ONLINE SHOPPING PLATFORM

OOPS MINI PROJECT REPORT

Submitted by

PAVITHRA.S
(231801123)

In partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY IN ARTIFICIAL INTELLIGENCE AND DATA
SCIENCE

RAJALAKSHMI ENGINEERING

COLLEGE, ANNA UNIVERSITY,

CHENNAI: 602 105

MAY 2024

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI 602 105

BONAFIDE CERTIFICATE

Certified that this report title **"ONLINE SHOPPING PLATFORM"** is the Bonafide work of **PAVITHRA.S(231801123)** who carried out the mini project work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE   SIGNATURE

Hod Name with designation   Supervisor name with designation
HEAD OF THE DEPARTMENT,   SUPERVISOR,
Professor,   Assistant Professor,
Department of AI&DS,   Department of AI&DS,
Rajalakshmi Engineering College   Rajalakshmi Engineering
Chennai – 602 105.   College, Chennai – 602 105.

Submitted for the OOPS Mini project review held on_____

Internal Examiner   External Examiner

TABLE OF

CONTENTS

1.1 Abstract:

The primary objective of the JDBC-powered Online Shopping Platform in Java is to provide a reliable and efficient platform for users to seamlessly browse products, add them to their cart, and complete purchases, all while utilizing Java's database connectivity capabilities. This system ensures data integrity and consistency through JDBC transactions, establishing a secure connection with a relational database for real-time updates on product availability, order status, and user information. The user-friendly interface, coupled with scalability, makes it an effective solution for optimizing the online shopping experience, catering to various customer

.

1.2 Introduction:

The Online Shopping Platform is an innovative software solution designed to revolutionize tra ditional retail and e-commerce experiences. This system automates the purchasing
process, significantly reducing time and operational costs. Administered through a secure authentication system, the platform ensures that only authorized personnel can manage product listings, customer orders, inventory, and other critical functionalities. By integrating advanced technology into the online shopping experience, the platform provides a seamless, efficient, and user-friendly environment, offering a marked improvement over traditional manual retail methods and enhancing the overall customer journey.

1.3 Purpose:

Th e purpose of this project is to create an efficient and user-friendly online shopping platform that benefits both customers and administrators. The system aims to: -Simplify the process of browsing and purchasing products online. -Enhance the shopping experience by providing users with detailed product information, personalized recommendations, and easy checkout options. - Enable administrators to manage product listings, customer orders, inventory, and payment processing effectively. -Store and analyze customer and sales data for business insights and informed decision- making

1.4 Scope of the Project:

The envisioned Online Shopping Platform aims to seamlessly interact with administrators and efficiently fulfill all proposed functionalities. Built on Java (JDBC) with a MySQL database, the system ensures smooth management of product details, reducing response time for customer queries and providing real-time updates on inventory. This project addresses the complexities associated with manual shopping processes, storing comprehensive information about products, prices, and customer orders. Emphasizing features like verification, validation, security, and user-friendliness, the system strives to optimize the online shopping experience, offering a robust solution that encompasses the entire spectrum of e-commerce management.

1.5 Software Requirement Specification:

The Software Requirement Specification (SRS) outlines the functional and non-functional requirements of the Online Shopping Platform. This document serves as a blueprint for the development of the system, detailing the technologies, tools, and architecture required to build a robust, efficient, and user-friendly e-commerce solution. The Online Shopping Platform is designed to offer a seamless shopping experience to users while providing administrators with easy management capabilities. The front-end ensures that users can browse products, view their shopping carts, and complete transactions. The back-end handles complex business logic, integrates with the database for data storage, and manages user sessions and transactions. By combining robust technologies like Java, MySQL, and modern web development tools, the system aims to provide a scalable, secure, and responsive platform that meets the needs of both customers and administrators. This SRS ensures that the final system will be efficient, reliable, and easy to maintain.

Definitions, Acronyms, and Abbreviations:

SRS - Software Requirements Specification.

References and Acknowledgement:

[1] https://www.javatpoint.com/java-awt
[2] https://www.javatpoint.com/java-swing

Overall Description:

The Online Shopping Platform provides authorized users with seamless access to product listings, customer orders, and transaction records, streamlining the online shopping process for customers worldwide. The system simplifies operations for e-commerce businesses, offering an efficient and user-friendly interface for browsing products , managing inventory, processing payments, and handling customer interactions, making it an ideal solution for modern retail environments.

Product Perspective:

Utilizing a client/server architecture, the system is designed to be compatible with the Microsoft Windows Operating System. The front end is developed using Java AWT and SWING, while the backend leverages the MySQL server for efficient data management.

Product Functionality:

The Online Shopping Platform provides user-friendly, menu-driven interfaces for: a) Admin Register: Registering new administrators. b) Admin Login: Logging in existing administrators.
c) Add Product: Storing new product details, including descriptions, prices, and categories. d) View Product: Viewing and updating existing product information. e) Delete Product: Deleting existing products from the inventory. f) Add Order: Creating new orders for customers. g) Update Order: Viewing and modifying existing customer orders. h) Remove Admin: Deleting specific administrator accounts.

User and Characteristics:

Qualification: Users should have at least basic educational qualifications, such as matriculation, and be comfortable with English. Experience: Familiarity with the university registration process is advantageous. Technical Experience: Users are expected to have elementary knowledge of computers for optimal system interaction.

Operating Environment: *Hardware Requirements: -*
Processor: Any Processor over i3 - Operating System:
Windows 8, 10, 11 - Processor Speed: 2.0 GHz - RAM:
4GB - Hard Disk: 500GB

Software Requirements:

- Database: MySQL - Frontend: Java (SWING, AWT) - Technology: Java
(JDBC)

Constraints:

- System access limited to administrators. - Delete operation restricted to
administrators without additional checks for simplicity. - Administrators must
exercise caution during deletion to maintain data consistency.

Assumptions and Dependencies:

- System administrators create and confidentially communicate login IDs and
passwords to users.

User Interface:

For your online shopping platform, here's a modified set of driven interfaces:

a) Admin Register: Registering new administrators.

b) Admin Login: Logging in existing administrators.

c) Add Product: Storing new product details in the inventory.

d) View Product: Viewing and updating existing product information.

e) Delete Product: Deleting products from the inventory.

f) Add Order: Creating new orders for users.

g) Update Order: Viewing and modifying existing order details.

h) Remove Admin: Deleting specific administrator accounts.

Hardware Interface:

- Screen resolution of at least 640 x 480 or above. -
Compatible with any version of Windows 8, 10, 11.

Software Interface:

a) MS-Windows Operating System b) Java AWT and SWING for designing the front end c) MySQL for the backend d) Platform: Java Language e) Integrated Development Environment (IDE): Eclipse EE

Functional Requirements:

1. Log in Module (LM):
○ Users (administrators) access the Login Module. LM supports user login with a username and password. Passwords are masked for security. Successful login verification by the database administrator is required for access.

2. Registered Users Module (RUM):
○ After successful login, users (administrators) can navigate through the application. Users can view detailed information about products, orders, and inventory. Users can update and maintain product details, including modifying inventory levels and prices.

3. Administrator Module (AM):
○ Upon successful login, the system displays administrative functions. Functions include adding, updating, and deleting product details. The "Add" function allows administrators to input new product details, such as descriptions, prices, and categories. The "Update" function enables administrators to modify existing product details in the database. The "Delete" function allows administrators to remove discontinued or unwanted products from the inventory. All add, update, or delete requests trigger the AM module to communicate with the Server Module (SM) for necessary database changes.

4. Server Module (SM):
○ SM acts as an intermediary between various modules and the database (DB). Receives requests from different modules and formats pages for display. Validates and executes requests received from other modules. Handles communication with the database, ensuring data consistency and integrity, especially regarding product details, orders, and inventory management.

Non-Functional Requirements:

Performance:

- The system must handle real-time product browsing and order placement requests efficiently, ensuring a response time of less than 2 seconds for adding items to the cart and completing a purchase. Critical failures, such as payment processing errors, must be addressed instantly to ensure a smooth and uninterrupted shopping experience.

Reliability:

- The system is mission-critical; in case of abnormal operation or downtime, immediate measures should be taken to resolve the issue and restore normal functionality. The platform should ensure consistent and accurate performance, handling peak loads effectively, such as during flash sales or promotional events.

Availability:

- Under normal operating conditions, user requests for browsing products, viewing inventory , and processing orders should be handled within 2 seconds to maintain a seamless shopping experience. Immediate feedback on order status, payment confirmation, and delivery details should be provided to enhance user satisfaction.

Security:

- A robust security mechanism must be implemented on the server side to prevent unauthorized access, protect user payment and personal information, and ensure the integrity of the shopping platform. User data, including account details, addresses, and transaction history, must be securely stored and managed to maintain confidentiality and comply with privacy regulations.
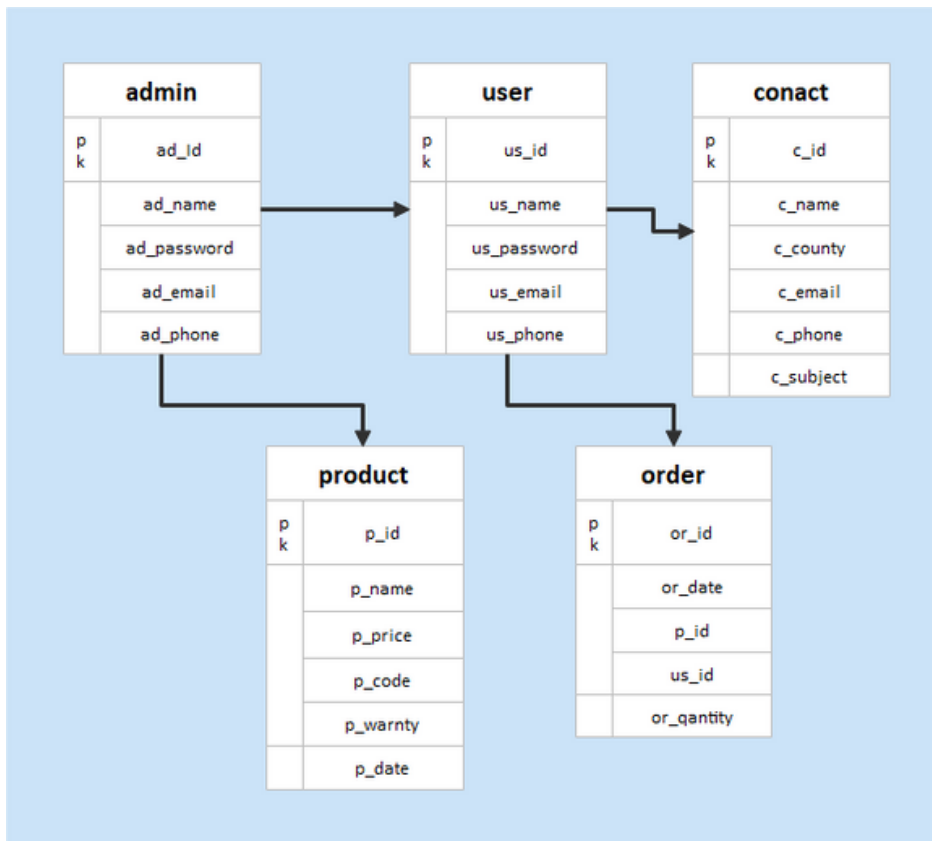
Maintainability:

- Comprehensive design documents outlining software and database maintenance procedures must be created to facilitate regular updates and modifications to the online shopping platform. Administrative access should be provided for maintaining the platform at both the front end and back end, ensuring long-term functionality, scalability, and adaptability to evolving business needs.
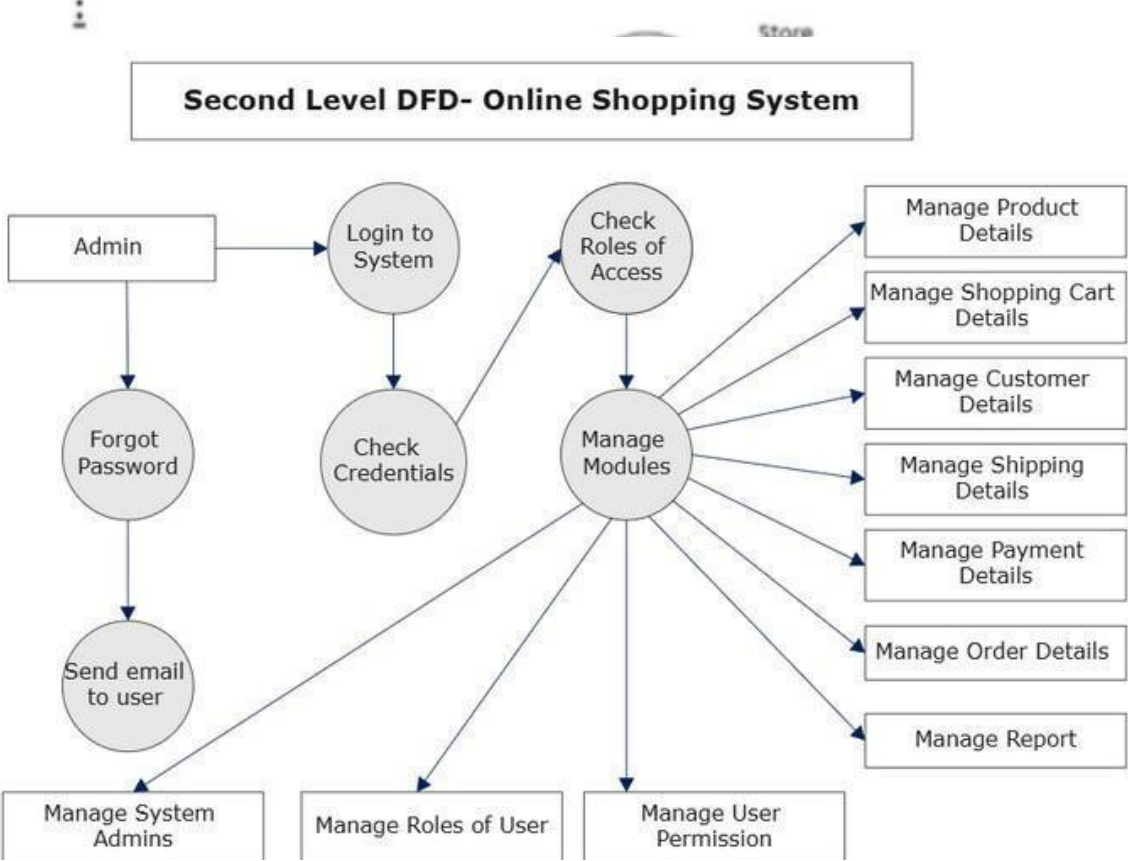
«Subsystem»
Online Shopping

View Items

«include»

Make Purchase

«include»

Checkout

Client Register

Registered Customer

Web Customer

New Customer

«Service»
Authentication

Identity Provider

Credit Payment Service

PayPal
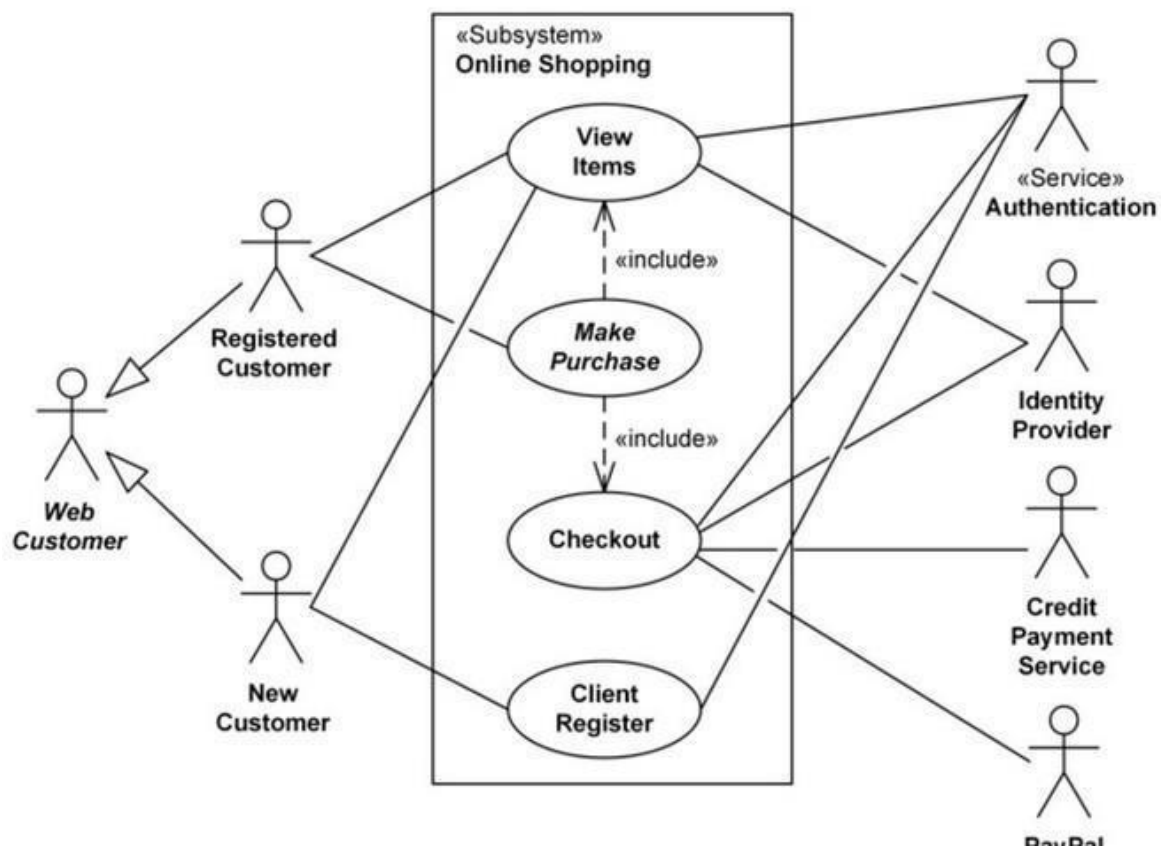
2.2 Entity-relationship diagram:

E-R (Entity-Relationship) Diagram is used to represents the relationship between entities in the table.

Entity -relationship diagram:

admin	
p k	ad_Id
	ad_name
	ad_password
	ad_email
	ad_phone

user	
p k	us_id
	us_name
	us_password
	us_email
	us_phone

conact	
p k	c_id
	c_name
	c_county
	c_email
	c_phone
	c_subject

product	
p k	p_id
	p_name
	p_price
	p_code
	p_warnty
	p_date

order	
p k	or_id
	or_date
	p_id
	us_id
	or_qantity

2.3 Data-flow diagram:

Second Level DFD- Online Shopping System

3. Module description
Admin: Register:

Admin can register with a username and password to gain access to the platform's administrative functions.

Login:
Admin can log in using their registered username and password.

After Login:
1. Add Product:

o Admin can add new products to the platform, including details such as product name, category, price, stock quantity, and description.

2. View Product:

o Admin can view and update product details such as the name, price, stock levels, or description to keep the inventory accurate.

3. Delete Product:

o Admin can remove outdated or discontinued products from the system.

4. Add Order:

o Admin can manually add orders for customers, including details such as product selection, quantity, and payment status.

5. Update Order:

o Admin can modify existing order details, such as updating the shipping address, product quantity, or order status.

6. Remove Admin:

o Admin details can be deleted if necessary, ensuring proper user management and security compliance.

This structure ensures efficient management of the platform and streamlines both customer and administrator interactions.

4.Implementation: 4.1
Design: Home Page:
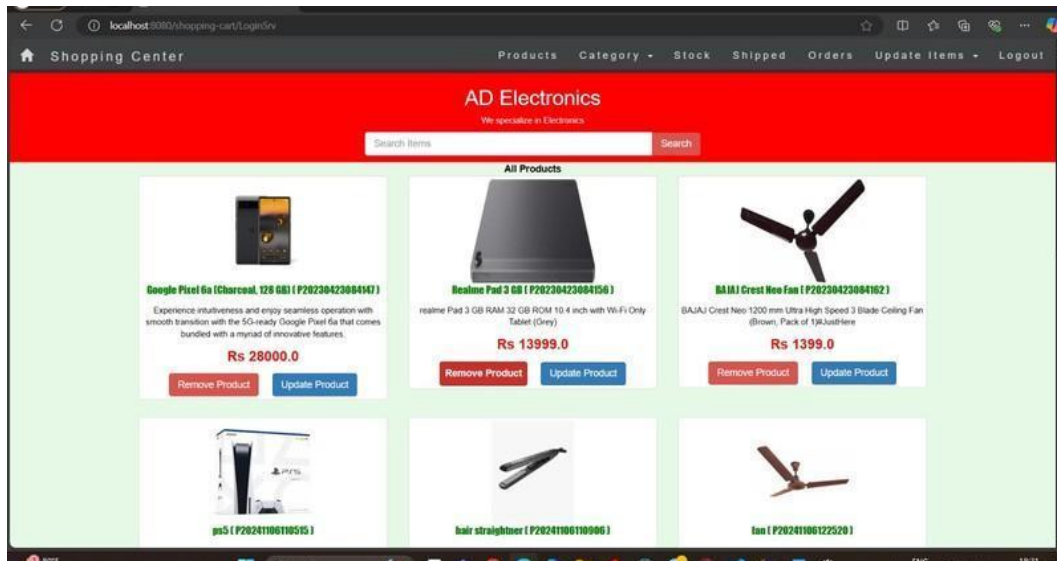
Login page:
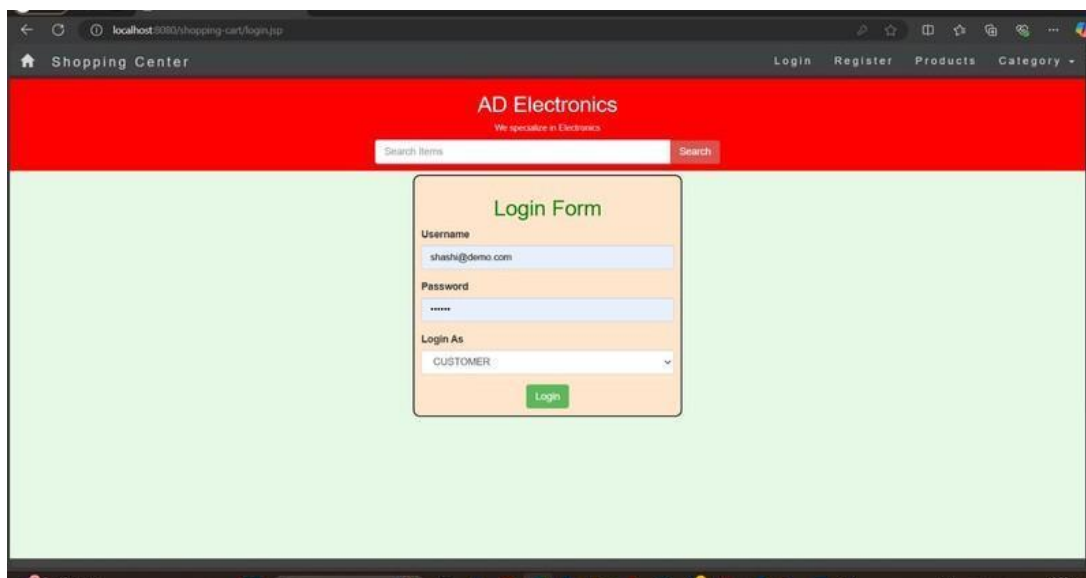
Cart Items

Credit Card Payment


Order Details & Status

4.2 Database Design:

The data in the Online Shopping Platform system is stored and retrieved from a relational database. Designing the database is an essential part of the system design process. Data elements and data structures identified during the analysis stage are organized and structured to design the storage and retrieval system effectively. A database is a collection of interrelated data stored with minimal redundancy to serve many users efficiently. The objective is to provide fast, reliable, and flexible access to data for users while keeping storage costs low. Relationships between data entities are carefully established, and unnecessary data is removed. Normalization is applied to achieve internal consistency of data, minimize redundancy, and ensure maximum stability. This process reduces the storage required, minimizes the chances of data inconsistencies, and optimizes the performance of data updates and retrieval. For the Online Shopping Platform, the MySQL database has been chosen to develop and manage relevant databases. This ensures efficient handling of critical data, such as product details, customer information, order transactions, and inventory levels. The database design als o supports scalability and reliability to accommodate the platform's growth and evolving requirements.

Online Shopping Platform MySQL tables :
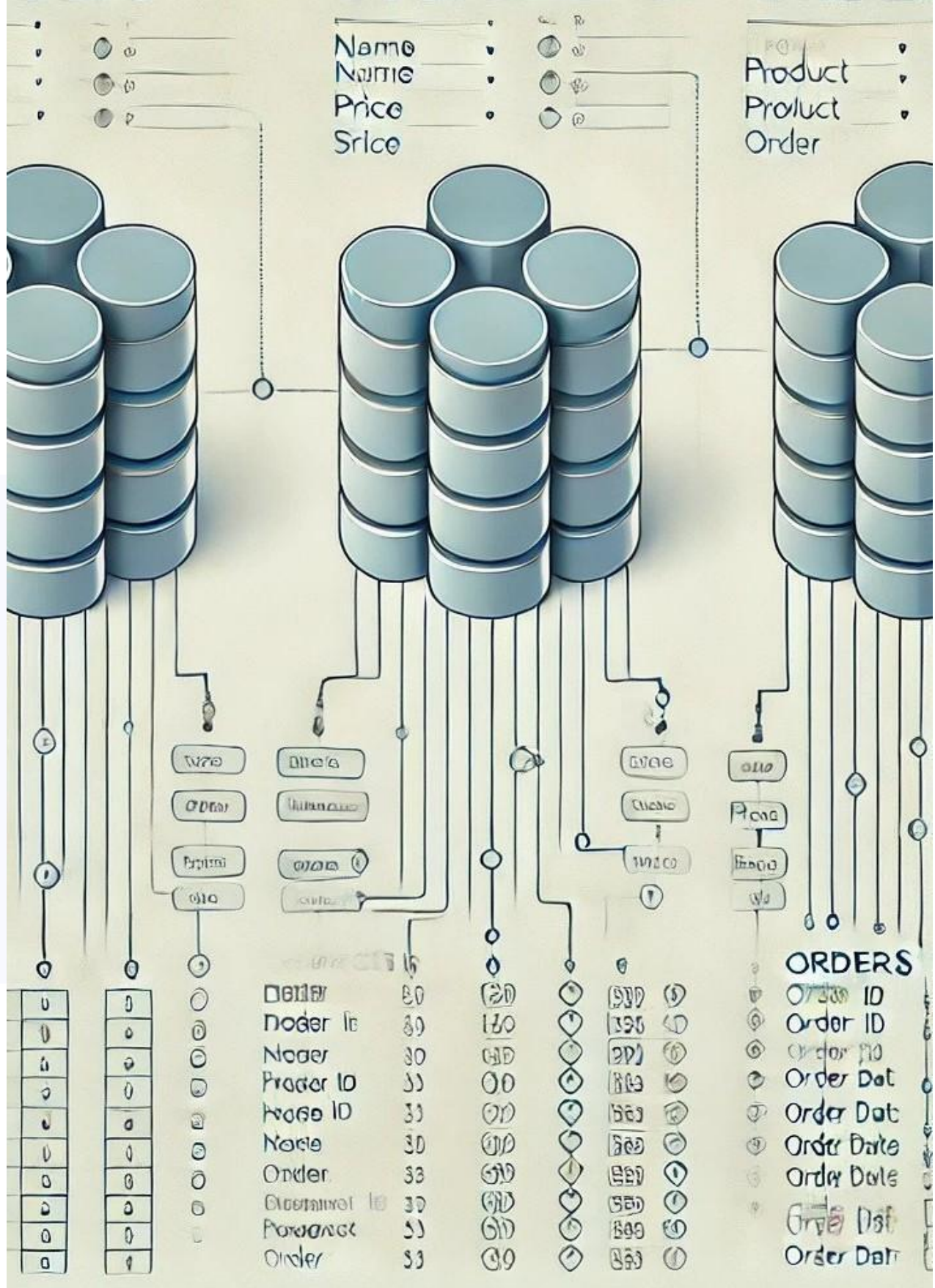
```java
Code:
package com.DA.service.impl;

import java.sql.Connection;

import java.sql.PreparedStatement; import
java.sql.ResultSet;
import java.sql.SQLException;

import com.shashi.beans.UserBean;
import com.shashi.constants.IUserConstants;
import com.shashi.service.UserService; import
com.shashi.utility.DBUtil;
import com.shashi.utility.MailMessage;
public class UserServiceImpl implements UserService {
@Override
public String registerUser(String userName, Long mobileNo, String emailId, String address, int pinCode, String
password) {

UserBean user = new UserBean(userName, mobileNo, emailId, address, pinCode, password); String

status = registerUser(user);

return status;
}

@Override
public String registerUser(UserBean user) { String

status = "User Registration Failed!"; boolean isRegtd

= isRegistered(user.getEmail());

if (isRegtd) {
status = "Email Id Already Registered!";
return status;
}
Connection conn = DBUtil.provideConnection();
PreparedStatement ps = null;
```

```
if (conn != null) {
```

```
if (conn != null) {
```

```java
try
{
ps = conn.prepareStatement("insert into " + IUserConstants.TABLE_USER + " values(?,?,?,?,?,?)"); ps.setString(1,
user.getEmail());
ps.setString(2, user.getName());
ps.setLong(3, user.getMobile());
ps.setString(4, user.getAddress());
ps.setInt(5, user.getPinCode());
ps.setString(6, user.getPassword());
int k = ps.executeUpdate(); if


(k > 0) {


status = "User Registered Successfully!";
MailMessage.registrationSuccess(user.getEmail(), user.getName().split(" ")[0]);
}


} catch (SQLException e) {
status = "Error: " + e.getMessage();
e.printStackTrace();
}


DBUtil.closeConnection(ps); DBUtil.closeConnection(ps);


return status;
}


@Override
public boolean isRegistered(String emailId) {
boolean flag = false;


Connection con =


DBUtil.provideConnection();


PreparedStatement ps = null;
ResultSet rs = null;


try {
ps = con.prepareStatement("select * from user where email=?");
```

```java
if (rs.next())
flag = true;

} catch (SQLException e) {
// TODO Auto-generated catch block e.printStackTrace();
}

DBUtil.closeConnection(con); DBUtil.closeConnection(ps);
DBUtil.closeConnection(rs);

return flag;
}

@Override
public String isValidCredential(String emailId, String password) {
String status = "Login Denied! Incorrect Username or Password";
Connection con = DBUtil.provideConnection();

PreparedStatement ps = null;

ResultSet rs = null;

try

{
ps = con.prepareStatement("select * from user where email=? and password=?");

ps.setString(1, emailId);
ps.setString(2, password);
rs = ps.executeQuery(); if

(rs.next())

status = "valid";
} catch (SQLException e) {
status = "Error: " + e.getMessage();
e.printStackTrace();
}
```

```
DBUtil.closeConnection(con);
return status
```

```java
@Override public UserBean getUserDetails(String emailId, String password)
{

    User Bean user = null;
    Connection con = DBUtil.provideConnection();
    PreparedStatement ps = null;
    ResultSet rs = null;

    try {
    ps = con.prepareStatement("select * from user where email=? and password=?");
    ps.setString(1, emailId);
    ps.setString(2, password); rs
    = ps.executeQuery();

    if (rs.next()) {
    user = new UserBean();
    user.setName(rs.getString("name"));
    user.setMobile(rs.getLong("mobile"));
    user.setEmail(rs.getString("email"));
    user.setAddress(rs.getString("address"));
    user.setPinCode(rs.getInt("pincode"));
    user.setPassword(rs.getString("password"));

    return user;
    }

    } catch (SQLException e)

    {
    e.printStackTrace();
    }
    DBUtil.closeConnection(con);
    DBUtil.closeConnection(ps);
    DBUtil.closeConnection(rs);

    return user;
    }

    @Override
    public String getFName(String emailId) { String
    fname = "";
```

```java
try {

ps = con.prepareStatement("select name from user where email=?");

ps.setString(1, emailId);


rs = ps.executeQuery();


if (rs.next()) {

fname = rs.getString(1);


fname = fname.split(" ")[0];


}


} catch (SQLException e) {


e.printStackTrace();

}


return fname;

}


@Override

public String getUserAddr(String userId) { String

userAddr = "";


Connection con = DBUtil.provideConnection();

PreparedStatement ps = null;

ResultSet rs = null;


try {

ps = con.prepareStatement("select address from user where email=?");


ps.setString(1, userId);


rs = ps.executeQuery();


if (rs.next())

userAddr = rs.getString(1);


} catch (SQLException e) {
```

```
            e.printStackTrace();
        }
```

Conclusion:

The Online Shopping Platform project, developed under experienced guidance, exemplifies a meticulous approach to design and implementation. With a focus on user-friendly functionalities such as product management, order tracking, and user registration, the system ensures a seamless shopping experience. Robust security measures, particularly in the "Remove Admin" and "Order Management" modules, underscore a commitment to data integrity and system protection. This project stands as a comprehensive solution, addressing current e-commerce needs while being adaptable for future enhancements and scalability.

Reference links: [1]

https://www.javatpoint.com/java-awt [2]
https://www.javatpoint.com/java-swing