**Database Design and Development Semester Project**

Project Name:

ShopSphere: The E-Commerce Ecosystem

Group Name:

ShopSphere

Team Members:

Jack Krantz: Team Lead

Bal Acharya: Programmer

Online retail store database.

Our target users will be small online retail businesses. Our database will manage:

* Customer (Customer name, customer ID, email address, shipping address, billing address, payment information)
* Product (Product name, product ID, price, stock quantity, description, category ID)
* Category (Category name, category ID)
* Order (Order ID, product ID, quantity, price)
* Shipping (Shipping ID, order ID, carrier, tracking number, estimated delivery date)
* Payment Transactions (Transaction ID, order ID, payment amount, date, method, status)

**Coding: Bal**

1. shopspheres/add\_customer.html

<!DOCTYPE html>

<html>

<head>

<title>Add Customer</title>

</head>

<body>

<h2>Add Customer</h2>

<form action="insert\_customer.php" method="post">

<label for="customerName">Name:</label><br>

<input type="text" id="customerName" name="customerName" required><br>

<label for="emailAddress">Email:</label><br>

<input type="email" id="emailAddress" name="emailAddress" required><br>

<label for="shippingAddress">Shipping Address:</label><br>

<input type="text" id="shippingAddress" name="shippingAddress" required><br>

<label for="billingAddress">Billing Address:</label><br>

<input type="text" id="billingAddress" name="billingAddress" required><br>

<label for="paymentInformation">Payment Information:</label><br>

<input type="text" id="paymentInformation" name="paymentInformation" required><br>

<input type="submit" value="Submit">

</form>

</body>

</html>

1. shopspheres/db\_connect.php

<?php

$servername = "localhost";

$username = "root"; // Your MySQL username, 'root' is the default for XAMPP

$password = "3541"; // Your MySQL password, default is no password in XAMPP

$dbname = "shopsphere"; // Your database name

// Create connection

$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

?>

1. shopspheres/delete\_customer.php

<?php

require 'db\_connect.php';

if (isset($\_GET['id']) && is\_numeric($\_GET['id'])) {

$customerID = $\_GET['id'];

$sql = "DELETE FROM Customer WHERE CustomerID=?";

if ($stmt = $conn->prepare($sql)) {

$stmt->bind\_param("i", $customerID);

if ($stmt->execute()) {

echo "Customer deleted successfully.";

} else {

echo "Error deleting record: " . $stmt->error;

}

$stmt->close();

} else {

echo "Error preparing statement: " . $conn->error;

}

$conn->close();

header("Location: read\_customer.php"); // Redirect back to the main customer page

exit;

} else {

echo "Invalid request.";

}

?>

1. shopspheres/insert\_customer.php

<?php

ini\_set('display\_errors', 1);

error\_reporting(E\_ALL);

require 'db\_connect.php'; // Ensure this points to the correct file where you establish a database connection

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

// Check and sanitize input before insertion

$customerName = isset($\_POST['customerName']) ? mysqli\_real\_escape\_string($conn, $\_POST['customerName']) : '';

$emailAddress = isset($\_POST['emailAddress']) ? mysqli\_real\_escape\_string($conn, $\_POST['emailAddress']) : '';

$shippingAddress = isset($\_POST['shippingAddress']) ? mysqli\_real\_escape\_string($conn, $\_POST['shippingAddress']) : '';

$billingAddress = isset($\_POST['billingAddress']) ? mysqli\_real\_escape\_string($conn, $\_POST['billingAddress']) : '';

$paymentInformation = isset($\_POST['paymentInformation']) ? mysqli\_real\_escape\_string($conn, $\_POST['paymentInformation']) : '';

// Prepare SQL and bind parameters

$sql = "INSERT INTO Customer (CustomerName, EmailAddress, ShippingAddress, BillingAddress, PaymentInformation)

VALUES (?, ?, ?, ?, ?)";

if ($stmt = $conn->prepare($sql)) {

$stmt->bind\_param("sssss", $customerName, $emailAddress, $shippingAddress, $billingAddress, $paymentInformation);

if ($stmt->execute()) {

echo "New record created successfully.";

header("Location: read\_customer.php"); // Redirect back to the main customer page

exit;

} else {

echo "Error: " . $stmt->error;

}

$stmt->close();

} else {

echo "Error preparing statement: " . $conn->error;

}

$conn->close();

} else {

echo "No data submitted.";

}

?>

**Coding: Jack**

1. shopspheres/read\_customer.php

<?php

require 'db\_connect.php'; // Ensure this points to the correct file where you establish a database connection

echo "<!DOCTYPE html>

<html>

<head>

<title>List of Customers</title>

<style>

table {

width: 100%;

border-collapse: collapse;

}

table, th, td {

border: 1px solid black;

padding: 8px;

text-align: middle;

}

th {

background-color: #f2f2f2;

}

.add-button {

margin-bottom: 20px;

padding: 10px;

background-color: #4CAF50;

color: white;

border: none;

border-radius: 5px;

cursor: pointer;

text-decoration: none; /\* Ensures it looks good as a link \*/

}

</style>

</head>

<body>

<h2>List of Customers</h2>

<a href='add\_customer.html'><button class='add-button'>Add New Customer</button></a>";

// SQL to fetch all customer records

$sql = "SELECT CustomerID, CustomerName, EmailAddress, ShippingAddress, BillingAddress, PaymentInformation FROM Customer";

$result = $conn->query($sql);

// Check if there are results

if ($result->num\_rows > 0) {

// Table headers

echo "<table><tr><th>ID</th><th>Name</th><th>Email</th><th>Shipping Address</th><th>Billing Address</th><th>Payment Information</th><th>Actions</th></tr>";

// Output data of each row

while($row = $result->fetch\_assoc()) {

echo "<tr><td>".$row["CustomerID"]."</td><td>".$row["CustomerName"]."</td><td>".$row["EmailAddress"]."</td><td>".$row["ShippingAddress"]."</td><td>".$row["BillingAddress"]."</td><td>".$row["PaymentInformation"]."</td><td><a href='update\_customer\_form.php?id=".$row["CustomerID"]."'>Edit</a> | <a href='delete\_customer.php?id=".$row["CustomerID"]."' onclick='return confirm(\"Are you sure you want to delete this customer?\")'>Delete</a></td></tr>";

}

echo "</table>";

} else {

echo "No customers found.";

}

// End HTML output

echo "</body></html>";

$conn->close();

?>

1. shopspheres/update\_customer.php

<?php

ini\_set('display\_errors', 1);

error\_reporting(E\_ALL);

require 'db\_connect.php';

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

$customerID = $\_POST['customerID'];

$customerName = mysqli\_real\_escape\_string($conn, $\_POST['customerName']);

$emailAddress = mysqli\_real\_escape\_string($conn, $\_POST['emailAddress']);

$shippingAddress = mysqli\_real\_escape\_string($conn, $\_POST['shippingAddress']);

$billingAddress = mysqli\_real\_escape\_string($conn, $\_POST['billingAddress']);

$paymentInformation = mysqli\_real\_escape\_string($conn, $\_POST['paymentInformation']);

$sql = "UPDATE Customer SET

CustomerName=?,

EmailAddress=?,

ShippingAddress=?,

BillingAddress=?,

PaymentInformation=?

WHERE CustomerID=?";

if ($stmt = $conn->prepare($sql)) {

$stmt->bind\_param("sssssi", $customerName, $emailAddress, $shippingAddress, $billingAddress, $paymentInformation, $customerID);

if ($stmt->execute()) {

echo "Record updated successfully.";

} else {

echo "Error updating record: " . $stmt->error;

}

$stmt->close();

} else {

echo "Error preparing statement: " . $conn->error;

}

$conn->close();

header("Location: read\_customer.php"); // Redirect back to the main customer page

exit;

} else {

echo "No data submitted.";

}

?>

1. shopspheres/update\_customer\_form.php

<?php

require 'db\_connect.php'; // Make sure this points correctly to your database connection script

// Check if 'id' is set in the URL

if(isset($\_GET['id']) && is\_numeric($\_GET['id'])) {

// Get the customer ID from the URL

$customerID = $\_GET['id'];

// Prepare SQL to fetch the customer's data

$sql = "SELECT \* FROM Customer WHERE CustomerID = $customerID";

$result = $conn->query($sql);

if ($result->num\_rows > 0) {

// Fetch the customer's data

$row = $result->fetch\_assoc();

?>

<!DOCTYPE html>

<html>

<head>

<title>Edit Customer</title>

</head>

<body>

<h2>Edit Customer</h2>

<form action="update\_customer.php" method="post">

<input type="hidden" name="customerID" value="<?php echo $row['CustomerID']; ?>">

<label for="customerName">Name:</label><br>

<input type="text" id="customerName" name="customerName" value="<?php echo $row['CustomerName']; ?>" required><br>

<label for="emailAddress">Email:</label><br>

<input type="email" id="emailAddress" name="emailAddress" value="<?php echo $row['EmailAddress']; ?>" required><br>

<label for="shippingAddress">Shipping Address:</label><br>

<input type="text" id="shippingAddress" name="shippingAddress" value="<?php echo $row['ShippingAddress']; ?>" required><br>

<label for="billingAddress">Billing Address:</label><br>

<input type="text" id="billingAddress" name="billingAddress" value="<?php echo $row['BillingAddress']; ?>" required><br>

<label for="paymentInformation">Payment Information:</label><br>

<input type="text" id="paymentInformation" name="paymentInformation" value="<?php echo $row['PaymentInformation']; ?>" required><br>

<input type="submit" value="Update">

</form>

</body>

</html>

<?php

} else {

echo "No customer found with ID $customerID";

}

} else {

echo "Invalid request.";

}

$conn->close();

?>

1. shopspheres/.htaccess

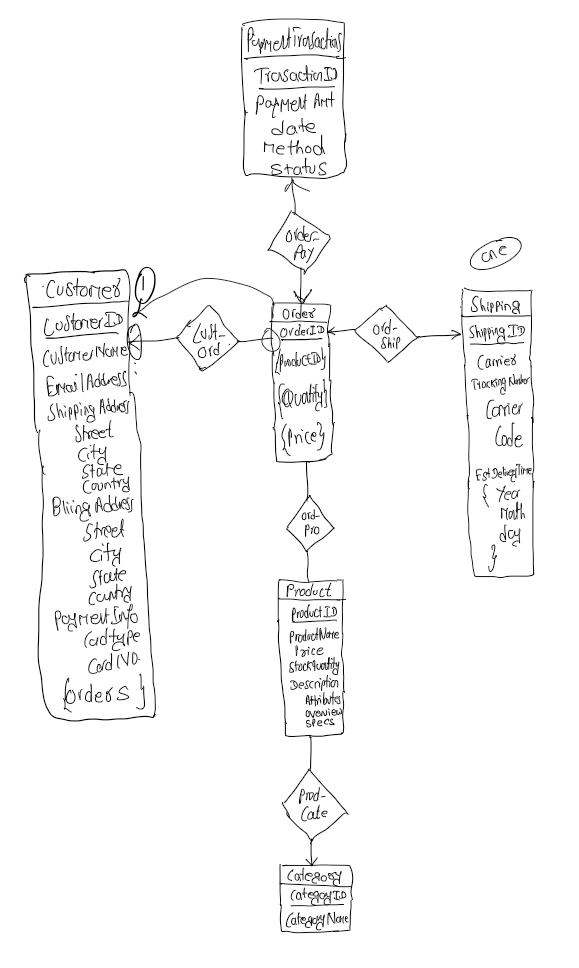
RewriteEngine On

RewriteBase /shopspheres/

# Rewrite "customers" to "read\_customer.php"

RewriteRule ^customers$ read\_customer.php [NC,L]

**Bal & Jack – Schema Diagram**



**Bal & Jack: UI Design**

**A screenshot of a computer

Description automatically generated**

**Experiments and Results:**

**Select, Create , Read, Update, Delete Functionalities for Customers Database**

**A close-up of a white sheet

Description automatically generated**

**Discussion and Conclusion**

For our semester project titled "ShopSphere: The E-Commerce Ecosystem," we aimed to design and develop a comprehensive database system tailored to small online retail businesses. This system is intended to manage various essential entities including customers, products, orders, shipping details, and payment transactions. Our objective was to create a user-friendly interface that facilitates efficient management of these entities through well-designed web forms and back-end scripts.

Database Design and Implementation

Schema Design

We meticulously designed a relational database schema to capture all necessary information across six main tables: Customer, Product, Category, Order, Shipping, and Payment Transactions. Each table was designed to not only store specific data relevant to its function but also to support efficient data retrieval and manipulation. We ensured that primary keys were uniquely identifying each entry, and foreign keys were used to establish relationships between tables, promoting data integrity and relational management.

Normalization

To optimize database efficiency and maintain data integrity, we applied normalization principles up to the third normal form (3NF). This approach helped in reducing data redundancy and improving data dependency by ensuring that every non-key attribute is only dependent on the primary key.

Web Interface Development

Front-end Design

The front-end of our application was built using HTML and enhanced with CSS for styling. We focused on creating intuitive and responsive forms for data entry and modification, ensuring they are accessible and easy to use.

Back-end Implementation

Our back-end logic was implemented using PHP, facilitating the interaction between the user interface and the MySQL database. We developed scripts for creating, reading, updating, and deleting records, ensuring that all operations were performed securely and efficiently. We emphasized secure programming practices, such as using prepared statements in PHP to prevent SQL injection attacks.

Security Measures

Recognizing the importance of security, especially when dealing with payment information, we implemented several measures to protect our data. This included sanitizing and validating all user inputs to prevent common web vulnerabilities. Additionally, we considered secure methods for handling sensitive payment details, though integrating with a dedicated payment gateway would be a suitable enhancement for future iterations.

Testing and Validation

We conducted comprehensive testing across all components of our system. This included unit testing individual functions and integration testing to ensure that all parts of the application worked cohesively. We also implemented error-handling mechanisms to manage and respond to unexpected user inputs or database errors effectively.

Challenges and Solutions

Throughout the development process, we encountered several challenges, including scalability concerns as data volume grows and ensuring data security. We addressed these by optimizing SQL queries, considering caching solutions, and keeping our software stack updated. Regular security audits and adhering to best practices helped in maintaining a secure environment.

Conclusion

The development of the ShopSphere system provided us with valuable insights into the practical aspects of designing and implementing a functional and secure e-commerce database system. It reinforced the importance of thoughtful design, rigorous testing, and continuous security practices. While the current implementation meets the project requirements effectively, future work could explore advanced features such as real-time data analytics and integration with external APIs for enhanced functionality.

**Reference**

Most of the coding we did ourselves. Out of that we use ChatGPT for ideas and debug.