Graphic Era Hill University, Haldwani

MCA Project Report

For

Online Shopping Website Using Java

Submitted to Graphic Era Hill University, Haldwani for the partial fulfillment of the requirement for the Award of degree for

MASTER OF COMPUTER APPLICATIONS

Submitted by :-

Kamlesh Pandey

Student id - 20711166

Under the Guidance of :-

Mrs. Sujata Negi

Faculty of GEHU, Haldwani

DECLARATION

I hereby declare that the work which is being present in this project report **"Online Shopping Website using Java",** in partial fulfilment of the requirement for the Award of the degree of **MASTER OF COMPUTER APPLICATION**, submitted at **GRAPHIC ERA HILL UNIVERSITY, HALDWANI** is an authentic work done by me during period from 1st April 2022 to 1st June 2022.

Project Guide:

Mrs. Sujata Negi

Faculty

Faculty of GEHU

Signature of the Student:

Kamlesh Pandey

Roll No 2098006

Graphic Era Hill University, Haldwani

BONAFIDE CERTIFICATE

Certified that this project report **ONLINE SHOPPING WEBSITE USING JAVA** is the bonafide work of **Kamlesh Pandey** who carried out the project work under my supervision.

Name – Mrs Sujata Negi

HEAD OF THE DEPARTMENT SUPERVISOR

Computer Application

Graphic Era Hill University

Haldwani

ACKNOWLEDGEMENTS

I would like to extend our thanks and appreciation to all those who have assisted us either directly or indirectly and participated in the success of this project. I would like to thank my guide Mrs. Sujata Negi for his constant support in the making of the project. As a part of University Curriculum, a 4th semester project is a paramount importance to an MCA student's curriculum and being our native effort into this project undertook by us, we faced a lot of impediments on our way to the completion of this project but constant guidance and able support of concerned software engineer members lend us a great help in successful completion of the project. I am thankful to all staff members of Graphic Era Hill University who helped me whenever required, during my project. Even though I expressed my gratitude to every person who helped me in reaching this stage, there might be a few, who'd been left out, who helped me without my knowledge. I would like to thank all of them. Last but not least, to all my friends and fellow students for giving me suggestions and helping us in debugging the code errors and above all the faculty of my Department of Computer Science Graphic Era Hill University who have always provided their guidance, support and well wishes.

Kamlesh Pandey

ABSTRACT

The Online Shopping is a web based application intended for online retailers. The main objective of this application is to make it interactive and its ease of use. It would make searching, viewing and selection of a product easier. It contains a sophisticated search engine for user's to search for products specific to their needs. The search engine provides an easy and convenient way to search for products where a user can Search for a product interactively and the search engine would refine the products available based on the user's input. The user can then view the complete specification of each product. They can also view the product reviews and also write their own reviews. The application also provides a drag and drop feature so that a user can add a product to the shopping cart by dragging the item in to the shopping cart. The main emphasis lies in providing a user-friendly search engine for effectively showing the desired results and its drag and drop behaviour.

TABLE OF CONTENT

ONLINE SHOPPING

1. INTRODUCTION

Online Shopping System (Online Shopping) is the simple shopping solution. It's a full-featured website and shopping cart system that bends over backwards to give you the flexibility you need to run your online store. This project is a web based shopping system for an existing shop. The project objective is to deliver the online shopping application. The basic concept of the application is to allow the customer to shop virtually using the Internet and allow customers to buy the items and articles of their desire from the store. The information pertaining to the products are store on an RDBMS at the server side(store). The server process the customers and the items are shipped to the address submitted by them. The details of the items are brought forward from the database for the customer view based on the selection through the menu and database of all products are updated at the end of each transaction. Online shopping is the process whereby consumers directly buy goods or services from a seller in real-time, without an intermediary service, over the Internet. It is a form of electronic commerce. The process is called business-to-consumer (B2C) online shopping. The Project entitled "ONLINE SHOPPING" is a web-based application Software developed in Java LANGUAGE using JSP as front end and MYSQL as backend with the help of hibernate. The main aim of "Online Shopping" is to improve the services of Customers and vendors. It maintains the details of customer payments, product receipts, addition of new customers, products and also updating, deletion for the same. The main aim of "Online Shopping" is to improve the services of Customers and vendors It also stores the details of invoices generated by customer and payments made by them with all Payments details like credit card. The primary features of the project entitled "ONLINE SHOPPING" are high accuracy, design flexibility and easy availability. And also it uses database tables Representing entities and relationships between entities. The application was designed into two modules first for the customers (USER) who wish to buy the articles. Second is for the storekeepers (ADMIN) who maintains and updates the information pertaining to the articles and those of the customers.

1.1 PROJECT OVERVIEW:

The central concept of the application is to allow the customer to shop virtually using the Internet and allow customers to buy the items and articles of their desire from the store. The information pertaining to the products are stores on an RDBMS at the server side (store). The Server process the customers and the items are shipped to the address submitted by them. The application was designed into two modules first is for the customers who wish to buy the articles. Second is for the storekeepers who maintains and updates the information pertaining to the articles and those of the customers. The end user of this product is a departmental store where the application is hosted on the web and the administrator maintains the database. The application which is deployed at the customer database, the details of the items are brought forward from the database for the customer view based on the selection through the menu and the database of all the products are updated at the end of each transaction. Data entry into the application can be done through various screens designed for various levels of users. Once the authorized personnel feed the relevant data into the system, several reports could be generated as per the security.

1.2 PROJECT SCOPE:

This system can be implemented to any shop in the locality or to multinational branded shops having retail outlet chains. The system recommends a facility to accept the orders 24*7 and a home delivery system which can make customers happy.

If shops are providing an online portal where their customers can enjoy easy shopping from anywhere, the shops won't be losing any more customers to the trending online shops such as flipcart or ebay. Since the application is available in the Smartphone it is easily accessible and always available.

2.System Requirement Analysis

2.1 Information Gathering

As the goal of the application is ease of use and to provide an interactive interface, extensive research has been done to gain an insight into the needs and behaviours of various users. The working of the application is made convenient and easy to use for the end user.. Users can be classified into two types based on their knowledge of the products that suit their needs. They can be classified as users who know about the product that would satisfy their needs and users who have to figure out the product that would satisfy their needs. Users who know about the product should be able to find the product easily with the click of a button. Such users can search for the product by using the product name as the search term. Users who have to figure out the product that would satisfy their needs could use a search term to find a list of products and then should be able to filter the results based on various parameters like product type, manufacturer, price range, platform supported etc. The users should be able to view the complete specification of the product and various images at different Zoom levels. The user should be able to read the customer reviews for the product and the ratings provided. They should be able to write their own reviews. They should be able to print out the specifications for a product or email the product page to a friends etc. To increase the ease of use the user should be able to add a product to the shopping cart by dragging a product and dropping it in the shopping cart. A user should able to edit the contents of a shopping cart. They should be able to update the quantities of the products added to the cart and remove the products from the cart. The user should be able to remove the product from the shopping cart by dragging the product and dropping it outside the cart. The application can be made interactive by pop up messages when a product has been dropped in to the shopping cart or out of the shopping cart. The user can be notified 4 if the cursor enters a drop area and the object that could be dropped. Also users are impatient making it important to load pages soon. Other than this, I did a lot of research on various other methods of building this application which and was able to incorporate a few stronger features into the application.

2.2 System Feasibility

The system feasibility can be divided into the following sections:

Technical Feasibility Study

The Project can be developed simply by using two platforms i.e. JAVA and Tomcat as frontend and MySQL Server as back-end. All the functions of a Online Shopping System can be implemented in the new system. Hence it is technically feasible.

Economic Feasibility study

This feasibility study present tangible and intangible benefits from the project by comparing the development and operational cost. The technique of cost benefit analysis is often used as a basis for assessing economic feasibility. This system needs some more initial investment than the existing system, but it can be justifiable that it will improve quality of service.

Thus feasibility study should centre along the following points:

- Improvement resulting over the existing method in terms of accuracy, timeliness.
- Cost comparison
- Estimate on the life expectancy of the hardware.
- Overall objective Our project is economically feasible.

It does not require much cost to be involved in the overall process. The overall objective is in easing out the recruitment processes.

Time Feasibility Study

it has been more probable (as per the requirements and functions specifications of the system) that the project can be completed within the given timeframe, it is considered that the undertaking this project is feasible in the context of time.

Operational Feasibility Study

If the system meets the requirements of the customers and the administrator we can say that the system is operationally feasible. The proposed system will be beneficial only if it can be turned into a system which will meet the requirements of the store when it is developed and installed and there is sufficient support from the users. The proposed system will provide a better market for different dealers.

Social Feasibility Study

This analysis involves how it will work when it is installed and the assessment of political and managerial environment in which it is implemented. People are inherently resistant to change and computers have been known to facilitate change. The new proposed system is very much useful to the users and Social feasibility is determination of whether a proposed project will be acceptable to people or not, So this project is totally Social and Feasible.

3.SYSTEM DESIGN

System design is the solution for the creation of a new system. This phase focuses on the detailed implementation of the feasible system. It emphasis on translating design. Specifications to performance specification. System design has two phases of development

- Logical design
- Physical design

During logical design phase the analyst describes inputs (sources), output s(destinations), databases (data sores) and procedures (data flows) all in a format that meets the user requirements. The analyst also specifies the needs of the user at a level that virtually determines the information flow in and out of the system and the data resources. Here the logical design is done through data flow diagrams and database design. The physical design is followed by physical design or coding. Physical design produces the working system by defining the design specifications, which specify exactly what the candidate system must do. The programmers write the necessary programs that accept input from the user, perform necessary processing on accepted data and produce the required report on a hard copy or display it on the screen.

3.1 INPUT AND OUTPUT DESIGN

INPUT DESIGN:

Input design is the link that ties the information system into the world of its users. The input design involves determining the inputs, validating the data, minimizing the data entry and provides a multi-user facility. Inaccurate inputs are the most common cause of errors in data processing. Errors entered by the data entry operators can be controlled by input design. The user-originated inputs are converted to a computer based format in the input design. Input data are collected and organized into groups of similar data. Once identified, the appropriate input media are selected for processing. All the input data are validated and if any data violates any conditions, the user is warned by a message. If the data satisfies

all the conditions, it is transferred to the appropriate tables in the database. In this project the student details are to be entered at the time of registration. A page is designed for this purpose which is user friendly and easy to use. The design is done such that users get appropriate messages when exceptions occur.

OUTPUT DESIGN:

Computer output is the most important and direct source of information to the user. Output design is a very important phase since the output needs to be in an efficient manner. Efficient and intelligible output design improves the system relationship with the user and helps in decision making. Allowing the user to view the sample screen is important because the user is the ultimate judge of the quality of output. The output module of this system is the selected notifications.

DATABASE

DATABASE DESIGN:

Databases are the storehouses of data used in the software systems. The data is stored in tables inside the database. Several tables are created for the manipulation of the data for the system. Two essential settings for a database are

- Primary Key - the field that is unique for all the record occurrences.
- Foreign Key -the field used to set relation between tables. Normalization is a technique to avoid redundancy in the tables.

SYSTEM TOOLS

The various system tools that have been used in developing both the front end and the back end of the project are being discussed in this chapter.

FRONT END:

JSP, HTML, CSS and JAVA SCRIPT are utilized to implement the frontend.

Java Server Pages (JSP)

Different pages in the applications are designed using jsp. A Java Server Pages component is a type of Java servlet that is designed to fulfil the role of a user interface for a Java web application. Web developers write JSPs as text files that combine HTML or XHTML code, XML elements, and embedded JSP actions and commands. Using JSP, one can collect input from users through web page.

HTML (Hyper Text Markup Language)

HTML is a syntax used to format a text document on the web.

CSS (Cascading Style Sheets)

CSS is a style sheet language used for describing the look and formatting of a document written in a markup language. Java Script JS is a dynamic computer programming language. It is most commonly used as part of web browsers, whose implementations allow client-side.

Java Script

JS is a dynamic computer programming language. It is most commonly used as part of web browsers, whose implementations allow client-side scripts to interact with the user, control the browser, communicate asynchronously, and alter the document content that is displayed. Java Script is used to create pop up windows displaying different alerts in the system like “User registered successfully”, ”Product added to cart” etc.

BACK END

Servlet

- Servlet is a technology which is used to create a web application.
- Servlet is an API that provides many interfaces and classes including documentation.
- Servlet is an interface that must be implemented for creating any Servlet.
- Servlet is a class that extends the capabilities of the servers and responds to the incoming requests. It can respond to any requests.
- Servlet is a web component that is deployed on the server to create a dynamic web page.

Hibernate

Hibernate is a Java framework that simplifies the development of Java application to interact with the database. It is an open source, lightweight, ORM (Object Relational Mapping) tool. Hibernate implements the specifications of JPA (Java Persistence API) for data persistence.

MySQL

MySQL is the world's second most widely used open-source relational database management system (RDBMS). The SQL phrase stands for Structured Query Language. An application software called Navicat was used to design the tables in MySQL.

List of Tables

User Table

```
mysql> desc user;
+---------------+--------------+------+-----+---------+----------------+
| Field         | Type         | Null | Key | Default | Extra          |
+---------------+--------------+------+-----+---------+----------------+
| user_id       | int          | NO   | PRI | NULL    | auto_increment |
| user_address  | varchar(1500) | YES  |     | NULL    |                |
| user_email    | varchar(100) | YES  |     | NULL    |                |
| user_name     | varchar(100) | YES  |     | NULL    |                |
| user_password | varchar(100) | YES  |     | NULL    |                |
| user_phone    | varchar(12)  | YES  |     | NULL    |                |
| user_pic      | varchar(1500) | YES  |     | NULL    |                |
| user_type     | varchar(255) | YES  |     | NULL    |                |
+---------------+--------------+------+-----+---------+----------------+
8 rows in set (0.01 sec)
```

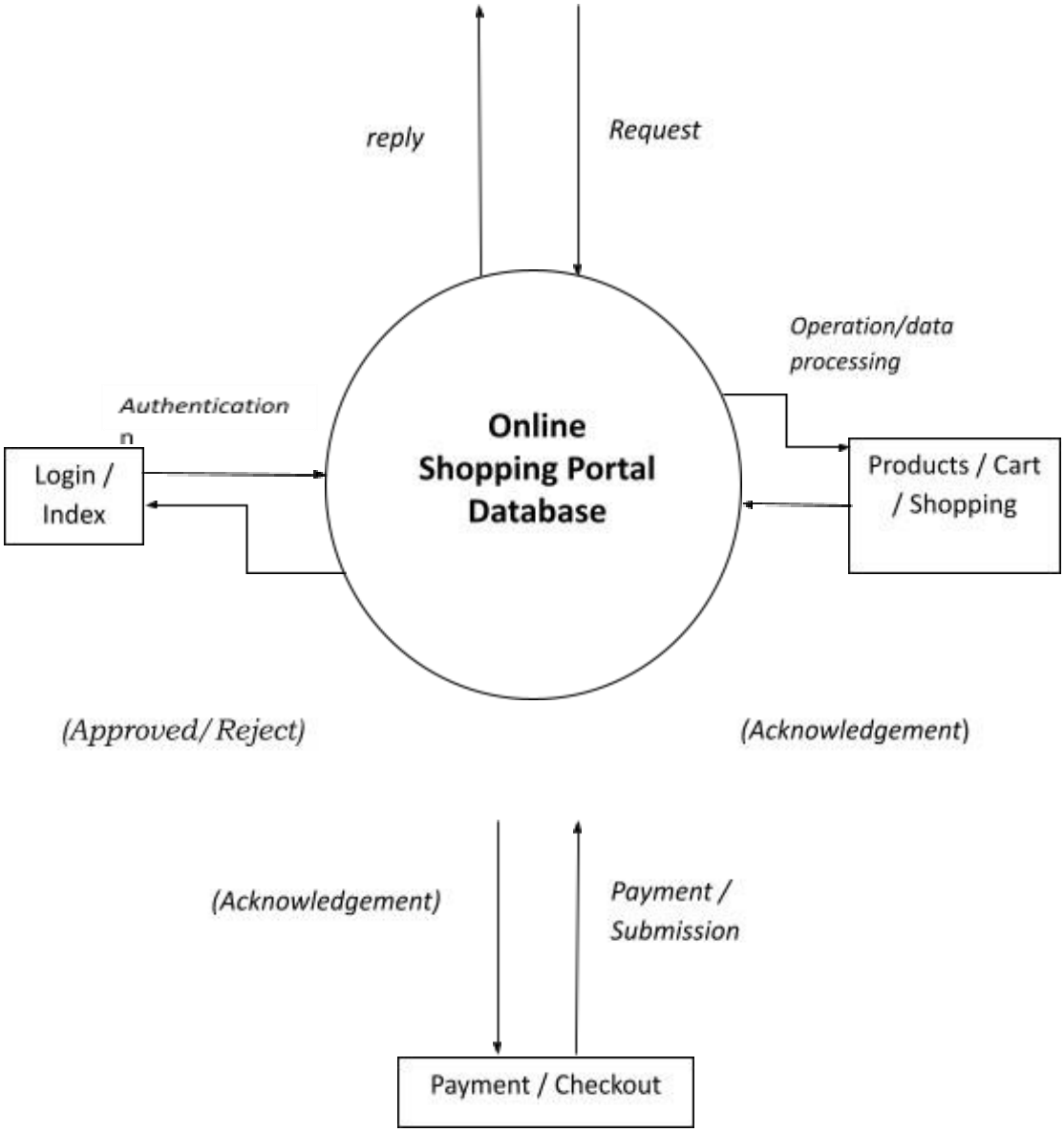
Product Table

```
mysql> desc product;
+--------------------+--------------+------+-----+---------+----------------+
| Field              | Type         | Null | Key | Default | Extra          |
+--------------------+--------------+------+-----+---------+----------------+
| pId                | int          | NO   | PRI | NULL    | auto_increment |
| pDesc              | varchar(3000) | YES  |     | NULL    |                |
| pDiscount          | int          | NO   |     | NULL    |                |
| pName              | varchar(255) | YES  |     | NULL    |                |
| pPhoto             | varchar(255) | YES  |     | NULL    |                |
| pPrice             | int          | NO   |     | NULL    |                |
| pQuantity          | int          | NO   |     | NULL    |                |
| category_categoryId | int          | YES  | MUL | NULL    |                |
+--------------------+--------------+------+-----+---------+----------------+
8 rows in set (0.00 sec)
```

Category Table

```
mysql> desc category;
+--------------------+-------------+------+-----+---------+----------------+
| Field              | Type        | Null | Key | Default | Extra          |
+--------------------+-------------+------+-----+---------+----------------+
| categoryId         | int         | NO   | PRI | NULL    | auto_increment |
| categoryDescription | varchar(255) | YES  |     | NULL    |                |
| categoryTitle      | varchar(255) | YES  |     | NULL    |                |
+--------------------+-------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)
```

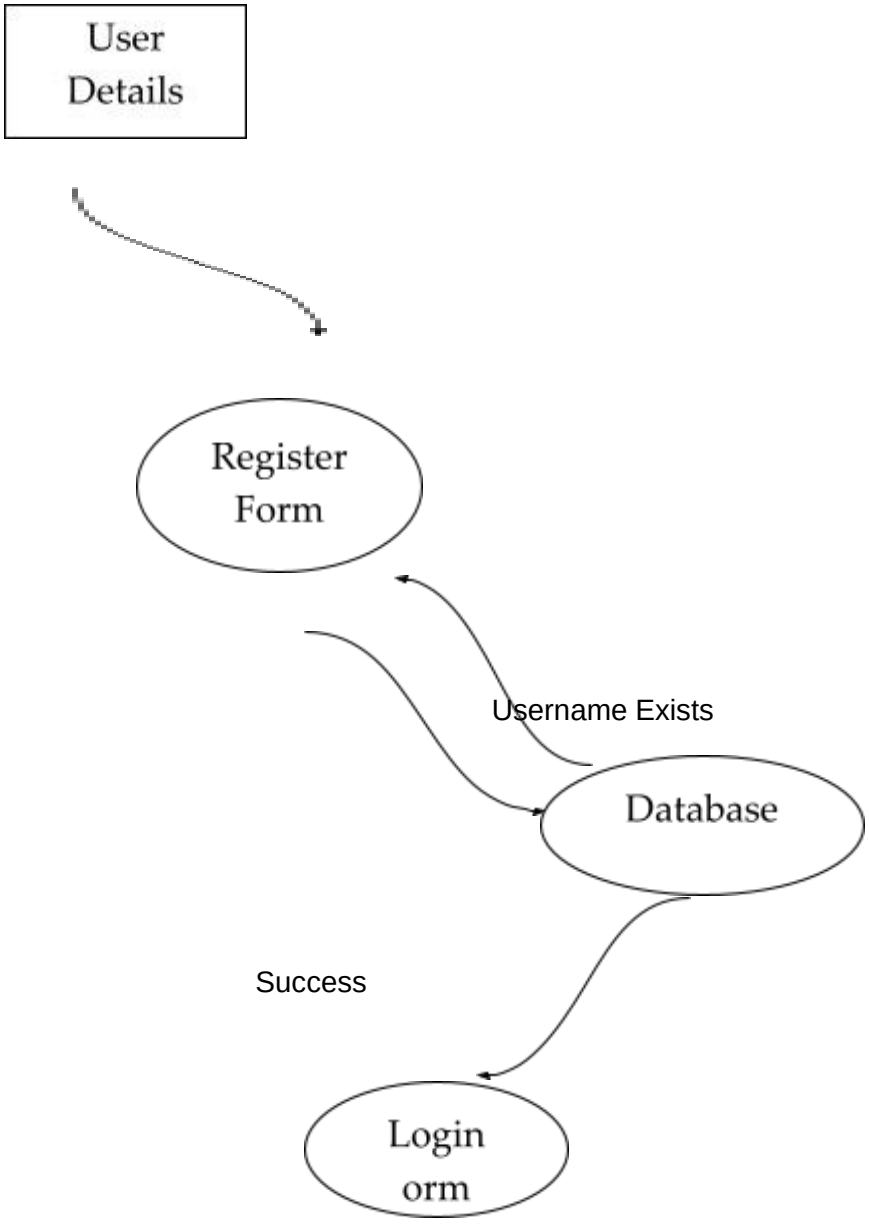
3.2 FLOWCHART



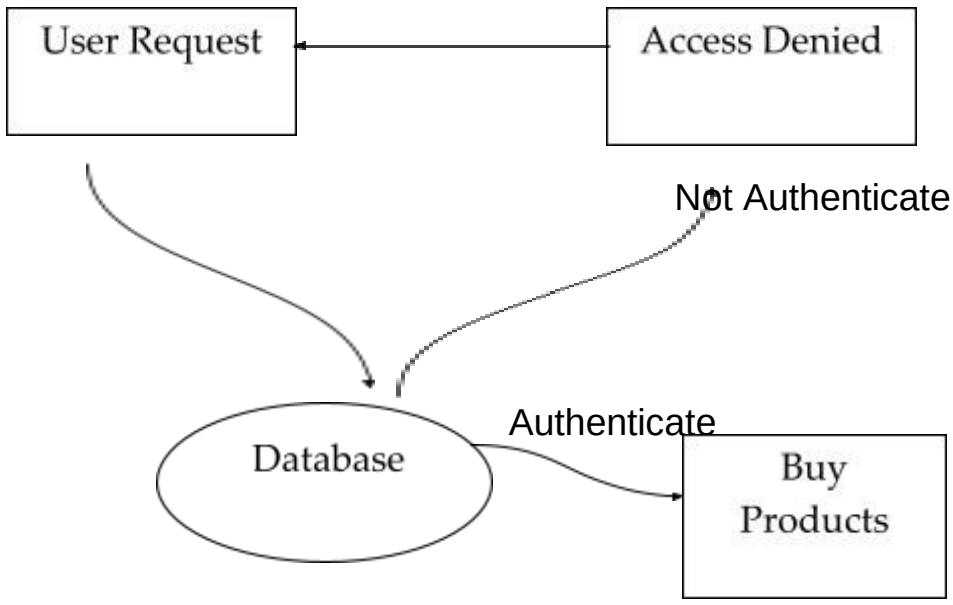
3.3 DFD

(0 level DFD)

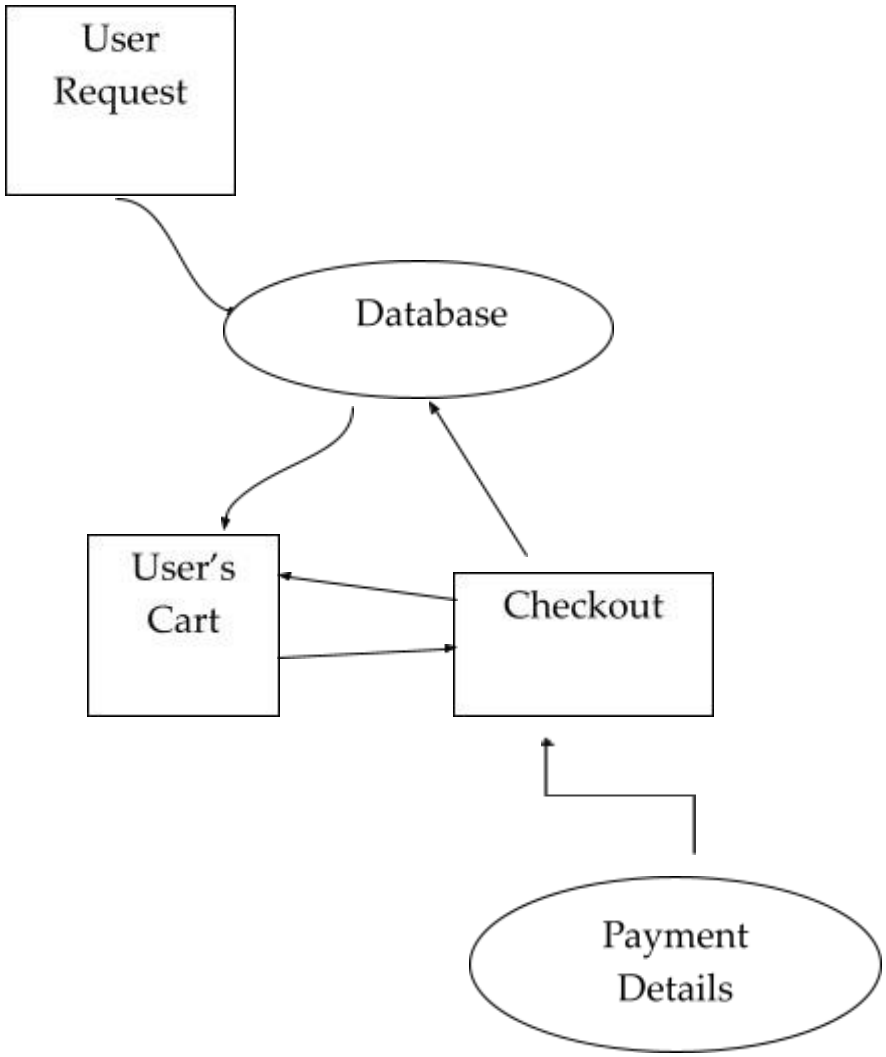
For Registration



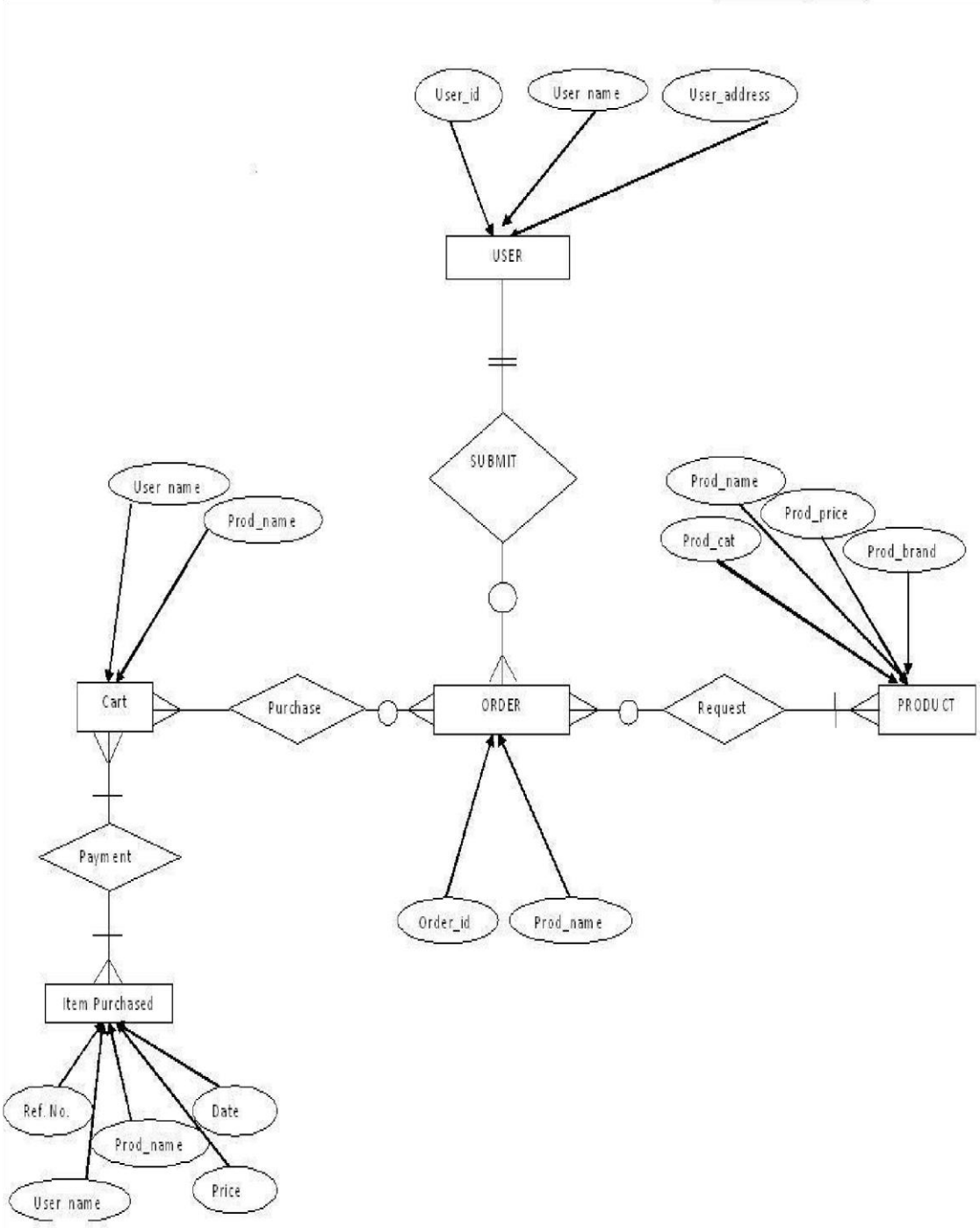
For Login


User Request
Access Denied
Not Authenticate
Database
Authenticate
Buy
Products

For Buying Product:



3.4 E-R Diagram



4 Hardware And Software Requirement

Hardware Requirement

- PC with 250 GB or more Hard disk.
- PC with 2 GB RAM.
- PC with Pentium or higher.

Software Requirement

- Operating System - Windows XP / Windows, Android
- Automation Tool – Apache maven
- Server - Tomcat

5. Limitation of the Online shopping website

First, the physical and photos of the gap is too big

Net purchase only is seen pictures of goods, to really get your hands, you will feel and the objects are not the same as. This is not in the mall to buy the rest assured.

Second, do not try

online shopping is just to see pictures and articles on the simple introduction, like clothes or shoes and the like, you can directly see the suitable for you, and if in the mall to buy, you can try it on, his body, immediately buy, not so much to consider, however, online shopping is more trouble.

Third, online payment security

Can be peeping, stolen passwords. Online shopping is most worried about is that he needs to use a bank account, some friends of the computer there is pilfer date trojan, can cause some serious account loss occurs, so everyone in the shopping time try not to choose the Internet cafes and other public places, your computer must also ensure the antivirus software can be installed network transaction.

Fourth, good faith question

Is the owner 's credit, if encountered poor service quality of the owner, asked a few questions appear impatient. Also in the online shopping appearance deceived happens.

Fifth, the speed of delivery problem

Online purchases, but also after the distribution of the link, the fast one or two days, would slow to a week or more, sometimes, there are still some problems in the process of distribution, and, if the goods are not satisfied with, and through distribution link, change the items, so much trouble; while in the mall, see you want, just get, if not satisfied, instead of directly.

Sixth, return the problem of inconvenient

Although the reality of shopping return requires a complex procedure, and even on the product to be protected, but the net return is relatively more difficult. Even proposed various unreasonable request refused to return and buck.

6. Appendices

code

Index.jsp

```
<%@page import="com.website.onlineshopping.helper.Helper"%>
<%@page import="com.website.onlineshopping.entities.Category"%>
<%@page import="com.website.onlineshopping.dao.CategoryDao"%>
<%@page import="java.util.List"%>
<%@page import="com.website.onlineshopping.entities.Product"%>
<%@page import="com.website.onlineshopping.dao.ProductDao"%>
<%@page import="com.website.onlineshopping.helper.FactoryProvider"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Online Shopping</title>
        <%@include file="components/common_css_js.jsp" %>
    </head>
    <body>
```

```
<%@include file="components/navbar.jsp" %>



<div class="container-fluid">


<div class="row mt-2 ">


   <%
      String cat =request.getParameter("category");



      ProductDao dao = new ProductDao(FactoryProvider.getFactory());
      List<Product> list = null;
      if(cat==null || cat.trim().equals("all"))
      {


      list = dao.getAllProducts();


      }else {
         int cid = Integer.parseInt(cat.trim());
         list = dao.getAllProductsById(cid);
      }
```

```
CategoryDao cdao = new CategoryDao(FactoryProvider.getFactory());

List<Category> clist = cdao.getCategories();




%>


<!-- show categories-->

<div class="col-md-2">


<div class="list-group mt-4">


<a href="index.jsp?category=all" class="list-group-item list-group-item-action
active">

All Products

</a>



<%


for(Category c : clist){


%>
```

```
            <a href="index.jsp?category=<%= c.getCategoryId() %> " class="list-group-item
list-group-item-action"><%=c.getCategoryTitle() %></a>

            <% }

            if(list.size()==0){

                    out.println("<h3>No item in this category</h3>");

                     }
%>
          </div>
        </div>

        <!--show products-->
        <div class="col-md-10">

          <div class="row  mt-4 ">

            <div class="col-md-12 ">

              <div class="card-columns product-card ">
```

```
<%
for(Product p:list){
%>

<div class="card">

<div class=" text-center">
<img class="card-img-top mt-2" style="max-height: 200px; max-width: 200px; width: auto; " src="img/products/<%=p.getpPhoto() %>" alt="Card image cap"/>
</div>

<div class="card-body">

<h5 class="card-title"> <%= p.getpName() %></h5>
<p class="card-text">
<a href="#" style="color:brown"><%= Helper.get10Words(p.getpDesc()) %></a>
</p>

</div>

<div class="card-footer text-center">
```

```
                        <h6> Product Id : <%= p.getpId() %></h6>

                        <button class="btn custom-bg text-white"
onclick="add_to_cart(<%=p.getpId() %> , '<%=p.getpName() %>', <%=p.getpPrice()
%>)">Add to Cart</button>

                        <button class="btn btn-outline-success">
&#8377;<%=p.getPriceAfterApplyingDiscount() %>/- <span class="text-secondary discount-
label"> <strike>&#8377;<%= p.getpPrice() %></strike>  <%= p.getpDiscount() %>%
off</span></button>



                    </div>

                </div>

                    <% } %>

                </div>

            </div>

        </div>

    </div>


    </div>


    <div>

        <%@include file="components/footer.jsp" %>
```

```
        </div>

       <%@include file="components/common_modals.jsp" %>

    </body>

</html>
```

- **admin.jsp**

```
<%@page import="com.website.onlineshopping.helper.Helper"%>

<%@page import="java.util.Map"%>



<%@page import="java.util.List"%>

<%@page import="com.website.onlineshopping.entities.Category"%>

<%@page import="com.website.onlineshopping.dao.CategoryDao"%>

<%@page import="com.website.onlineshopping.helper.FactoryProvider"%>

<%@page import="com.website.onlineshopping.entities.User"%>
```

```
<%



   User user = (User) session.getAttribute("current-user");
   if(user==null){
   session.setAttribute("message","You are not logged in !!!");
   response.sendRedirect("login.jsp");
   return;
   }else
   {
      if(user.getUserType().equals("normal")){
       session.setAttribute("message","You are not admin !!!");
        response.sendRedirect("login.jsp");
          return;
   }
   }
%>
<%
            CategoryDao cdao = new CategoryDao(FactoryProvider.getFactory());
            List<Category> list = cdao.getCategories();
```

```
            Map<String,Long> m = Helper.getCounts(FactoryProvider.getFactory());


        %>


<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Admin Panel</title>
    <%@include file="components/common_css_js.jsp" %>
  </head>
  <body>
    <%@include file="components/navbar.jsp" %>
    <div class="container admin">


      <div class="container-fluid mt-3">
        <%@include file="components/message.jsp" %>


      </div>
```

```
<div class="row mt-3">
  <div class="col-md-4">
    <!--first call-->
    <div class="card" >
    <div class="card-body text-center">
      <div class="container">
        <img style="max-width: 125px;" class="img-fluid rounded-circle"
src="img/team.png" alt="">
      </div>
      <h1><%= m.get("userCount")%></h1>
      <h2 class="text-uppercase text-muted">Users</h2>
    </div>
  </div>
  </div>

  <div class="col-md-4">
    <!--second call-->
    <div class="card" >
    <div class="card-body text-center">
      <div class="container">
```

```
                <img style="max-width: 125px;" class="img-fluid rounded-circle"
src="img/list.png" alt="">
                </div>
                <h1><%=list.size() %></h1>
                <h2 class="text-uppercase text-muted">Categories</h2>
            </div>
        </div>
        </div>

        <div class="col-md-4">
            <!--third call-->
            <div class="card" >
            <div class="card-body text-center">
                <div class="container">
                <img style="max-width: 125px;" class="img-fluid rounded-circle"
src="img/list.png" alt="">
                </div>
                <h1><%= m.get("productCount")%></h1>
                <h2 class="text-uppercase text-muted">Products</h2>
            </div>
        </div>
        </div>
```

```
</div>


<!--second row-->


<div class ="row mt-3">
  <div class="col-md-4">
    <div class="card" data-toggle="modal" data-target="#add-category-modal" >
    <div class="card-body text-center">
      <div class="container">
        <img style="max-width: 125px;" class="img-fluid rounded-circle" src="img/calculator.png" alt="">
      </div>
      <p class="mt-2">click here for add new category</p>
      <h2 class="text-uppercase text-muted">Add Category</h2>          </div>
    </div>
  </div>


  <div class="col-md-4">
    <div class="card" data-toggle="modal" data-target="#add-product-modal">
    <div class="card-body text-center">
      <div class="container">
```

```
                <img style="max-width: 125px;" class="img-fluid rounded-circle"
src="img/plus.png" alt="">
                </div>
                 <p class="mt-2">click here for add new product</p>
                <h2 class="text-uppercase text-muted">Add Products</h2>
              </div>
              </div>
            </div>




            <div class="col-md-4">
              <div class="card" data-toggle="modal" data-target="#delete-product-modal">
              <div class="card-body text-center">
                <div class="container">
                <img style="max-width: 125px;" class="img-fluid rounded-circle"
src="img/plus.png" alt="">
                </div>
                 <p class="mt-2">click here for delete existing product</p>
                <h2 class="text-uppercase text-muted">Delete Products</h2>
              </div>
              </div>
```

```
            </div>



          </div>
        </div>

        <!-- start Button trigger modal for Category -->


<!-- add Category Modal -->


<div class="modal fade" id="add-category-modal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true">
  <div class="modal-dialog modal-lg" role="document">
    <div class="modal-content">
      <div class="modal-header custom-bg text-white">
        <h5 class="modal-title" id="exampleModalLabel">Fill category details</h5>
        <button type="button" class="close" data-dismiss="modal" aria-label="Close">
          <span aria-hidden="true">&times;</span>
        </button>
      </div>
      <div class="modal-body">
```

```
<form action="ProductOperationServlet" method="post">

<input type="hidden" name="operation" value="addcategory">

<div class="form-group">

<input type="text" class="form-control" name="catTitle" placeholder="Enter category title" required />

</div>

<div class="form-group">

<input style type="text" class="form-control" name="catDescription" placeholder="Enter category description" required />

</div>

<div class="container text-center">

<button class="btn btn-outline-success">Add Category</button>

<button type="button" class="btn btn-secondary" data-dismiss="modal">Close</button>

</div>
```

```
        </form>
      </div>


    </div>
  </div>
</div>

<!-- end of add category model -->



<!-- start of add product model -->


<!-- Modal -->
<div class="modal fade" id="add-product-modal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true">
  <div class="modal-dialog modal-lg" role="document">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="exampleModalLabel">Product details</h5>
        <button type="button" class="close" data-dismiss="modal" aria-label="Close">
```

```
        <span aria-hidden="true">&times;</span>
      </button>
    </div>
    <div class="modal-body">
      <form action="ProductOperationServlet" method="post" enctype="multipart/form-data">

        <input type="hidden" name="operation" value="addproduct"/>

        <div class="form-group">
          <input type="text" class="form-control" placeholder="Enter title of product" name="pName" required/>

        </div>

        <div class="form-group">
          <input type="text" style="height: 200px;" class="form-control" placeholder="Enter product description" name="pDesc" required/>

        </div>

        <div class="form-group">
```

```
<input type="number" class="form-control" placeholder="Enter price of product" name="pPrice" required/>

</div>

<div class="form-group">
<input type="number" class="form-control" placeholder="Enter product discount " name="pDiscount" required/>

</div>

<div class="form-group">
<input type="number" class="form-control" placeholder="Enter product quantity" name="pQuantity" required/>

</div>
<div class="form-group">
<select name="catId" class="form-control" >
<%
for(Category c:list){
%>
```

```
            <option value="<%= c.getCategoryId()%>" > <%= c.getCategoryTitle()%>
</option>
                <% } %>
            </select>
        </div>

        <div class="form-group">
            <label for="pPic">Select picture of product label</label>
            <br>
            <input type="file" id="pPic" name="pPic" required/>
        </div>

        <div class="container text-center">
            <button class="btn btn-outline-success">Add product</button>
        </div>
```

```
        </form>
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-secondary" data-dismiss="modal">Close</button>


      </div>


    </div>
  </div>
</div>


          <!-- delete product Modal -->


<div class="modal fade" id="delete-product-modal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true">
  <div class="modal-dialog modal-lg" role="document">
    <div class="modal-content">
      <div class="modal-header custom-bg text-white">
        <h5 class="modal-title" id="exampleModalLabel">Fill category details</h5>
        <button type="button" class="close" data-dismiss="modal" aria-label="Close">
          <span aria-hidden="true">&times;</span>
```

```
      </button>
    </div>
    <div class="modal-body">
      <form action="ProductOperationServlet" method="post">

        <input type="hidden" name="operation" value="deleteproduct">

        <div class="form-group">
          <input type="number" class="form-control" name="pName" placeholder="Enter product id" required />

        </div>

        <div class="container text-center">

          <button class="btn btn-outline-success">Delete Product</button>
          <button type="button" class="btn btn-secondary" data-dismiss="modal">Close</button>

        </div>
```

```
        </form>
      </div>


    </div>
  </div>
</div>


<!-- end of add category model -->


                <%@include file="components/common_modals.jsp" %>
    </body>
</html>
```

- **login.jsp**

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
```

```
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Login</title>

    <%@include file="components/common_css_js.jsp" %>
</head>
<body>

    <%@include file="components/navbar.jsp" %>
    <div class="container">
        <div class="row">
            <div class="col-md-6 offset-md-3">

                <div class="card mt-3">

                    <div class="card-header custom-bg text-white">
                        <h3> Login here </h3>
                    </div>

                    <div class="card-body">
```

```
<%@include file="components/message.jsp" %>

<form action="LoginServlet" method="post">
  <div class="form-group">
    <label for="exampleInputEmail1">Email address</label>
    <input type="email" name="email" class="form-control" id="exampleInputEmail1" aria-describedby="emailHelp" placeholder="Enter email" required>
    <small id="emailHelp" class="form-text text-muted">We'll never share your email with anyone else.</small>
  </div>
  <div class="form-group">
    <label for="exampleInputPassword1">Password</label>
    <input type="password" name="password" class="form-control" id="exampleInputPassword1" placeholder="Password" required>
  </div>

  <a href="register.jsp" class="text-center d-block mb-2"/>If not registered click here</a>

  <div class="container text-center">
    <button type="submit" class="btn btn-primary custom-bg border-0">Submit</button>
```

```
                    <button type="reset" class="btn btn-primary custom-bg border-0">Reset</button>
                </div>
            </form>
        </div>
    </div>
</div>
</div>
</div>

<%@include file="components/common_modals.jsp" %>

</body>
</html>
```

- **normal.jsp**

```
<%
   User user = (User) session.getAttribute("current-user");
   if(user==null){
   session.setAttribute("message","You are not logged in !!!");
   response.sendRedirect("login.jsp");
   return;
   }
%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Normal user</title>
      <%@include file="components/common_css_js.jsp" %>
   </head>
   <body>
       <%@include file="components/navbar.jsp" %>

       <div class="container-fluid">
```

```
<div class="row">

  <div class="col-md-6 mt-5">

    <div class="card">

      <div class="card-header custom-bg1 ">
      <h3>Your details</h3>
    </div>

      <div class="card-body mt-5">
      <div style="color: gold"><h4> Name - <%= user.getUserName()%></h4></div><br>

      <div style="color: gold"><h4> Email address - <%= user.getUserEmail()%></h4></div><br>

      <div style="color: gold"><h4> address - <%= user.getUserAddress()%></h4></div><br>
```

```
                <div style="color: gold"><h4> contact number - <%= user.getUserPhone()
%></h4></div><br>


                 <div class="container text-center">

                     <button class="btn btn-outline-success">Change your details </button>

                   </div>

               </div>

             </div>

           </div>

           <div class="col-md-6 mt-5">

            <!--card-->


            <div class="card">

              <div class="card-header custom-bg1">

                  <h3> Your Previous item</h3>

               </div>


              <div class="card-body text-center">

                 <div class='mt-5'>

                    <h4 style="color:red;"> No transaction history !!</h4>
```

```
                </div>
              </div>
            </div>
          </div>
        </div>
        <%@include file="components/common_modals.jsp" %>
    </body>
</html>
```

- **register .jsp**

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>New User</title>

        <%@include file="components/common_css_js.jsp" %>
    </head>
```

```
<body>

<%@include file="components/navbar.jsp" %>

<div class="container-fluid">
<div class="row mt-5">
<div class="col-md-4 offset-md-4" >
<div class="card">

<%@include file="components/message.jsp" %>

<div class="card-body px-5">

<div class="container text-center">
<img src="img/signUp.jpg" style="max-width: 200px;" class="img-fluid "
alt="..." >

</div>
<h3 class="text-center my-3">Sign up here!!</h3>
<form action="RegisterServlet" method="post">
<div class="form-group">
```

```
                <label for="name">User Name</label>

                <input type="text" name="user_name" class="form-control" id="name" placeholder="Enter here" required >

              </div>

             <div class="form-group">

                <label for="name">User Email</label>

                <input type="text" name="user_email" class="form-control" id="email" placeholder="Enter here" required >

              </div>

             <div class="form-group">

                <label for="password">User password</label>

                <input type="password" name="user_password" class="form-control" id="password" placeholder="Enter here" required >

              </div>

             <div class="form-group">

                <label for="phone">User phone</label>

                <input type="number" name="user_phone" class="form-control" id="phone" placeholder="Enter here" required>

              </div>

             <div class="form-group">

                <label for="phone">User Address</label>
```

```
                <textarea name="user_address" style="height:100px;" class="form-control"
placeholder="Enter your address" required>
                </textarea>
               </div>

              <div class="container text-center">
                <button type="submit" class="btn btn-outline-success">Register here</button>
                <button type="reset" class="btn btn-outline-warning">Reset</button>
              </div>
            </form>
              </div>
            </div>
          </div>
        </div>
        </div>
                <%@include file="components/common_modals.jsp" %>
    </body>
</html>
```

- **checkout.jsp**

```
<%
    User user = (User) session.getAttribute("current-user");
   if(user==null){
   session.setAttribute("message","You are not logged in !! Login first to access checkout page");
   response.sendRedirect("login.jsp");
   return;
   }
   else{


   }
%>


<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Checkout</title>
       <%@include file="components/common_css_js.jsp" %>
```

```
</head>
<body>
   <%@include file="components/navbar.jsp" %>
   <div class="container">

      <div class="row mt-5">

         <div class="col-md-6">
          <!--card-->

          <div class="card">

             <div class="card-body">
                <h3 class="text-center mb-5">Your Selected items</h3>

                <div class="cart-body">

                </div>
             </div>
          </div>
         </div>
```

```
<div class="col-md-6">
  <!--form-->

  <div class="card">
    <div class="card-body">

      <h3 class="text-center mb-5">Your details for order </h3>
      <form action="#!">

        <div class="form-group">
          <label for="exampleInputEmail1">Email address</label>
          <input value="<%= user.getUserEmail() %>" type="email" class="form-control" id="exampleInputEmail1" aria-describedby="emailHelp" placeholder="Enter email">

          <small id="emailHelp" class="form-text text-muted">We'll never share your email with anyone else.</small>
        </div>

        <div class="form-group">
          <label for="name">Your name</label>
```

<input value="<%= user.getUserName() %>" type="text" class="form-control" id="name" aria-describedby="emailHelp" placeholder="Enter name">

** **

** **

<label for="phone">Your contact number</label>

<input value="<%= user.getUserPhone() %>" type="text" class="form-control" id="name" aria-describedby="emailHelp" placeholder="Enter contact number">

** **

** **

<label for="name" >Your shipping address</label>

<textarea value="<%= user.getUserAddress() %>" class="form-control" rows="3" placeholder="enter your address"> </textarea>

** **

** **

<button class="btn btn-outline-success">Order Now </button>

<button class="btn btn-outline-primary">Continue Shopping</button>

** **

</form>

```
                </div>
            </div>
        </div>
    </div>
</div>
<%@include file="components/common_modals.jsp" %>
</body>
</html>
```

- **LoginServlet.java**

```
package com.website.onlineshopping.servlets;

import com.website.onlineshopping.dao.UserDao;
import com.website.onlineshopping.entities.User;
import com.website.onlineshopping.helper.FactoryProvider;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
```

```
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import jakarta.servlet.http.HttpSession;
import java.io.IOException;
import java.io.PrintWriter;

@WebServlet(name = "LoginServlet", urlPatterns = {"/LoginServlet"})
public class LoginServlet extends HttpServlet {
  // private static final long serialVersionUID = 1L;

   protected void processRequest(HttpServletRequest request, HttpServletResponse response)
           throws ServletException, IOException {
       response.setContentType("text/html;charset=UTF-8");
      try ( PrintWriter out = response.getWriter()) {

           String email = request.getParameter("email");
           String password = request.getParameter("password");

           //validation
           //authenticating the user
           UserDao userDao = new UserDao(FactoryProvider.getFactory());
```

```
User user = userDao.getUserByEmailAndPassword(email, password);
//System.out.println(user);
HttpSession httpSession = request.getSession();

if(user==null){

   httpSession.setAttribute("message","Invalid Details !! Try with another one");
   response.sendRedirect("login.jsp");

}else{
   out.println("<h1>Welcome "+ user.getUserName() +" </h1>");

   httpSession.setAttribute("current-user",user);

   if(user.getUserType().equals("admin")){
      response.sendRedirect("admin.jsp");
   }else if(user.getUserType().equals("normal")){
     response.sendRedirect("normal.jsp");
}
   else{
      out.println("we havve not identified user type");
```

```
        }
      }
   }
}

@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
      throws ServletException, IOException {
   processRequest(request, response);
}

@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
      throws ServletException, IOException {
       processRequest(request, response);
}

@Override
public String getServletInfo() {
   return "Short description";
```

}// </editor-fold>}

LogoutServlet.java

```
package com.website.onlineshopping.servlets;

import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import jakarta.servlet.http.HttpSession;
import java.io.IOException;
import java.io.PrintWriter;


@WebServlet(name = "LogoutServlet", urlPatterns = {"/LogoutServlet"})
public class LogoutServlet extends HttpServlet {

    private static final long serialVersionUID = 1L;
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
```

```
        response.setContentType("text/html;charset=UTF-8");
        try ( PrintWriter out = response.getWriter()) {

            HttpSession httpSession = request.getSession();
            httpSession.removeAttribute("current-user");
            response.sendRedirect("login.jsp");
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
```

```
        throws ServletException, IOException {
    processRequest(request, response);
}

/**
 * Handles the HTTP <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    processRequest(request, response);
}

/**
 * Returns a short description of the servlet.
 *
```

```
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

- **ProductOperationServlet.java**

```
package com.website.onlineshopping.servlets;


import com.website.onlineshopping.dao.CategoryDao;
import com.website.onlineshopping.dao.ProductDao;
import com.website.onlineshopping.entities.Category;
import com.website.onlineshopping.entities.Product;
import com.website.onlineshopping.helper.FactoryProvider;
import jakarta.servlet.ServletException;
```

```
import jakarta.servlet.annotation.MultipartConfig;

import jakarta.servlet.annotation.WebServlet;

import jakarta.servlet.http.HttpServlet;

import jakarta.servlet.http.HttpServletRequest;

import jakarta.servlet.http.HttpServletResponse;

import jakarta.servlet.http.HttpSession;

import jakarta.servlet.http.Part;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;



import java.io.IOException;

import java.io.InputStream;

import java.io.PrintWriter;



@MultipartConfig

@WebServlet(name = "ProductOperationServlet", urlPatterns =
{"/ProductOperationServlet"})

public class ProductOperationServlet extends HttpServlet {
```

```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    try ( PrintWriter out = response.getWriter()) {

        String op = request.getParameter("operation");
        if(op.trim().equals("addcategory")){

            //fetching category data

       String title = request.getParameter("catTitle");
       String description = request.getParameter("catDescription");

            Category category = new Category();
            category.setCategoryTitle(title);
            category.setCategoryDescription(description);

            CategoryDao categoryDao = new CategoryDao(FactoryProvider.getFactory());
           int catId = categoryDao.saveCategory(category);
            //out.println("Category saved");
            HttpSession httpSession = request.getSession();
```

```
    httpSession.setAttribute("message", "Category added succesfully: "+catId);
    response.sendRedirect("admin.jsp");
    return;

}else if(op.trim().equals("addproduct")){
    //add product
    String pName = request.getParameter("pName");
    String pDesc = request.getParameter("pDesc");
    int pPrice = Integer.parseInt(request.getParameter("pPrice"));
    int pDiscount = Integer.parseInt(request.getParameter("pDiscount"));
    int pQuantity = Integer.parseInt(request.getParameter("pQuantity"));
    int catId = Integer.parseInt(request.getParameter("catId"));
    Part part = request.getPart("pPic");

    Product p = new Product();
    p.setpName(pName);

    p.setpDesc(pDesc);
    p.setpPrice(pPrice);
    p.setpDiscount(pDiscount);
    p.setpQuantity(pQuantity);
```

```
p.setpPhoto(part.getSubmittedFileName());


//get Category by id
CategoryDao cdoa = new CategoryDao(FactoryProvider.getFactory());
Category category = cdoa.getCategoryById(catId);
p.setCategory(category);



//product save

ProductDao pdao = new ProductDao(FactoryProvider.getFactory());
pdao.saveProduct(p);


//upload photo in folder

String path = request.getRealPath("img")+File.separator + "products"
+File.separator+ part.getSubmittedFileName();


try{
FileOutputStream fos = new FileOutputStream(path);
```

```
InputStream is = part.getInputStream();


 byte []data = new byte[is.available()];
 is.read(data);


 fos.write(data);


 fos.close();
} catch (Exception e){
 e.printStackTrace();
}



//out.println("Product save Success.....");


HttpSession httpSession = request.getSession();
httpSession.setAttribute("message", "Product is added successfully : " + catId);
response.sendRedirect("admin.jsp");
return;
```

```
}
else if(op.equals("deleteproduct"))
{
   int pName = Integer.parseInt(request.getParameter("pName"));




   ProductDao pdao = new ProductDao(FactoryProvider.getFactory());


  String f= pdao.deleteProduct(pName);


   HttpSession httpSession = request.getSession();


   httpSession.setAttribute("message", "Delete Operation on Product is  : "+f);}
```

```
            response.sendRedirect("admin.jsp");

            return;

        }

    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
```

```
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    processRequest(request, response);
}

/**
 * Handles the HTTP <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    processRequest(request, response);
}

/**
```

```
     * Returns a short description of the servlet.
     *
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

- **RegisterServlet.java**

```
package com.website.onlineshopping.servlets;

import com.website.onlineshopping.entities.User;
import com.website.onlineshopping.helper.FactoryProvider;
import jakarta.servlet.ServletException;
import java.io.IOException;
```

```
import java.io.PrintWriter;

import jakarta.servlet.annotation.WebServlet;

import jakarta.servlet.http.HttpServlet;

import jakarta.servlet.http.HttpServletRequest;

import jakarta.servlet.http.HttpServletResponse;

import jakarta.servlet.http.HttpSession;

import org.hibernate.Session;

import org.hibernate.Transaction;



/**

 *

 * @author me

 */

@WebServlet(name = "RegisterServlet", urlPatterns = {"/RegisterServlet"})

public class RegisterServlet extends HttpServlet {



    /**

     * Processes requests for both HTTP <code>GET</code> and <code>POST</code>

     * methods.

     *

     * @param request servlet request
```

```
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    try ( PrintWriter out = response.getWriter()) {

        try{

            String userName = request.getParameter("user_name");
            String userEmail = request.getParameter("user_email");
            String userPassword = request.getParameter("user_password");
            String userPhone = request.getParameter("user_phone");
            String userAddress = request.getParameter("user_address");

            //validation

            if(userName.isEmpty()){
                out.println("Name is blank");
```

```
            return;
        }
        //creating user object to store data

        User user = new User(userName, userEmail, userPassword, userPhone, "signUp.jpg",
userAddress,"normal");

        int userId;
        try (Session hibernateSession = FactoryProvider.getFactory().openSession()) {
            Transaction tx = hibernateSession.beginTransaction();
            userId = (int) hibernateSession.save(user);
            tx.commit();
        }

        out.println("succesfully");

        HttpSession httpSession = request.getSession();
        httpSession.setAttribute("message","Resgistration Succesful !!" +userId);

        response.sendRedirect("register.jsp");
        return;
```

```
        }catch(Exception e){
            e.printStackTrace();
        }
    }
}

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        processRequest(request, response);
```

```
}

/**
 * Handles the HTTP <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    processRequest(request, response);
}

/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
```

```
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>
}
```

- **FactoryProvider.java**

```
package com.website.onlineshopping.helper;

import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class FactoryProvider {
    private static SessionFactory factory;

    public static SessionFactory getFactory()
    {
        try{
```

```
            if(factory==null){
                factory  = new Configuration()
                        .configure("hibernate.cfg.xml")
                        .buildSessionFactory();
            }

        }catch(Exception e){
            e.printStackTrace();
        }
        return factory;
    }

}
```

- **Helper.java**

```
package com.website.onlineshopping.helper;

import java.util.HashMap;
```

```
import java.util.Map;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.query.Query;



public class Helper {

   public static String get10Words(String desc){

      String[] strs = desc.split(" ");

      if(strs.length>10){

         String res="";

         for(int i=0;i<10;i++){

            res=res+strs[i]+" ";



         }

         return (res+"....");

      }else{

         return (desc+"....");

      }
```

```
    }

    public static Map<String,Long> getCounts(SessionFactory factory){

        Session session = factory.openSession();
        String q1 ="select count(*) from User";
        String q2 ="select count(*) from Product";

        Query query1 = session.createQuery(q1);
        Query query2 = session.createQuery(q2);

        Long userCount = (Long)query1.list().get(0);
        Long productCount = (Long)query2.list().get(0);

        Map<String,Long> map = new HashMap<>();
        map.put("userCount",userCount);
        map.put("productCount",productCount);
        session.close();
        return map;
    }
}
```

- **Category.java**

```
package com.website.onlineshopping.entities;

import java.util.ArrayList;

import java.util.List;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.OneToMany;

import org.hibernate.SessionFactory;


@Entity

public class Category {

    @Id

   @GeneratedValue(strategy = GenerationType.IDENTITY)

   private int categoryId;

   private String categoryTitle;

   private String categoryDescription;

   @OneToMany(mappedBy = "category")

   private List<Product> products = new ArrayList<>();
```

```
public Category() {

}


public Category(int categoryId, String categoryTitle, String categoryDescription) {

    this.categoryId = categoryId;

    this.categoryTitle = categoryTitle;

    this.categoryDescription = categoryDescription;

}


public Category(String categoryTitle, String categoryDescription,List<Product> products) {

    this.categoryTitle = categoryTitle;

    this.categoryDescription = categoryDescription;

    this.products = products;

}
```

```
public int getCategoryId() {
    return categoryId;
}

public void setCategoryId(int categoryId) {
    this.categoryId = categoryId;
}

public String getCategoryTitle() {
    return categoryTitle;
}

public void setCategoryTitle(String categoryTitle) {
    this.categoryTitle = categoryTitle;
}

public String getCategoryDescription() {
    return categoryDescription;
}
public void setCategoryDescription(String categoryDescription) {
    this.categoryDescription = categoryDescription;
```

```
    }

    public List<Product> getProducts() {

        return products;

    }


    public void setProducts(List<Product> products) {

        this.products = products;

    }


    @Override

    public String toString() {

        return "Product{" + "categoryId=" + categoryId + ", categoryTitle=" + categoryTitle + ",
categoryDescription=" + categoryDescription + '}';

    }

    public Category getCategoryById(int catId) {

        throw new UnsupportedOperationException("Not supported yet."); // Generated from
nbfs://nbhost/SystemFileSystem/Templates/Classes/Code/GeneratedMethodBody

    }

}
```

Product.java

```
package com.website.onlineshopping.entities;


import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.ManyToOne;


@Entity

public class Product {

   @Id

   @GeneratedValue(strategy = GenerationType.IDENTITY)

  private int pId;

  private String pName;

  @Column(length = 3000)

  private String pDesc;

  private String pPhoto;

  private int pPrice;

  private int pDiscount;
```

```
    private int pQuantity;
    @ManyToOne
    private Category category;


    public Product() {
    }




    public Product(String pName, String pDesc, String pPhoto, int pPrice, int pDiscount, int pQuantity, Category category) {
        this.pName = pName;
        this.pDesc = pDesc;
        this.pPhoto = pPhoto;
        this.pPrice = pPrice;
        this.pDiscount = pDiscount;
        this.pQuantity = pQuantity;
        this.category = category;
    }


    public int getpId() {
```

```
    return pId;
}

public void setpId(int pId) {
    this.pId = pId;
}

public String getpName() {
    return pName;
}

public void setpName(String pName) {
    this.pName = pName;
}

public String getpDesc() {
    return pDesc;
}

public void setpDesc(String pDesc) {
    this.pDesc = pDesc;
```

```
}

public String getpPhoto() {
    return pPhoto;
}

public void setpPhoto(String pPhoto) {
    this.pPhoto = pPhoto;
}

public int getpPrice() {
    return pPrice;
}

public void setpPrice(int pPrice) {
    this.pPrice = pPrice;
}

public int getpDiscount() {
    return pDiscount;
}
```

```
public void setpDiscount(int pDiscount) {
    this.pDiscount = pDiscount;
}

public int getpQuantity() {
    return pQuantity;
}

public void setpQuantity(int pQuantity) {
    this.pQuantity = pQuantity;
}

public Category getCategory() {
    return category;
}

public void setCategory(Category category) {
    this.category = category;
}
```

```
    @Override
    public String toString() {
        return "Product{" + "pId=" + pId + ", pName=" + pName + ", pDesc=" + pDesc + ",
pPhoto=" + pPhoto + ", pPrice=" + pPrice + ", pDiscount=" + pDiscount + ", pQuantity=" +
pQuantity + '}';
    }

    public int getPriceAfterApplyingDiscount(){
        int d = (int)((this.getpDiscount()/100)*this.getpPrice());
        return this.getpPrice() - d;
    }
}
```

- **Product.java**

```
package com.website.onlineshopping.entities;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
```

```
import javax.persistence.ManyToOne;


@Entity
public class Product {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
   private int pId;
   private String pName;
   @Column(length = 3000)
   private String pDesc;
   private String pPhoto;
   private int pPrice;
   private int pDiscount;
   private int pQuantity;
   @ManyToOne
   private Category category;


    public Product() {
    }
```

```
    public Product(String pName, String pDesc, String pPhoto, int pPrice, int pDiscount, int pQuantity, Category category) {
        this.pName = pName;
        this.pDesc = pDesc;
        this.pPhoto = pPhoto;
        this.pPrice = pPrice;
        this.pDiscount = pDiscount;
        this.pQuantity = pQuantity;
        this.category = category;
    }

    public int getpId() {
        return pId;
    }

    public void setpId(int pId) {
        this.pId = pId;
    }

    public String getpName() {
```

```
    return pName;
}
public void setpName(String pName) {
    this.pName = pName;
}

public String getpDesc() {
    return pDesc;
}

public void setpDesc(String pDesc) {
    this.pDesc = pDesc;
}

public String getpPhoto() {
    return pPhoto;
}

public void setpPhoto(String pPhoto) {
    this.pPhoto = pPhoto;
}
```

```
public int getpPrice() {
    return pPrice;
}

public void setpPrice(int pPrice) {
    this.pPrice = pPrice;
}

public int getpDiscount() {
    return pDiscount;
}

public void setpDiscount(int pDiscount) {
    this.pDiscount = pDiscount;
}

public int getpQuantity() {
    return pQuantity;
}
```

```
    public void setpQuantity(int pQuantity) {
        this.pQuantity = pQuantity;
    }

    public Category getCategory() {
        return category;
    }

    public void setCategory(Category category) {
        this.category = category;
    }

    @Override
    public String toString() {
        return "Product{" + "pId=" + pId + ", pName=" + pName + ", pDesc=" + pDesc + ", pPhoto=" + pPhoto + ", pPrice=" + pPrice + ", pDiscount=" + pDiscount + ", pQuantity=" + pQuantity + '}';
    }

    public int getPriceAfterApplyingDiscount(){
        int d = (int)((this.getpDiscount()/100)*this.getpPrice());
        return this.getpPrice() - d;
```

```
    }
}
```

- **User.java**

```
package com.website.onlineshopping.entities;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

@Entity
public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(length =10, name = "user_id")
    private int userId;
    @Column(length =100, name = "user_name")
```

```
    private String userName;

    @Column(length =100, name = "user_email")

    private String userEmail;

    @Column(length =100, name = "user_password")

    private String userPassword;

    @Column(length =12, name = "user_phone")

    private String userPhone;

    @Column(length =1500, name = "user_pic")

    private String userPic;

    @Column(length =1500, name = "user_address")

    private String userAddress;


    @Column(name = "user_type")

    private String userType;


    public User(int userId, String userName, String userEmail, String userPassword, String
userPhone, String userPic, String userAddress, String userType) {

        this.userId = userId;

        this.userName = userName;

        this.userEmail = userEmail;

        this.userPassword = userPassword;
```

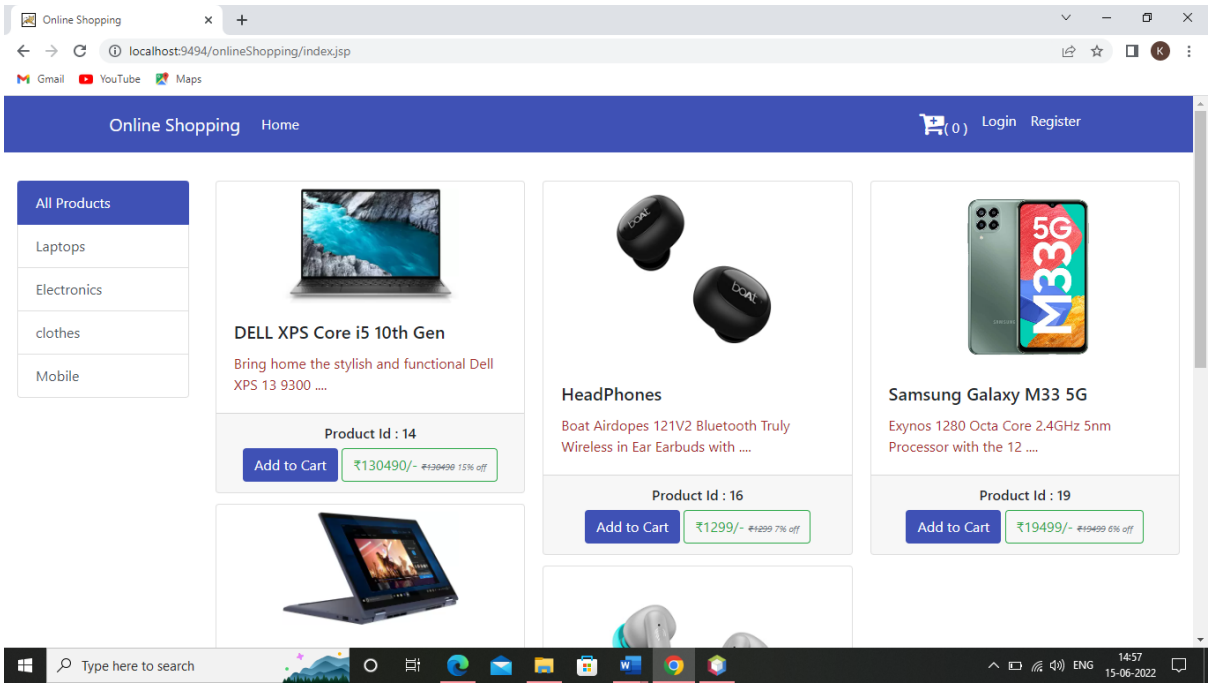
```
        this.userPhone = userPhone;
        this.userPic = userPic;
        this.userAddress = userAddress;
        this.userType = userType;
    }

    public User(String userName, String userEmail, String userPassword, String userPhone, String userPic, String userAddress, String userType) {
        this.userName = userName;
        this.userEmail = userEmail;
        this.userPassword = userPassword;
        this.userPhone = userPhone;
        this.userPic = userPic;
        this.userAddress = userAddress;
        this.userType = userType;
    }

    public User() {
    }

    public int getUserId() {
```

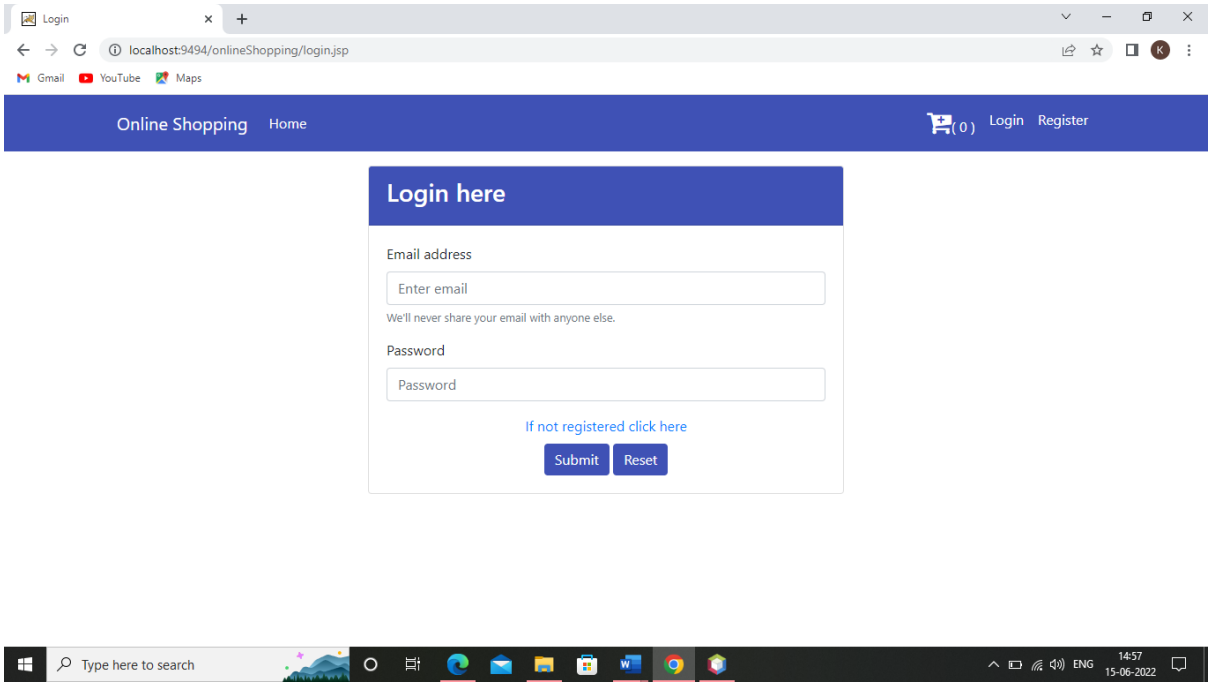
```
    return userId;
}

public void setUserId(int userId) {
    this.userId = userId;
}

public String getUserName() {
    return userName;
}

public void setUserName(String userName) {
    this.userName = userName;
}

public String getUserEmail() {
    return userEmail;
}

public void setUserEmail(String userEmail) {
    this.userEmail = userEmail;
```

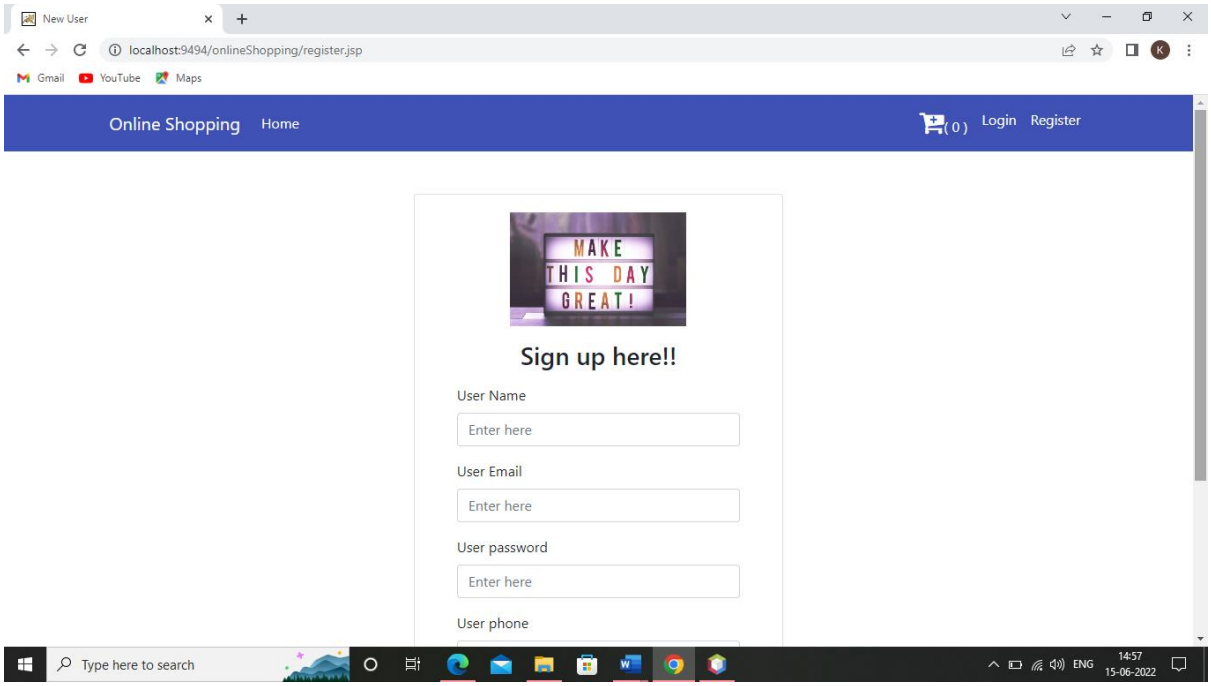
```
}

public String getUserPassword() {
    return userPassword;
}

public void setUserPassword(String userPassword) {
    this.userPassword = userPassword;
}

public String getUserPhone() {
    return userPhone;
}

public void setUserPhone(String userPhone) {
    this.userPhone = userPhone;
}

public String getUserPic() {
    return userPic;
}
```

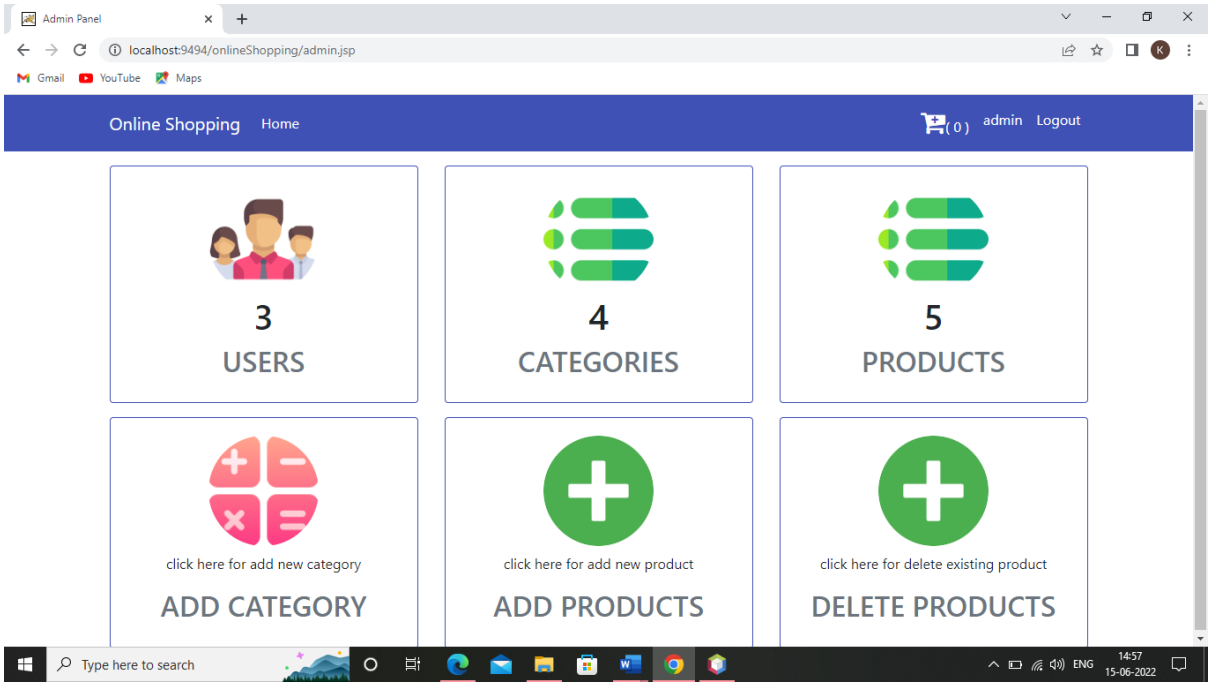
```
    public void setUserPic(String userPic) {
        this.userPic = userPic;
    }
    public String getUserAddress() {
        return userAddress;
    }
    public void setUserAddress(String userAddress) {
        this.userAddress = userAddress;
    }
    public String getUserType() {
        return userType;
    }
    public void setUserType(String userType) {
        this.userType = userType;
    }
    @Override
    public String toString() {
        return "User{" + "userId=" + userId + ", userName=" + userName + ", userEmail=" +
userEmail + ", userPassword=" + userPassword + ", userPhone=" + userPhone + ", userPic=" +
userPic + ", userAddress=" + userAddress + '}';
    }
}
```

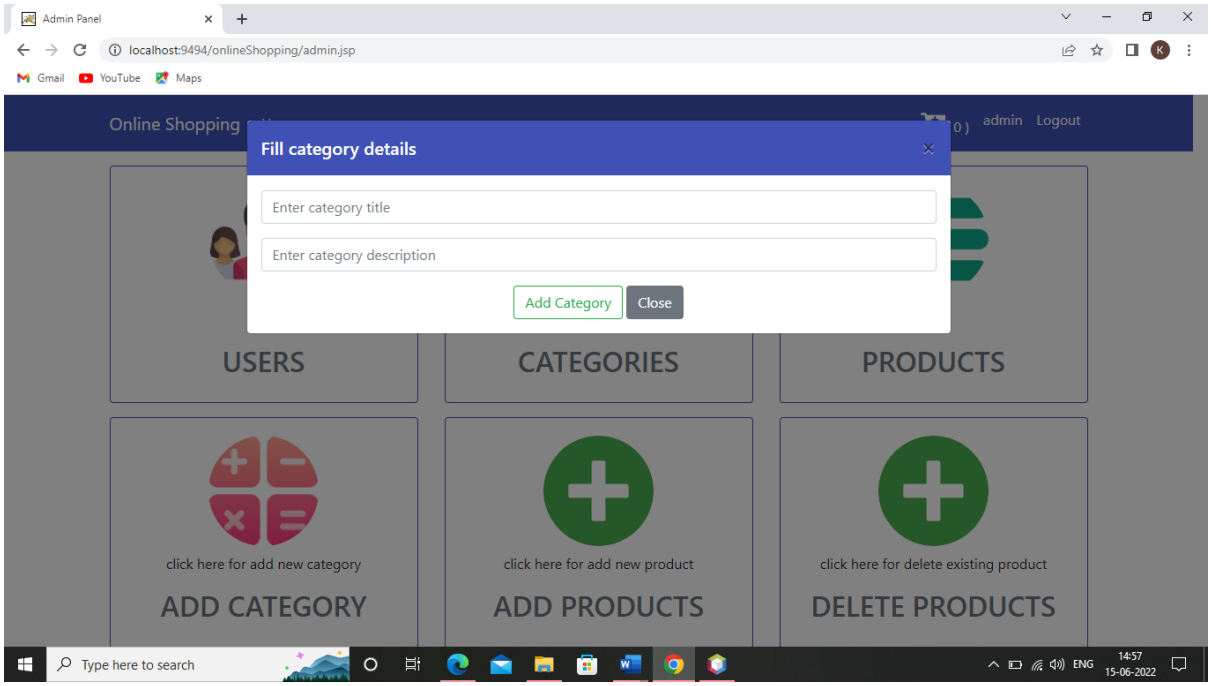
Results

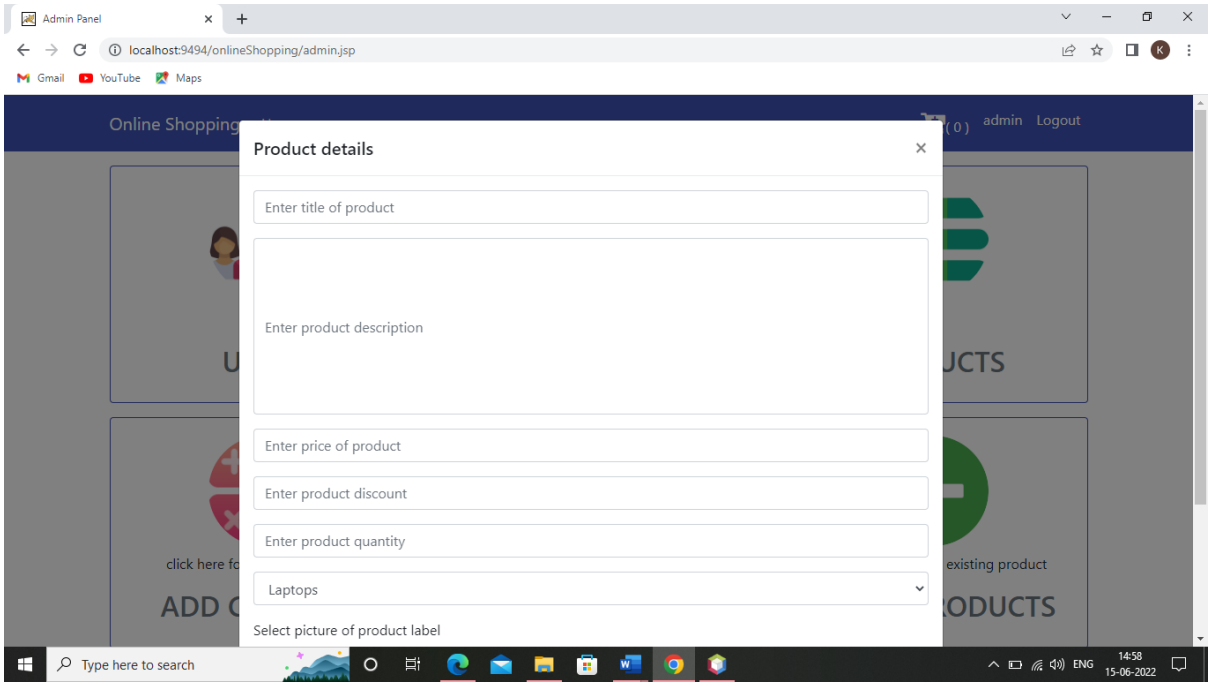

Online Shopping
localhost:9494/onlineShopping/index.jsp
Gmail
YouTube
Maps
Online Shopping
Home
(0)
Login
Register
All Products
Laptops
Electronics
clothes
Mobile
DELL XPS Core i5 10th Gen
Bring home the stylish and functional Dell XPS 13 9300 ....
Product Id : 14
Add to Cart
₹130490/-
HeadPhones
Boat Airdopes 121V2 Bluetooth Truly Wireless in Ear Earbuds with ....
Product Id : 16
Add to Cart
₹1299/-
Samsung Galaxy M33 5G
Exynos 1280 Octa Core 2.4GHz 5nm Processor with the 12 ....
Product Id : 19
Add to Cart
₹19499/-
Type here to search
ENG
14:57
15-06-2022

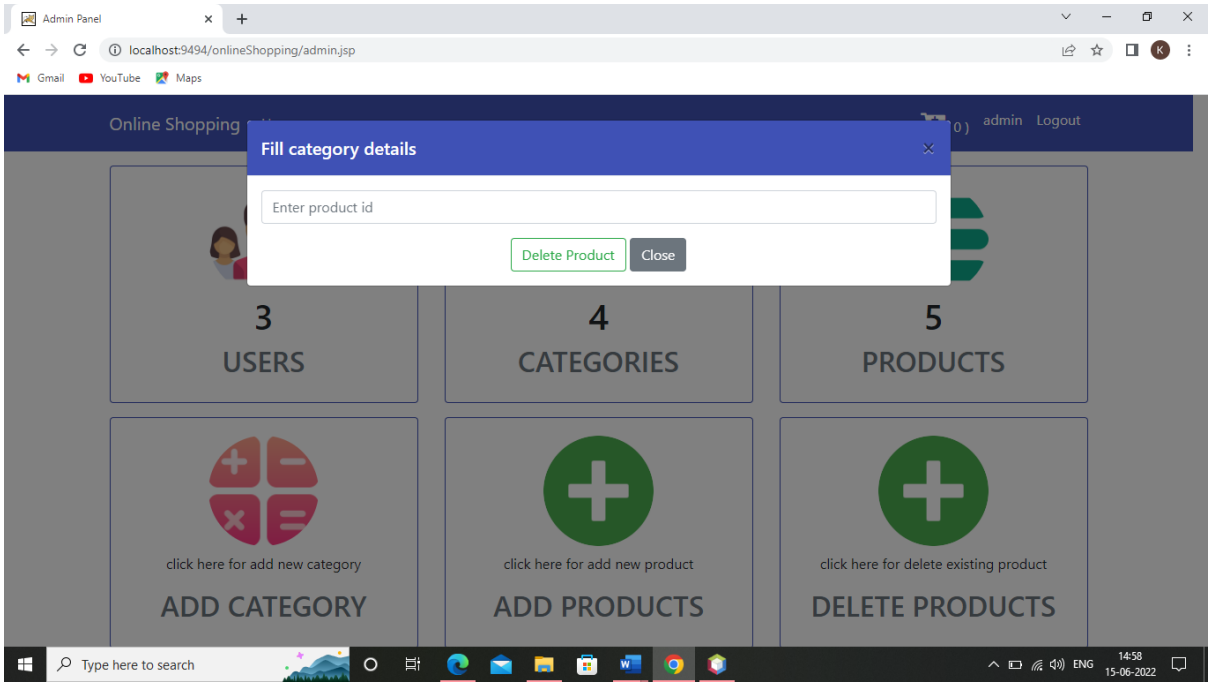

Login
localhost:9494/onlineShopping/login.jsp
Gmail
YouTube
Maps
Online Shopping
Home
(0)
Login
Register
Login here
Email address
Enter email
We'll never share your email with anyone else.
Password
Password
If not registered click here
Submit
Reset
Type here to search
ENG
14:57
15-06-2022

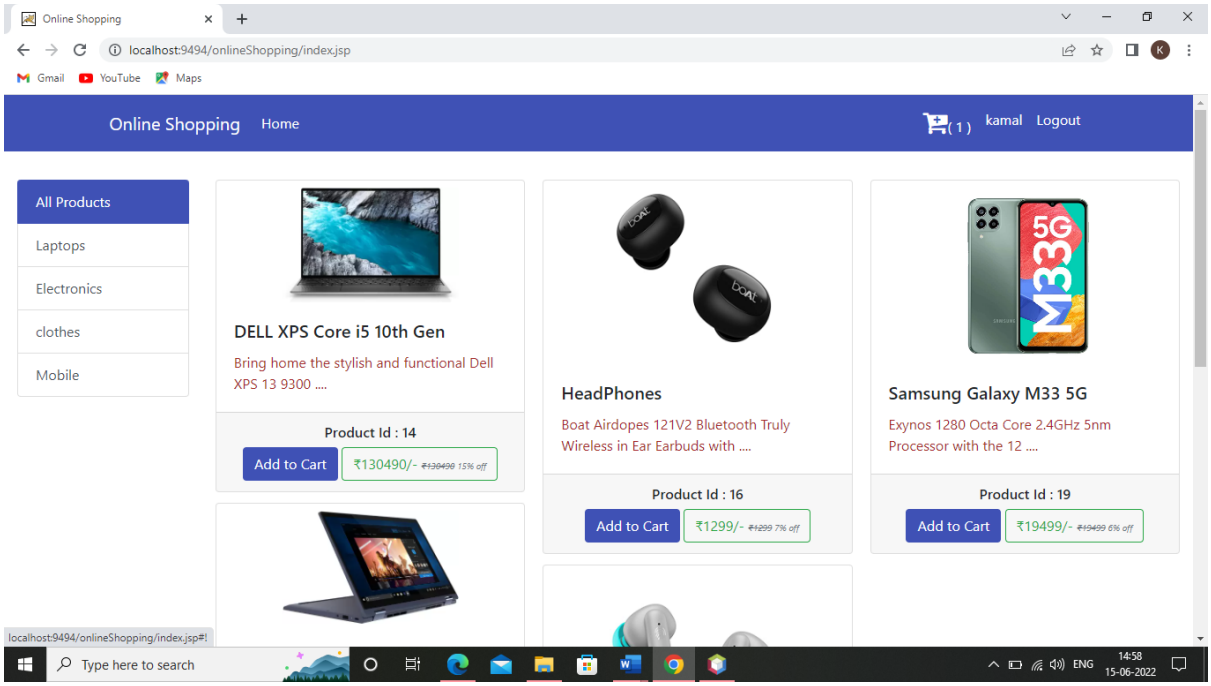

New User
localhost:9494/onlineShopping/register.jsp
Gmail
YouTube
Maps
Online Shopping
Home
(0)
Login
Register
MAKE
THIS DAY
GREAT!
Sign up here!!
User Name
Enter here
User Email
Enter here
User password
Enter here
User phone
Type here to search
ENG
14:57
15-06-2022

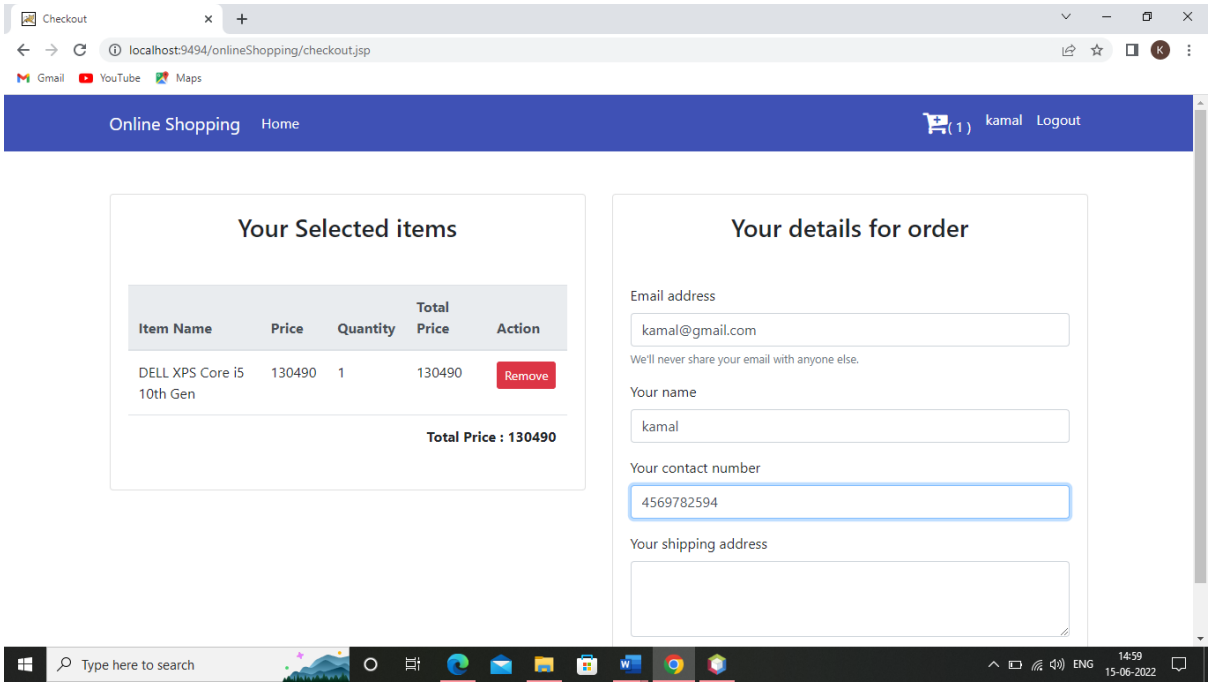

Admin Panel
localhost:9494/onlineShopping/admin.jsp
Gmail
YouTube
Maps
Online Shopping
Home
(0)
admin
Logout
3
USERS
4
CATEGORIES
5
PRODUCTS
click here for add new category
ADD CATEGORY
click here for add new product
ADD PRODUCTS
click here for delete existing product
DELETE PRODUCTS
Type here to search
ENG
14:57
15-06-2022


Admin Panel
localhost:9494/onlineShopping/admin.jsp
Gmail
YouTube
Maps
Online Shopping
admin
Logout
Fill category details
Enter category title
Enter category description
Add Category
Close
USERS
CATEGORIES
PRODUCTS
click here for add new category
ADD CATEGORY
click here for add new product
ADD PRODUCTS
click here for delete existing product
DELETE PRODUCTS
Type here to search
ENG
14:57
15-06-2022


Admin Panel
localhost:9494/onlineShopping/admin.jsp
Gmail
YouTube
Maps
Online Shopping
admin
Logout
Product details
Enter title of product
Enter product description
Enter price of product
Enter product discount
Enter product quantity
Laptops
Select picture of product label
Type here to search
ENG
14:58
15-06-2022


Admin Panel
localhost:9494/onlineShopping/admin.jsp
Gmail
YouTube
Maps
Online Shopping
admin
Logout
Fill category details
Enter product id
Delete Product
Close
3
USERS
4
CATEGORIES
5
PRODUCTS
click here for add new category
ADD CATEGORY
click here for add new product
ADD PRODUCTS
click here for delete existing product
DELETE PRODUCTS
Type here to search
ENG
14:58
15-06-2022


Online Shopping
localhost:9494/onlineShopping/index.jsp
Gmail
YouTube
Maps
Online Shopping
Home
kamal
Logout
All Products
Laptops
Electronics
clothes
Mobile
DELL XPS Core i5 10th Gen
Bring home the stylish and functional Dell XPS 13 9300 ....
Product Id : 14
Add to Cart
₹130490/-
HeadPhones
Boat Airdopes 121V2 Bluetooth Truly Wireless in Ear Earbuds with ....
Product Id : 16
Add to Cart
₹1299/-
Samsung Galaxy M33 5G
Exynos 1280 Octa Core 2.4GHz 5nm Processor with the 12 ....
Product Id : 19
Add to Cart
₹19499/-
Type here to search
ENG
14:58
15-06-2022


Checkout
localhost:9494/onlineShopping/checkout.jsp
Gmail
YouTube
Maps
Online Shopping
Home
(1)
kamal
Logout
Your Selected items
Item Name
Price
Quantity
Total Price
Action
DELL XPS Core i5 10th Gen
130490
1
130490
Remove
Total Price : 130490
Your details for order
Email address
kamal@gmail.com
We'll never share your email with anyone else.
Your name
kamal
Your contact number
4569782594
Your shipping address
Type here to search
ENG
14:59
15-06-2022

7. Conclusion

The project entitled Online shopping system was completed successfully. The system has been developed with much care and free of errors and at the same time it is efficient and less time consuming. The purpose of this project was to develop a web application and an android application for purchasing items from a shop. This project helped us in gaining valuable information and practical knowledge on several topics like designing web pages using html & css, usage of responsive templates, designing of android applications, and management of database using mysql . The entire system is secured. Also the project helped us understanding about the development phases of a project and software development life cycle. We learned how to test different features of a project. This project has given us great satisfaction in having designed an application which can be implemented to any nearby shops or branded shops selling various kinds of products by simple modifications. There is a scope for further development in our project to a great extend. A number of features can be added to this system in future like providing moderator more control over products so that each moderator can maintain their own products. Another feature we wished to implement was providing classes for customers so that different offers can be given to each class. System may keep track of history of purchases of each customer and provide suggestions based on their history. These features could have implemented unless the time did not limited us.

8. Bibliography

- **https://www.javatpoint.com/jsp-tutorial**
- **https://www.javatpoint.com/servlet-tutorial**
- **https://www.javatpoint.com/hibernate-tutorial**
- **https://www.w3schools.com/js/**