



# **LOVE IT BUY IT**

## **E-commerce-Website**

Database Management System  
Semester Project

# ***Table of Contents***

Introduction

Abstract

1. User Manual
  - 1.1 Admin Pannel
  - 1.2 Customer-End
2. ERD
3. Database Schema Documentation
  - 3.1 Schema
  - 3.2 Normalization
- 4.Code Documentation
  - 4.1. CRUD
  - 4.2. Image Load and Store

## **Introduction:**

Welcome to our cutting-edge project in Database Management Systems, where innovation converges with seamless e-commerce functionality. Our platform revolutionizes the online shopping experience by offering an extensive array of products, ranging from everyday grocery items to trendy clothing and accessories. With a user-friendly interface, customers can effortlessly fill their carts and proceed to a secure online payment system, ensuring a smooth transaction process. Additionally, the incorporation of an admin access feature empowers administrators to dynamically manage and update the website's inventory while maintaining comprehensive records of customer interactions. Experience the future of e-commerce with our robust and efficient Database Management Systems project.

## **Abstract:**

This project presents the design and implementation of a robust Database Management System (DBMS) for a user-friendly e-commerce platform "LOVE IT BUY IT". The system facilitates a wide range of product categories, encompassing groceries to apparel, providing users with a comprehensive and seamless shopping experience. Key features include a user-friendly interface that allows customers to populate their carts effortlessly along with admin friendly as he/she can see the stats and figures of their website.

The system architecture employs a relational database model to manage and organize vast product inventories efficiently. A dedicated admin access feature also empowers administrators to dynamically update, add, or remove items, ensuring real-time inventory control. The database records and maintains customer information, contributing to personalized services and data-driven decision-making.

The implementation adopts industry-standard facility measures to provide ease to the customers by providing an online shopping platform and home delivery service. The report delves into the intricacies of the system's architecture, and database schema, highlighting the synergy between efficient e-commerce functionality and robust database management. This project serves as a testament to the fusion of technology and user-centric design, redefining the landscape of contemporary e-commerce platforms.

# 1. User Manual (Love It Buy It)

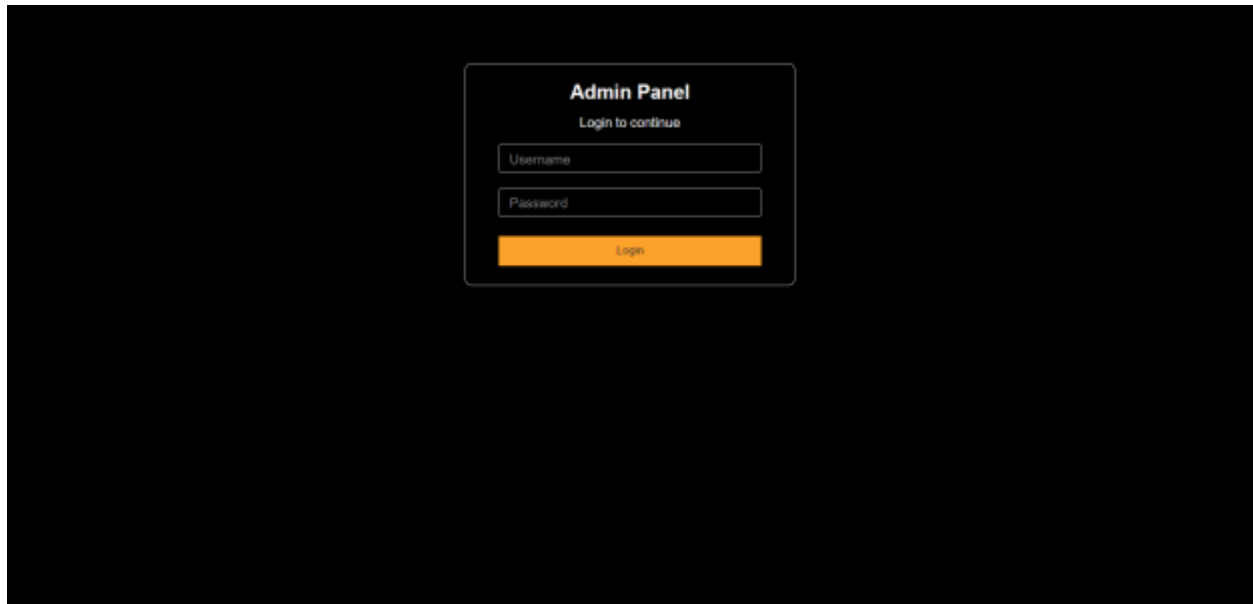
## 1.1 Admin Panel

### Connecting Database:

To connect database to the admin panel, open “includeDB.php” and change the value of \$database to desired database name. To connect database without issues please upload the provided “database.sql” to the database named “shopping\_site”.

### Login to Admin Panel:

Please use “admin” as username and “admin” as password to login to admin panel.

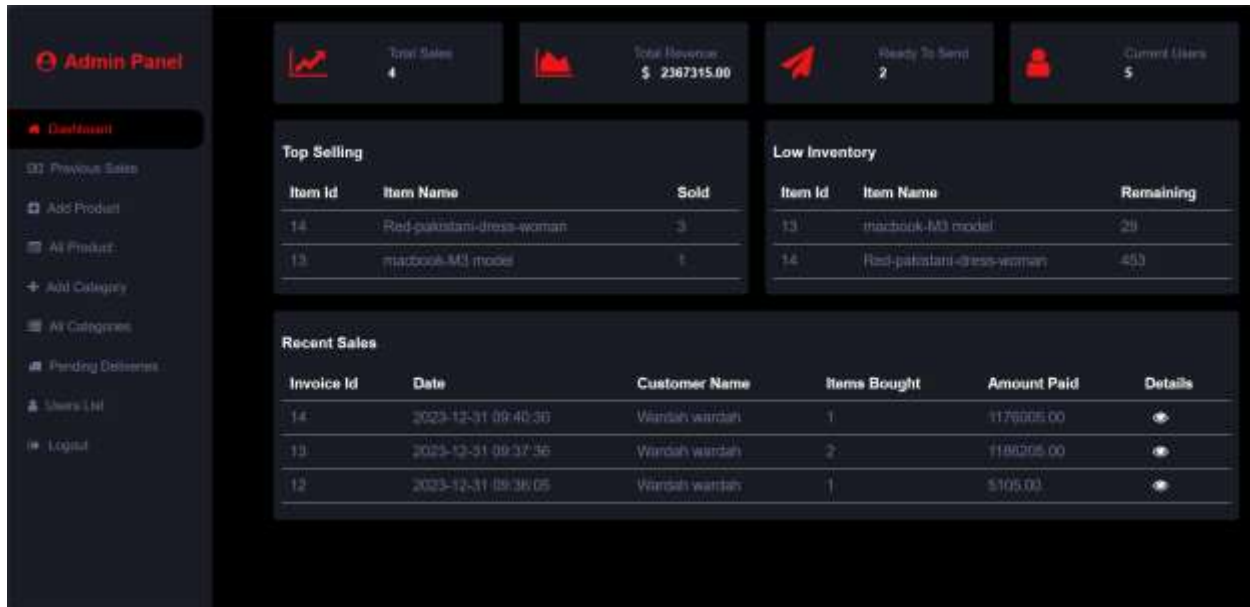


If you wish to change username and/or password, go to admin table of database and update the values.

A screenshot of a MySQL database management tool showing the structure of the 'admin' table. The table has two columns: 'admin\_name' and 'admin\_password'. The 'admin\_name' column is of type 'varchar(20)' with a 'utf8mb4\_0900\_ai\_ci' collation, and the 'admin\_password' column is of type 'varchar(10)' with the same collation. Both columns are 'No' null and have 'None' as the default value. The 'Action' column for each row contains links for 'Change', 'Drop', and 'More'.

### Dashboard:

After logging in, you will be greeted with dashboard panel. Dashboard shows number of total sales made on site till now, total revenue generated, number of pending deliveries and current users.



It also shows top 5 selling products, 5 products with least available items and recent sales which are completed.

### Previous Sales:

This page shows all the completed previous sales made on the site along with who bought it and a button for sale details. Upon clicking the eye icon, you will be redirected to the details page of that sale.

The Previous Sales page displays a table of completed sales. The sidebar navigation is updated to highlight 'Previous Sales'.

**Admin Panel**

- Dashboard
- Previous Sales
- Add Product
- All Product
- Add Category
- All Categories
- Pending Deliveries
- Users List
- Logout

**Past Sales**

Invoice Id	Customer Name	Customer Email	Date	Invoice Details
14	Wardah wardah	naveed3@gmail.com	2023-12-31 09:40:30	👁
13	Wardah wardah	naveed3@gmail.com	2023-12-31 09:37:36	👁
12	Wardah wardah	naveed3@gmail.com	2023-12-31 09:36:05	👁

### Add Product:

New product can be added to product table using this page. You will have to provide details such as product name, product category, price, inventory, description along with 3 product images.

**Admin Panel**

- Dashboard
- Previous Sales
- Add Product**
- All Product
- Add Category
- All Categories
- Pending Deliveries
- User List
- Logout

### Add New Product

Product Name:

Select Category:

Price:

Inventory:

Product Image 1:

Product Image 2:

Product Image 3:

Description:

### All Products:

This page shows all the available products along with an option to edit and discontinue the product.

**Admin Panel**

- Dashboard
- Previous Sales
- All Product
- All Product**
- Add Category
- All Categories
- Pending Deliveries
- User List
- Logout

### Products List

Product Id	Product Name	Price	Sold	Inventory	Discount	Edit	Discontinued
15	VR Gear	1500.00	0	10	0.00	<input type="button" value="Edit"/>	<input type="button" value="Discontinue"/>
14	Red pakistani dress woman	6000.00	3	453	15.00	<input type="button" value="Edit"/>	<input type="button" value="Discontinue"/>
13	macbook-M3 model	600000.00	1	29	2.00	<input type="button" value="Edit"/>	<input type="button" value="Discontinue"/>

### Edit Product:

You can update everything of the product on this page as well as add discount to your product.

The screenshot shows the 'Update Product' form in the Admin Panel. The left sidebar contains the following menu items: Dashboard, Previous Sales, Add Product, All Product, Add Category, All Categories, Pending Deliveries, Users List, and Logout. The 'Add Category' item is highlighted. The main content area is titled 'Update Product' and contains the following fields: Product Name (text input with 'VR Gear'), Select Category (dropdown menu with 'Gaming'), Price (text input with '1500.00'), Inventory (text input with '10'), Discount %age (text input with '0.00'), Product Image 1, Product Image 2, and Product Image 3 (each with a 'Choose File' button and a 'No file chosen' message), and Description (text area with 'Get yourself some virtual reality glasses to escape reality.'). An 'Update' button is located at the bottom right of the form.

### Add Category:

To add more variety of products in your shop, you can add new category on this page. Simply write the new category name and post it.

The screenshot shows the 'Add New Category' form in the Admin Panel. The left sidebar contains the following menu items: Dashboard, Previous Sales, Add Product, All Product, Add Category, All Categories, Pending Deliveries, Users List, and Logout. The 'Add Category' item is highlighted. The main content area is titled 'Add New Category' and contains a single field: Category Name (text input with 'Enter Category name.'). A 'Post' button is located at the bottom right of the form.

### All Categories:

To view all categories, go to all categories page. You can edit the category here as well.

Admin Panel																																																		
<ul style="list-style-type: none"> <li>Dashboard</li> <li>Previous Sales</li> <li>Add Product</li> <li>All Product</li> <li>+ Add Category</li> <li><b>All Categories</b></li> <li>Pending Deliveries</li> <li>Users List</li> <li>Logout</li> </ul>	<h3>Categories List</h3> <table> <tr> <th>Category Id</th><th>Category Name</th><th>Edit</th></tr> <tr><td>1</td><td>Electronics</td><td></td></tr> <tr><td>2</td><td>Clothing</td><td></td></tr> <tr><td>3</td><td>Home &amp; Furniture</td><td></td></tr> <tr><td>4</td><td>Books</td><td></td></tr> <tr><td>5</td><td>Sports &amp; Outdoors</td><td></td></tr> <tr><td>6</td><td>Beauty &amp; Personal Care</td><td></td></tr> <tr><td>7</td><td>Toys &amp; Games</td><td></td></tr> <tr><td>8</td><td>Automotive</td><td></td></tr> <tr><td>9</td><td>Health &amp; Wellness</td><td></td></tr> <tr><td>10</td><td>Accessories</td><td></td></tr> <tr><td>11</td><td>Appliances</td><td></td></tr> <tr><td>12</td><td>Pet Supplies</td><td></td></tr> <tr><td>13</td><td>Outdoor Gear</td><td></td></tr> <tr><td>14</td><td>Office Supplies</td><td></td></tr> <tr><td>15</td><td>Stationery</td><td></td></tr> </table>		Category Id	Category Name	Edit	1	Electronics		2	Clothing		3	Home & Furniture		4	Books		5	Sports & Outdoors		6	Beauty & Personal Care		7	Toys & Games		8	Automotive		9	Health & Wellness		10	Accessories		11	Appliances		12	Pet Supplies		13	Outdoor Gear		14	Office Supplies		15	Stationery	
Category Id	Category Name	Edit																																																
1	Electronics																																																	
2	Clothing																																																	
3	Home & Furniture																																																	
4	Books																																																	
5	Sports & Outdoors																																																	
6	Beauty & Personal Care																																																	
7	Toys & Games																																																	
8	Automotive																																																	
9	Health & Wellness																																																	
10	Accessories																																																	
11	Appliances																																																	
12	Pet Supplies																																																	
13	Outdoor Gear																																																	
14	Office Supplies																																																	
15	Stationery																																																	

### Pending Deliveries:

When you are ready to deliver the product to the customer, simply go to this page and click on truck icon to complete the order. You can check all the pending deliveries here as well as a link to the order details page by clicking on eye icon.

Admin Panel

Dashboard

Previous Sales

Add Product

All Product

+ Add Category

All Categories

Pending Deliveries

Users List

Logout

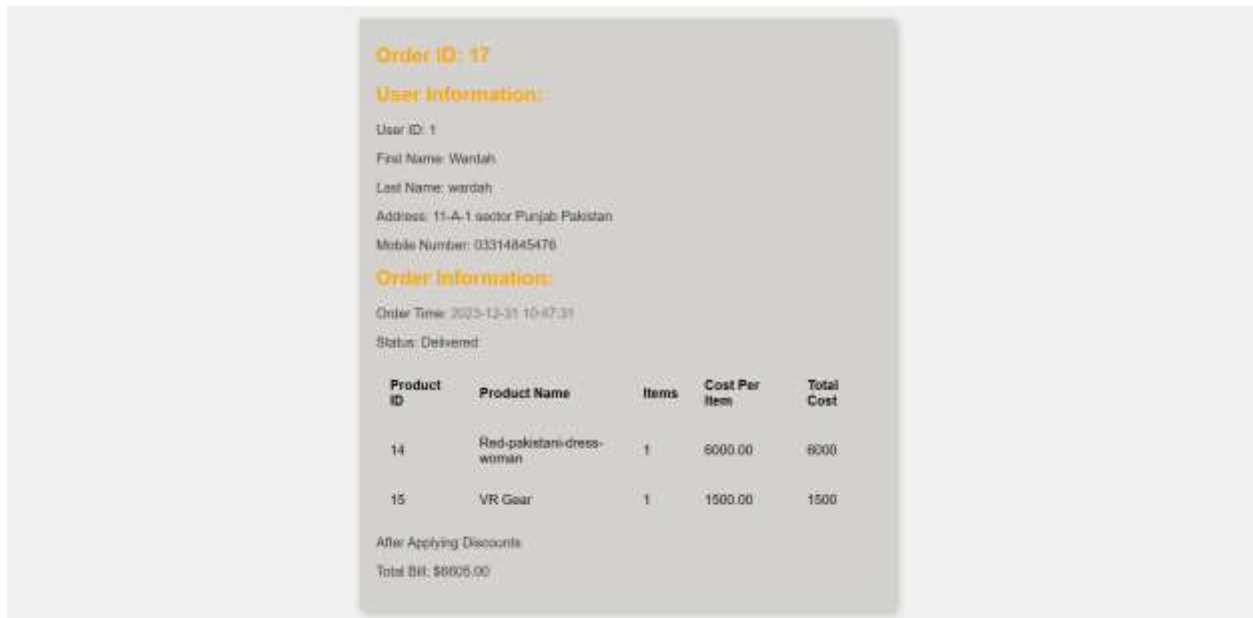
Pending Deliveries

Delivery Id	Date	Customer Name	Address	Number	Bill	Details	Deliver
15	2023-12-31 09:53:25	Wardah wardah	11-A-1 sector Punjab Pakistan	03314845476	580005.00		
16	2023-12-31 09:54:00	Wardah wardah	11-A-1 sector Punjab Pakistan	03314845476	15705.00		

### Order Details:

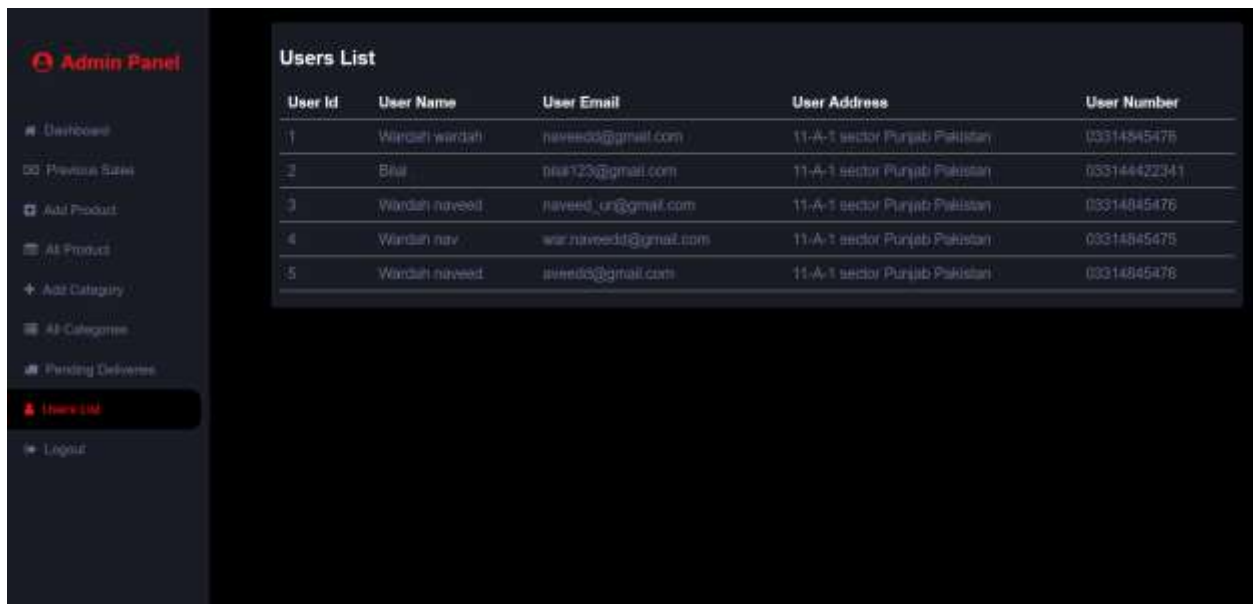
When you click on eye icon on pending deliveries page or on previous sales page, you will be redirected to the order details page which shows what was total bill and which products were ordered along with customer details and its status.





### Users List:

Users list page shows all the registered users along with their emails, addresses and phone numbers.



### Logout:

To logout to admin panel simply click on Logout button. It will run a php script to reset the started session.

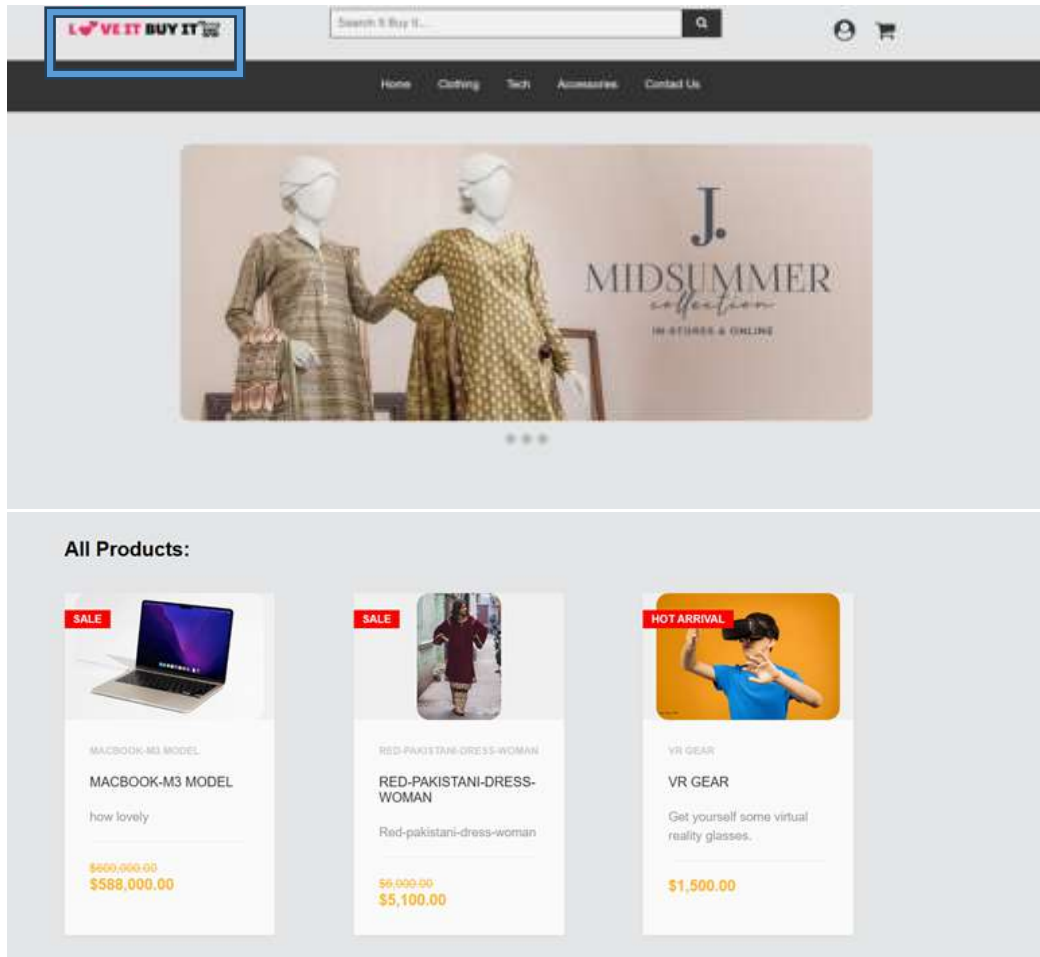
## 1.2 Customer-END:

### Connecting Database:

To connect the database to the admin panel, open “includeDB.php” and change the value of \$database to the desired database name. To connect the database without issues please upload the provided “database.sql” to the database named “shopping\_site”.

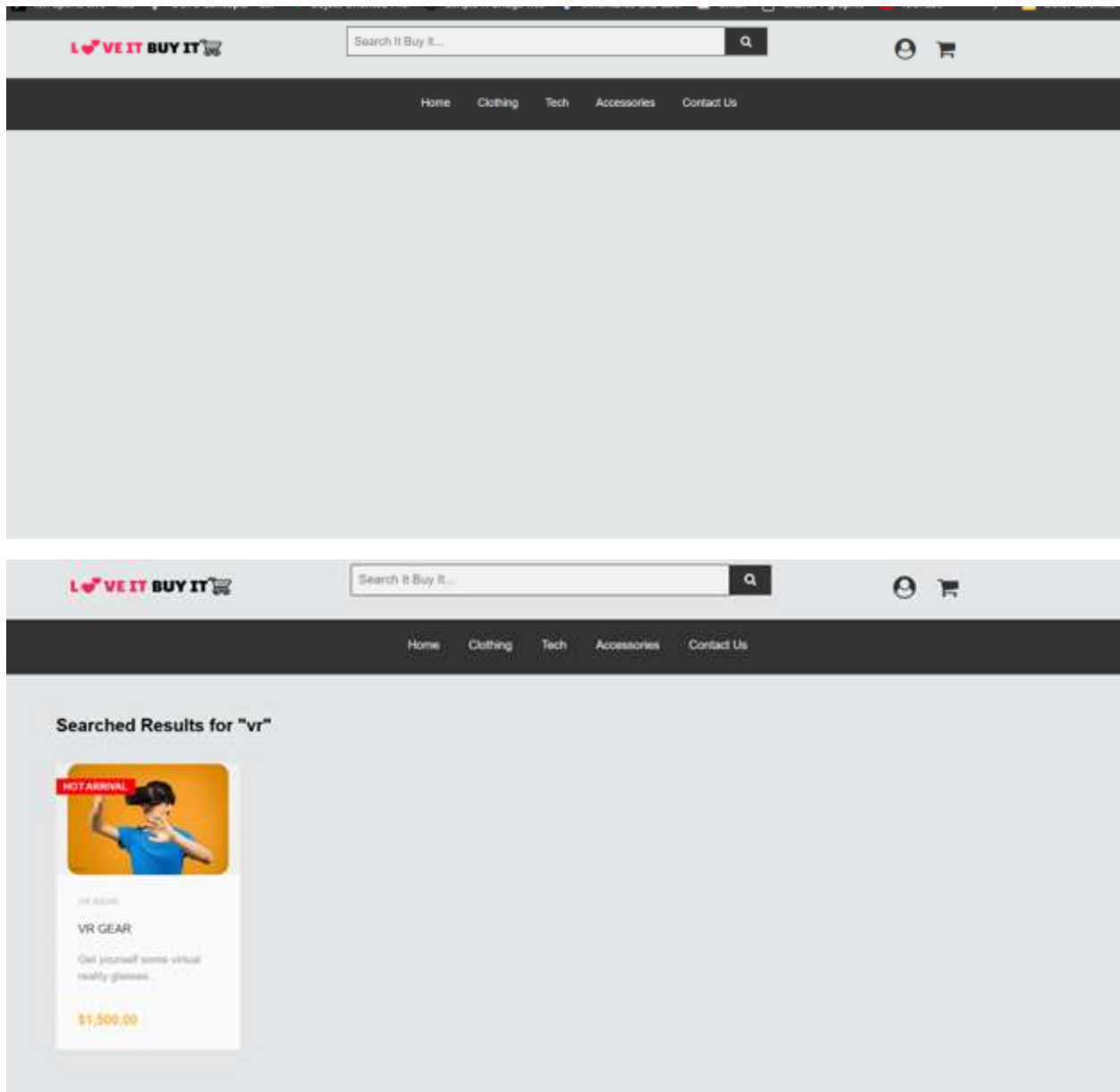
### Home Page:

When you enter our website dear customer we say warm welcome to you with our home page, displaying some brands that our store has. Then there is the logo of our website on the header along with the search bar that takes you to the search page and an icon for the profile and the cart. Then there are some products displayed below. When clicked on the product’s name it take you to the product page.



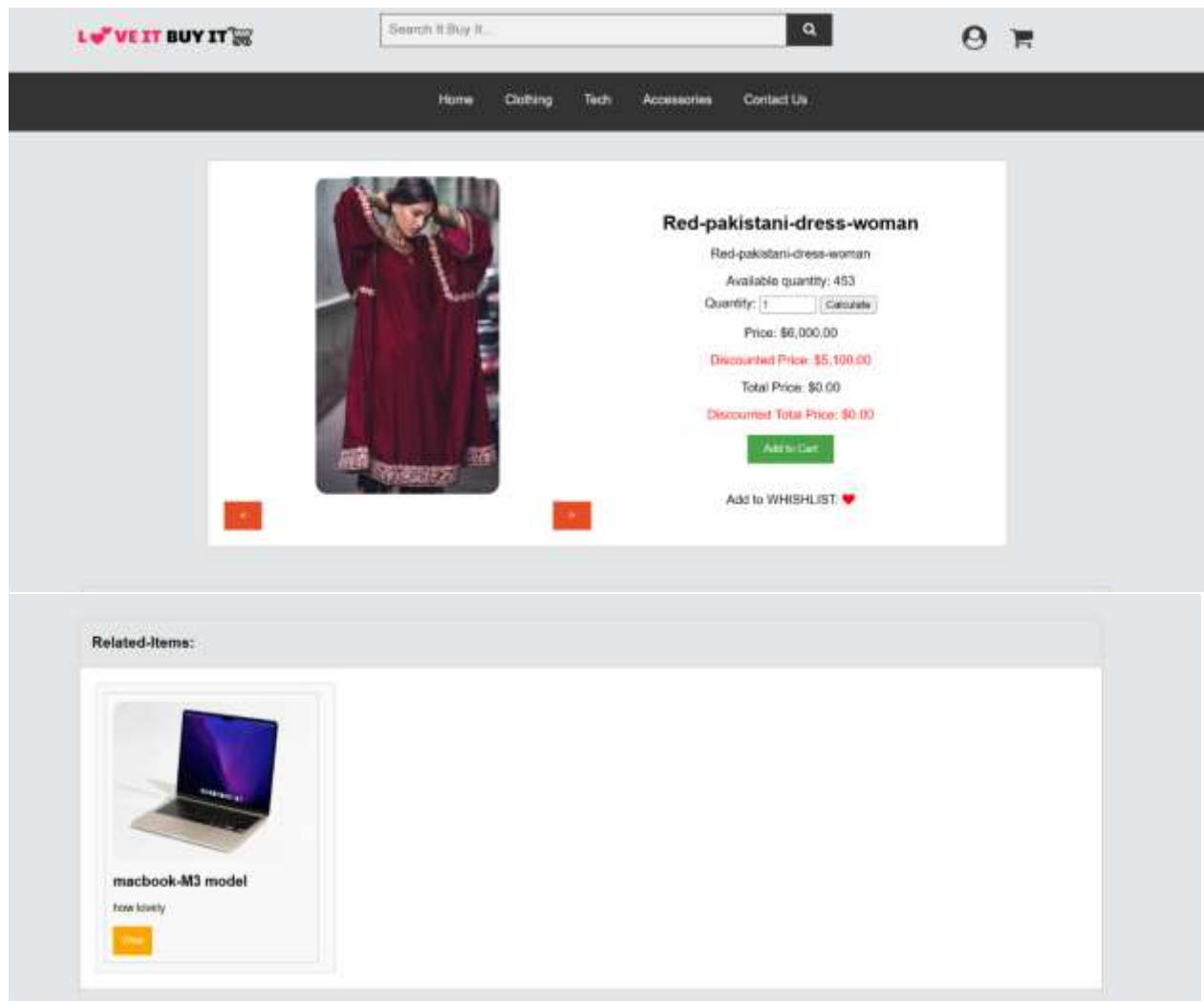
### Search Page:

When clicking on the search bar it takes you to the search page. Enter the words on it and press enter or on the search icon. The products having the same name or the same category or when if that word is present in either of them appears.



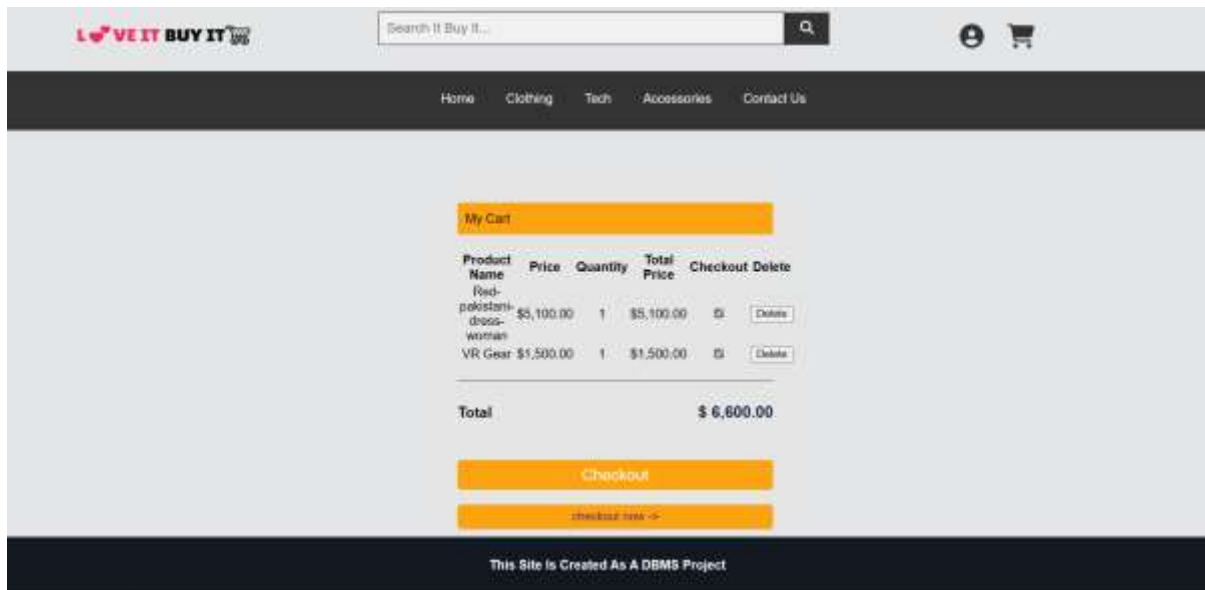
## Product Page:

The product is displayed fetching data from the database and the product link is pressed it redirects id and this page is pressed. It displays 3 product pictures of that product, name, and description, asks the user to enter the quantity and when the quantity is entered, if the user presses on the calculate button it displays the price for that number of items. Similarly, after entering the quantity the cart button is pressed which adds to the no. of items to that cart. Clicked on the Wishlist it adds to the wishlist of the user. The product that has the same category as the above product is shown in the related scrollable table. When pressed on view button it opens that product.



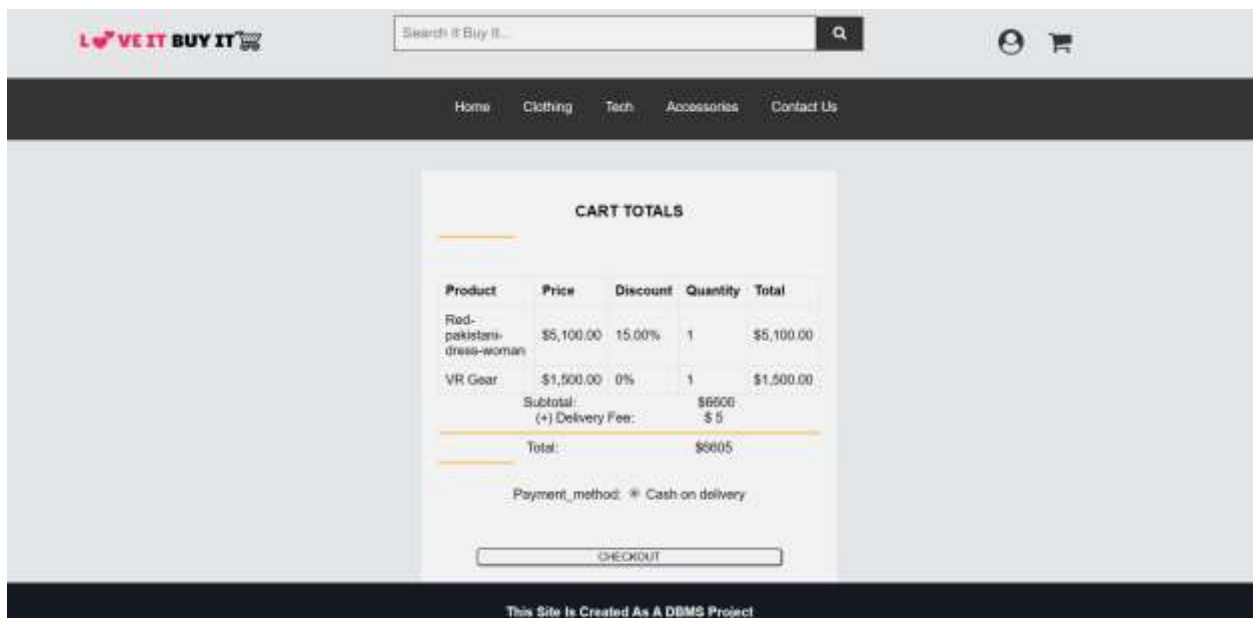
## Cart Page:

On pressing the delete, the product deletes from the user cart. To checkout check the select boxes of the rows that you want to check out with and press on the button checkout. Now click on the link checkout now-> that will take you to the checkout page.



### Checkout Page:

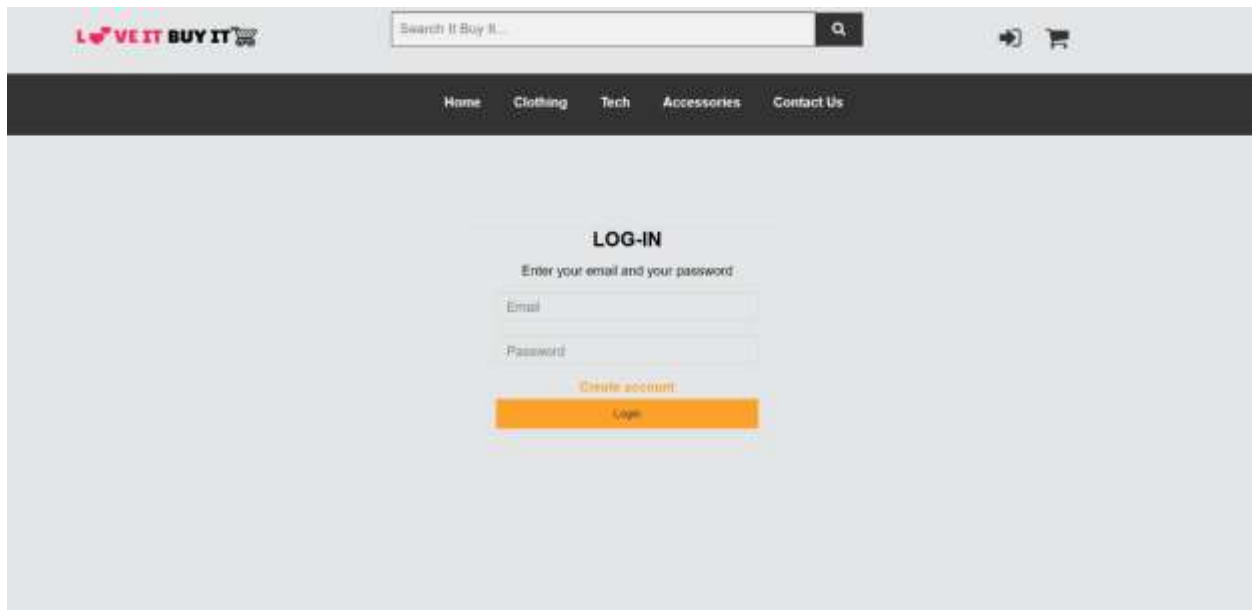
This displays the total price of the selected items from the cart and adds the delivery charges to it. On clicking Checkout the user places the order.



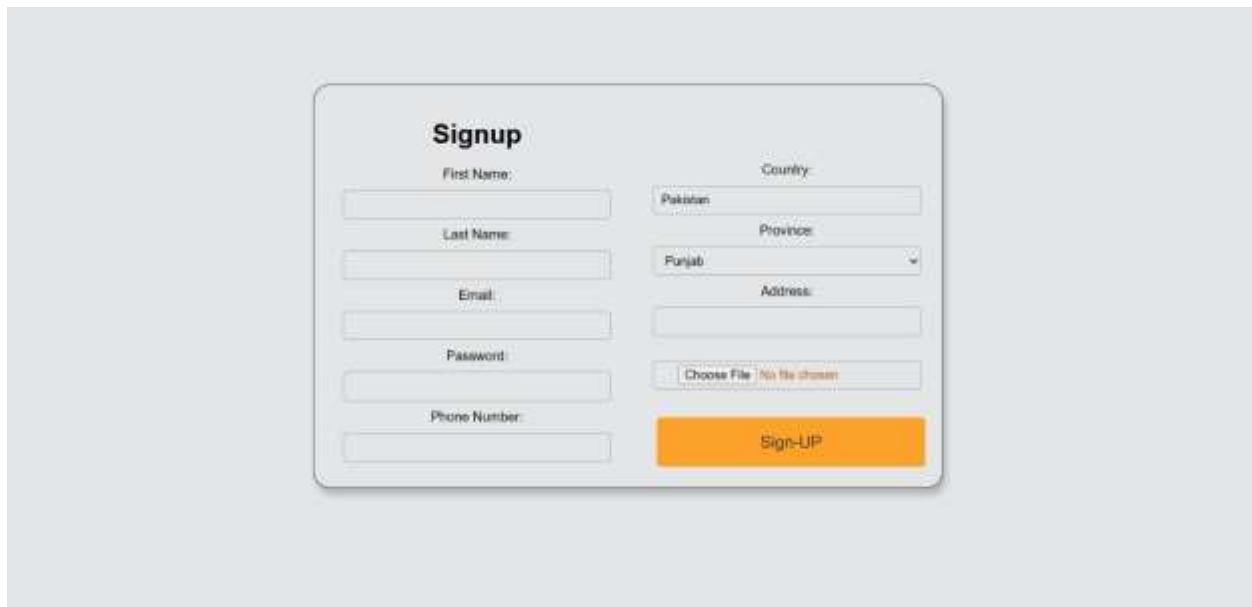
### Login and Signup Page:

If you already have an account enter your registered email enter the password and enter login. If the login is correct, then you will proceed to profile oof the login is not correct it will ask you to enter the registered email and password if you have an account or Sign up if you don't have the account. Click on the signup and it will take you to the

sign-up page there you fill in the data in the fields accordingly and click on sign up, your account will be successfully created.



The screenshot shows the top section of a website with a header. The header includes a logo "LOVE IT BUY IT" with a shopping cart icon, a search bar with the placeholder text "Search It Buy It...", and navigation links: Home, Clothing, Tech, Accessories, and Contact Us. Below the header is a "LOG-IN" form. The form has a title "LOG-IN", a subtitle "Enter your email and your password", and two input fields for "Email" and "Password". Below these fields are two buttons: "Create account" (in orange text) and "Login" (in white text on an orange background).



The screenshot shows a "Signup" form. The form has a title "Signup" and several input fields: "First Name:", "Last Name:", "Email:", "Password:", "Phone Number:", "Country:", "Province:", "Address:", and a "Choose File" button. The "Country" field is set to "Pakistan" and the "Province" field is set to "Punjab". The "Address" field is empty. The "Choose File" button is labeled "Choose File" and "No file chosen". Below the input fields is a large orange "Sign-UP" button.

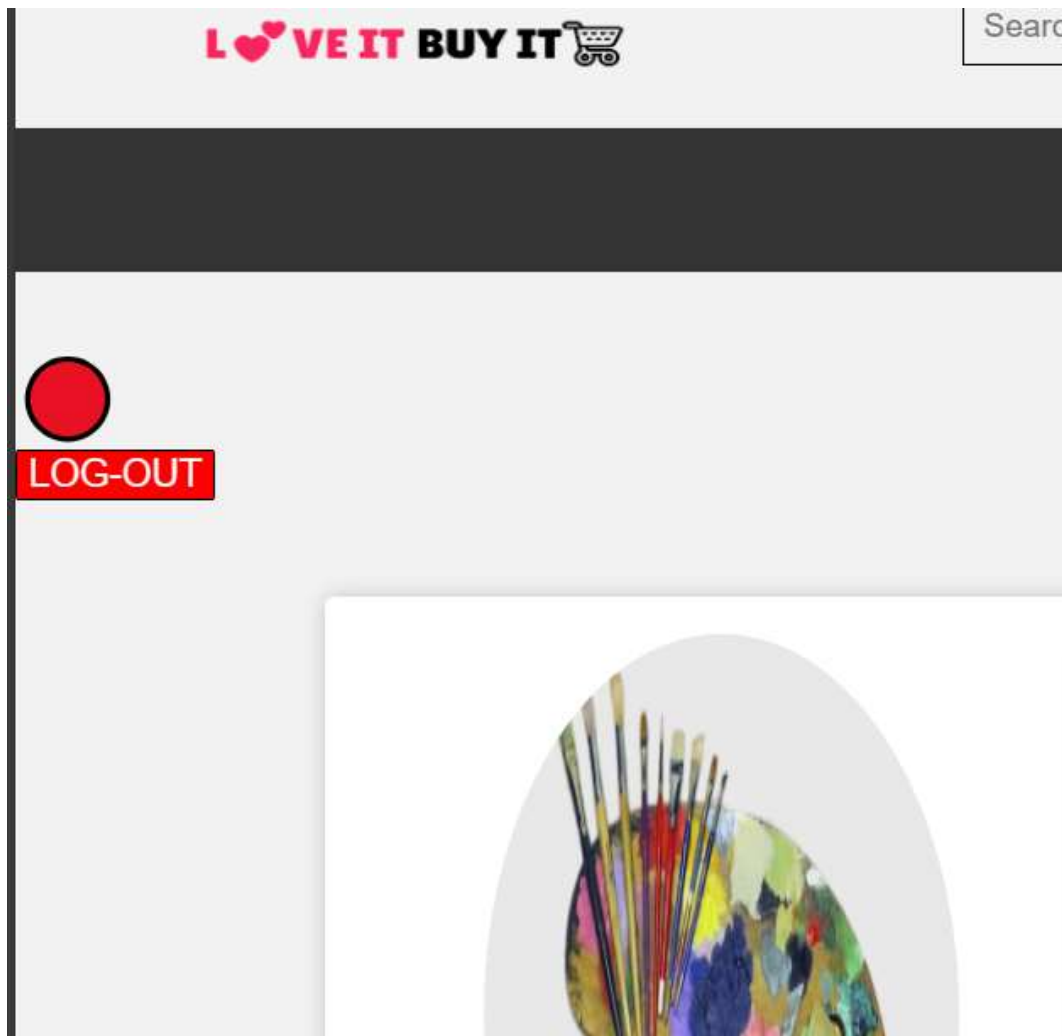
### Profile Page:

It displays the profile of the user along with his order history, order pending, and wishlist. On Pressing the Edit it takes you to the Edit page where the user can edit his data.



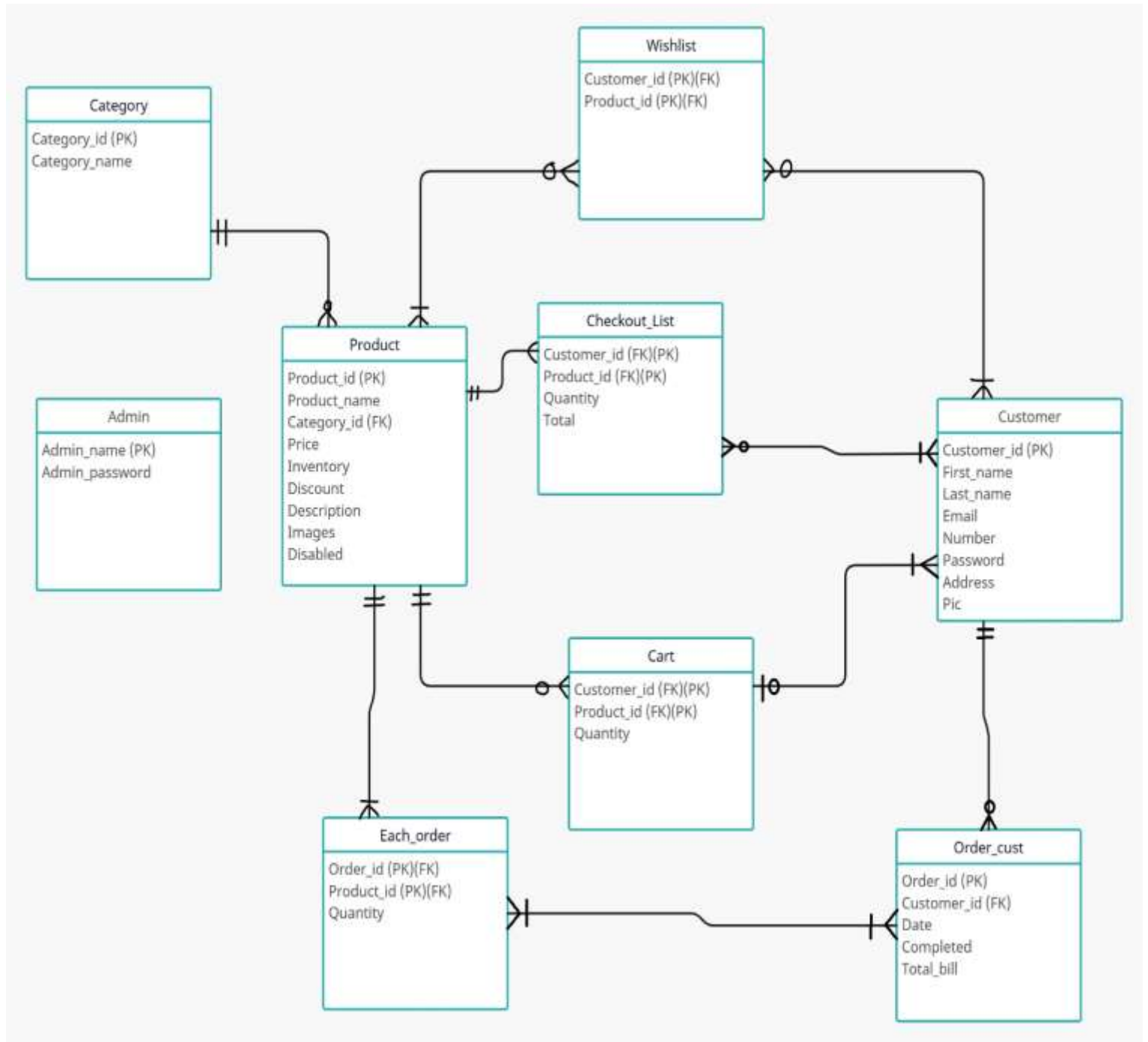
## LOG-OUT:

Click on the red button and then the log-out button will appear. Click on it to log out.





## 2. ERD:



### 3. Database Schema Documentation:

#### 3.1 Schema

Table: **admin**



The screenshot shows the MySQL Table Structure window for the 'admin' table in the 'shopping\_site' database. The table has two columns: 'admin\_name' (varchar(20)) and 'admin\_password' (varchar(10)). Both columns are using the 'utf8mb4\_0900\_ai\_ci' collation and are set to 'No' for null and 'None' for default. The 'admin\_name' column is marked as the primary key with a key icon. The interface includes tabs for Browse, Structure, SQL, Search, Insert, Export, Import, Privileges, and Operations.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	admin_name	varchar(20)	utf8mb4_0900_ai_ci		No	None			Change Drop More
2	admin_password	varchar(10)	utf8mb4_0900_ai_ci		No	None			Change Drop More

Description:

This table stores information about administrators, created for the login for admin panel.

Columns:

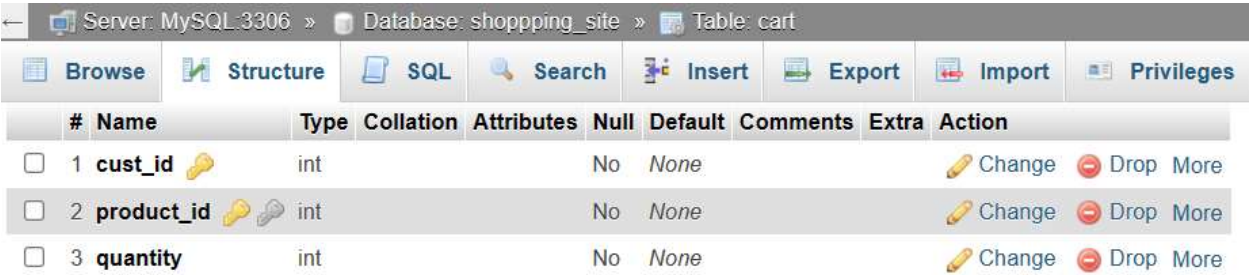
- admin\_name (varchar (20), NOT NULL)
- admin\_password (varchar (10), NOT NULL)

Primary Key (admin\_name)

Explanation:

The **admin\_name** is created for storing the username of the administrator. The **admin\_password** is created to hold the password for the administrator. The Primary Key is admin\_name, in our schema. No index is explicitly defined for this table. Since the primary key (admin\_name) is used for quick data retrieval and uniqueness.

Table: **cart**



The screenshot shows the MySQL Table Structure window for the 'cart' table in the 'shopping\_site' database. The table has three columns: 'cust\_id' (int), 'product\_id' (int), and 'quantity' (int). All columns are using the default collation and are set to 'No' for null and 'None' for default. The 'cust\_id' and 'product\_id' columns are marked as primary keys with key icons. The interface includes tabs for Browse, Structure, SQL, Search, Insert, Export, Import, and Privileges.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	cust_id	int			No	None			Change Drop More
2	product_id	int			No	None			Change Drop More
3	quantity	int			No	None			Change Drop More

Description:

This table represents the shopping carts of customers.

Columns:

- cust\_id (int, NOT NULL): Customer ID.
- product\_id (int, NOT NULL): Product ID.

- quantity (int, NOT NULL): Quantity of the product in the cart.

Composite Key (cust\_id, product\_id)

#### FOREIGN KEYS

- cust\_id Refers to the customer(cust\_id)
- product\_id Refers to the product(product\_id)

Explanation:

The **cust\_id** column stores the unique identifier of the customer. The **product\_id** column holds the unique identifier of the product in the cart. The **quantity** column indicates the quantity of the respective product in the customer's cart. The composite key is defined on **cust\_id** and **product\_id** to ensure **uniqueness and facilitate efficient data retrieval** for a specific customer and product..

#### Table: category



The screenshot shows the MySQL Table Structure window for the 'category' table in the 'shopping\_site' database. The table has two columns: 'category\_id' (int, AUTO\_INCREMENT, NOT NULL) and 'category\_name' (varchar(50), NOT NULL). The 'category\_id' column is the primary key.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	category_id	int			No	None		AUTO_INCREMENT	Change Drop More
2	category_name	varchar(50)	utf8mb4_0900_ai_ci		No	None			Change Drop More

Description:

This table stores information about product categories.

Columns:

- category\_id (int, AUTO\_INCREMENT, NOT NULL): Category ID.
- category\_name (varchar (50), NOT NULL): Name of the category.

Primary Key (category\_id)

Explanation:

The **category\_id** column is an automatically incrementing **unique identifier** for each category. The **category\_name** column stores the name of the product category. The primary key is defined on category\_id to ensure each category has a unique identifier.

#### Table: checkout\_list

Server: MySQL 3306 » Database: shopping_sde » Table: checkout_list									
Browse Structure SQL Search Insert Export Import Privileges									
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	cust_id	int			No	None			Change Drop More
<input type="checkbox"/> 2	product_id	int			No	None			Change Drop More
<input type="checkbox"/> 3	quantity	int			No	None			Change Drop More
<input type="checkbox"/> 4	total	decimal(10,2)			No	None			Change Drop More

Description:

This table records the checkout details of customers.

Columns:

- cust\_id (int, NOT NULL): Customer ID.
- product\_id (int, NOT NULL): Product ID.
- quantity (int, NOT NULL): Quantity of the product.
- total (decimal (10,2), NOT NULL): Total cost of the product in the checkout list.

Composite Key (cust\_id, product\_id).

FOREIGN KEYS

- cust\_id Refers to the customer(cust\_id)
- product\_id Refers to the product(product\_id)

Explanation:

The **cust\_id** column stores the unique identifier of the customer. The **product\_id** column holds the unique identifier of the product in the checkout list. The **quantity** column indicates the quantity of the respective product in the checkout list. The **total** column represents the total cost of the product in the checkout list. The composite key is composed of cust\_id and product\_id. Similar to the `cart` table, the composite key is defined on **cust\_id and product\_id** to ensure uniqueness and efficient retrieval of checkout details for a specific customer and product.

Table: **customer**

Server: MySQL 3306 * Database: shopping_site * Table: customer										
Browse Structure SQL Search Insert Export Import Privileges Operations Triggers										
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action	
<input type="checkbox"/>	1 cust_id	int			No	None		AUTO_INCREMENT		Change  Drop  More
<input type="checkbox"/>	2 PASSWORD	varchar(255)	utf8mb4_0900_ai_ci		No	None				Change  Drop  More
<input type="checkbox"/>	3 first_name	varchar(50)	utf8mb4_0900_ai_ci		No	None				Change  Drop  More
<input type="checkbox"/>	4 last_name	varchar(50)	utf8mb4_0900_ai_ci		No	None				Change  Drop  More
<input type="checkbox"/>	5 email	varchar(100)	utf8mb4_0900_ai_ci		No	None				Change  Drop  More
<input type="checkbox"/>	6 NUMBER	varchar(20)	utf8mb4_0900_ai_ci		No	None				Change  Drop  More
<input type="checkbox"/>	7 country	varchar(50)	utf8mb4_0900_ai_ci		No	None				Change  Drop  More
<input type="checkbox"/>	8 province	varchar(50)	utf8mb4_0900_ai_ci		No	None				Change  Drop  More
<input type="checkbox"/>	9 address	varchar(255)	utf8mb4_0900_ai_ci		No	None				Change  Drop  More
<input type="checkbox"/>	10 profile_picture	blob			Yes	NULL				Change  Drop  More

Description:

This table stores customer information.

Columns:

- cust\_id (int, AUTO\_INCREMENT, NOT NULL): Customer ID.
- PASSWORD (varchar (255), NOT NULL): Customer password.
- first\_name (varchar (50), NOT NULL): Customer's first name.
- last\_name (varchar (50), NOT NULL): Customer's last name.
- email (varchar (100), NOT NULL): Customer's email address (unique).
- NUMBER (varchar (20), NOT NULL): Customer's phone number.
- country (varchar (50), NOT NULL): Customer's country.
- province (varchar (50), NOT NULL): Customer's province.
- address (varchar (255), NOT NULL): Customer's address.
- profile\_picture (blob): Customer's profile picture.

PRIMARY KEY (cust\_id)

Unique Key (email)

Explanation:

The **cust\_id** column is an automatically incrementing unique identifier for each customer. The **PASSWORD** column stores the customer's password. The **email** column ensures the uniqueness of each customer's email address. The primary key is defined on **cust\_id** to ensure each customer has a unique identifier. The unique key on **email** enforces the **uniqueness of email addresses**. These keys facilitate efficient data retrieval and ensure data integrity.

Table: each\_order

Server: MySQL:3306 » Database: shopping_site » Table: each_order									
<a href="#">Browse</a> <a href="#">Structure</a> <a href="#">SQL</a> <a href="#">Search</a> <a href="#">Insert</a> <a href="#">Export</a> <a href="#">Import</a> <a href="#">Privileges</a>									
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	<b>order_id</b>	int			No	None			Change  Drop <a href="#">More</a>
<input type="checkbox"/> 2	<b>product_id</b>	int			No	None			Change  Drop <a href="#">More</a>
<input type="checkbox"/> 3	<b>quantity</b>	int			Yes	NULL			Change  Drop <a href="#">More</a>

Description:

This table records individual product details in each order.

Columns:

- **order\_id** (int, NOT NULL): Order ID.
- **product\_id** (int, NOT NULL): Product ID.
- **quantity** (int, DEFAULT NULL): Quantity of the product.

Composite Key (cust\_id, product\_id).

FOREIGN KEYS

- **cust\_id** Refers to the customer(cust\_id)
- **product\_id** Refers to the product(product\_id)

Explanation:

The **order\_id** column stores the unique identifier of the order. The **product\_id** column holds the unique identifier of the product in the order. The **quantity** column indicates the quantity of the respective product in the order. The composite key is defined on order\_id and product\_id to ensure uniqueness and efficient retrieval of individual product details within each order.

Table: **order\_cust**

Server: MySQL:3306 » Database: shopping_site » Table: order_cust									
<a href="#">Browse</a> <a href="#">Structure</a> <a href="#">SQL</a> <a href="#">Search</a> <a href="#">Insert</a> <a href="#">Export</a> <a href="#">Import</a> <a href="#">Privileges</a> <a href="#">Operations</a>									
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	<b>order_id</b>	int			No	None		AUTO_INCREMENT	Change  Drop <a href="#">More</a>
<input type="checkbox"/> 2	<b>cust_id</b>	int			Yes	NULL			Change  Drop <a href="#">More</a>
<input type="checkbox"/> 3	<b>date_time</b>	datetime			Yes	NULL			Change  Drop <a href="#">More</a>
<input type="checkbox"/> 4	<b>no_of_products</b>	int			Yes	NULL			Change  Drop <a href="#">More</a>
<input type="checkbox"/> 5	<b>completed</b>	tinyint(1)			Yes	NULL			Change  Drop <a href="#">More</a>
<input type="checkbox"/> 6	<b>price_total</b>	decimal(10,2)			Yes	NULL			Change  Drop <a href="#">More</a>

Description:



This table stores information about customer orders.

Columns:

- `order_id` (int, AUTO\_INCREMENT, NOT NULL): Order ID.
- `cust_id` (int, DEFAULT NULL): Customer ID.
- `date_time` (datetime, DEFAULT NULL): Date and time of the order.
- `no_of_products` (int, DEFAULT NULL): Number of products in the order.
- `completed` (tinyint(1), DEFAULT NULL): Status of order completion (1 for completed, 0 for not completed).
- `price_total` (decimal(10,2), DEFAULT NULL): Total price of the order.

PRIMARY Key (`order_id`).


FOREIGN KEYS

- `cust_id` Refers to the customer(`cust_id`)

Explanation:

The **order\_id** column is an automatically incrementing unique identifier for each order. The **cust\_id** column stores the unique identifier of the customer placing the order. The `date_time` column records the date and time of the order. The **completed** column indicates whether the order is completed or not. The **price\_total** column represents the total price of the order. The primary key is **order\_id**.

Table: **product**



The screenshot shows the MySQL Table Structure for the 'product' table. The table has 13 columns with the following details:

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	product_id	int			No	None		AUTO_INCREMENT	Change Drop More
2	product_name	varchar(100)	utf8mb4_0900_ai_ci		No	None			Change Drop More
3	category_id	int			Yes	NULL			Change Drop More
4	inventory	int			Yes	0			Change Drop More
5	price	decimal(10,2)			No	None			Change Drop More
6	product_description	text	utf8mb4_0900_ai_ci		No	None			Change Drop More
7	image1	blob			No	None			Change Drop More
8	image2	blob			No	None			Change Drop More
9	image3	blob			No	None			Change Drop More
10	discount	decimal(4,2)			Yes	0.00			Change Drop More
11	time_arrival	timestamp			Yes	CURRENT_TIMESTAMP		DEFAULT_GENERATED	Change Drop More
12	sold	int			Yes	0			Change Drop More
13	disabled	tinyint(1)			No	0			Change Drop More

Description:

This table stores information about products.

Columns:

- **product\_id** (int, AUTO\_INCREMENT, NOT NULL): Product ID.
- **product\_name** (varchar (100), NOT NULL): Name of the product.
- **category\_id** (int, DEFAULT NULL): Category ID to which the product belongs.
- **inventory** (int, DEFAULT '0'): Quantity of the product in stock.
- **price** (decimal (10,2), NOT NULL): Price of the product.
- **product\_description** (text, NOT NULL): Description of the product.
- **image1** (blob, NOT NULL)
- **image2** (blob, NOT NULL)
- **image3** (blob, NOT NULL): Images of the product.
- **discount** (decimal (4,2), DEFAULT '0.00'): Discount on the product.
- **time\_arrival** (timestamp, NULL DEFAULT CURRENT\_TIMESTAMP): Timestamp of product arrival.
- **sold** (int, DEFAULT '0'): Quantity of the product sold.
- **disabled** (tinyint (1), NOT NULL DEFAULT '0'): Product availability status (1 for disabled, 0 for enabled).

PRIMARY KEY (product\_id)

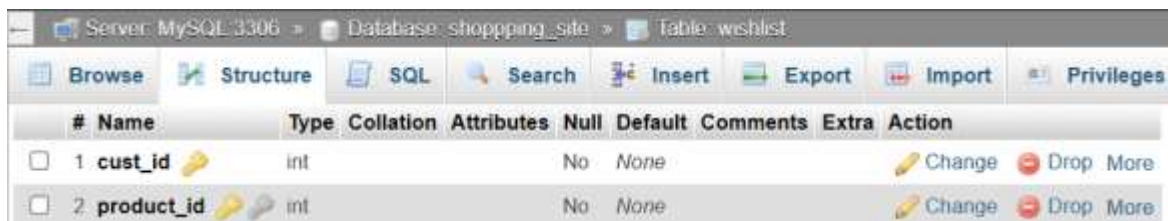
FOREIGN KEY

- **category\_id** REFERS to category(category\_id)

Explanation:

The **product\_id** column is an automatically incrementing unique identifier for each product. The **product\_name** column stores the name of the product. The **category\_id** column indicates the category to which the product belongs. The **inventory** column represents the quantity of the product in stock. The **price** column holds the price of the product. The **product\_description** column provides a textual description of the product. The **image1**, **image2**, and **image3** columns store blob data representing images of the product. The **discount** column represents any discount applied to the product. The **time\_arrival** column records the timestamp of the product's arrival. The **sold** column indicates the quantity of the product sold. The **disabled** column determines the availability status of the product. The primary key is product\_id.

Table: wishlist



The screenshot shows the MySQL Table Structure window for the 'wishlist' table in the 'shopping\_site' database. The table has two columns: 'cust\_id' and 'product\_id', both of type 'int'. 'cust\_id' is the first column and 'product\_id' is the second column. Both columns are 'NOT NULL' and have a 'Default' value of 'None'. The 'product\_id' column is marked as the primary key with a key icon. The 'Action' column for each row contains links for 'Change', 'Drop', and 'More'.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	cust_id	int			No	None			Change Drop More
2	product_id	int			No	None			Change Drop More

Description:



This table represents the wishlist of customers.

Columns:

- `cust_id` (int, NOT NULL): Customer ID.
- `product_id` (int, NOT NULL): Product ID.

Composite Key (`cust_id`, `product_id`).

FOREIGN KEYS

- `cust_id` Refers to the customer(`cust_id`)
- `product_id` Refers to the product(`product_id`)

Explanation:

The **`cust_id`** column stores the unique identifier of the customer. The **`product_id`** column holds the unique identifier of the product in the wishlist. The primary key is composed of `cust_id` and `product_id`.

---

### ***3.2 Normalization:***

Our schema aligns with normalization principles, specifically targeting the Third Normal Form (3NF) and Boyce-Codd Normal Form (BCNF) requirements:

#### **1. Table: admin**

The admin table meets 3NF and BCNF as it possesses a single primary key (`admin_name`), eliminating transitive dependencies.

#### **2. Table: cart**

The cart table adheres to 3NF and BCNF with its composite primary key (`cust_id`, `product_id`), preventing dependencies on non-prime attributes.

#### **3. Table: category**

The category table satisfies 3NF and BCNF by having a primary key (`category_id`), ensuring no transitive dependencies or non-prime attribute dependencies.

#### **4. Table: checkout\_list**

The checkout\_list table aligns with 3NF and BCNF due to its composite primary key (`cust_id`, `product_id`), preventing dependencies on non-prime attributes.

#### **5. Table: customer**

The customer table meets 3NF and BCNF criteria with its primary key (`cust_id`) and unique key (`email`), eliminating transitive dependencies.

## 6. Table: each\_order

The each\_order table adheres to 3NF and BCNF with its composite primary key (order\_id, product\_id), preventing dependencies on non-prime attributes.

## 7. Table: order\_cust

The order\_cust table satisfies 3NF and BCNF requirements with its primary key (order\_id), ensuring no transitive dependencies or non-prime attribute dependencies.

## 8. Table: product

The product table aligns with 3NF and BCNF by having a primary key (product\_id) and establishing foreign key relationships (category\_id), eliminating transitive dependencies.

## 9. Table: wishlist

The wishlist table meets 3NF and BCNF criteria through its composite primary key (cust\_id, product\_id), preventing dependencies on non-prime attributes.

In essence, our schema follows normalization principles, organizing data to eliminate redundancy and ensuring that each table meets the criteria of 3NF and BCNF, which enhances data integrity and minimizes anomalies during database operations.

---

## 4. Code Documentation:

### 4.1 *CRUD OPERATIONS:*

Crud operations in DBMS stands for:

- Create
- Read
- Update
- Delete

is foundational for database management. These operations involve adding new data, retrieving information, modifying existing records, and deleting entries, forming the core functions in data manipulation within a database system.

In our Code, we also implemented Crud operations many times some example instances are given below.

#### 4.1.1 CREATE:

- SQL Command: INSERT INTO table\_name (column1, column2, column3, ...) VALUES (value1, value2, value3, ...);

- **Description:** Adds new records (rows) to a table. It involves specifying the table, columns, and corresponding values.

Okay now analyzing our code

product.php (Product Page)

```
if ($result->num_rows > 0) {
    while ($row = $result->fetch_assoc()) {
        if ($row['product_id'] == $product_id) {
            //remove from wishlist
            $sqlWishlist = "DELETE FROM wishlist WHERE cust_id = '$user_id' AND product_id = '$product_id'";
            $stmtWishlist = $conn->prepare($sqlWishlist);
            $stmtWishlist->execute();
            break;
        }
    }
}
```

Explanation: Adds or removes products from the user's wishlist. If the product is already in the wishlist, it removes it; otherwise, it adds it. This involves both INSERT and DELETE operations on the "wishlist" table

```
if (isset($_POST['insert_data'])) {
    if (isset($_SESSION['user_id'])) {
        $user_id = $_SESSION['user_id'];
    } else {
        header('Location: login.php');
        exit();
    }

    $quantity = $_POST['quantity'];

    $sqlCart = "INSERT INTO cart (product_id, quantity, cust_id)
                VALUES ('$product_id', '$quantity', '$user_id')
                ON DUPLICATE KEY UPDATE quantity = quantity + VALUES(quantity)";

    $stmtCart = $conn->query($sqlCart);

    if ($stmtCart) {
        header("Location: cartpage.php");
        exit();
    } else {
        echo "<script>alert('Category Addition Failed!')</script>";
    }
}
?>
```

Explanation: Inserts or updates the quantity of a product in the user's shopping cart. It uses an INSERT ON DUPLICATE KEY UPDATE query to handle both scenarios.

Signup.php:

```

if (empty($signup_error)) {
    $sql_insert = "INSERT INTO customer (PASSWORD, first_name, last_name, email, NUMBER, country, province, a
        VALUES (?, ?, ?, ?, ?, ?, ?, ?)";

    $stmt = $conn->prepare($sql_insert);
    $stmt->bind_param('ssssssss', $password, $first_name_value, $last_name_value, $email_value, $number_valu

```

Explanation: signing in the user and user entering the data and the database storing it.

```

// Prepare and execute insert statement for each selected item
if (isset($_POST['checkout_items']) && is_array($_POST['checkout_items'])) {
    $sqlInsert = "INSERT INTO checkout_list (cust_id, product_id, quantity, total)
        SELECT ?, c.product_id, c.quantity,
            (CASE WHEN p.discount > 0 THEN (p.price - (p.price * p.discount / 100)) * c.quantity
            ELSE p.price * c.quantity END)
        FROM cart c
        JOIN product p ON c.product_id = p.product_id
        WHERE c.cust_id = ? AND c.product_id = ?";
    $stmtInsert = $conn->prepare($sqlInsert);

    foreach ($_POST['checkout_items'] as $productId) {
        if ($stmtInsert) {
            $stmtInsert->bind_param("iii", $user_id, $user_id, $productId);
            $stmtInsert->execute();
        }
    }
}

```

Explanation: the checkout\_list table being inserted.

editproduct.php:

```

if(isset($_POST['category']))
{
    $category = $_POST['category'];
    $sql = "INSERT INTO category (category_name) VALUES ('$category')";
    $result = $conn->query($sql);
    if($result)
    {
        echo "<script>alert('Category Added Successfully!')</script>";
    }
    else
    {
        echo "<script>alert('Category Addition Failed!')</script>";
    }
}

```

category is created here.

Whenever we make a user sign up, the admin enters the product details and adds the category, and when the user uploads things in the cart and wishlist or checks out the order; it inserts data in the cart, wishlist, in checkout\_list then in further in from checkout\_list to order\_cust and then to each\_order.

#### 4.1.2 READ:

- **Description:** Retrieves data from the database. The SELECT statement is used to specify the columns to be retrieved, the table from which to retrieve them, and any conditions for filtering the data.

Okay now analyzing some parts of our code.

index.php (HOME PAGE):

```
// Fetch products from the database
$sql = "SELECT p.*, COUNT(eo.order_id) AS order_count
        FROM product p
        LEFT JOIN each_order eo ON p.product_id = eo.product_id
        GROUP BY p.product_id";
$result = $conn->query($sql);
```

Explanation: The SELECT SQL command retrieves product information, including the count of orders (order\_count) for each product from the product and each\_order tables. The result is stored in the \$result variable.

```
// Store the results in an array
$products = [];
while ($row = $result->fetch_assoc()) {
    $products[] = $row;
}
```

Explanation: The code reads each row from the result set (\$result) and appends it to the \$products array. This loop fetches data from the database and stores it in the PHP array for later use.

```
<?php foreach ($products as $product): ?>
    <div class="product-card">
        <?php if ($product['discount'] > 0): ?>
            <div class="badge">Sale</div>
        <?php elseif (date('Y-m-d') == date('Y-m-d', strtotime($product['time_arrival']))): ?>
            <div class="badge">Hot Arrival</div>
        <?php elseif ($product['order_count'] >= 5 && $product['order_count'] <= 10): ?>
            <div class="badge">Hot</div>
    </div>
</foreach>
```

Explanation: Within the HTML section, a loop iterates through the \$products array, displaying product details dynamically. This operation involves reading and presenting the data retrieved from the database to the webpage.

profile.php (Profile Page):

```
// Transform the result into a structured array
$order_history = [];
foreach ($rows as $row) {
    $order_id = $row['order_id'];
    if (!isset($order_history[$order_id])) {
        $order_history[$order_id] = [
            'order_id' => $order_id,
            'date_time' => $row['date_time'],
            'no_of_products' => $row['no_of_products'],
            'price_total' => $row['price_total'],
            'products' => [],
        ];
    }
}
```

Explanation: Fetches order history with product details for the logged-in user from the database. It involves a complex SQL query with JOIN operations to retrieve relevant data.

```
// Fetch wishlist items
$sql_wishlist = "SELECT p.product_id, p.product_name, p.product_description, p.price, p.image1
FROM wishlist w
JOIN product p ON w.product_id = p.product_id
WHERE w.cust_id = $user_id";

$result_wishlist = $conn->query($sql_wishlist);

if ($result_wishlist->num_rows > 0) {
    $wishlist_items = $result_wishlist->fetch_all(MYSQLI_ASSOC);
} else {
    $wishlist_items = [];
}
```

Explanation: Retrieves user data (excluding the password) for the logged-in user from the database based on the session's user ID.

allproduct.php:

```
$sql_check = "SELECT * FROM product WHERE disabled = 0 ORDER BY product_id DESC";
$result = $conn->query($sql_check);
while ($row = mysqli_fetch_array($result)) { ?>
    <tr>
        <td><?php echo $row['product_id']; ?></td>
        <td><?php echo $row['product_name']; ?></td>
        <td><?php echo $row['price']; ?></td>
        <td><?php echo $row['sold']; ?></td>
        <td><?php echo $row['inventory']; ?></td>
        <td><?php echo $row['discount']; ?></td>
        <td><a href="editproduct.php?product_id=<?php echo $row['product_id']; ?>"><i class="fa fa-edit" style = "color:
        <td><a href="disableproduct.php?product_id=<?php echo $row['product_id']; ?>" target="_blank"><i class="fa fa-
    </td>
    </tr>
?php } ?>
</tbody>
</table>
```

Explanation: Displaying data of the product

**Overall whenever we have to read data from the database to display Read operation is used; almost in every file we use it as we need to display is the data**



related to products, customers, customer carts, customer wishlist, order-list, sales made; Thus, it is most used by us.

#### 4.1.3 UPDATE:

- SQL Command: UPDATE table\_name SET column1 = value1, column2 = value2, ... WHERE condition;
- **Description:** Modifies existing records in a table. It involves specifying the table, columns to be updated, new values, and conditions for identifying the records to be updated.

Okay now analyzing our code

profile.php (Profile Page):

```
if (isset($_POST['insert_data'])) {  
    if (isset($_SESSION['user_id'])) {  
        $user_id = $_SESSION['user_id'];  
    } else {  
        header('Location: login.php');  
        exit();  
    }  
  
    $quantity = $_POST['quantity'];  
  
    $sqlCart = "INSERT INTO cart (product_id, quantity, cust_id)  
                VALUES ('$product_id', '$quantity', '$user_id')  
                ON DUPLICATE KEY UPDATE quantity = quantity + VALUES(quantity)";  
  
    $stmtCart = $conn->query($sqlCart);  
  
    if ($stmtCart) {  
        header("Location: cartpage.php");  
        exit();  
    } else {  
        echo "<script>alert('Category Addition Failed!')</script>";  
    }  
}
```

Explanation: Inserts or updates the quantity of a product in the user's shopping cart. It uses an INSERT ON DUPLICATE KEY UPDATE query to handle both scenarios.

update\_profile.php:

```
// Update customer data
$updateQuery = "UPDATE `customer` SET `first_name` = ?, `last_name` = ?, `email` = ?, `NUMBER` = ?, `province` = ?,
$updateStmt = $conn->prepare($updateQuery);
$updateStmt->bind_param('ssssssi', $first_name, $last_name, $email, $NUMBER, $province, $address, $profilePictureDe

if ($updateStmt->execute()) {
    echo "Customer data updated successfully.<br>";

    // Redirect to profile.php
    header('Location: profile.php');
    exit();
} else {
    echo "Error updating customer data: " . $updateStmt->error . "<br>";
}
}
```

Explanation: updating the valued of the customer with the cust\_id.

edit\_product.php:

```
if(!empty($_FILES['image1']['tmp_name'])) {
    // get image data
    $image1Data = addslashes(file_get_contents($_FILES['image1']['tmp_name']));
    $sql = "UPDATE product SET image1 = '$image1Data' WHERE product_id = '$product_id'";
    $result = $conn->query($sql);
}
if(!empty($_FILES['image2']['tmp_name'])) {
    // get image data
    $image1Data = addslashes(file_get_contents($_FILES['image2']['tmp_name']));
    $sql = "UPDATE product SET image2 = '$image1Data' WHERE product_id = '$product_id'";
    $result = $conn->query($sql);
}
}
```

Overall, when we update the cart item value, we deliver the order to the user the order value of complete is updated for that order, in the updated profile, or in updating the category from the admin panel.

#### 4.1.1 DELETE:

- SQL Command: DELETE FROM table\_name WHERE condition;
- **Description:** Removes records from a table based on specified conditions. The DELETE statement can delete specific records or all records in a table.

Okay now analyzing our code



### product.php (Product Page)

```
if ($result->num_rows > 0) {  
    while ($row = $result->fetch_assoc()) {  
        if ($row['product_id'] == $product_id) {  
            //remove from wishlist  
            $sqlWishlist = "DELETE FROM wishlist WHERE cust_id = '$user_id' AND product_id = '$product_id'";  
            $stmtWishlist = $conn->prepare($sqlWishlist);  
            $stmtWishlist->execute();  
            break;  
        }  
    }  
}
```

Description: Adds or removes products from the user's wishlist. If the product is already in the wishlist, it removes it; otherwise, it adds it. This involves both INSERT and DELETE operations on the "wishlist" table

### Cart.php (CART page):

```
// Clear the cart for the selected items  
$sqlClearCart = "DELETE FROM cart WHERE cust_id = ? AND product_id IN (" . implode(",", $_POST['checkout_item'] . ")";  
$stmtClearCart = $conn->prepare($sqlClearCart);  
$stmtClearCart->bind_param("i", $user_id);  
$stmtClearCart->execute();  
$stmtClearCart->close();  
}
```

When added to checkout\_list it deletes from the cart

**So, we performed this operation in wishlist handling, cart when checkout, checkout\_list when finally checked out, and deleting an item or product from the cart.**

---

## **4.2 Image-Storing and Displaying:**

We take user input only in the form of png, jpeg and jpg

```
<label>Profile Picture: <input type="file" name="profile_picture"  
accept="image/jpeg, image/jpg, image/png" required></label>
```

For storing the image we stored it in the blob format in the database and resized each image so that the complete image is stored and storing the image blob in string format.

```
$profilePictureData =  
resizeImage(file_get_contents($_FILES['profile_picture']['tmp_name']), 800);
```

```

$updateQuery = "UPDATE `customer` SET `first_name` = ?, `last_name` = ?,
`email` = ?, `NUMBER` = ?, `province` = ?, `address` = ?, `profile_picture` = ?
WHERE `cust_id` = ?";
$updateStmt = $conn->prepare($updateQuery);
$updateStmt->bind_param('sssssssi', $first_name, $last_name, $email, $NUMBER,
$province, $address, $profilePictureData, $user_id);

```

Here s stands for string and 'i' stands for integer.

For loading of the image is converted back to image format from the blob file to display

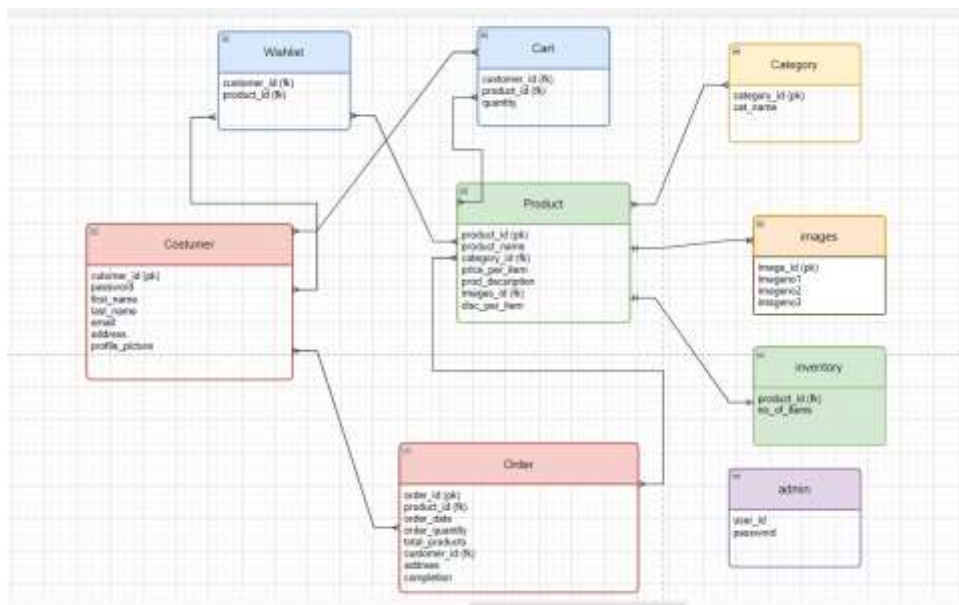
```

// Display profile picture stored in BLOB format
echo "<img src='data:image/png;base64,'" .
base64_encode($existingCustomerData['profile_picture']) . "' alt='Profile
Picture'><br>";

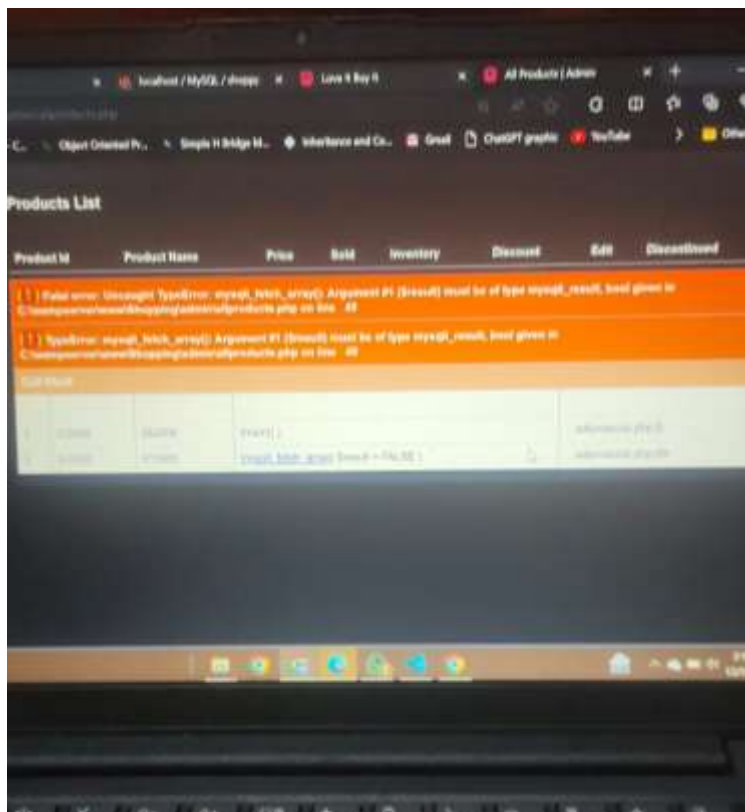
```

## Improvements:

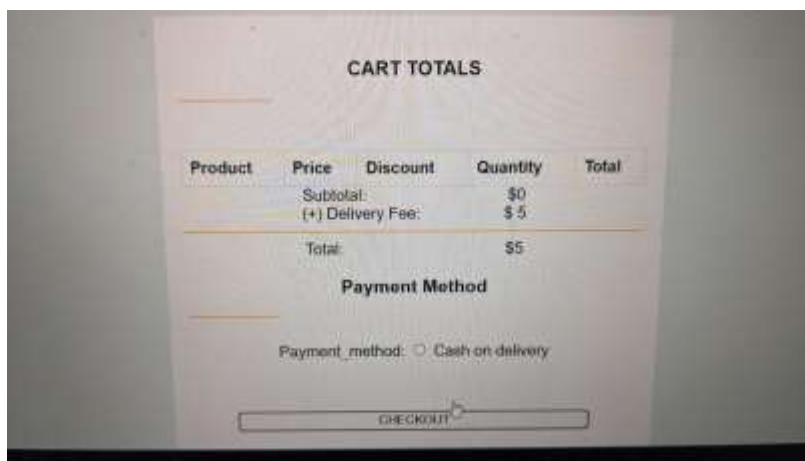
Firstly, we were making schema like these:



Also, we had some errors which we resolved later:



Checkout page giving no value issue in storing the data



Total price being calculated wrong:

### My Cart

Product Name	Price	Quantity	Total Price	Checkout	Delete
macbook-M3 model	\$588,000.00	2	\$1,176,000.00	<input checked="" type="checkbox"/>	<button>Delete</button>

---

<b>Total</b>	<b>\$ 1,176,000.00</b>
--------------	------------------------