

CSCE 5350
FUNDAMENTALS OF DATABASE SYSTEMS

SUPPLY CHAIN MANAGEMENT SYSTEM

PROJECT REPORT



Team Members

About System

The Supply Chain Management System is designed to improve a company's supply chain operations. It represents a critical initiative for business striving to maintain competitiveness in the global market. The database serves as the cornerstone of our system, housing critical information about products, customers, orders, payments, and various other entities. It forms the backbone upon which our platform thrives, impacting user experiences, data security, and the overall performance of our application.

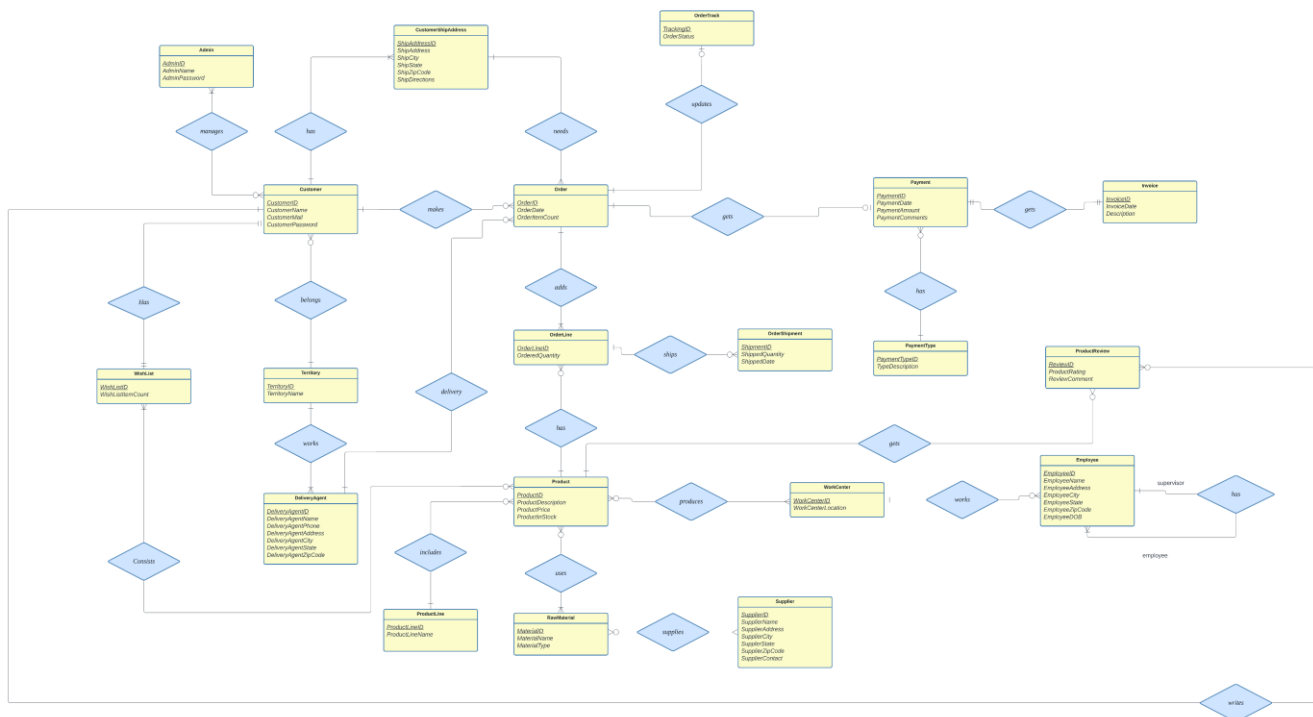
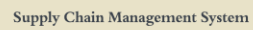
The operations include:

- A Customer can have an account associated with him.
- Employee works at a Work Center and manages products availability.
- Delivery Agent delivers the orders to the customer.
- It involves managing Order fulfilment.
- Payments and Payment types are available with the system.
- Raw Material data required for product development is also managed.
- It also allows customers to provide ratings and reviews.
- Status of Order is also made available.
- Wishlist is available for customer to add products of his interest.

Assumptions

1. Customer can have one or more shipping addresses for order delivery.
2. A customer can make zero or more product reviews.
3. Only one Wishlist is associated with each customer.
4. A Product can have zero or more reviews.
5. A customer can be associated with one territory.
6. Only one invoice must be associated with one payment.
7. Employee can act as general employee and supervisor.
8. Order Shipment provides shipment details about each product in the order.
9. Product Line means Category and many products can be associated.

Entity Relationship Diagram



Lucid Chart : [Access Link](#)

Relational Schema:

Our Supply Chain Management System has a total of 20 entities. They are listed below.

- 1) User
- 2) Admin
- 3) Cart
- 4) DeliveryAgent
- 5) Employee
- 6) Invoice
- 7) Message
- 8) Order
- 9) OrderDetails (i.e., OrderLine)
- 10) OrderTrack
- 11) Payment
- 12) PaymentType
- 13) Product
- 14) ProductLine
- 15) RawMaterial
- 16) Review
- 17) Supplier
- 18) Territory
- 19) Wishlist
- 20) WorkCenter

Table	Action	Rows	Type	Collation	Size
admins	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	16.0 KiB
cart	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	48.0 KiB
delivery_agent	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	32.0 KiB
employees	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	48.0 KiB
invoices	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	48.0 KiB
messages	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	32.0 KiB
orders	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	32.0 KiB
order_details	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	48.0 KiB
order_track	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	48.0 KiB
payment	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	48.0 KiB
payment_type	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	16.0 KiB
products	Browse Structure Search Insert Empty Drop	6	InnoDB	utf8mb4_general_ci	16.0 KiB
product_line	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	16.0 KiB
raw_materials	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	16.0 KiB
reviews	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	64.0 KiB
supplier	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	16.0 KiB
territory	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	16.0 KiB
users	Browse Structure Search Insert Empty Drop	7	InnoDB	utf8mb4_general_ci	16.0 KiB
wishlist	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	32.0 KiB
workcenter	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	16.0 KiB
20 tables	Sum	35	InnoDB	utf8mb4_general_ci	624.0 KiB

Admin

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	admin_id 🔑	int(100)			No	None		AUTO_INCREMENT
2	name	varchar(20)	utf8mb4_general_ci		No	None		
3	password	varchar(50)	utf8mb4_general_ci		No	None		

```
DROP TABLE IF EXISTS `admins`;  
CREATE TABLE IF NOT EXISTS `admins` (  
  `admin_id` int(100) NOT NULL AUTO_INCREMENT,  
  `name` varchar(20) NOT NULL,  
  `password` varchar(50) NOT NULL,  
  PRIMARY KEY (`admin_id`));
```

The `admins` table has three columns: `admin_id` (auto-incremented integer), `name` (varchar up to 20 characters), and `password` (varchar up to 50 characters). The `admin_id` column is the primary key for the table.

User

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	user_id 🔑	int(100)			No	None		AUTO_INCREMENT
2	name	varchar(20)	utf8mb4_general_ci		No	None		
3	email	varchar(50)	utf8mb4_general_ci		No	None		
4	password	varchar(50)	utf8mb4_general_ci		No	None		

```
DROP TABLE IF EXISTS `users`;  
CREATE TABLE IF NOT EXISTS `users` (  
  `user_id` int(100) NOT NULL AUTO_INCREMENT,  
  `name` varchar(20) NOT NULL,  
  `email` varchar(50) NOT NULL,  
  `password` varchar(50) NOT NULL,  
  PRIMARY KEY (`user_id`));
```

The users table consists of three columns: user_id (auto-incremented integer), name (varchar up to 20 characters), and email (varchar up to 50 characters), with an additional column password (varchar up to 50 characters). The user_id column serves as the primary key for the table.

➡ Wishlist

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	wishlist_id 🔑	int(100)			No	None		AUTO_INCREMENT
2	user_id 🔑	int(100)			No	None		
3	pid	int(100)			No	None		
4	name	varchar(100)	utf8mb4_general_ci		No	None		
5	price	int(100)			No	None		
6	image	varchar(100)	utf8mb4_general_ci		No	None		

```

DROP TABLE IF EXISTS `wishlist`;
CREATE TABLE IF NOT EXISTS `wishlist` (
  `wishlist_id` int(100) NOT NULL AUTO_INCREMENT,
  `user_id` int(100) NOT NULL,
  `pid` int(100) NOT NULL,
  `name` varchar(100) NOT NULL,
  `price` int(100) NOT NULL,
  `image` varchar(100) NOT NULL,
  PRIMARY KEY (`wishlist_id`),
  KEY `fk_user_wishlist` (`user_id`));

```

➡ Cart

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	cart_id 🔑	int(100)			No	None		AUTO_INCREMENT
2	user_id 🔑	int(100)			No	None		
3	pid 🔑	int(100)			No	None		
4	name	varchar(100)	utf8mb4_general_ci		No	None		
5	price	int(10)			No	None		
6	quantity	int(10)			No	None		
7	image	varchar(100)	utf8mb4_general_ci		No	None		

```

DROP TABLE IF EXISTS `cart`;
CREATE TABLE IF NOT EXISTS `cart` (
  `cart_id` int(100) NOT NULL AUTO_INCREMENT,
  `user_id` int(100) NOT NULL,
  `pid` int(100) NOT NULL,
  `name` varchar(100) NOT NULL,
  `price` int(10) NOT NULL,
  `quantity` int(10) NOT NULL,
  `image` varchar(100) NOT NULL,
  PRIMARY KEY (`cart_id`),
  KEY `fk_user` (`user_id`),
  KEY `fk_product` (`pid`));

```

Employee

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	employee_id 🔑	int(100)			No	None		AUTO_INCREMENT
2	first_name	varchar(100)	utf8mb4_general_ci		No	None		
3	last_name	varchar(100)	utf8mb4_general_ci		No	None		
4	address	varchar(200)	utf8mb4_general_ci		No	None		
5	city	varchar(50)	utf8mb4_general_ci		No	None		
6	state	varchar(50)	utf8mb4_general_ci		No	None		
7	zipcode	int(5)			No	None		
8	contact	varchar(20)	utf8mb4_general_ci		No	None		
9	supervisor_id 🔑	int(100)			Yes	NULL		
10	workcenter_id 🔑	int(100)			Yes	NULL		

```

DROP TABLE IF EXISTS `employees`;
CREATE TABLE IF NOT EXISTS `employees` (
  `employee_id` int(100) NOT NULL AUTO_INCREMENT,
  `first_name` varchar(100) NOT NULL,
  `last_name` varchar(100) NOT NULL,
  `address` varchar(200) NOT NULL,
  `city` varchar(50) NOT NULL,
  `state` varchar(50) NOT NULL,
  `zipcode` int(5) NOT NULL,
  `contact` varchar(20) NOT NULL,
  `supervisor_id` int(100) DEFAULT NULL,
  `workcenter_id` int(100) DEFAULT NULL,
  PRIMARY KEY (`employee_id`),
  KEY `fk_supervisor` (`supervisor_id`),
  KEY `fk_workcenter` (`workcenter_id`));

```

➔ Supplier

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	supplier_id 🔑	int(100)			No	None		AUTO_INCREMENT
2	name	varchar(50)	utf8mb4_general_ci		No	None		
3	address	varchar(100)	utf8mb4_general_ci		No	None		
4	city	varchar(20)	utf8mb4_general_ci		No	None		
5	state	varchar(20)	utf8mb4_general_ci		No	None		
6	zipcode	int(5)			No	None		
7	contact	varchar(20)	utf8mb4_general_ci		No	None		

```

DROP TABLE IF EXISTS `supplier`;
CREATE TABLE IF NOT EXISTS `supplier` (
  `supplier_id` int(100) NOT NULL AUTO_INCREMENT,
  `name` varchar(50) NOT NULL,
  `address` varchar(100) NOT NULL,
  `city` varchar(20) NOT NULL,
  `state` varchar(20) NOT NULL,
  `zipcode` int(5) NOT NULL,
  `contact` varchar(20) NOT NULL,
  PRIMARY KEY (`supplier_id`));

```

➔ Territory

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	territory_id 🔑	int(100)			No	None		AUTO_INCREMENT
2	territory_name	varchar(50)	utf8mb4_general_ci		No	None		
3	territory_location	varchar(50)	utf8mb4_general_ci		No	None		

```

DROP TABLE IF EXISTS `territory`;
CREATE TABLE IF NOT EXISTS `territory` (
  `territory_id` int(100) NOT NULL AUTO_INCREMENT,
  `territory_name` varchar(50) NOT NULL,
  `territory_location` varchar(50) NOT NULL,
  PRIMARY KEY (`territory_id`));

```


🔙 DeliveryAgent

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	delivery_agent_id 🔑	int(100)			No	None		AUTO_INCREMENT
2	name	varchar(100)	utf8mb4_general_ci		No	None		
3	contact	varchar(20)	utf8mb4_general_ci		No	None		
4	email	varchar(100)	utf8mb4_general_ci		No	None		
5	address	varchar(100)	utf8mb4_general_ci		No	None		
6	city	varchar(20)	utf8mb4_general_ci		No	None		
7	state	varchar(20)	utf8mb4_general_ci		No	None		
8	zipcode	int(5)			No	None		
9	territory_id 🔑	int(100)			No	None		

```

DROP TABLE IF EXISTS `delivery_agent`;
CREATE TABLE IF NOT EXISTS `delivery_agent` (
  `delivery_agent_id` int(100) NOT NULL AUTO_INCREMENT,
    `name` varchar(100) NOT NULL,
    `contact` varchar(20) NOT NULL,
    `email` varchar(100) NOT NULL,
    `address` varchar(100) NOT NULL,
    `city` varchar(20) NOT NULL,
    `state` varchar(20) NOT NULL,
    `zipcode` int(5) NOT NULL,
    `territory_id` int(100) NOT NULL,
  PRIMARY KEY (`delivery_agent_id`),
  KEY `FK_territory` (`territory_id`));

```

🔙 Product

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	product_id 🔑	int(100)			No	None		AUTO_INCREMENT
2	name	varchar(100)	utf8mb4_general_ci		No	None		
3	details	varchar(500)	utf8mb4_general_ci		No	None		
4	price	decimal(10,2)			No	None		
5	image_01	varchar(100)	utf8mb4_general_ci		No	None		
6	image_02	varchar(100)	utf8mb4_general_ci		No	None		
7	image_03	varchar(100)	utf8mb4_general_ci		No	None		

```

DROP TABLE IF EXISTS `products`;
CREATE TABLE IF NOT EXISTS `products` (
  `product_id` int(100) NOT NULL AUTO_INCREMENT,
  `name` varchar(100) NOT NULL,
  `details` varchar(500) NOT NULL,
  `price` decimal(10,2) NOT NULL,
  `image_01` varchar(100) NOT NULL,
  `image_02` varchar(100) NOT NULL,
  `image_03` varchar(100) NOT NULL,
  PRIMARY KEY (`product_id`));

```

➡ Order

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	order_id 🔑	int(100)			No	None		AUTO_INCREMENT
2	user_id 🔑	int(100)			No	None		
3	name	varchar(20)	utf8mb4_general_ci		No	None		
4	number	varchar(10)	utf8mb4_general_ci		No	None		
5	email	varchar(50)	utf8mb4_general_ci		No	None		
6	method	varchar(50)	utf8mb4_general_ci		No	None		
7	address	varchar(500)	utf8mb4_general_ci		No	None		
8	total_products	varchar(1000)	utf8mb4_general_ci		No	None		
9	total_price	int(100)			No	None		
10	placed_on	date			No	current_timestamp()		
11	payment_status	varchar(20)	utf8mb4_general_ci		No	pending		

```

DROP TABLE IF EXISTS `orders`;
CREATE TABLE IF NOT EXISTS `orders` (
  `order_id` int(100) NOT NULL AUTO_INCREMENT,
  `user_id` int(100) NOT NULL,
  `name` varchar(20) NOT NULL,
  `number` varchar(10) NOT NULL,
  `email` varchar(50) NOT NULL,
  `method` varchar(50) NOT NULL,
  `address` varchar(500) NOT NULL,
  `total_products` varchar(1000) NOT NULL,
  `total_price` int(100) NOT NULL,
  `placed_on` date NOT NULL DEFAULT current_timestamp(),
  `payment_status` varchar(20) NOT NULL DEFAULT 'pending',
  PRIMARY KEY (`order_id`),
  KEY `fk_userid_order` (`user_id`));

```

➡ OrderLine

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	order_detail_id 🔑	int(11)			No	None		AUTO_INCREMENT
2	order_id 🔑	int(11)			Yes	NULL		
3	product_id 🔑	int(11)			Yes	NULL		
4	product_name	varchar(255)	utf8mb4_general_ci		Yes	NULL		
5	quantity	int(11)			Yes	NULL		
6	price	decimal(10,2)			Yes	NULL		
7	has_review	tinyint(1)			No	0		

```

DROP TABLE IF EXISTS `order_details`;
CREATE TABLE IF NOT EXISTS `order_details` (
  `order_detail_id` int(11) NOT NULL AUTO_INCREMENT,
  `order_id` int(11) DEFAULT NULL,
  `product_id` int(11) DEFAULT NULL,
  `product_name` varchar(255) DEFAULT NULL,
  `quantity` int(11) DEFAULT NULL,
  `price` decimal(10,2) DEFAULT NULL,
  `has_review` tinyint(1) NOT NULL DEFAULT 0,
  PRIMARY KEY (`order_detail_id`),
  KEY `fk_order_details_order` (`order_id`),
  KEY `fk_order_details_product` (`product_id`));

```

➡ OrderTrack


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	track_id 🔑	int(11)			No	None		AUTO_INCREMENT
2	order_id 🔑	int(11)			Yes	NULL		
3	user_id 🔑	int(11)			Yes	NULL		
4	order_date	date			Yes	NULL		
5	status	varchar(50)	utf8mb4_general_ci		Yes	NULL		

```

DROP TABLE IF EXISTS `order_track`;
CREATE TABLE IF NOT EXISTS `order_track` (
  `track_id` int(11) NOT NULL AUTO_INCREMENT,
  `order_id` int(11) DEFAULT NULL,
  `user_id` int(11) DEFAULT NULL,
  `order_date` date DEFAULT NULL,
  `status` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`track_id`),
  KEY `fk_user_track` (`user_id`),
  KEY `fk_order_track` (`order_id`));

```

➡ ProductLine


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	product_line_id 	int(100)			No	None		AUTO_INCREMENT
2	product_line_name	varchar(100)	utf8mb4_general_ci		No	None		
3	number_of_products	int(100)			No	None		

```

DROP TABLE IF EXISTS `product_line`;
CREATE TABLE IF NOT EXISTS `product_line` (
  `product_line_id` int(100) NOT NULL AUTO_INCREMENT,
  `product_line_name` varchar(100) NOT NULL,
  `number_of_products` int(100) NOT NULL,
  PRIMARY KEY (`product_line_id`));

```

➡ WorkCenter

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	workcenter_id 	int(100)			No	None		AUTO_INCREMENT
2	workcenter_name	varchar(100)	utf8mb4_general_ci		No	None		
3	workcenter_location	varchar(100)	utf8mb4_general_ci		No	None		

```

DROP TABLE IF EXISTS `workcenter`;
CREATE TABLE IF NOT EXISTS `workcenter` (
  `workcenter_id` int(100) NOT NULL AUTO_INCREMENT,
  `workcenter_name` varchar(100) NOT NULL,
  `workcenter_location` varchar(100) NOT NULL,
  PRIMARY KEY (`workcenter_id`));

```

Review

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	review_id 🔑	int(100)			No	None		AUTO_INCREMENT
2	product_id 🔑	int(100)			No	None		
3	user_id 🔑	int(100)			No	None		
4	product_rating	decimal(2,1)			No	None		
5	comments	varchar(500)	utf8mb4_general_ci		Yes	NULL		
6	order_id 🔑	int(100)			No	None		

```

DROP TABLE IF EXISTS `reviews`;
CREATE TABLE IF NOT EXISTS `reviews` (
  `review_id` int(100) NOT NULL AUTO_INCREMENT,
  `product_id` int(100) NOT NULL,
  `user_id` int(100) NOT NULL,
  `product_rating` decimal(2,1) NOT NULL CHECK (`product_rating` >= 0 and `product_rating` <= 5),
  `comments` varchar(500) DEFAULT NULL,
  `order_id` int(100) NOT NULL,
  PRIMARY KEY (`review_id`),
  KEY `product_id` (`product_id`),
  KEY `user_id` (`user_id`),
  KEY `fk_order` (`order_id`));

```

RawMaterial

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	material_id 🔑	int(100)			No	None		AUTO_INCREMENT
2	material_name	varchar(100)	utf8mb4_general_ci		No	None		
3	material_standard_price	decimal(10,2)			No	None		
4	unit_of_measure	varchar(50)	utf8mb4_general_ci		No	None		
5	quantity_available	int(100)			No	None		




```

DROP TABLE IF EXISTS `raw_materials`;
CREATE TABLE IF NOT EXISTS `raw_materials` (
  `material_id` int(100) NOT NULL AUTO_INCREMENT,
  `material_name` varchar(100) NOT NULL,
  `material_standard_price` decimal(10,2) NOT NULL,
  `unit_of_measure` varchar(50) NOT NULL,
  `quantity_available` int(100) NOT NULL,

```



PRIMARY KEY (`material_id`));

Invoice

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	invoice_id 	int(100)			No	None		AUTO_INCREMENT
2	user_id 	int(100)			Yes	NULL		
3	order_id 	int(100)			Yes	NULL		
4	total_amount	decimal(10,2)			Yes	NULL		
5	invoice_date	date			Yes	NULL		

```
DROP TABLE IF EXISTS `invoices`;
CREATE TABLE IF NOT EXISTS `invoices` (
  `invoice_id` int(100) NOT NULL AUTO_INCREMENT,
  `user_id` int(100) DEFAULT NULL,
  `order_id` int(100) DEFAULT NULL,
  `total_amount` decimal(10,2) DEFAULT NULL,
  `invoice_date` date DEFAULT NULL,
  PRIMARY KEY (`invoice_id`),
  KEY `user_id` (`user_id`),
  KEY `order_id` (`order_id`));
```

Message

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	message_id 	int(100)			No	None		AUTO_INCREMENT
2	user_id 	int(100)			No	None		
3	name	varchar(100)	utf8mb4_general_ci		No	None		
4	email	varchar(100)	utf8mb4_general_ci		No	None		
5	number	varchar(12)	utf8mb4_general_ci		No	None		
6	message	varchar(500)	utf8mb4_general_ci		No	None		

```
DROP TABLE IF EXISTS `messages`;
CREATE TABLE IF NOT EXISTS `messages` (
  `message_id` int(100) NOT NULL AUTO_INCREMENT,
  `user_id` int(100) NOT NULL,
  `name` varchar(100) NOT NULL,
  `email` varchar(100) NOT NULL,
```

```

`number` varchar(12) NOT NULL,
`message` varchar(500) NOT NULL,
PRIMARY KEY (`message_id`),
KEY `fk_userid` (`user_id`));

```

➡ Payment

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	payment_id 🔑	int(100)			No	None		AUTO_INCREMENT
2	order_id 🔑	int(100)			No	None		
3	payment_type_id 🔑	int(100)			No	None		
4	payment_date	date			No	None		
5	payment_amount	decimal(10,2)			No	None		
6	payment_status	varchar(50)	utf8mb4_general_ci		No	None		

```

DROP TABLE IF EXISTS `payment`;
CREATE TABLE IF NOT EXISTS `payment` (
  `payment_id` int(100) NOT NULL AUTO_INCREMENT,
  `order_id` int(100) NOT NULL,
  `payment_type_id` int(100) NOT NULL,
  `payment_date` date NOT NULL,
  `payment_amount` decimal(10,2) NOT NULL,
  `payment_status` varchar(50) NOT NULL,
  PRIMARY KEY (`payment_id`),
  KEY `fk_payment_type` (`payment_type_id`),
  KEY `fk_order_payment` (`order_id`));

```

➡ PaymentType

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	payment_type_id 🔑	int(100)			No	None		AUTO_INCREMENT
2	payment_type_name	varchar(100)	utf8mb4_general_ci		No	None		
3	description	varchar(200)	utf8mb4_general_ci		No	None		

```

DROP TABLE IF EXISTS `payment_type`;
CREATE TABLE IF NOT EXISTS `payment_type` (
  `payment_type_id` int(100) NOT NULL AUTO_INCREMENT,
  `payment_type_name` varchar(100) NOT NULL,
  `description` varchar(200) NOT NULL,
  PRIMARY KEY (`payment_type_id`));

```

Normalization:

In the context of normalization:

1. First Normal Form (1NF):

- Each table in our relational schema has a unique primary key attribute, ensuring that each record can be identified uniquely.
- Every piece of data in our tables is atomic, meaning it cannot be further divided. Therefore, our tables are already in 1st normal form.

2. Second Normal Form (2NF):

- All non-prime attributes (attributes not part of the primary key) are fully dependent on the primary key.
- We have eliminated partial dependencies in our relations, ensuring that each non-prime attribute is directly related to the primary key. Hence, our tables are already in 2nd normal form.

3. Third Normal Form (3NF):

- Our relations do not contain non-prime attributes that depend on other non-prime attributes, which, in turn, are dependent on the primary key.
- There are no transitive dependencies; all non-prime attributes are fully dependent on the primary key. Therefore, our relations are already in 3rd normal form.

In summary, our database design meets the criteria of 1NF, 2NF, and 3NF, indicating a high level of normalization, and no further normalization is deemed necessary.

Technologies:

- ➡ Visual Studio Code
- ➡ PhpMyAdmin
- ➡ Xampp
- ➡ MySQL Work bench

To Run the Supply Chain Management System, Follow These Steps:

1. Unzip the Project File:

Unzip the project file and locate your XAMPP directory.

2. Copy to "htdocs" Directory:

Within the XAMPP directory, find the "htdocs" folder. Paste the extracted project folder into this directory.

3. Open Google Chrome Browser:

Launch your Google Chrome browser.

4. Access phpMyAdmin:

Go to the URL "http://localhost/phpmyadmin" in your browser.

5. Create Database:

Create a database with the name 'scmdb'.

6. Import Database File:

Click on the "Import" tab in phpMyAdmin. Choose the database file (scmdb.sql) provided

7. Configure Database:

Set up the database by following the import process.

8. Access the Application:

Once configured, for accessing User system visit "<http://localhost/SCM/>"

Admin system visit "<http://localhost/SCM/admin>" in your browser.

9. Retrieve Login Details:

Refer to the user and admin tables in phpMyAdmin to login and access the system.

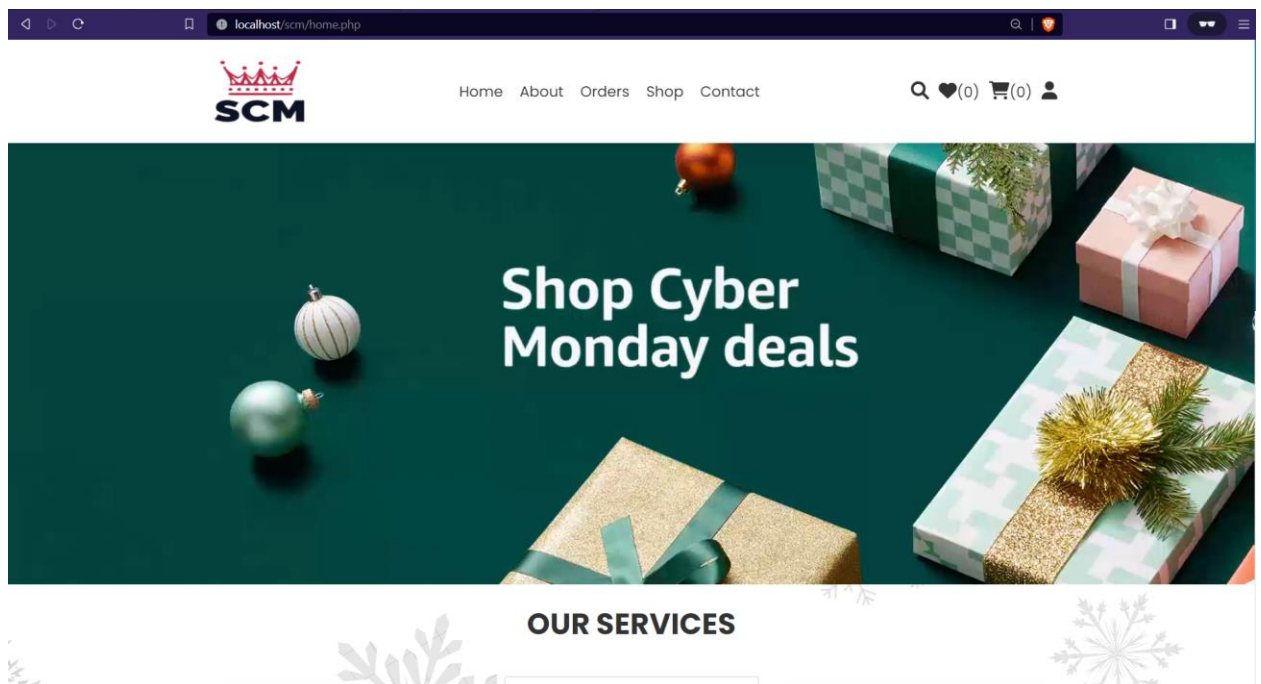
10. Login and Utilize:

Enter the obtained login details to access and use the Supply Chain Management System.

Follow these steps meticulously to ensure a smooth setup and operation of the supply chain management application.

Application Walkthrough:

1. Access the application through specified link and home screen images are provided below.





please login or register first!

Register

Login



LOGIN NOW

Login Now

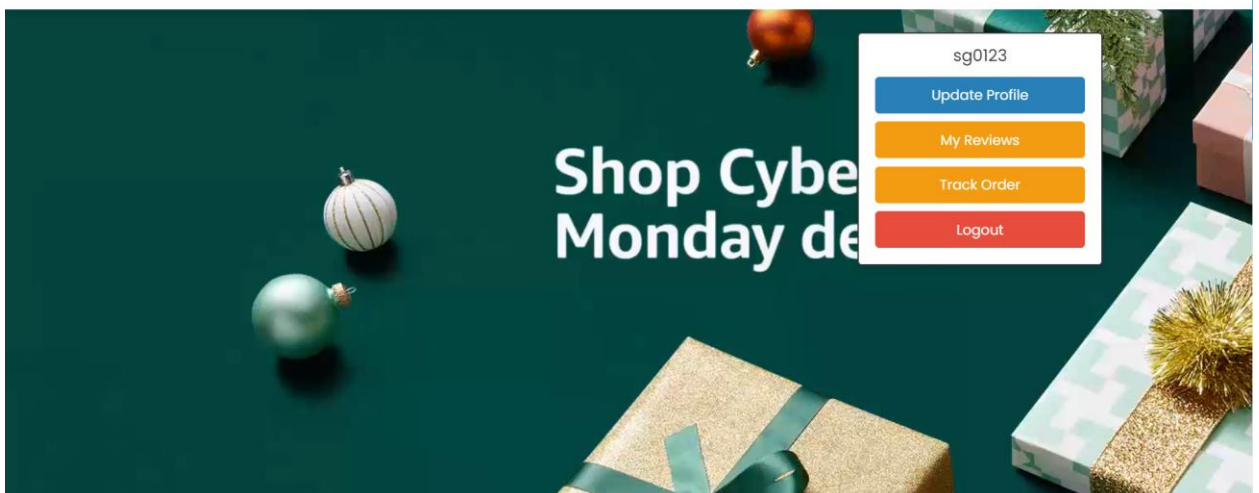
Register Now

Quick Links

- > [Home](#)
- > [About](#)
- > [Shop](#)
- > [Contact](#)

Contact Us

+1 (987) 654-3210
fdbgroup16@my.unt.edu



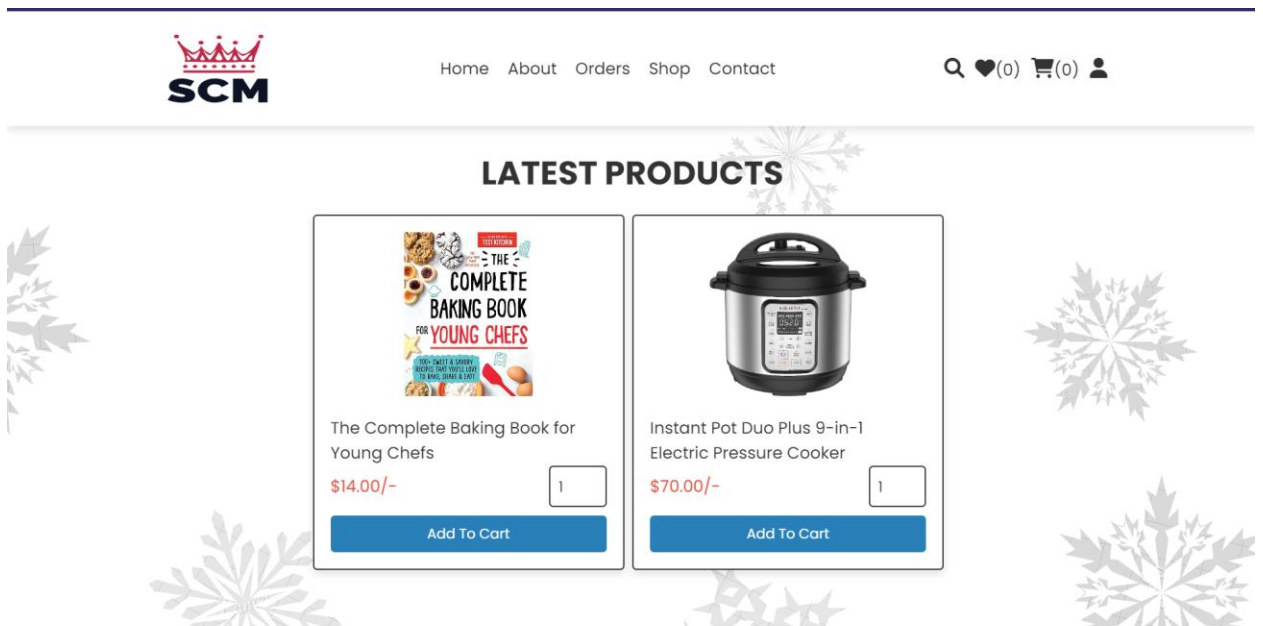
sg0123

Update Profile

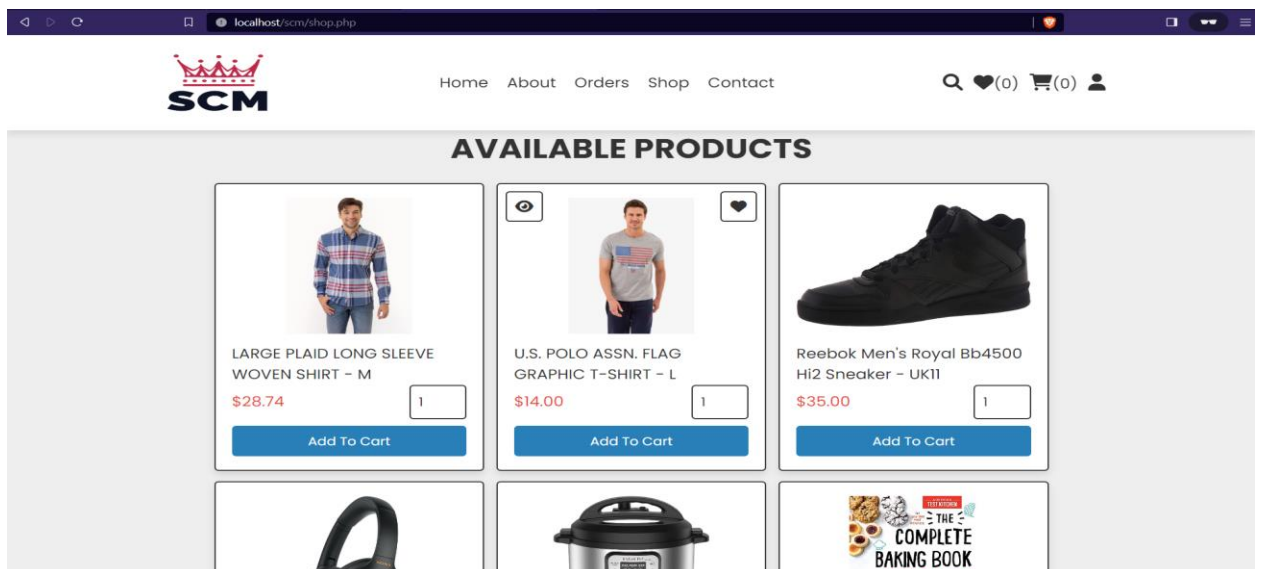
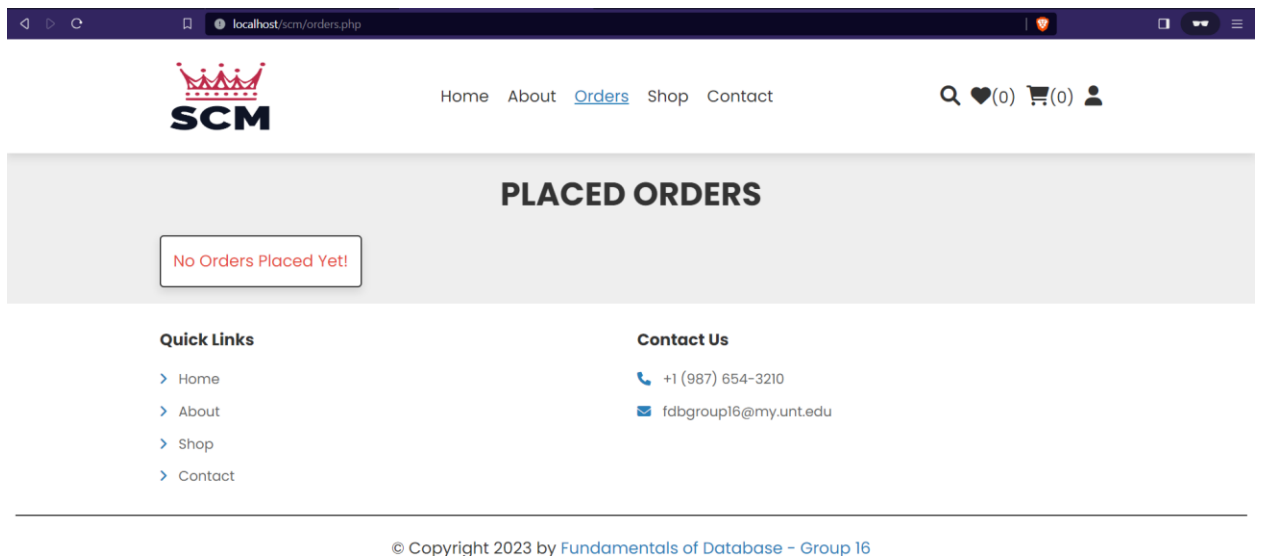
My Reviews

Track Order

Logout



2. The navigation and making orders in the system is demonstrated below.




localhost/scm/quick_view.php?pid=107

SCM

Home About Orders Shop Contact

🔍❤️(0)🛒(0)👤

QUICK VIEW






U.S. POLO ASSN. FLAG GRAPHIC T-SHIRT - L

\$14.00/-

U.S. POLO ASSN. FLAG GRAPHIC T-SHIRT

Add To Cart Add To Wishlist

1



localhost/scm/shop.php


added to wishlist!

SCM

Home About Orders Shop Contact

🔍❤️(1)🛒(0)👤


AVAILABLE PRODUCTS



LARGE PLAID LONG SLEEVE
WOVEN SHIRT - M

\$28.74


Add To Cart



U.S. POLO ASSN. FLAG
GRAPHIC T-SHIRT - L

\$14.00

Add To Cart



Reebok Men's Royal Bb4500
Hi2 Sneaker - UK11

\$35.00

Add To Cart

localhost/scm/shop.php


added to cart!

SCM

Home About Orders Shop Contact

🔍❤️(0)🛒(1)👤


AVAILABLE PRODUCTS



LARGE PLAID LONG SLEEVE
WOVEN SHIRT - M

\$28.74


Add To Cart



U.S. POLO ASSN. FLAG
GRAPHIC T-SHIRT - L

\$14.00

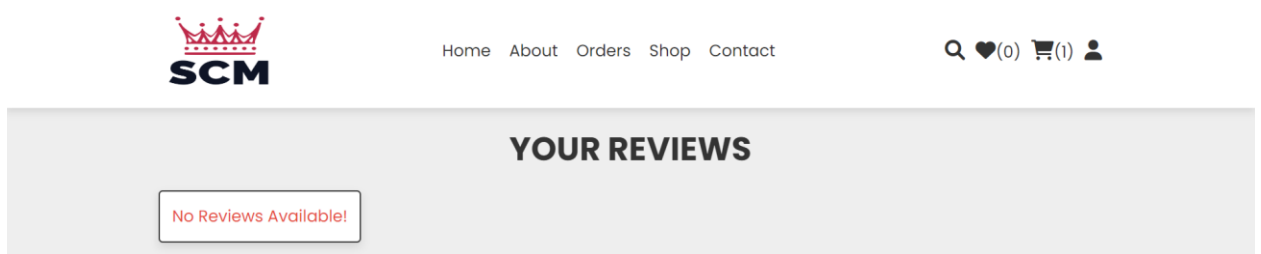
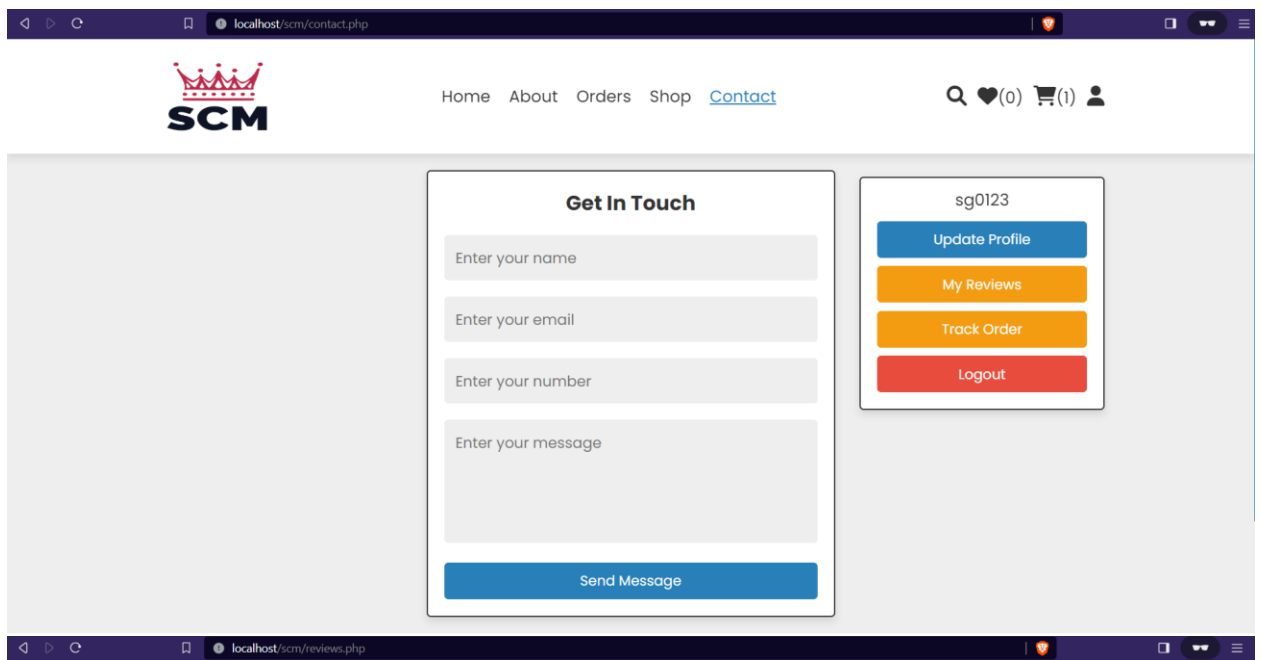
Add To Cart



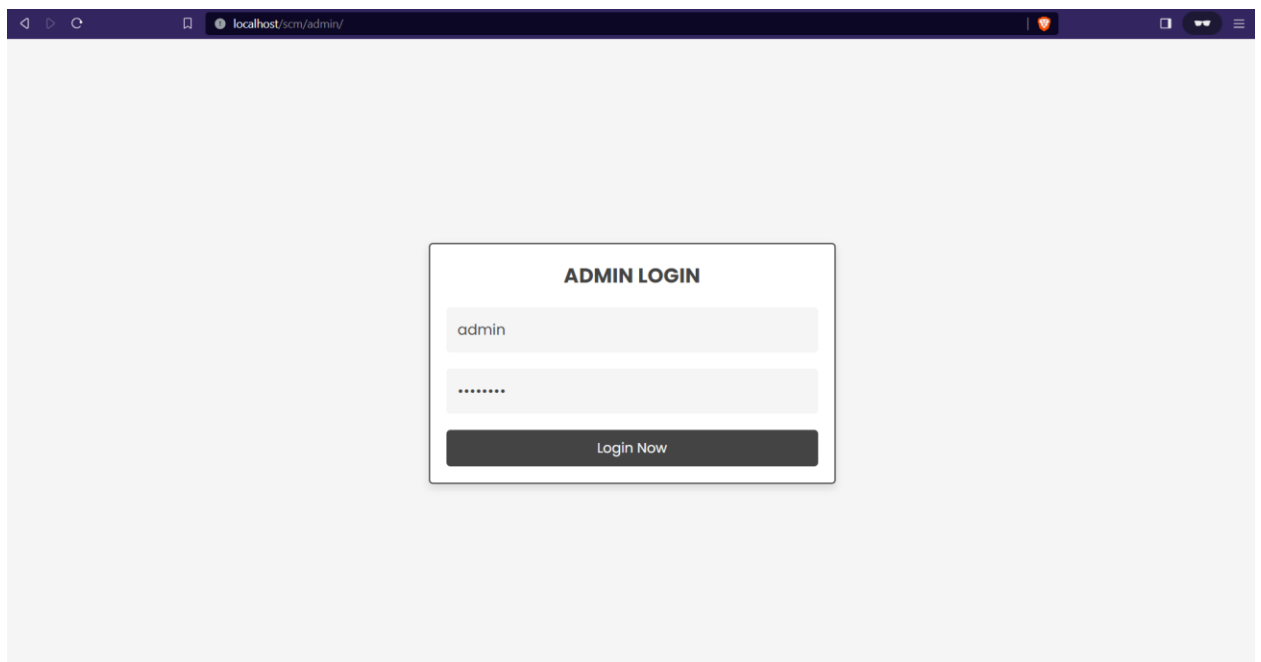
Reebok Men's Royal Bb4500
Hi2 Sneaker - UK11

\$35.00

Add To Cart



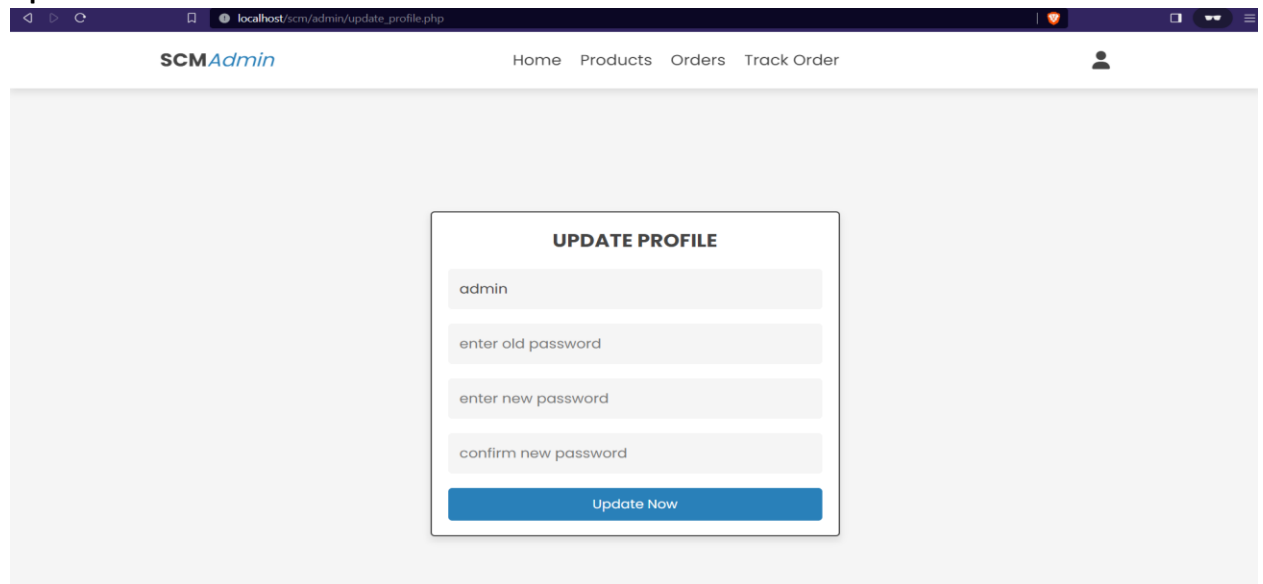
3. The admin panel provides more options to handle data and tables present in the database, below attached is the home screen of admin panel.



ADMIN DASHBOARD



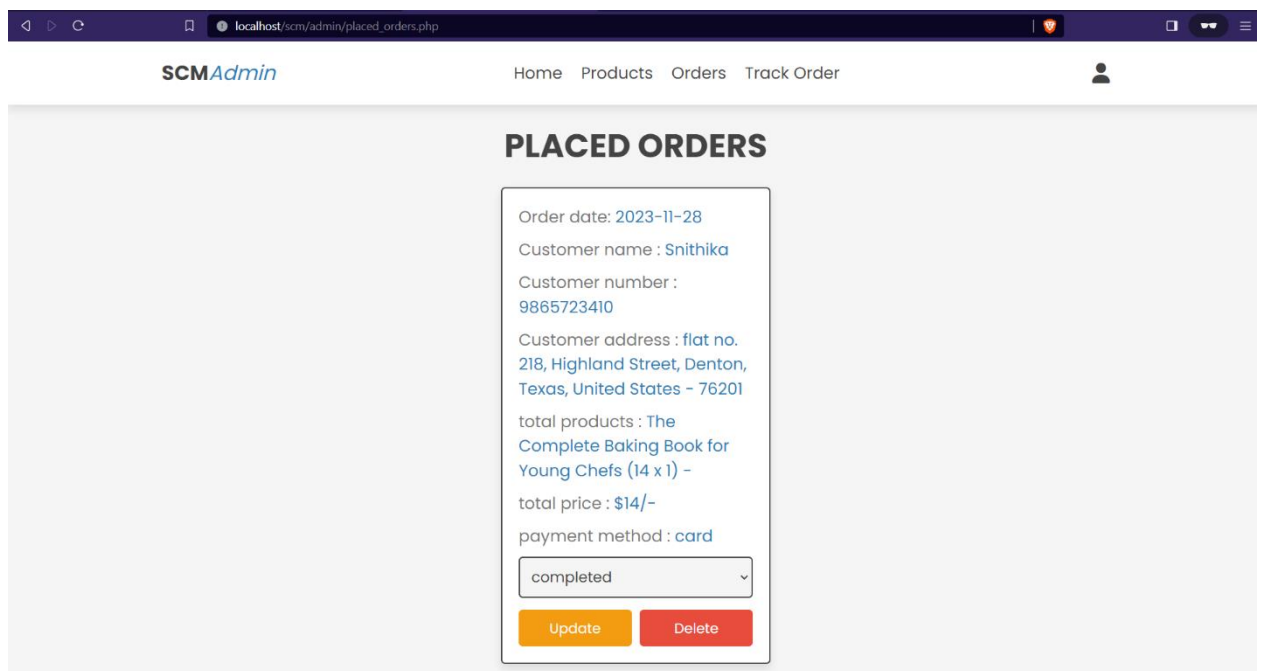
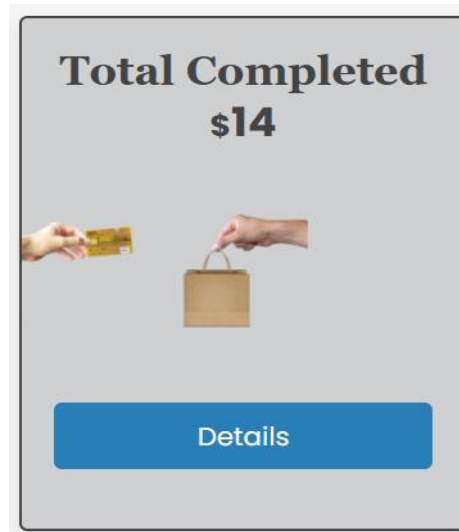
Update Profile



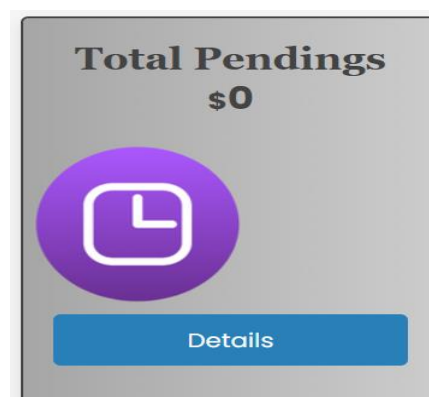
The Update Profile form is displayed in a modal window. It contains the following fields and a button:

- Username: admin
- Old Password: enter old password
- New Password: enter new password
- Confirm New Password: confirm new password
- Update Now button

The admin is asked to enter details for updating the profile, it includes username, new password and confirm new password. The old password is also asked to ensure that authorized updating action.



We can observe that the orders with payment completed is listed under Total completed section with value being sum of the total prices of all the completed orders.\



Total Pending is zero because there are no orders available at present with pending payments. The only order available is already paid.






localhost/scm/admin/products.php

SCMAdmin Home Products Orders Track Order

ADD PRODUCT

product name (required) <input type="text" value="enter product name"/>	product price (required) <input type="text" value="enter product price"/>
image 01 (required) <input type="button" value="Choose File"/> No file chosen	image 02 (required) <input type="button" value="Choose File"/> No file chosen
image 03 (required) <input type="button" value="Choose File"/> No file chosen	product details (required) <input type="text" value="enter product details"/>

PRODUCTS ADDED

 <p>LARGE PLAID LONG SLEEVE WOVEN SHIRT - M \$28.74/- An embodiment of tasteful, timeless style, this LARGE PLAID LONG SLEEVE WOVEN SHIRT.</p> <p><input type="button" value="Update"/> <input type="button" value="Delete"/></p>	 <p>U.S. POLO ASSN. FLAG GRAPHIC T-SHIRT - L \$14.00/- U.S. POLO ASSN. FLAG GRAPHIC T-SHIRT</p> <p><input type="button" value="Update"/> <input type="button" value="Delete"/></p>	 <p>Reebok Men's Royal Bb4500 Hi2 Sneaker - UK11 \$35.00/- These mid-cut, basketball-inspired men's shoes are sure to be an instant favorite, with a blended leather and mesh upper weighing in for heritage comfort.</p>
--	---	---

Here we can observe that there are DML operations like update, delete and insert are available with products in the products page. The total number of products available in the system is presented using count under the label in home screen.

SCMAdmin

Home Products Orders Track Order

User Name (required) Email (required)

Enter username Enter email

Password (required) Confirm Password (required)

Enter password Confirm password

Add User

User ID: 1 Username: sg0123 Email: user@mail.com Delete	User ID: 3 Username: ng0123 Email: niharika@mail.com Delete	User ID: 4 Username: sp0456 Email: snithika@mail.com Delete
User ID: 5 Username: sd0789 Email: sumanth@mail.com Delete	User ID: 6 Username: bg0345 Email: balaji@mail.com Delete	User ID: 7 Username: sv0234 Email: suchitha@mail.com Delete
User ID: 8 Username: pp0789 Email: pranitha@mail.com Delete		

Keeping in mind, the user already has update profile option and the credentials are confidential, we did not provide update option with the Admin panel.

SCMAdmin

Home Products Orders Track Order

ADMIN ACCOUNTS

add new admin
Register Admin

admin id : 1 admin name : admin Delete Update	admin id : 2 admin name : admin11 Delete
---	--

In the similar way, an admin can update his own profile but not other admins with same privileges.

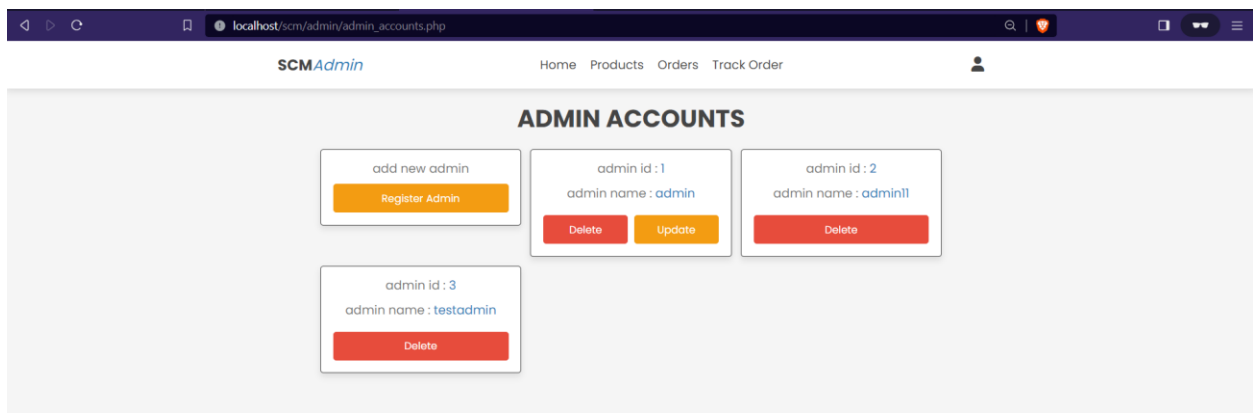
SCMAdmin

Home Products Orders Track Order

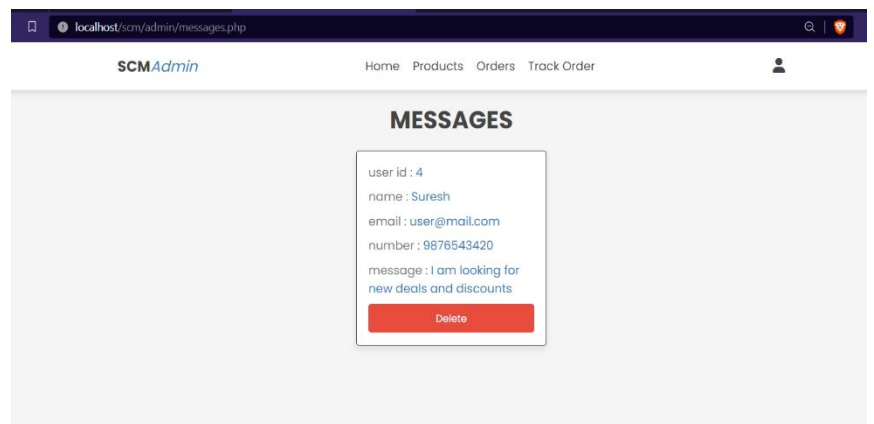
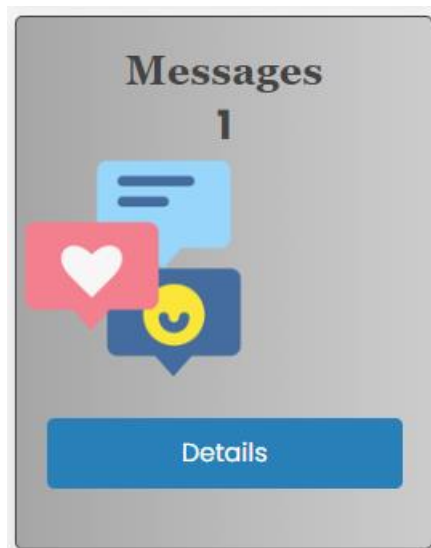
REGISTER NOW

testadmin

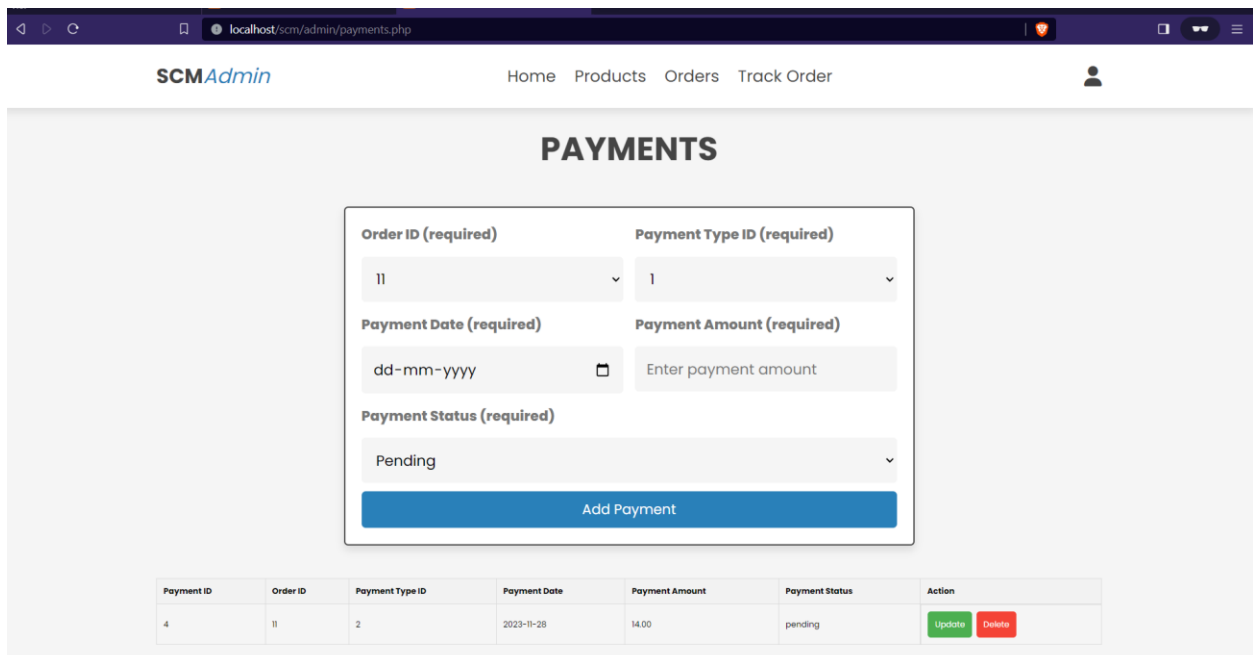
Register Now



The addition of new admin to the table in database is successful.



Any feedback or contact messages sent by the user is populated here and admin can read and delete these messages.



Here dynamic javascript is used, such that only associated ids will be displayed for selecting from drop down menu. The records in the table are attached below along with action column having update and delete options.

```

if(isset($_POST['update_payment'])){
    $order_id = $_POST['order_id'];
    $payment_status = $_POST['payment_status'];
    $payment_status = filter_var($payment_status, FILTER_SANITIZE_STRING);
    $update_payment = $conn->prepare("UPDATE `orders` SET payment_status = ? WHERE order_id = ?");
    $update_payment->execute([$payment_status, $order_id]);
    $message[] = 'payment status updated!';
}

if(isset($_GET['delete'])){
    $delete_id = $_GET['delete'];
    $delete_review = $conn->prepare("DELETE FROM `reviews` WHERE order_id = ?");
    $delete_review->execute([$delete_id]);
    $delete_invoice = $conn->prepare("DELETE FROM `invoices` WHERE order_id = ?");
    $delete_invoice->execute([$delete_id]);
    $delete_payment = $conn->prepare("DELETE FROM `payment` WHERE order_id = ?");
    $delete_payment->execute([$delete_id]);
    $delete_order_details = $conn->prepare("DELETE FROM `order_details` WHERE order_id = ?");
    $delete_order_details->execute([$delete_id]);
    $delete_order_track = $conn->prepare("DELETE FROM `order_track` WHERE order_id = ?");
    $delete_order_track->execute([$delete_id]);
    $delete_order = $conn->prepare("DELETE FROM `orders` WHERE order_id = ?");
    $delete_order->execute([$delete_id]);
    header('location:placed_orders.php');
}

```

Above attached code snippet is a sample scenario, where all the associated tables and records are updated having references.

```

if(isset($_POST['update'])){
    $pid = $_POST['pid'];
    $name = $_POST['name'];
    $name = filter_var($name, FILTER_SANITIZE_STRING);
    $price = $_POST['price'];
    $price = filter_var($price, FILTER_SANITIZE_STRING);
    $details = $_POST['details'];
    $details = filter_var($details, FILTER_SANITIZE_STRING);

    $update_product = $conn->prepare("UPDATE `products` SET name = ?, price = ?, details = ? WHERE product_id = ?");
    $update_product->execute([$name, $price, $details, $pid]);

    $message[] = 'product updated successfully!';

    $old_image_01 = $_POST['old_image_01'];
    $image_01 = $_FILES['image_01']['name'];
    $image_01 = filter_var($image_01, FILTER_SANITIZE_STRING);
    $image_size_01 = $_FILES['image_01']['size'];
    $image_tmp_name_01 = $_FILES['image_01']['tmp_name'];
    $image_folder_01 = '../uploaded_img/'.$image_01;
}

```

In the above image, update operation is performed on product table with the option to change records and provide new images for product. All the values are sanitized such that any unnecessary tags, special characters and additional spaces are stripped from the string.

```

// Check if a workcenter is being added
if(isset($_POST['add_workcenter'])){
    $workcenter_name = $_POST['workcenter_name'];
    $workcenter_location = $_POST['workcenter_location'];

    $insert_workcenter = $conn->prepare("INSERT INTO `workcenter` (workcenter_name, workcenter_location) VALUES (?, ?)");
    $insert_workcenter->execute([$workcenter_name, $workcenter_location]);
    header('location:workcenters.php');
}

// Check if a workcenter is being updated
if(isset($_POST['update_workcenter'])){
    $update_id = $_POST['update_id'];
    $new_workcenter_name = $_POST['new_workcenter_name'];
    $new_workcenter_location = $_POST['new_workcenter_location'];

    $update_workcenter = $conn->prepare("UPDATE `workcenter` SET workcenter_name = ?, workcenter_location = ? WHERE workcenter_id = ?");
    $update_workcenter->execute([$new_workcenter_name, $new_workcenter_location, $update_id]);
    header('location:workcenters.php');
}

// Check if a workcenter is being deleted
if(isset($_GET['delete'])){
    $delete_id = $_GET['delete'];
    $delete_workcenter = $conn->prepare("DELETE FROM `workcenter` WHERE workcenter_id = ?");
    $delete_workcenter->execute([$delete_id]);
    header('location:workcenters.php');
}

```

All the three DML operations on records in WorkCenter table is provided here.

In conclusion, the Supply Chain Management System exhibits a well-designed database that serves as the foundation for seamless operations, encompassing diverse functionalities such as customer management, order fulfilment, payment processing, and product development. The user interface is not only intuitive but also aesthetically appealing, enhancing user experiences and contributing to the overall efficiency of the application.