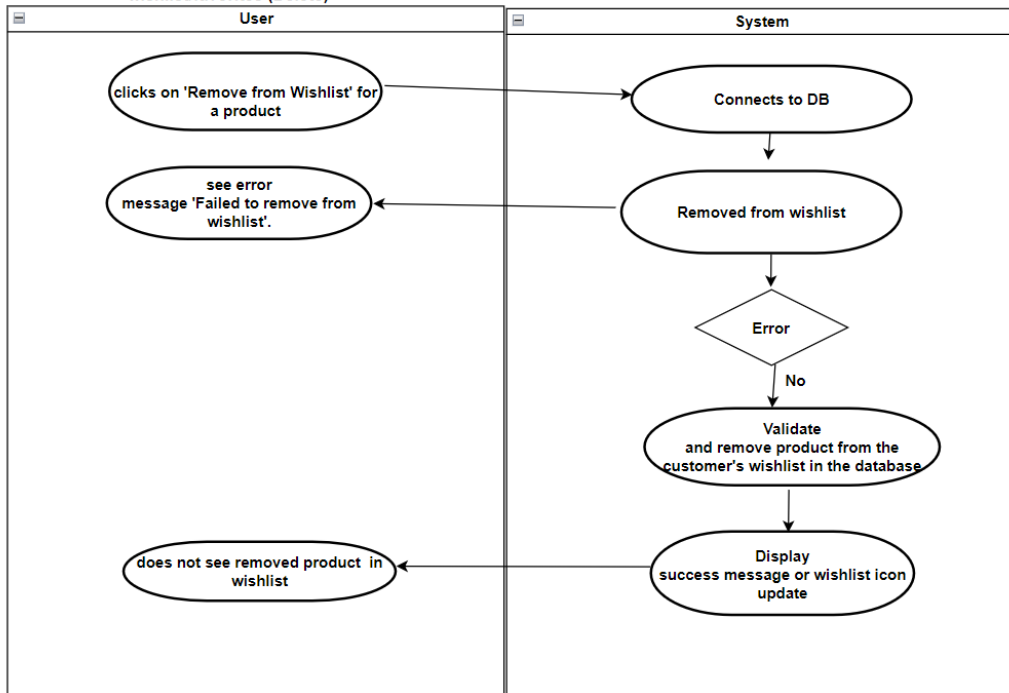
#### 2.14 Use Case: Activity View all products in the wishlist/favorites(Read)



#### 2.15 Use Case: Move a product from wishlist to cart update



2.16 Use Case Activity: Remove a product from the wishlist/favorites (Delete)



## 4. Wire Frame Diagrams

### 4.1 Register user

The wireframe diagram illustrates a user registration page. At the top, a browser window is shown with a single tab titled "Suburban'n Register" and a URL bar containing "https://www.suburbanclothingoutfitters.com/register". The page content is centered and includes the following elements:

- Name**: A text input field with the placeholder text "Enter your full name".
- Username**: A text input field with the placeholder text "Enter a user name".
- Username**: A text input field with the placeholder text "Enter your email address".
- Password**: A text input field with the placeholder text "Enter your password".
- Confirm Password**: A text input field with the placeholder text "Enter your password again".
- Register**: A blue, rounded rectangular button.
- Already have an account?**: A text label followed by a link labeled "Sign In".

## 4.2 Login User

Suburban'n Login

https://www.suburbanclothingoutfitters.com

### Suburban Outfitters

User Name

Password

[Sign in](#) [Register](#)

## 4.3 Product List

Product-list

https://www.suburbanclothingoutfitters.com/product-list

# Suburban Outfitters

Shop By Category

Reorder My Items

Reorder

Favourite List


Hello! Account

Purchase History

Account

Profile

Sign Out




+ save with coupon\*

Jackets and coats for the family.  
Select styles.

Shop Jackets & Coats for the Family

40% off



+ save extra 15% off with coupon\*

Women & Junior Clothes

Shop Women's & Juniors' clothes

50% off

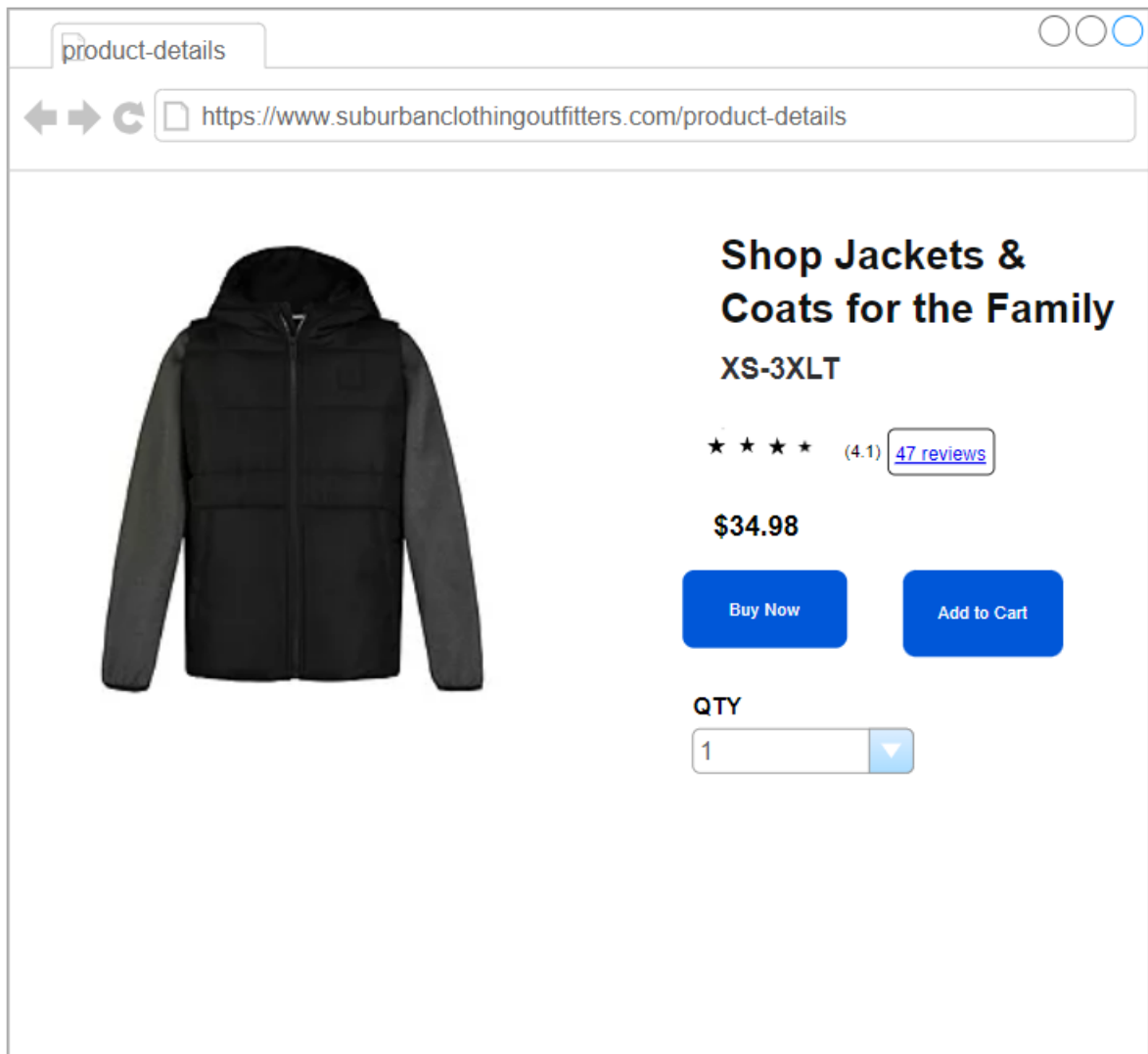
Clearance

Clearance & Closeout

Shop All Clearance

Clearance 70% off

#### 4.4 Product Details





## 4.5 User Details

Suburban'n Profile

← → ↻

https://www.suburbanclothingoutfitters.com/profile

### Hi, Customer

Thanks for being a Suburban's customer for .. years

#### Your personal information:

Full Name

John Smith

Email Address

1234@gmail.com

Address

1245 sand drive

Password

.....

Update

Delete

#### 4.6 Order Details

### Welcome to your Account!

**Customer ID:** 47514584555

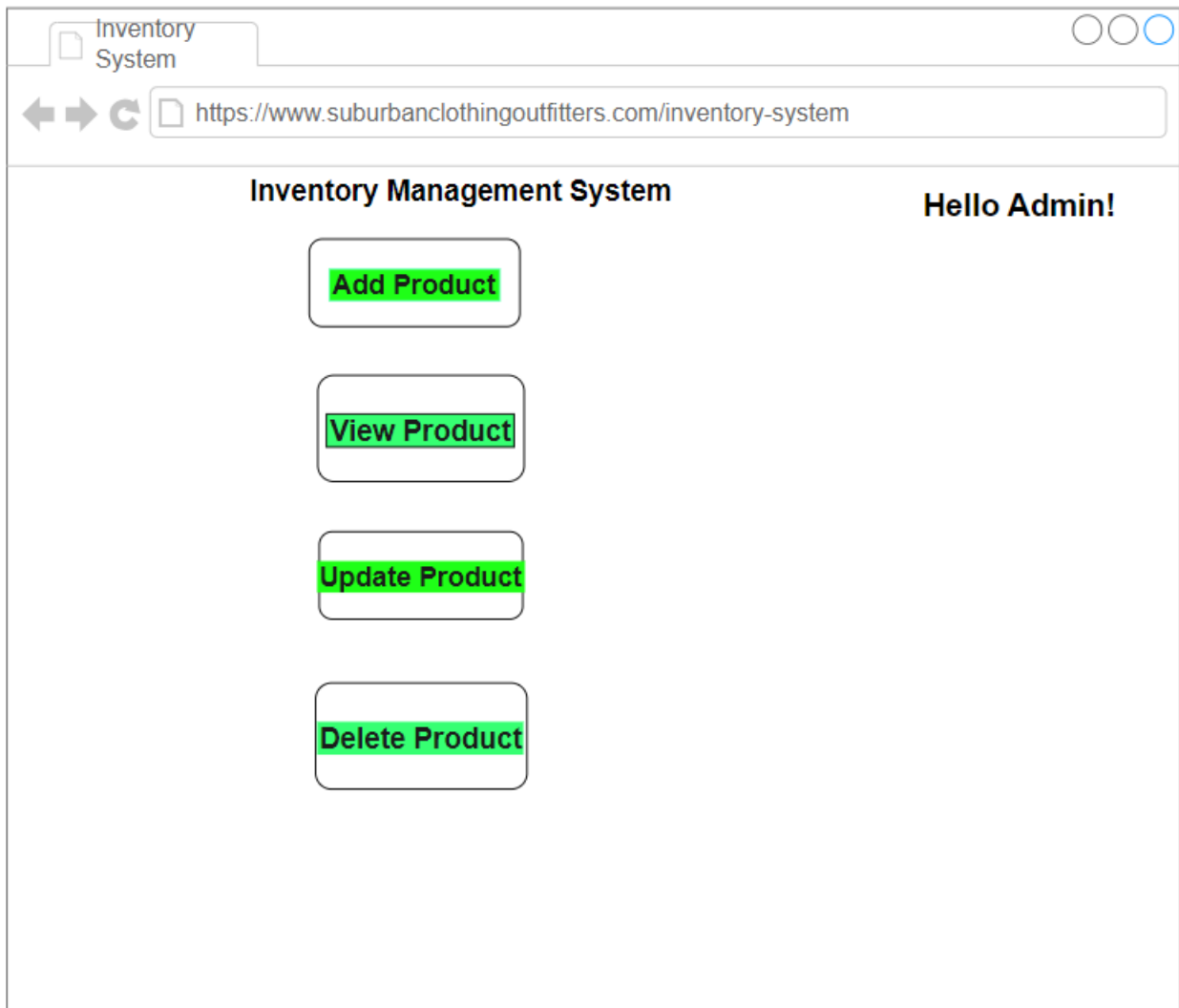
Purchase History:

Payment Method:

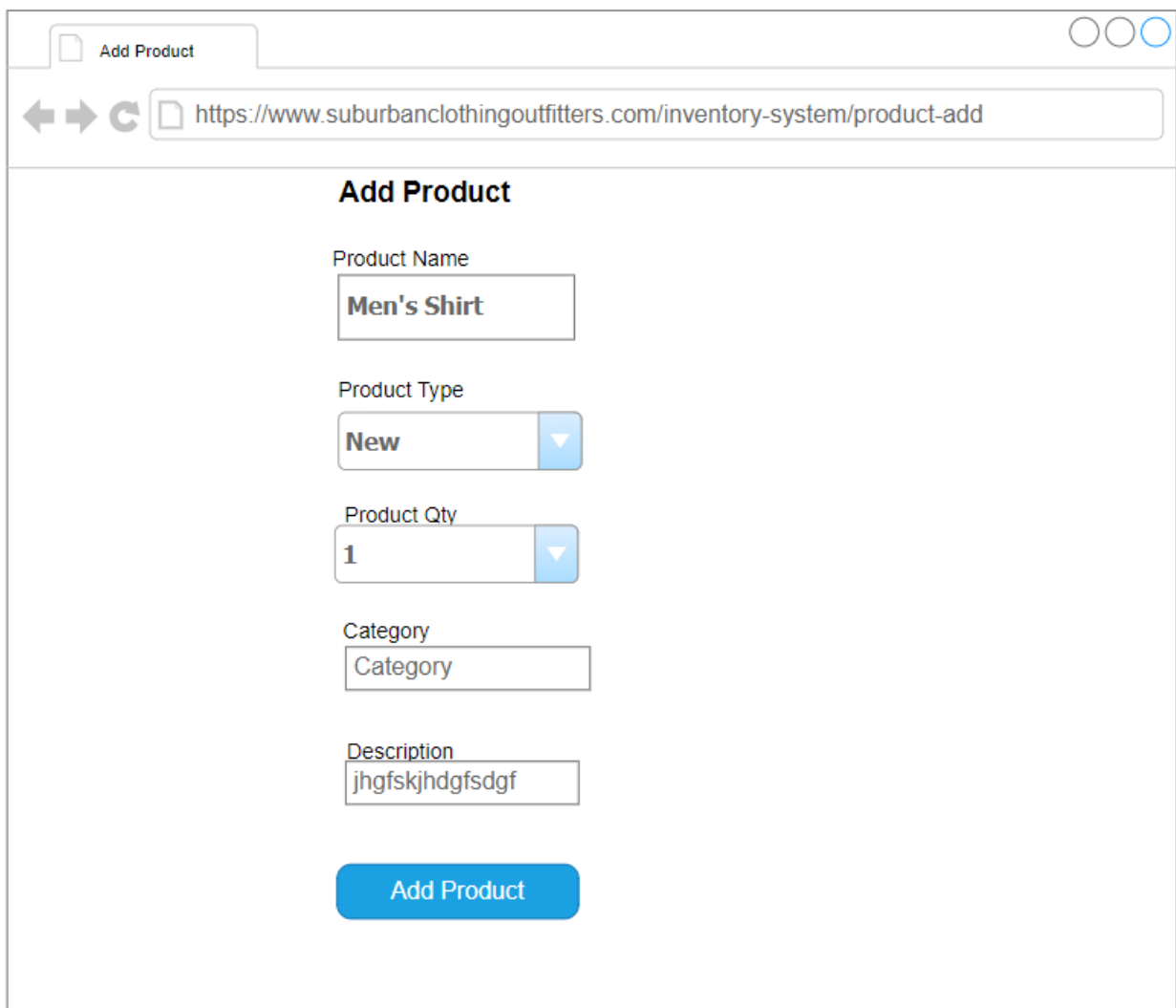
Delete

Deactivate

## 4.7 Inventory System



## 4.8 Add Product



The screenshot shows a web browser window with a single tab titled 'Add Product'. The address bar displays the URL 'https://www.suburbanclothingoutfitters.com/inventory-system/product-add'. The page content features a form titled 'Add Product' with the following fields:

- Product Name:** A text input field containing 'Men's Shirt'.
- Product Type:** A dropdown menu with 'New' selected.
- Product Qty:** A dropdown menu with '1' selected.
- Category:** A text input field containing 'Category'.
- Description:** A text input field containing 'jhgfskjhdgfsdgf'.

At the bottom of the form is a blue button labeled 'Add Product'.

## 4.9 View Product

View Product

https://www.suburbanclothingoutfitters.com/inventory-system/product-view

### View Product

Product details

Men's Shirt

Availability Status

Yes

Category

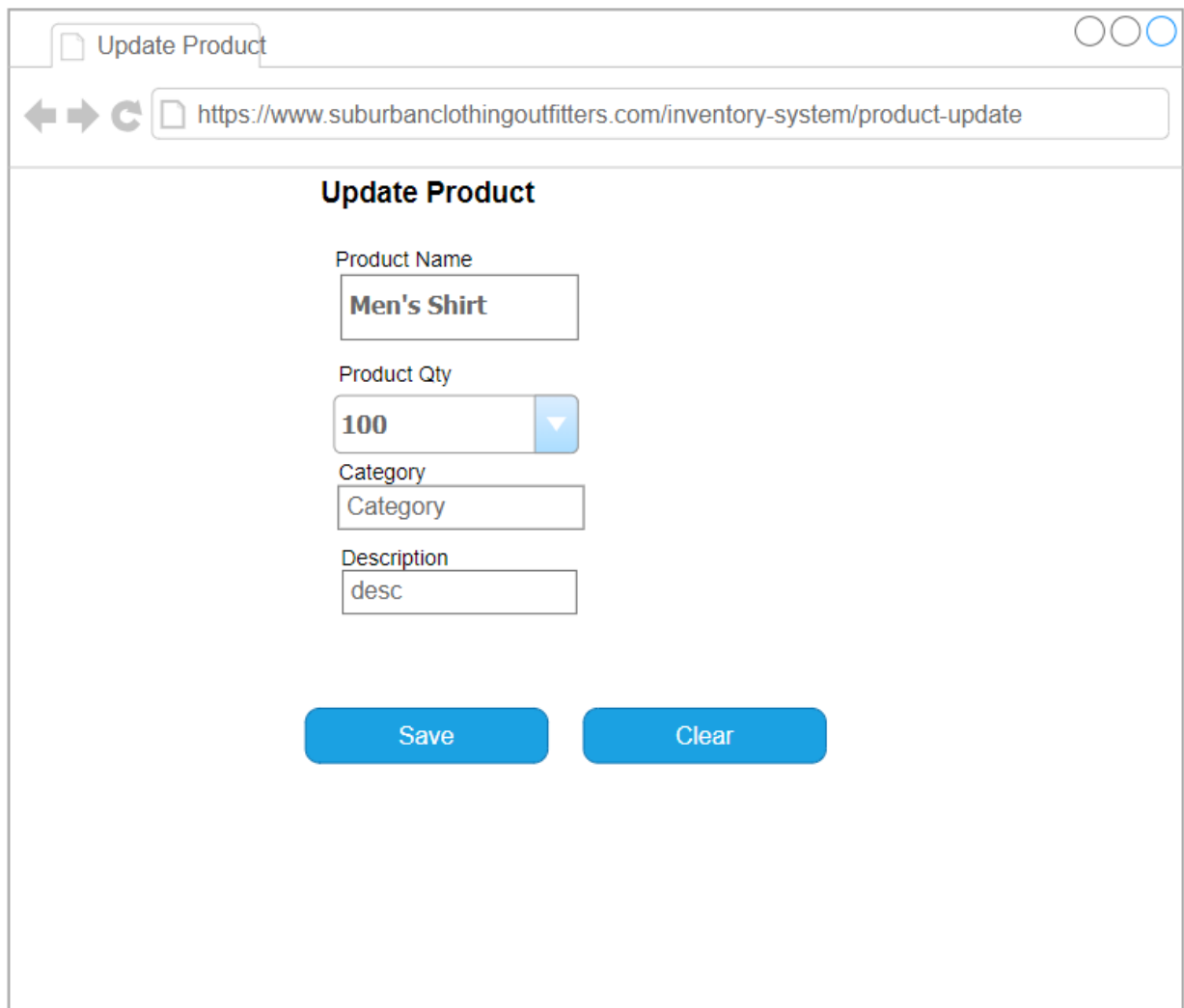
Category

Description

desc

Update Delete

#### 4.10 Update Product



The screenshot shows a web browser window with a single tab titled "Update Product". The address bar displays the URL "https://www.suburbanclothingoutfitters.com/inventory-system/product-update". The main content area features a form titled "Update Product" with the following fields:

- Product Name:** A text input field containing "Men's Shirt".
- Product Qty:** A numeric input field containing "100" with a dropdown arrow on the right.
- Category:** A text input field containing "Category".
- Description:** A text input field containing "desc".

At the bottom of the form are two blue buttons: "Save" and "Clear".

#### 4.11 Delete Product

Product Name

**Men's Shirt**

Product Type

**New**

Product Qty

**100**

Category

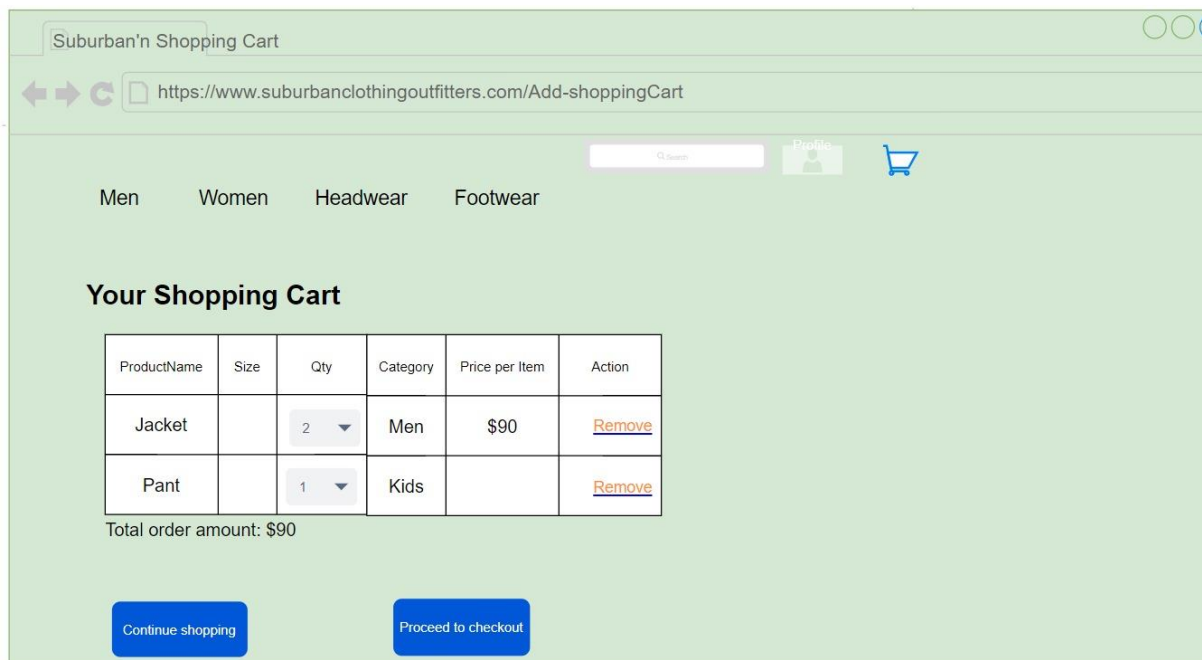
Category

Description

jhgfskjhdgfsdgf

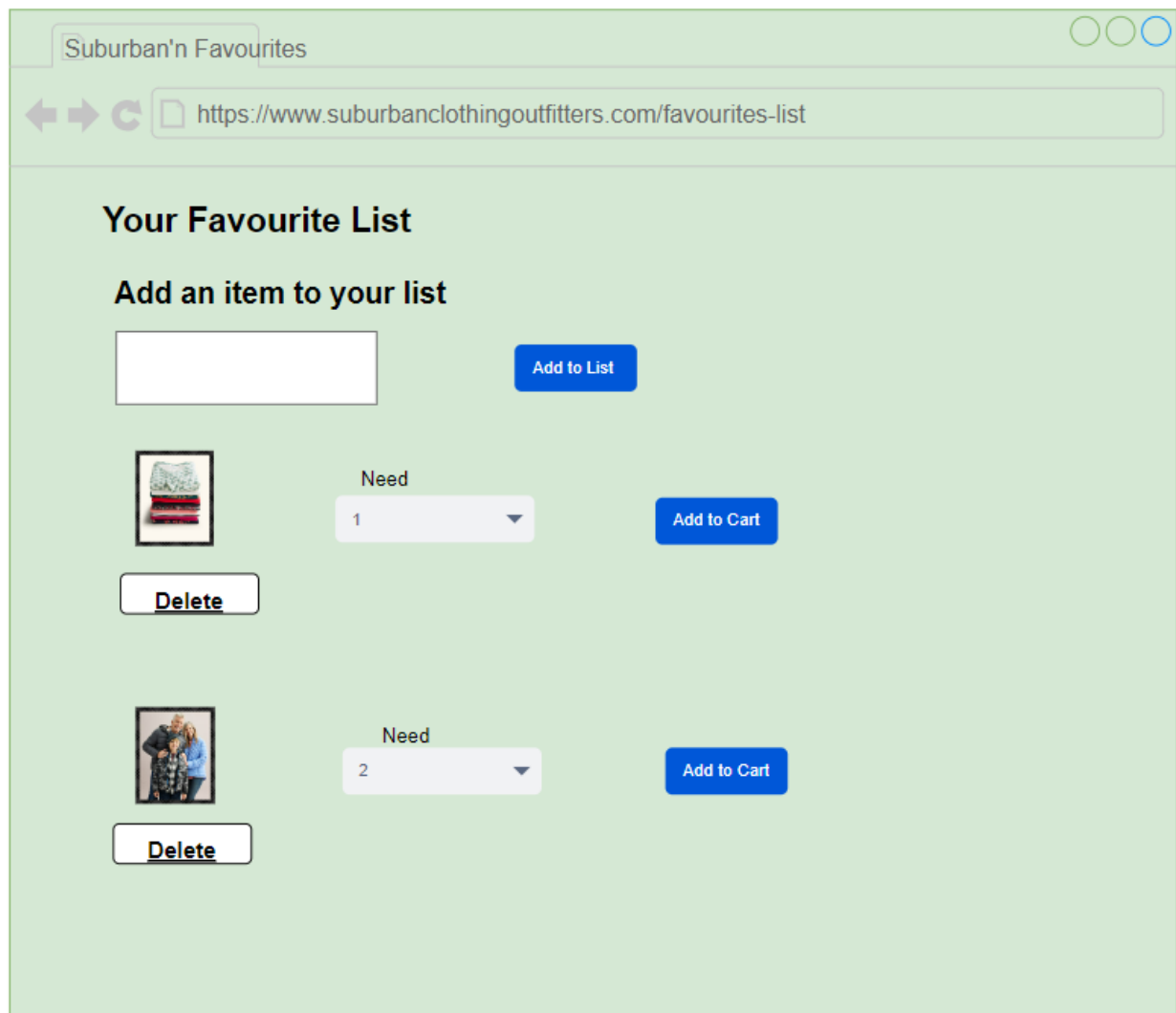
**Delete**

## 4.12 Customer Shopping Cart

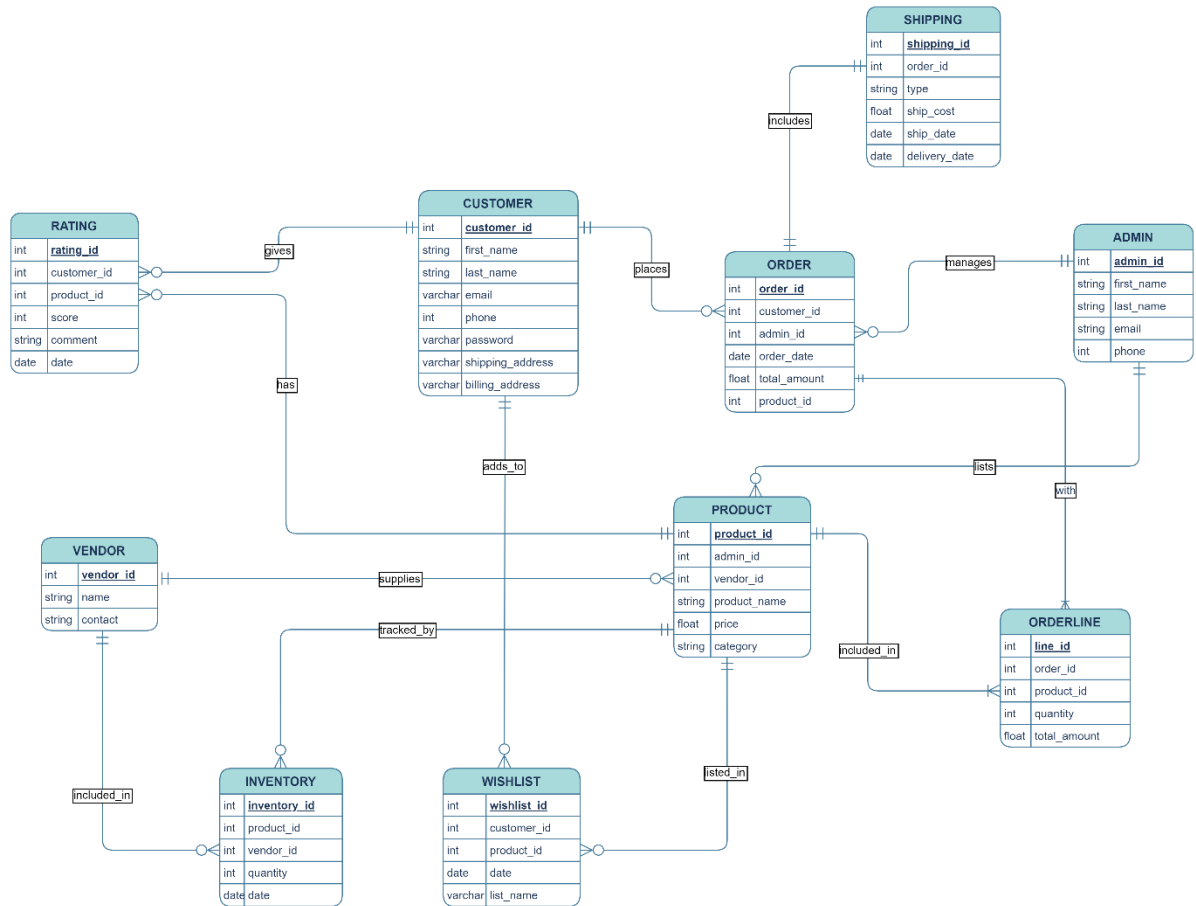




#### 4.13 Customer Wishlist



## 5. ERD



## **Business/Organizational Objectives:**

In developing the Suburban Outfitters online shopping portal, we've closely aligned our application's functionalities with our key business objectives. We've designed an intuitive user interface that simplifies the shopping process, ensuring customers experience seamless navigation and transaction completion. Accurate data management is at the core of our platform, enabling real-time tracking of inventory and orders—a vital feature that ensures customers and admins are informed about product availability and order status promptly.

We've integrated KPI dashboards that offer comprehensive sales reports, aiding in making informed business decisions. Personalization is another cornerstone of our strategy; by leveraging customer data, we provide personalized product recommendations and allow customers to manage their profiles and wishlists, enhancing the tailored shopping experience we promise.

Furthermore, we've embedded ethical practices into our development process, prioritizing privacy, accessibility, and transparency to protect customer data and ensure a trustworthy and inclusive platform. Our agile development approach, coupled with collaborative design efforts, ensures flexibility and the incorporation of feedback, keeping the portal adaptive to customer needs and market changes. By doing so, we aim to solidify Suburban Outfitters' market position, drive sales, and establish a modern shopping experience that resonates with today's consumers.

## **Ethics and Critical Thinking in Developing Front End Pages:**

- Using ethics in web development is crucial. Ethical considerations:
  - Ensuring user data is kept private and secure.
  - Making web pages accessible to all users, including those with disabilities.
  - Not using dark patterns that might deceive or force users into taking actions they didn't intend.
- Critical thinking in web development:
  - Anticipating user needs and designing for a user-friendly experience.
  - Testing and iterating designs based on feedback.
  - Keeping updated with the latest web standards and practices.

*End of Documentation*