Order and Cashier Management System

Overview

The Order and Cashier Management Project is a System that manage the employee management, order processing, inventory tracking, and sales monitoring. The system features role-based access, where administrators manage employees, stocks, and sales, while cashiers handle orders and payments.

Features

User Authentication : Secure log in functionality for users.

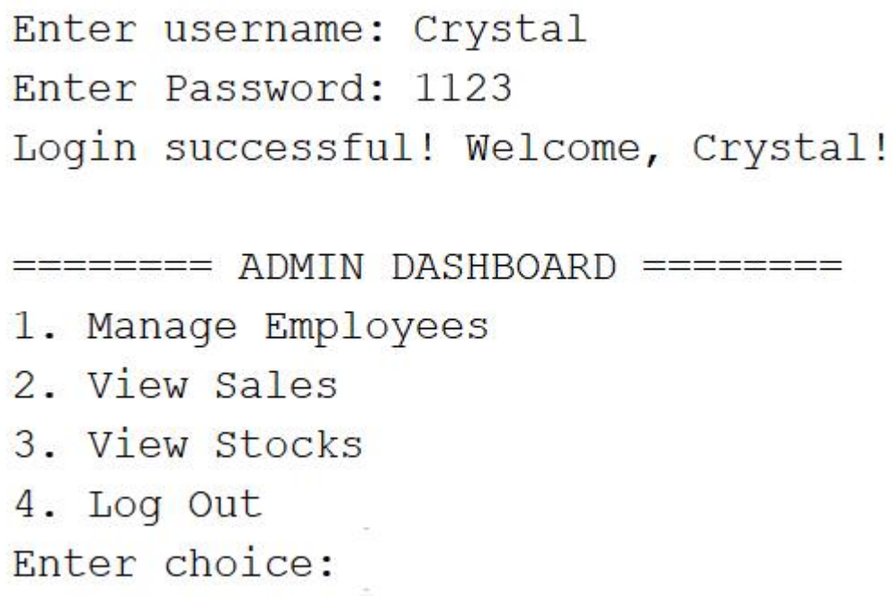
CRUD Operations for users: Create, Read, Update, Delete user information.

View Sales : functionality to view sales.

View Stocks : functionality to view and update stocks

Screenshots of CRUD Functionality

Log in



Add Employee

```
Enter choice: 1

======== MANAGE EMPLOYEES ========
1. Add Employee
2. View Employees
3. Edit Employee
4. Remove Employee
5. Back to Dashboard
Enter choice: 1
Enter Employee ID: 5
Enter Name: Tally
Enter Email: tally@gmail.com
Enter Password: 1234
Enter Role (admin/cashier): admin
 Employee added successfully with role: admin
```

View Employees

```
======== MANAGE EMPLOYEES ========
1. Add Employee
2. View Employees
3. Edit Employee
4. Remove Employee
5. Back to Dashboard
Enter choice: 2

========= EMPLOYEES LIST =========
ID          | Name            | Email                       | Role
-------------------------------------------------------------
1           | Crystal         | admin@example.com           | admin
2           | Ogille          | cashier1@example.com        | cashier
3           | Jesse           | cashier2@example.com        | cashier
4           | Jenie           | cashier3@example.com        | cashier
5           | Tally           | tally@gmail.com             | admin
```

Deleted[Crysta Joy Herda]:

Deleted[Crysta Joy Herda]: mployees

Edit

Employee

```
======== MANAGE EMPLOYEES ========
1. Add Employee
2. View Employees
3. Edit Employee
4. Remove Employee
5. Back to Dashboard
Enter choice: 3
Enter Employee ID to edit: 5
Enter new Name: Taltal
Enter new Email: Taltal@example.com
Enter new Password: 1234
Enter new Role (admin/cashier): admin
 Employee details updated successfully!
```

```
======== MANAGE EMP
1. Add Employee
2. View Employees
3. Edit Employee
4. Remove Employee
5. Back to Dashboar
Enter choice: 2

========= EMPLOYEES
ID          | Name
-------------------
1           | Crysta
2           | Ogille
3           | Jesse
4           | Jenie
5           | Tally
```

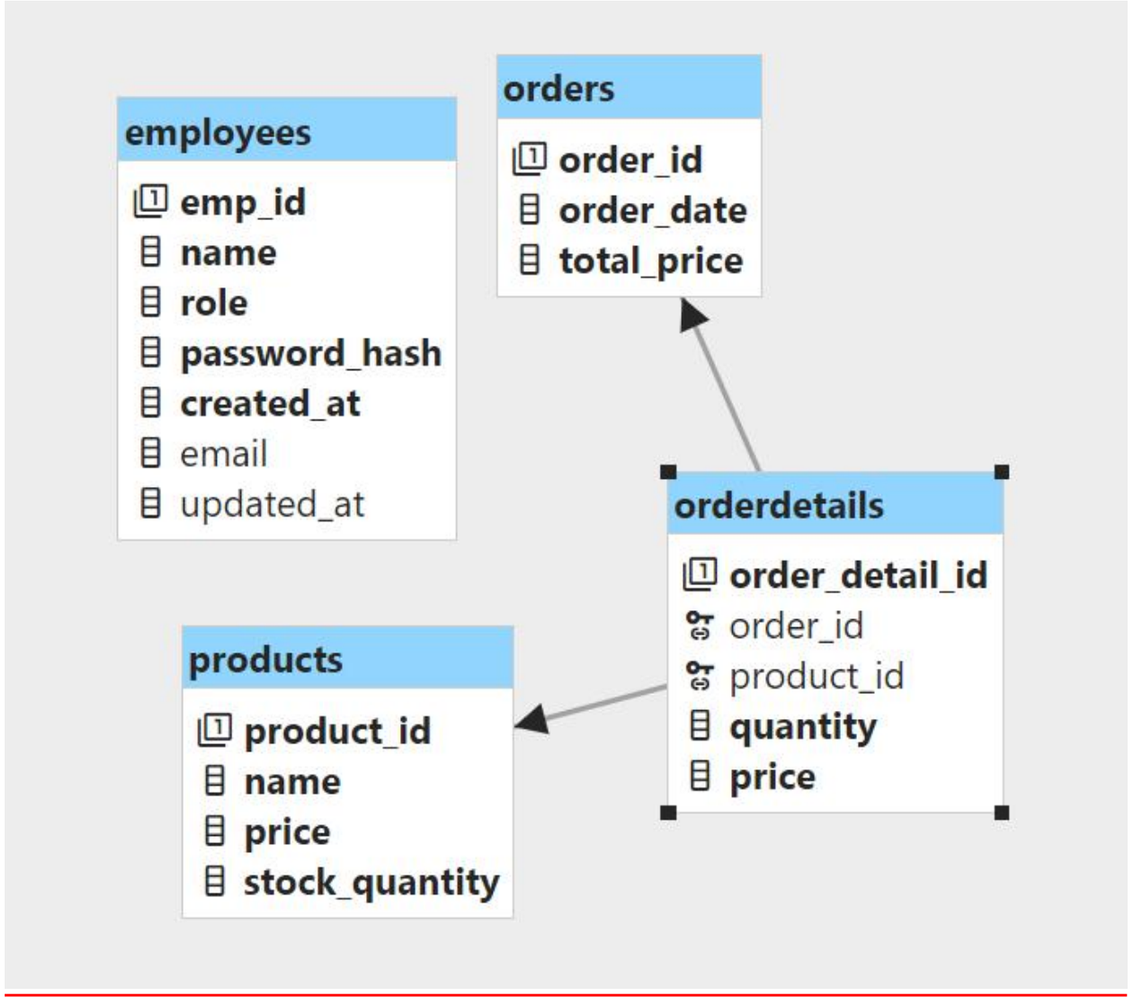
Deleted[Crysta Joy Herda]:

Deleted[Crysta Joy Herda]:

Remove Employee

```
======== MANAGE EMPLOYEES ========
1. Add Employee
2. View Employees
3. Edit Employee
4. Remove Employee
5. Back to Dashboard
Enter choice: 4
Enter Employee ID to remove: 5
 Employee removed successfully!
```

DataBase Schema Diagram


employees
emp_id
name
role
password_hash
created_at
email
updated_at
orders
order_id
order_date
total_price
orderdetails
order_detail_id
order_id
product_id
quantity
price
products
product_id
name
price
stock_quantity

Code Snippet

Formatted[Crysta Joy Herda]: Font: 20 pt, Bold

Add Employee

```
   public boolean addEmployee(String empId, String name, String email, String password, String role) {
    String query = "INSERT INTO employees (emp_id, name, email, password_hash, role) VALUES (?, ?, ?, ?, ?)";

    try (Connection conn = EmployeeDB.getConnection();
         PreparedStatement stmt = conn.prepareStatement(query)) {
        stmt.setString(1, empId);
        stmt.setString(2, name);
        stmt.setString(3, email);
        stmt.setString(4, PasswordUtil.hashPassword(password));
        stmt.setString(5, role);

        return stmt.executeUpdate() > 0;
    } catch (SQLException e) {

        return false;
    }
}
```

View Employee

```
public void viewEmployees() {
    String query = "SELECT emp_id, name, email, role FROM employees";

    try (Connection conn = EmployeeDB.getConnection();
         PreparedStatement stmt = conn.prepareStatement(query);
         ResultSet rs = stmt.executeQuery()) {

        System.out.println("\n========= EMPLOYEES LIST =========");
        System.out.printf("%-10s | %-15s | %-25s | %-10s%n", "ID", "Name", "Email", "Role");
        System.out.println("-----------------------------------------------------------------");

        while (rs.next()) {
            System.out.printf("%-10s | %-15s | %-25s | %-10s%n",
                rs.getString("emp_id"),
                rs.getString("name"),
                rs.getString("email"),
                rs.getString("role"));
        }
    } catch (SQLException e) {

    }
}
```

Edit Employee

```
    public boolean editEmployee(String empId, String newName, String newEmail, String newPassword, String newRole) {
    String query = "UPDATE employees SET name = ?, email = ?, password_hash = ?, role = ? WHERE emp_id = ?";

    try (Connection conn = EmployeeDB.getConnection();
         PreparedStatement stmt = conn.prepareStatement(query)) {
        stmt.setString(1, newName);
        stmt.setString(2, newEmail);
        stmt.setString(3, PasswordUtil.hashPassword(newPassword));
        stmt.setString(4, newRole);
        stmt.setString(5, empId);

        return stmt.executeUpdate() > 0;
    } catch (SQLException e) {

        return false;
    }
}
    }
```

Remove Employee

```
public boolean removeEmployee(String name) {
    String query = "DELETE FROM employees WHERE emp_id = ?";

    try (Connection conn = EmployeeDB.getConnection();
         PreparedStatement stmt = conn.prepareStatement(query)) {
        stmt.setString(1, name);
        return stmt.executeUpdate() > 0;
    } catch (SQLException e) {

        return false;
    }
}
```

Formatted[Crysta Joy Herda]: Font: 20 pt