



Cranes Varsity

'Where Technology Meets Excellence'

Team	
Name	RAVEESHA H
Name	RUCHITHA S
College Name:	M V J COLLEGE OF ENGINEERING
Project Title	ONLINE SHOPPING SYSTEM
Program:	1 Month Internship on Full Stack Java
Start Date:	17/07/2022
End Date:	11/08/2022
Project Is Handled By	The project was adeptly supervised by our experienced trainer Chama Tiwari mam providing guidance, insights, and direction throughout its lifecycle. Their mentorship ensured the successful development and implementation of the online shopping System.

ABSTRACT

This project is a web based shopping system for an existing shop. The project objective is to deliver the online shopping application into an android platform. This project is an attempt to provide the advantages of online shopping to customers of a real shop. It helps buying the products in the shop anywhere through internet by using an android device. Thus the customer will get the service of online shopping and home delivery from his favorite shop. This system can be implemented to any shop in the locality or to multinational branded shops having retail outlet chains. 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.

INTRODUCTION TO PROJECT

1. Goal

Shopping has long been considered a recreational activity by many. Shopping online is no exception. The goal of this application is to develop a web based interface for online retailers. The system would be easy to use and hence make the shopping experience pleasant for the users. The goal of this application is

- To develop an easy to use web based interface where users can search for products, view a complete description of the products and order the products.
- A search engine that provides an easy and convenient way to search for products specific to their needs. The search engine would list a set of products based on the search term and the user can further filter the list based on various parameters.
- An AJAX enabled website with the latest AJAX controls giving attractive and interactive look to the web pages and prevents the annoying post backs.
- Drag and Drop feature which would allow the users to add a product to or remove a product from the shopping cart by dragging the product in to the shopping cart or out of the shopping cart.
- A user can view the complete specification of the product along with various images and also view the customer reviews of the product. They can also write their own reviews.

2 Need of the application

There are large numbers of commercial Online Shopping websites offering large number of products tailored to meet the shopping interests of large number of customers. These online marketplaces have thousands of products listed under various categories.

Problem:

- The basic problems with the existing systems are the non-interactive environment they provide to the users.
- The use of traditional user interfaces which make continuous post backs to the server; each post back makes a call to the server, gets the response and then refreshes the entire web form to display the result. This scenario adds an extra

trade off causing a delay in displaying the results

- A search engine that would display the results without allowing the users to further filter the results based on various parameters.
- Use of traditional and non user friendly interfaces that are hard to use

3.Solution:

- The motive of this Online Shopping Web Application is to allow the user to play with the search tool and create different combinatorial search criterion to perform exhaustive search.
- Making the application AJAX enabled gets rid of these unnecessary delays letting the user to perform exhaustive search. The users of this application can easily feel the difference between the Ajax empowered user interfaces vs. traditional user interfaces.
- Provide Interactive interface through which a user can interact with different areas of application easily.
- A search engine that provides an easy and convenient way to search for products specific to their needs. The search engine would list a set of products based on the search term and the user can further filter the list based on various parameters.
- Provide Drag and Drop feature thereby allowing the user to add products to or remove products from the shopping cart by dragging the products in to or out of the shopping cart.

4.Scope

- The current system can be extended to allow the users to create accounts and save products in to wish list.
- The users could subscribe for price alerts which would enable them to receive messages when price for products fall below a particular level.
- The current system is confined only to the shopping cart process. It can be extended to have a easy to use check out process.
- Users can have multiple shipping and billing information saved. During checkout they can use the drag and drop feature to select shipping and billing information.

Online Shopping

1. Connection

```
package cn.techtutorial.connection;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class DbCon {
    private static Connection connection = null;

    public static Connection getConnection() throws ClassNotFoundException,
SQLException{
        if(connection == null){
            Class.forName("com.mysql.cj.jdbc.Driver");

            connection=DriverManager.getConnection("jdbc:mysql://localhost:3306/ecommerce_cart","root","root");

            System.out.print("connected");
        }
        return connection;
    }
}
```

2. orderDao.java

```
package cn.techtutorial.dao;

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

```

import java.util.*;

import cn.techtutorial.model.*;

public class OrderDao {

    private Connection con;

    private String query;
    private PreparedStatement pst;
    private ResultSet rs;

    public OrderDao(Connection con) {
        super();
        this.con = con;
    }

    public boolean insertOrder(Order model) {
        boolean result = false;
        try {
            query = "insert into orders (p_id, u_id, o_quantity, o_date) values(?,?,?,?)";
            pst = this.con.prepareStatement(query);
            pst.setInt(1, model.getId());
            pst.setInt(2, model.getUid());
            pst.setInt(3, model.getQunatity());
            pst.setString(4, model.getDate());
            pst.executeUpdate();
            result = true;
        } catch (SQLException e) {
            System.out.println(e.getMessage());
        }
        return result;
    }

```

```

    }

    public List<Order> userOrders(int id) {
        List<Order> list = new ArrayList<>();

        try {
            query = "select * from orders where u_id=? order by orders.o_id desc";
            pst = this.con.prepareStatement(query);
            pst.setInt(1, id);
            rs = pst.executeQuery();
            while (rs.next()) {
                Order order = new Order();

                ProductDao productDao = new ProductDao(this.con);

                int pId = rs.getInt("p_id");
                Product product = productDao.getSingleProduct(pId);
                order.setOrderId(rs.getInt("o_id"));
                order.setId(pId);
                order.setName(product.getName());
                order.setCategory(product.getCategory());
                order.setPrice(product.getPrice()*rs.getInt("o_quantity"));
                order.setQunatity(rs.getInt("o_quantity"));
                order.setDate(rs.getString("o_date"));
                list.add(order);
            }
        } catch (Exception e) {
            e.printStackTrace();
            System.out.println(e.getMessage());
        }

        return list;
    }

    public void cancelOrder(int id) {

```

```

//boolean result = false;

try {
    query = "delete from orders where o_id=?";
    pst = this.con.prepareStatement(query);
    pst.setInt(1, id);
    pst.execute();
    //result = true;
} catch (SQLException e) {
    e.printStackTrace();
    System.out.print(e.getMessage());
}
//return result;
}
}

```

3. ProductDao.java

```

package cn.techtutorial.dao;

import java.sql.*;
import java.util.*;

import cn.techtutorial.model.Cart;
import cn.techtutorial.model.Product;

public class ProductDao {
    private Connection con;

    private String query;
    private PreparedStatement pst;

```



```

private ResultSet rs;

    public ProductDao(Connection con) {
        super();
        this.con = con;
    }

    public List<Product> getAllProducts() {
        List<Product> book = new ArrayList<>();
        try {

            query = "select * from products";
            pst = this.con.prepareStatement(query);
            rs = pst.executeQuery();

            while (rs.next()) {
                Product row = new Product();
                row.setId(rs.getInt("id"));
                row.setName(rs.getString("name"));
                row.setCategory(rs.getString("category"));
                row.setPrice(rs.getDouble("price"));
                row.setImage(rs.getString("image"));

                book.add(row);
            }

        } catch (SQLException e) {
            e.printStackTrace();
            System.out.println(e.getMessage());
        }

        return book;
    }

```

```
}
```

```

public Product getSingleProduct(int id) {
    Product row = null;
    try {
        query = "select * from products where id=? ";

        pst = this.con.prepareStatement(query);
        pst.setInt(1, id);
        ResultSet rs = pst.executeQuery();

        while (rs.next()) {
            row = new Product();
            row.setId(rs.getInt("id"));
            row.setName(rs.getString("name"));
            row.setCategory(rs.getString("category"));
            row.setPrice(rs.getDouble("price"));
            row.setImage(rs.getString("image"));
        }
    } catch (Exception e) {
        e.printStackTrace();
        System.out.println(e.getMessage());
    }

    return row;
}

public double getTotalCartPrice(ArrayList<Cart> cartList) {
    double sum = 0;

```

```

try {
    if (cartList.size() > 0) {
        for (Cart item : cartList) {
            query = "select price from products where id=?";
            pst = this.con.prepareStatement(query);
            pst.setInt(1, item.getId());
            rs = pst.executeQuery();
            while (rs.next()) {
                sum+=rs.getDouble("price")*item.getQuantity();
            }
        }
    }
} catch (SQLException e) {
    e.printStackTrace();
    System.out.println(e.getMessage());
}
return sum;
}

public List<Cart> getCartProducts(ArrayList<Cart> cartList) {
    List<Cart> book = new ArrayList<>();
    try {
        if (cartList.size() > 0) {
            for (Cart item : cartList) {
                query = "select * from products where id=?";
                pst = this.con.prepareStatement(query);
                pst.setInt(1, item.getId());
                rs = pst.executeQuery();
                while (rs.next()) {

```

```

        Cart row = new Cart();
        row.setId(rs.getInt("id"));
        row.setName(rs.getString("name"));
        row.setCategory(rs.getString("category"));
        row.setPrice(rs.getDouble("price")*item.getQuantity());
        row.setQuantity(item.getQuantity());
        book.add(row);
    }

}

}

} catch (SQLException e) {
    e.printStackTrace();
    System.out.println(e.getMessage());
}
return book;
}

}

```

4. ProductDao2.java

```

package cn.techtutorial.dao;

import java.sql.*;
import java.util.*;
import cn.techtutorial.model.Cart;
import cn.techtutorial.model.Product2;

public class ProductDao2 {

```

```

    private Connection con;

    private String query;
private PreparedStatement pst;
private ResultSet rs;

    public ProductDao2(Connection con) {
        super();
        this.con = con;
    }

    public List<Product2> getAllProducts2() {
List<Product2> book = new ArrayList<>();
    try {
        query = "select * from products2";
        pst = this.con.prepareStatement(query);
        rs = pst.executeQuery();
        while (rs.next()) {
            Product2 row = new Product2();
            row.setId(rs.getInt("id"));
            row.setName(rs.getString("name"));
            row.setCategory(rs.getString("category"));
            row.setPrice(rs.getDouble("price"));
            row.setImage(rs.getString("image"));
            book.add(row);
        }
    } catch (SQLException e) {
        e.printStackTrace();
        System.out.println(e.getMessage());
    }
    return book;
}

    public Product2 getSingleProduct2(int id) {

```

```

        Product2 row = null;
    try {
        query = "select * from products2 where id=? ";
        pst = this.con.prepareStatement(query);
        pst.setInt(1, id);
        ResultSet rs = pst.executeQuery();
        while (rs.next()) {
            row = new Product2();
            row.setId(rs.getInt("id"));
            row.setName(rs.getString("name"));
            row.setCategory(rs.getString("category"));
            row.setPrice(rs.getDouble("price"));
            row.setImage(rs.getString("image"));
        }
    } catch (Exception e) {
        e.printStackTrace();
        System.out.println(e.getMessage());
    }
    return row;
}

public double getTotalCartPrice(ArrayList<Cart> cartList) {
    double sum = 0;
    try {
        if (cartList.size() > 0) {
            for (Cart item : cartList) {
                query = "select price from products2 where id=?";
                pst = this.con.prepareStatement(query);
                pst.setInt(1, item.getId());
                rs = pst.executeQuery();
                while (rs.next()) {

```

```

        sum+=rs.getDouble("price")*item.getQuantity();
    }

}

}

} catch (SQLException e) {
    e.printStackTrace();
    System.out.println(e.getMessage());
}

return sum;
}

public List<Cart> getCartProducts2(ArrayList<Cart> cartList) {
    List<Cart> book = new ArrayList<>();
    try {
        if (cartList.size() > 0) {
            for (Cart item : cartList) {
                query = "select * from products2 where id=?";
                pst = this.con.prepareStatement(query);
                pst.setInt(1, item.getId());
                rs = pst.executeQuery();
                while (rs.next()) {
                    Cart row = new Cart();
                    row.setId(rs.getInt("id"));
                    row.setName(rs.getString("name"));
                    row.setCategory(rs.getString("category"));
                    row.setPrice(rs.getDouble("price")*item.getQuantity());
                    row.setQuantity(item.getQuantity());
                    book.add(row);
                }
            }
        }
    }
}

```

```

        }
    }

    } catch (SQLException e) {
        e.printStackTrace();
        System.out.println(e.getMessage());
    }
    return book;
}
}

```

5. ProductDao3.java

```

package cn.techtutorial.dao;

import java.sql.*;
import java.util.*;
import cn.techtutorial.model.Cart;
import cn.techtutorial.model.*;

public class ProductDao3 {
    private Connection con;

    private String query;
    private PreparedStatement pst;
    private ResultSet rs;

    public ProductDao3(Connection con) {
        super();
        this.con = con;
    }
}

```



```

    public List<Product3> getAllProducts3() {
List<Product3> book = new ArrayList<>();
try {
    query = "select * from products3";
    pst = this.con.prepareStatement(query);
    rs = pst.executeQuery();
    while (rs.next()) {
        Product3 row = new Product3();
        row.setId(rs.getInt("id"));
        row.setName(rs.getString("name"));
        row.setCategory(rs.getString("category"));
        row.setPrice(rs.getDouble("price"));
        row.setImage(rs.getString("image"));
        book.add(row);
    }
} catch (SQLException e) {
    e.printStackTrace();
    System.out.println(e.getMessage());
}
return book;
}

    public Product3 getSingleProduct3(int id) {
        Product3 row = null;
        try {
            query = "select * from products3 where id=? ";
            pst = this.con.prepareStatement(query);
            pst.setInt(1, id);
            ResultSet rs = pst.executeQuery();

```

```

        while (rs.next()) {
            row = new Product3();
            row.setId(rs.getInt("id"));
            row.setName(rs.getString("name"));
            row.setCategory(rs.getString("category"));
            row.setPrice(rs.getDouble("price"));
            row.setImage(rs.getString("image"));
        }
    } catch (Exception e) {
        e.printStackTrace();
        System.out.println(e.getMessage());
    }
    return row;
}

public double getTotalCartPrice(ArrayList<Cart> cartList) {
    double sum = 0;
    try {
        if (cartList.size() > 0) {
            for (Cart item : cartList) {
                query = "select price from products3 where id=?";
                pst = this.con.prepareStatement(query);
                pst.setInt(1, item.getId());
                rs = pst.executeQuery();
                while (rs.next()) {
                    sum+=rs.getDouble("price")*item.getQuantity();
                }
            }
        }
    }

    } catch (SQLException e) {

```

```

        e.printStackTrace();
        System.out.println(e.getMessage());
    }
    return sum;
}

public List<Cart> getCartProducts3(ArrayList<Cart> cartList) {
    List<Cart> book = new ArrayList<>();
    try {
        if (cartList.size() > 0) {
            for (Cart item : cartList) {
                query = "select * from products3 where id=?";
                pst = this.con.prepareStatement(query);
                pst.setInt(1, item.getId());
                rs = pst.executeQuery();
                while (rs.next()) {
                    Cart row = new Cart();
                    row.setId(rs.getInt("id"));
                    row.setName(rs.getString("name"));
                    row.setCategory(rs.getString("category"));
                    row.setPrice(rs.getDouble("price")*item.getQuantity());
                    row.setQuantity(item.getQuantity());
                    book.add(row);
                }
            }
        }
    } catch (SQLException e) {
        e.printStackTrace();
        System.out.println(e.getMessage());
    }
    return book;
}

```

```

    }
}

```

6. ProductDao4.java

```

package cn.techtutorial.dao;

import java.sql.*;
import java.util.*;
import cn.techtutorial.model.Cart;
import cn.techtutorial.model.*;

public class ProductDao4 {
    private Connection con;
    private String query;
    private PreparedStatement pst;
    private ResultSet rs;

    public ProductDao4(Connection con) {
        super();
        this.con = con;
    }

    public List<Product4> getAllProducts4() {
        List<Product4> book = new ArrayList<>();
        try {
            query = "select * from products4";
            pst = this.con.prepareStatement(query);
            rs = pst.executeQuery();
            while (rs.next()) {
                Product4 row = new Product4();
                row.setId(rs.getInt("id"));
                row.setName(rs.getString("name"));
                row.setCategory(rs.getString("category"));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return book;
    }
}

```

```

        row.setPrice(rs.getDouble("price"));
        row.setImage(rs.getString("image"));
        book.add(row);
    }
} catch (SQLException e) {
    e.printStackTrace();
    System.out.println(e.getMessage());
}
return book;
}

public Product4 getSingleProduct4(int id) {
    Product4 row = null;
    try {
        query = "select * from products4 where id=? ";
        pst = this.con.prepareStatement(query);
        pst.setInt(1, id);
        ResultSet rs = pst.executeQuery();
        while (rs.next()) {
            row = new Product4();
            row.setId(rs.getInt("id"));
            row.setName(rs.getString("name"));
            row.setCategory(rs.getString("category"));
            row.setPrice(rs.getDouble("price"));
            row.setImage(rs.getString("image"));
        }
    } catch (Exception e) {
        e.printStackTrace();
        System.out.println(e.getMessage());
    }
    return row;
}

```

```

    }

    public double getTotalCartPrice(ArrayList<Cart> cartList) {
double sum = 0;
try {
    if (cartList.size() > 0) {
        for (Cart item : cartList) {
            query = "select price from products4 where id=?";
            pst = this.con.prepareStatement(query);
            pst.setInt(1, item.getId());
            rs = pst.executeQuery();
            while (rs.next()) {
                sum+=rs.getDouble("price")*item.getQuantity();
            }
        }
    }
}

    } catch (SQLException e) {
        e.printStackTrace();
        System.out.println(e.getMessage());
    }
    return sum;
}

public List<Cart> getCartProducts4(ArrayList<Cart> cartList) {
    List<Cart> book = new ArrayList<>();
    try {
        if (cartList.size() > 0) {
            for (Cart item : cartList) {
                query = "select * from products4 where id=?";
                pst = this.con.prepareStatement(query);

```

```

        pst.setInt(1, item.getId());
        rs = pst.executeQuery();
        while (rs.next()) {
            Cart row = new Cart();
            row.setId(rs.getInt("id"));
            row.setName(rs.getString("name"));
            row.setCategory(rs.getString("category"));
            row.setPrice(rs.getDouble("price")*item.getQuantity());
            row.setQuantity(item.getQuantity());
            book.add(row);
        }
    }
}
} catch (SQLException e) {
    e.printStackTrace();
    System.out.println(e.getMessage());
}
return book;
}
}

```

7. ProductDao5.java

```

package cn.techtutorial.dao;
import java.sql.*;
import java.util.*;
import cn.techtutorial.model.Cart;
import cn.techtutorial.model.*;
public class ProductDao5 {
    private Connection con;

```

```

        private String query;
private PreparedStatement pst;
private ResultSet rs;

        public ProductDao5(Connection con) {
            super();
            this.con = con;
        }

        public List<Product5> getAllProducts5() {
List<Product5> book = new ArrayList<>();
        try {
            query = "select * from products5";
            pst = this.con.prepareStatement(query);
            rs = pst.executeQuery();
            while (rs.next()) {
                Product5 row = new Product5();
                row.setId(rs.getInt("id"));
                row.setName(rs.getString("name"));
                row.setCategory(rs.getString("category"));
                row.setPrice(rs.getDouble("price"));
                row.setImage(rs.getString("image"));
                book.add(row);
            }
        } catch (SQLException e) {
            e.printStackTrace();
            System.out.println(e.getMessage());
        }
        return book;
    }

```



```

public Product5 getSingleProduct5(int id) {
    Product5 row = null;
    try {
        query = "select * from products5 where id=? ";
        pst = this.con.prepareStatement(query);
        pst.setInt(1, id);
        ResultSet rs = pst.executeQuery();
        while (rs.next()) {
            row = new Product5();
            row.setId(rs.getInt("id"));
            row.setName(rs.getString("name"));
            row.setCategory(rs.getString("category"));
            row.setPrice(rs.getDouble("price"));
            row.setImage(rs.getString("image"));
        }
    } catch (Exception e) {
        e.printStackTrace();
        System.out.println(e.getMessage());
    }
    return row;
}

public double getTotalCartPrice(ArrayList<Cart> cartList) {
    double sum = 0;
    try {
        if (cartList.size() > 0) {
            for (Cart item : cartList) {
                query = "select price from products5 where id=?";
                pst = this.con.prepareStatement(query);
                pst.setInt(1, item.getId());
            }
        }
    }
}

```

```

        rs = pst.executeQuery();
        while (rs.next()) {
            sum+=rs.getDouble("price")*item.getQuantity();
        }
    }
}

} catch (SQLException e) {
    e.printStackTrace();
    System.out.println(e.getMessage());
}

return sum;
}

public List<Cart> getCartProducts5(ArrayList<Cart> cartList) {
    List<Cart> book = new ArrayList<>();
    try {
        if (cartList.size() > 0) {
            for (Cart item : cartList) {
                query = "select * from products5 where id=?";
                pst = this.con.prepareStatement(query);
                pst.setInt(1, item.getId());
                rs = pst.executeQuery();
                while (rs.next()) {
                    Cart row = new Cart();
                    row.setId(rs.getInt("id"));
                    row.setName(rs.getString("name"));
                    row.setCategory(rs.getString("category"));
                    row.setPrice(rs.getDouble("price")*item.getQuantity());
                    row.setQuantity(item.getQuantity());
                    book.add(row);
                }
            }
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

```

```

        }
    }
}

} catch (SQLException e) {
    e.printStackTrace();
    System.out.println(e.getMessage());
}
return book;
}
}

```

8. ProductDao6.java

```

package cn.techtutorial.dao;

import java.sql.*;
import java.util.*;
import cn.techtutorial.model.Cart;
import cn.techtutorial.model.*;

public class ProductDao6 {
    private Connection con;
    private String query;
    private PreparedStatement pst;
    private ResultSet rs;

    public ProductDao6(Connection con) {
        super();
        this.con = con;
    }
}

```

```

    public List<Product6> getAllProducts6() {
List<Product6> book = new ArrayList<>();
try {
    query = "select * from products6";
    pst = this.con.prepareStatement(query);
    rs = pst.executeQuery();
    while (rs.next()) {
        Product6 row = new Product6();
        row.setId(rs.getInt("id"));
        row.setName(rs.getString("name"));
        row.setCategory(rs.getString("category"));
        row.setPrice(rs.getDouble("price"));
        row.setImage(rs.getString("image"));
        book.add(row);
    }
} catch (SQLException e) {
    e.printStackTrace();
    System.out.println(e.getMessage());
}
return book;
}

    public Product6 getSingleProduct6(int id) {
        Product6 row = null;
        try {
            query = "select * from products6 where id=? ";
            pst = this.con.prepareStatement(query);
            pst.setInt(1, id);
            ResultSet rs = pst.executeQuery();
            while (rs.next()) {
                row = new Product6();

```

```

        row.setId(rs.getInt("id"));
        row.setName(rs.getString("name"));
        row.setCategory(rs.getString("category"));
        row.setPrice(rs.getDouble("price"));
        row.setImage(rs.getString("image"));
    }
} catch (Exception e) {
    e.printStackTrace();
    System.out.println(e.getMessage());
}
return row;
}

public double getTotalCartPrice(ArrayList<Cart> cartList) {
double sum = 0;
try {
    if (cartList.size() > 0) {
        for (Cart item : cartList) {
            query = "select price from products6 where id=?";
            pst = this.con.prepareStatement(query);
            pst.setInt(1, item.getId());
            rs = pst.executeQuery();
            while (rs.next()) {
                sum+=rs.getDouble("price")*item.getQuantity();
            }
        }
    }
} catch (SQLException e) {
    e.printStackTrace();
    System.out.println(e.getMessage());
}
}

```

```

        return sum;
    }

    public List<Cart> getCartProducts6(ArrayList<Cart> cartList) {
        List<Cart> book = new ArrayList<>();
        try {
            if (cartList.size() > 0) {
                for (Cart item : cartList) {
                    query = "select * from products6 where id=?";
                    pst = this.con.prepareStatement(query);
                    pst.setInt(1, item.getId());
                    rs = pst.executeQuery();
                    while (rs.next()) {
                        Cart row = new Cart();
                        row.setId(rs.getInt("id"));
                        row.setName(rs.getString("name"));
                        row.setCategory(rs.getString("category"));
                        row.setPrice(rs.getDouble("price")*item.getQuantity());
                        row.setQuantity(item.getQuantity());
                        book.add(row);
                    }
                }
            }
        } catch (SQLException e) {
            e.printStackTrace();
            System.out.println(e.getMessage());
        }
        return book;
    }

```

```
}
```

9. ProductDao7.java

```
package cn.techtutorial.dao;

import java.sql.*;
import java.util.*;
import cn.techtutorial.model.Cart;
import cn.techtutorial.model.*;

public class ProductDao7 {

    private Connection con;

    private String query;

    private PreparedStatement pst;

    private ResultSet rs;

    public ProductDao7(Connection con) {

        super();

        this.con = con;

    }

    public List<Product7> getAllProducts7() {

        List<Product7> book = new ArrayList<>();

        try {

            query = "select * from products7";

            pst = this.con.prepareStatement(query);

            rs = pst.executeQuery();

            while (rs.next()) {

                Product7 row = new Product7();

                row.setId(rs.getInt("id"));

                row.setName(rs.getString("name"));

                row.setCategory(rs.getString("category"));

                row.setPrice(rs.getDouble("price"));

            }

        } catch (SQLException e) {

            e.printStackTrace();

        }

        return book;

    }

}
```

```

        row.setImage(rs.getString("image"));
        book.add(row);
    }
} catch (SQLException e) {
    e.printStackTrace();
    System.out.println(e.getMessage());
}
return book;
}

public Product7 getSingleProduct7(int id) {
    Product7 row = null;
    try {
        query = "select * from products7 where id=? ";
        pst = this.con.prepareStatement(query);
        pst.setInt(1, id);
        ResultSet rs = pst.executeQuery();
        while (rs.next()) {
            row = new Product7();
            row.setId(rs.getInt("id"));
            row.setName(rs.getString("name"));
            row.setCategory(rs.getString("category"));
            row.setPrice(rs.getDouble("price"));
            row.setImage(rs.getString("image"));
        }
    } catch (Exception e) {
        e.printStackTrace();
        System.out.println(e.getMessage());
    }
    return row;
}

```



```

    public double getTotalCartPrice(ArrayList<Cart> cartList) {
double sum = 0;
try {
    if (cartList.size() > 0) {
        for (Cart item : cartList) {
            query = "select price from products7 where id=?";
            pst = this.con.prepareStatement(query);
            pst.setInt(1, item.getId());
            rs = pst.executeQuery();
            while (rs.next()) {
                sum+=rs.getDouble("price")*item.getQuantity();
            }
        }
    }
} catch (SQLException e) {
    e.printStackTrace();
    System.out.println(e.getMessage());
}
return sum;
}

public List<Cart> getCartProducts7(ArrayList<Cart> cartList) {
List<Cart> book = new ArrayList<>();
try {
    if (cartList.size() > 0) {
        for (Cart item : cartList) {
            query = "select * from products7 where id=?";
            pst = this.con.prepareStatement(query);
            pst.setInt(1, item.getId());
            rs = pst.executeQuery();

```

```

        while (rs.next()) {
            Cart row = new Cart();
            row.setId(rs.getInt("id"));
            row.setName(rs.getString("name"));
            row.setCategory(rs.getString("category"));
            row.setPrice(rs.getDouble("price")*item.getQuantity());
            row.setQuantity(item.getQuantity());
            book.add(row);
        }
    }
}
} catch (SQLException e) {
    e.printStackTrace();
    System.out.println(e.getMessage());
}
return book;
}
}

```

10. ProductDao8.java

```

package cn.techtutorial.dao;

import java.sql.*;
import java.util.*;
import cn.techtutorial.model.Cart;
import cn.techtutorial.model.*;

public class ProductDao8 {
    private Connection con;
    private String query;
    private PreparedStatement pst;

```

```

private ResultSet rs;

    public ProductDao8(Connection con) {
        super();
        this.con = con;
    }

    public List<Product8> getAllProducts8() {
        List<Product8> book = new ArrayList<>();
        try {
            query = "select * from products8";
            pst = this.con.prepareStatement(query);
            rs = pst.executeQuery();
            while (rs.next()) {
                Product8 row = new Product8();
                row.setId(rs.getInt("id"));
                row.setName(rs.getString("name"));
                row.setCategory(rs.getString("category"));
                row.setPrice(rs.getDouble("price"));
                row.setImage(rs.getString("image"));
                book.add(row);
            }
        } catch (SQLException e) {
            e.printStackTrace();
            System.out.println(e.getMessage());
        }
        return book;
    }

    public Product8 getSingleProduct8(int id) {
        Product8 row = null;
        try {
            query = "select * from products8 where id=? ";

```

```

        pst = this.con.prepareStatement(query);
        pst.setInt(1, id);
        ResultSet rs = pst.executeQuery();
        while (rs.next()) {
            row = new Product8();
            row.setId(rs.getInt("id"));
            row.setName(rs.getString("name"));
            row.setCategory(rs.getString("category"));
            row.setPrice(rs.getDouble("price"));
            row.setImage(rs.getString("image"));
        }
    } catch (Exception e) {
        e.printStackTrace();
        System.out.println(e.getMessage());
    }
    return row;
}

public double getTotalCartPrice(ArrayList<Cart> cartList) {
double sum = 0;
try {
    if (cartList.size() > 0) {
        for (Cart item : cartList) {
            query = "select price from products8 where id=?";
            pst = this.con.prepareStatement(query);
            pst.setInt(1, item.getId());
            rs = pst.executeQuery();
            while (rs.next()) {
                sum+=rs.getDouble("price")*item.getQuantity();
            }
        }
    }
}

```

```

        }
    }
} catch (SQLException e) {
    e.printStackTrace();
    System.out.println(e.getMessage());
}
return sum;
}

public List<Cart> getCartProducts8(ArrayList<Cart> cartList) {
    List<Cart> book = new ArrayList<>();
    try {
        if (cartList.size() > 0) {
            for (Cart item : cartList) {
                query = "select * from products8 where id=?";
                pst = this.con.prepareStatement(query);
                pst.setInt(1, item.getId());
                rs = pst.executeQuery();
                while (rs.next()) {
                    Cart row = new Cart();
                    row.setId(rs.getInt("id"));
                    row.setName(rs.getString("name"));
                    row.setCategory(rs.getString("category"));
                    row.setPrice(rs.getDouble("price")*item.getQuantity());
                    row.setQuantity(item.getQuantity());
                    book.add(row);
                }
            }
        }
    }
}

```

```

    } catch (SQLException e) {
        e.printStackTrace();
        System.out.println(e.getMessage());
    }
    return book;
}
}

```

11. ProductDao9.java

```

package cn.techtutorial.dao;
import java.sql.*;
import java.util.*;
import cn.techtutorial.model.Cart;
import cn.techtutorial.model.*;
public class ProductDao9 {
    private Connection con;
    private String query;
    private PreparedStatement pst;
    private ResultSet rs;

    public ProductDao9(Connection con) {
        super();
        this.con = con;
    }

    public List<Product9> getAllProducts9() {
        List<Product9> book = new ArrayList<>();
        try {
            query = "select * from products9";
            pst = this.con.prepareStatement(query);
            rs = pst.executeQuery();

```

```

while (rs.next()) {
    Product9 row = new Product9();
    row.setId(rs.getInt("id"));
    row.setName(rs.getString("name"));
    row.setCategory(rs.getString("category"));
    row.setPrice(rs.getDouble("price"));
    row.setImage(rs.getString("image"));
    book.add(row);
}
} catch (SQLException e) {
    e.printStackTrace();
    System.out.println(e.getMessage());
}
return book;
}

public Product9 getSingleProduct9(int id) {
    Product9 row = null;
    try {
        query = "select * from products9 where id=? ";
        pst = this.con.prepareStatement(query);
        pst.setInt(1, id);
        ResultSet rs = pst.executeQuery();
        while (rs.next()) {
            row = new Product9();
            row.setId(rs.getInt("id"));
            row.setName(rs.getString("name"));
            row.setCategory(rs.getString("category"));
            row.setPrice(rs.getDouble("price"));
            row.setImage(rs.getString("image"));
        }
    }
}

```

```

    }
    } catch (Exception e) {
        e.printStackTrace();
        System.out.println(e.getMessage());
    }
    return row;
}

public double getTotalCartPrice(ArrayList<Cart> cartList) {
    double sum = 0;
    try {
        if (cartList.size() > 0) {
            for (Cart item : cartList) {
                query = "select price from products9 where id=?";
                pst = this.con.prepareStatement(query);
                pst.setInt(1, item.getId());
                rs = pst.executeQuery();
                while (rs.next()) {
                    sum+=rs.getDouble("price")*item.getQuantity();
                }
            }
        }
    } catch (SQLException e) {
        e.printStackTrace();
        System.out.println(e.getMessage());
    }
    return sum;
}

public List<Cart> getCartProducts9(ArrayList<Cart> cartList) {
    List<Cart> book = new ArrayList<>();
    try {

```



```

    if (cartList.size() > 0) {
        for (Cart item : cartList) {
            query = "select * from products9 where id=?";
            pst = this.con.prepareStatement(query);
            pst.setInt(1, item.getId());
            rs = pst.executeQuery();
            while (rs.next()) {
                Cart row = new Cart();
                row.setId(rs.getInt("id"));
                row.setName(rs.getString("name"));
                row.setCategory(rs.getString("category"));
                row.setPrice(rs.getDouble("price")*item.getQuantity());
                row.setQuantity(item.getQuantity());
                book.add(row);
            }
        }
    }
} catch (SQLException e) {
    e.printStackTrace();
    System.out.println(e.getMessage());
}
return book;
}
}

```

12. UserDao.java

```

package cn.techtutorial.dao;
import java.sql.*;
import cn.techtutorial.model.*;

```

```

public class UserDao {
    private Connection con;
    private String query;
    private PreparedStatement pst;
    private ResultSet rs;

    public UserDao(Connection con) {
        this.con = con;
    }

    public User userLogin(String email, String password) {
        User user = null;
        try {
            query = "select * from users where email=? and password=?";
            pst = this.con.prepareStatement(query);
            pst.setString(1, email);
            pst.setString(2, password);
            rs = pst.executeQuery();
            if(rs.next()){
                user = new User();
                user.setId(rs.getInt("id"));
                user.setName(rs.getString("name"));
                user.setEmail(rs.getString("email"));
            }
        } catch (SQLException e) {
            e.printStackTrace();
            System.out.print(e.getMessage());
        }
        return user;
    }
}

```

Model

1. Cart.java

```
package cn.techtutorial.model;

public class Cart extends Product{
    private int quantity;
    public Cart() {
    }
    public int getQuantity() {
        return quantity;
    }
    public void setQuantity(int quantity) {
        this.quantity = quantity;
    }
}
```

2. ModelEx.java

```
package cn.techtutorial.model;

import java.sql.Connection;
import java.sql.DriverManager;
import cn.techtutorial.connection.*;
import java.sql.PreparedStatement;
import java.sql.SQLException;

public class ModelEx {
    private String id;
    private String name;
    private String email;
    private String password;
```

```
private String confirmPassword;
private Connection connection;
private PreparedStatement pstmt;
public String getId() {
    return id;
}
public void setId(String id) {
    this.id = id;
}
public String getName() {
    return name;
}
public void setName(String name) {
    this.name = name;
}
public String getEmail() {
    return email;
}
public void setEmail(String email) {
    this.email = email;
}
public String getPassword() {
    return password;
}
public void setPassword(String password) {
    this.password = password;
}
public String getConfirmPassword() {
    return confirmPassword;
}
```

```

public void setConfirmPassword(String confirmPassword) {
    this.confirmPassword = confirmPassword;
}

public ModelEx() throws Exception
{
    Class.forName("com.mysql.jdbc.Driver");
    connection =
DriverManager.getConnection("jdbc:mysql://localhost:3306/ecommerce_cart", "root", "root");
    System.out.println("Loading the Driver and Establishing the Connection Completed");
}

public boolean register() throws SQLException
{
    String s="insert into users values(?,?,?,?,?)";
        pstmt = connection.prepareStatement(s);
        pstmt.setString(1, id);
        pstmt.setString(2, name);
        pstmt.setString(3, email);
        pstmt.setString(4, password);
        pstmt.setString(5, confirmPassword);
        int x = pstmt.executeUpdate();
        if(x>0)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
}

```

3. Order.java

```
package cn.techtutorial.model;

public class Order extends Product{

    private int orderId;

    private int uid;

    private int qunatity;

    private String date;

    public Order() {

    }

    public Order(int orderId, int uid, int qunatity, String date) {

        super();

        this.orderId = orderId;

        this.uid = uid;

        this.qunatity = qunatity;

        this.date = date;

    }

    public Order(int uid, int qunatity, String date) {

        super();

        this.uid = uid;

        this.qunatity = qunatity;

        this.date = date;

    }

    public int getOrderId() {

        return orderId;

    }

    public void setOrderId(int orderId) {

        this.orderId = orderId;

    }

}
```

```
public int getUid() {  
    return uid;  
}  
public void setUid(int uid) {  
    this.uid = uid;  
}  
public int getQunatity() {  
    return qunatity;  
}  
public void setQunatity(int qunatity) {  
    this.qunatity = qunatity;  
}  
public String getDate() {  
    return date;  
}  
public void setDate(String date) {  
    this.date = date;  
}  
}
```

4. Product.java

```
package cn.techtutorial.model;  
public class Product {  
    private int id;  
    private String name;  
    private String category;  
    private Double price;  
    private String image;  
    public Product() {
```

```
}  
  
public Product(int id, String name, String category, Double price, String image) {  
    super();  
    this.id = id;  
    this.name = name;  
    this.category = category;  
    this.price = price;  
    this.image = image;  
}  
  
public int getId() {  
    return id;  
}  
  
public void setId(int id) {  
    this.id = id;  
}  
  
public String getName() {  
    return name;  
}  
  
public void setName(String name) {  
    this.name = name;  
}  
  
public String getCategory() {  
    return category;  
}  
  
public void setCategory(String category) {  
    this.category = category;  
}  
  
public Double getPrice() {  
    return price;  
}
```



```

    }

    public void setPrice(Double price) {
        this.price = price;
    }

    public String getImage() {
        return image;
    }

    public void setImage(String image) {
        this.image = image;
    }

    @Override
    public String toString() {
        return "Product [id=" + id + ", name=" + name + ", category=" + category + ",
price=" + price + ", image="
            + image + "]";
    }
}

```

5. Product2.java

```

package cn.techtutorial.model;

public class Product2 {
    private int id;
    private String name;
    private String category;
    private Double price;
    private String image;

    public Product2() {
    }
}

```

```
public Product2(int id, String name, String category, Double price, String image) {  
    super();  
    this.id = id;  
    this.name = name;  
    this.category = category;  
    this.price = price;  
    this.image = image;  
}  
  
public int getId() {  
    return id;  
}  
  
public void setId(int id) {  
    this.id = id;  
}  
  
public String getName() {  
    return name;  
}  
  
public void setName(String name) {  
    this.name = name;  
}  
  
public String getCategory() {  
    return category;  
}  
  
public void setCategory(String category) {  
    this.category = category;  
}  
  
public Double getPrice() {  
    return price;  
}  
  
public void setPrice(Double price) {
```

```

        this.price = price;
    }
    public String getImage() {
        return image;
    }
    public void setImage(String image) {
        this.image = image;
    }
    @Override
    public String toString() {
        return "Product [id=" + id + ", name=" + name + ", category=" + category + ",
price=" + price + ", image="
            + image + "]";
    }
}

```

6. Product3.java

```

package cn.techtutorial.model;

public class Product3 {
    private int id;
    private String name;
    private String category;
    private Double price;
    private String image;
    public Product3() {
    }

    public Product3(int id, String name, String category, Double price, String image) {
        super();
    }
}

```

```
        this.id = id;

        this.name = name;

        this.category = category;

        this.price = price;

        this.image = image;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getCategory() {
        return category;
    }

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

    public Double getPrice() {
        return price;
    }

    public void setPrice(Double price) {
        this.price = price;
    }
}
```

```

    }

    public String getImage() {
        return image;
    }

    public void setImage(String image) {
        this.image = image;
    }

    @Override
    public String toString() {
        return "Product [id=" + id + ", name=" + name + ", category=" + category + ",
price=" + price + ", image="
            + image + "]";
    }
}

```

7. Product4.java

```

package cn.techtutorial.model;

public class Product4 {
    private int id;
    private String name;
    private String category;
    private Double price;
    private String image;
    public Product4() {
    }

    public Product4(int id, String name, String category, Double price, String image) {
        super();
        this.id = id;
        this.name = name;
        this.category = category;
    }
}

```

```
        this.price = price;
        this.image = image;
    }
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getCategory() {
        return category;
    }
    public void setCategory(String category) {
        this.category = category;
    }
    public Double getPrice() {
        return price;
    }
    public void setPrice(Double price) {
        this.price = price;
    }

    public String getImage() {
        return image;
    }
}
```

```

    }

    public void setImage(String image) {
        this.image = image;
    }

    @Override
    public String toString() {
        return "Product [id=" + id + ", name=" + name + ", category=" + category + ",
price=" + price + ", image="
                + image + "]";
    }
}

```

8. Product5.java

```

package cn.techtutorial.model;

public class Product5 {
    private int id;
    private String name;
    private String category;
    private Double price;
    private String image;
    public Product5() {
    }

    public Product5(int id, String name, String category, Double price, String image) {
        super();
        this.id = id;
        this.name = name;
        this.category = category;
        this.price = price;
        this.image = image;
    }
}

```

```
public int getId() {  
    return id;  
}  
public void setId(int id) {  
    this.id = id;  
}  
public String getName() {  
    return name;  
}  
public void setName(String name) {  
    this.name = name;  
}  
public String getCategory() {  
    return category;  
}  
public void setCategory(String category) {  
    this.category = category;  
}  
public Double getPrice() {  
    return price;  
}  
public void setPrice(Double price) {  
    this.price = price;  
}  
public String getImage() {  
    return image;  
}  
public void setImage(String image) {  
    this.image = image;  
}
```



```

@Override

public String toString() {
    return "Product [id=" + id + ", name=" + name + ", category=" + category + ",
price=" + price + ", image="
        + image + "]";
}
}

```

9. Product6.java

```

package cn.techtutorial.model;

public class Product6 {
    private int id;
    private String name;
    private String category;
    private Double price;
    private String image;
    public Product6() {
    }

    public Product6(int id, String name, String category, Double price, String image) {
        super();
        this.id = id;
        this.name = name;
        this.category = category;
        this.price = price;
        this.image = image;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {

```

```

        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getCategory() {
        return category;
    }
    public void setCategory(String category) {
        this.category = category;
    }
    public Double getPrice() {
        return price;
    }
    public void setPrice(Double price) {
        this.price = price;
    }
    public String getImage() {
        return image;
    }
    public void setImage(String image) {
        this.image = image;
    }
    @Override
    public String toString() {
        return "Product [id=" + id + ", name=" + name + ", category=" + category + ",
price=" + price + ", image="
        + image + "]";
    }

```

```

    }
}

```

10. Product7.java

```

package cn.techtutorial.model;

public class Product7 {
    private int id;
    private String name;
    private String category;
    private Double price;
    private String image;
    public Product7() {
    }
    public Product7(int id, String name, String category, Double price, String image) {
        super();
        this.id = id;
        this.name = name;
        this.category = category;
        this.price = price;
        this.image = image;
    }
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
}

```

```

    }

    public void setName(String name) {
        this.name = name;
    }

    public String getCategory() {
        return category;
    }

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

    public Double getPrice() {
        return price;
    }

    public void setPrice(Double price) {
        this.price = price;
    }

    public String getImage() {
        return image;
    }

    public void setImage(String image) {
        this.image = image;
    }

    @Override
    public String toString() {
        return "Product [id=" + id + ", name=" + name + ", category=" + category + ",
price=" + price + ", image="
            + image + "]";
    }
}

```

11. Product8.java

```
package cn.techtutorial.model;

public class Product8 {
    private int id;
    private String name;
    private String category;
    private Double price;
    private String image;
    public Product8() {
    }
    public Product8(int id, String name, String category, Double price, String image) {
        super();
        this.id = id;
        this.name = name;
        this.category = category;
        this.price = price;
        this.image = image;
    }
    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
}
```

```

    }

    public String getCategory() {
        return category;
    }

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

    public Double getPrice() {
        return price;
    }

    public void setPrice(Double price) {
        this.price = price;
    }

    public String getImage() {
        return image;
    }

    public void setImage(String image) {
        this.image = image;
    }

    @Override
    public String toString() {
        return "Product [id=" + id + ", name=" + name + ", category=" + category + ",
price=" + price + ", image="
        + image + "]";
    }
}

```

12. Product9.java

```

package cn.techtutorial.model;

public class Product9 {

```

```
private int id;
private String name;
private String category;
private Double price;
private String image;
public Product9() {
}
public Product9(int id, String name, String category, Double price, String image) {
    super();
    this.id = id;
    this.name = name;
    this.category = category;
    this.price = price;
    this.image = image;
}
public int getId() {
    return id;
}
public void setId(int id) {
    this.id = id;
}
public String getName() {
    return name;
}
public void setName(String name) {
    this.name = name;
}
public String getCategory() {
    return category;
}
```

```

    public void setCategory(String category) {
        this.category = category;
    }
    public Double getPrice() {
        return price;
    }
    public void setPrice(Double price) {
        this.price = price;
    }
    public String getImage() {
        return image;
    }
    public void setImage(String image) {
        this.image = image;
    }
    @Override
    public String toString() {
        return "Product [id=" + id + ", name=" + name + ", category=" + category + ",
price=" + price + ", image="
            + image + "]";
    }
}

```

13. User.java

```

package cn.techtutorial.model;

public class User{
    private int id;
    private String name;

```



```
private String email;
private String password;
public User() {
}
public User(int id, String name, String email, String password) {
    this.id = id;
    this.name = name;
    this.email = email;
    this.password = password;
}
public int getId() {
    return id;
}
public void setId(int id) {
    this.id = id;
}
public String getName() {
    return name;
}
public void setName(String name) {
    this.name = name;
}
public String getEmail() {
    return email;
}
public void setEmail(String email) {
    this.email = email;
}
public String getPassword() {
    return password;
}
```

```
}  
  
public void setPassword(String password) {  
    this.password = password;  
}  
  
@Override  
public String toString() {  
    return "User [id=" + id + ", name=" + name + ", email=" + email + ",  
password=" + password + "];"  
}  
}
```

Servlet

1. AddToCartServlet.java

```
package cn.techtutorial.servlet;  
  
import java.io.IOException;
```

```

import java.io.PrintWriter;
import java.util.ArrayList;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import cn.techtutorial.model.*;

@WebServlet(name = "AddToCartServlet", urlPatterns = "/add-to-cart")
public class AddToCartServlet extends HttpServlet {

    private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

        response.setContentType("text/html;charset=UTF-8");

        try (PrintWriter out = response.getWriter()) {
            // out.print("add to cart servlet");

            ArrayList<Cart> cartList = new ArrayList<>();

            int id = Integer.parseInt(request.getParameter("id"));

            Cart cm = new Cart();

            cm.setId(id);

            cm.setQuantity(1);

            HttpSession session = request.getSession();

            ArrayList<Cart> cart_list = (ArrayList<Cart>) session.getAttribute("cart-list");

            if (cart_list == null) {

                cartList.add(cm);

                session.setAttribute("cart-list", cartList);

                response.sendRedirect("index.jsp");

            } else {

                cartList = cart_list;

                boolean exist = false;

```

```

        for (Cart c : cart_list) {
            if (c.getId() == id) {
                exist = true;

                out.println("<h3 style='color:crimson; text-align: center'>Item Already in Cart.
<a href='cart.jsp'>GO to Cart Page</a></h3>");
            }
        }
        if (!exist) {
            cartList.add(cm);
            response.sendRedirect("index.jsp");
        }
    }
}
}
}
}

```

2. CancelOrderServlet

```

package cn.techtutorial.servlet;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.SQLException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import cn.techtutorial.connection.DbCon;
import cn.techtutorial.dao.OrderDao;

@WebServlet("/cancel-order")

```

```

public class CancelOrderServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
        try(PrintWriter out = response.getWriter()) {
            String id = request.getParameter("id");
            if(id != null) {
                OrderDao orderDao = new OrderDao(DbCon.getConnection());
                orderDao.cancelOrder(Integer.parseInt(id));
            }
            response.sendRedirect("orders.jsp");
        } catch (ClassNotFoundException|SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}

```

3. CheckoutServlet.java

```

package cn.techtutorial.servlet;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;

```

```

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import cn.techtutorial.connection.DbCon;
import cn.techtutorial.dao.OrderDao;
import cn.techtutorial.model.*;
@WebServlet("/cart-check-out")
public class CheckOutServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
        try(PrintWriter out = response.getWriter()){
            SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-
dd");
            Date date = new Date();

            ArrayList<Cart> cart_list = (ArrayList<Cart>)
request.getSession().getAttribute("cart-list");

            User auth = (User) request.getSession().getAttribute("auth");
            if(cart_list != null && auth!=null) {
                for(Cart c:cart_list) {
                    Order order = new Order();
                    order.setId(c.getId());
                    order.setUid(auth.getId());
                    order.setQunatity(c.getQuantity());
                    order.setDate(formatter.format(date));
                    OrderDao oDao = new
OrderDao(DbCon.getConnection());
                    boolean result = oDao.insertOrder(order);
                    if(!result) break;
                }
                cart_list.clear();
                response.sendRedirect("orders.jsp");
            }else {

```

```

        if(auth == null) response.sendRedirect("login.jsp");
        response.sendRedirect("cart.jsp");
    }
} catch (Exception e) {
    e.printStackTrace();
}
}

protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    // TODO Auto-generated method stub
    doGet(request, response);
}
}

```

4. LoginServlet.java

```

package cn.techtutorial.servlet;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.SQLException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import cn.techtutorial.connection.DbCon;
import cn.techtutorial.dao.*;
import cn.techtutorial.model.*;
@WebServlet("/user-login")
public class LoginServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

```

```

protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {
        String email = request.getParameter("login-email");
        String password = request.getParameter("login-password");
        UserDao udao = new UserDao(DbCon.getConnection());
        User user = udao.userLogin(email, password);
        if (user != null) {
            request.getSession().setAttribute("auth", user);
            System.out.print("user logged in");
            response.sendRedirect("index.jsp");
        } else {
            out.println("there is no user");
        }
    } catch (ClassNotFoundException|SQLException e) {
        e.printStackTrace();
    }
}

```

5. LogoutServlet.java

```

package cn.techtutorial.servlet;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.SQLException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;

```



```

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
@WebServlet("/log-out")
public class LogoutServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
    ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            if(request.getSession().getAttribute("auth")!=null) {
                request.getSession().removeAttribute("auth");
                response.sendRedirect("login.jsp");
            }else {
                response.sendRedirect("index.jsp");
            }
        }
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
        // TODO Auto-generated method stub
        doGet(request, response);
    }
}

```

6. OrderNowServlet.java

```

package cn.techtutorial.servlet;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.SQLException;
import java.util.*;

```

```

import java.text.SimpleDateFormat;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import cn.techtutorial.connection.DbCon;
import cn.techtutorial.dao.*;
import cn.techtutorial.model.*;
@WebServlet("/order-now")
public class OrderNowServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd");
            Date date = new Date();
            User auth = (User) request.getSession().getAttribute("auth");
            if (auth != null) {
                String productId = request.getParameter("id");
                int productQuantity = Integer.parseInt(request.getParameter("quantity"));
                if (productQuantity <= 0) {
                    productQuantity = 1;
                }
                Order orderModel = new Order();
                orderModel.setId(Integer.parseInt(productId));
                orderModel.setUid(auth.getId());
                orderModel.setQunatity(productQuantity);
                orderModel.setDate(formatter.format(date));
                OrderDao orderDao = new OrderDao(DbCon.getConnection());

```

```

        boolean result = orderDao.insertOrder(orderModel);

        if (result) {
            ArrayList<Cart> cart_list = (ArrayList<Cart>)
request.getSession().getAttribute("cart-list");

            if (cart_list != null) {
                for (Cart c : cart_list) {
                    if (c.getId() == Integer.parseInt(productId)) {
                        cart_list.remove(cart_list.indexOf(c));
                        break;
                    }
                }
            }

            response.sendRedirect("orders.jsp");
        } else {
            out.println("order failed");
        }
    } else {
        response.sendRedirect("login.jsp");
    }
} catch (ClassNotFoundException|SQLException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}

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

```

7. QuantityIncDecServlet.java

```

package cn.techtutorial.servlet;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.ArrayList;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import cn.techtutorial.model.Cart;

@WebServlet("/quantity-inc-dec")
public class QuantityIncDecServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            String action = request.getParameter("action");
            int id = Integer.parseInt(request.getParameter("id"));
            ArrayList<Cart> cart_list = (ArrayList<Cart>)
request.getSession().getAttribute("cart-list");
            if (action != null && id >= 1) {
                if (action.equals("inc")) {
                    for (Cart c : cart_list) {
                        if (c.getId() == id) {
                            int quantity = c.getQuantity();
                            quantity++;
                            c.setQuantity(quantity);
                            response.sendRedirect("cart.jsp");

```

```

    }
    }
}

if (action.equals("dec")) {
    for (Cart c : cart_list) {
        if (c.getId() == id && c.getQuantity() > 1) {
            int quantity = c.getQuantity();
            quantity--;
            c.setQuantity(quantity);
            break;
        }
    }
    response.sendRedirect("cart.jsp");
}
} else {
    response.sendRedirect("cart.jsp");
}
}
}
}

```

8. RemoveFromCartServlet.java

```
package cn.techtutorial.servlet;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.ArrayList;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
```

```

import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import cn.techtutorial.model.Cart;

@WebServlet("/remove-from-cart")
public class RemoveFromCartServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            String bookId = request.getParameter("id");
            if (bookId != null) {
                ArrayList<Cart> cart_list = (ArrayList<Cart>)
request.getSession().getAttribute("cart-list");
                if (cart_list != null) {
                    for (Cart c : cart_list) {
                        if (c.getId() == Integer.parseInt(bookId)) {
                            cart_list.remove(cart_list.indexOf(c));
                            break;
                        }
                    }
                }
                response.sendRedirect("cart.jsp");
            } else {
                response.sendRedirect("cart.jsp");
            }
        }
    }
}

```

9. Signup.java

```
package cn.techtutorial.servlet;
import java.io.IOException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import cn.techtutorial.model.ModelEx;
@WebServlet("/Signup")
public class Signup extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
    IOException
    {
        String id=request.getParameter("id");
        String name = request.getParameter("name");
        String email = request.getParameter("email");
        String password = request.getParameter("password");
        String confirmPassword = request.getParameter("confirmPassword");
        try
        {
            ModelEx m = new ModelEx();
            m.setId(id);
            m.setName(name);
            m.setEmail(email);
            m.setPassword(password);
            m.setConfirmPassword(confirmPassword);
```

```
System.out.println(name);  
boolean b = m.register();  
if(b==true)  
{  
    response.sendRedirect("successReg.html");  
}  
else  
{  
    response.sendRedirect("failure.html");  
}  
}  
catch(Exception e){  
    e.printStackTrace();  
}  
}  
}
```


Jsp

1. Cart.jsp

```

<%@page import="cn.techtutorial.connection.DbCon"%>
<%@page import="cn.techtutorial.dao.ProductDao"%>
<%@page import="cn.techtutorial.model.*"%>
<%@page import="java.util.*"%>
<%@page import="java.text.DecimalFormat"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%
DecimalFormat dcf = new DecimalFormat("#.##");
request.setAttribute("dcf", dcf);
User auth = (User) request.getSession().getAttribute("auth");
if (auth != null) {
    request.setAttribute("person", auth);
}
ArrayList<Cart> cart_list = (ArrayList<Cart>) session.getAttribute("cart-list");
List<Cart> cartProduct = null;
if (cart_list != null) {
    ProductDao pDao = new ProductDao(DbCon.getConnection());
    cartProduct = pDao.getCartProducts(cart_list);
    double total = pDao.getTotalCartPrice(cart_list);
    request.setAttribute("total", total);
    request.setAttribute("cart_list", cart_list);
}
%>
<!DOCTYPE html>
<html>
<head>
<%@include file="/head.jsp"%>
<title>E-Commerce Cart</title>
<style type="text/css">
.table tbody td {
    vertical-align: middle;
}

.btn-incre, .btn-decre {
    box-shadow: none;
    font-size: 25px;
}
body{
background-image: url("product-image/cartbg.jpg");
    background-repeat: no-repeat;

```

```

        background-size: cover;
    }
    .table{
        background-color:#4DD637;
        size:50px;}
    .d-flex{
        background-color:#E07C24;
    }
</style>
</head>
<body>
    <%@include file="/navbar.jsp"%>

    <div class="container my-3">
        <div class="d-flex py-4">
            <h3>Total Price: $ ${total>0}?dcf.format(total):0}</h3>
            <a class="mx-3 btn btn-primary" href="cart-check-out">Check
Out</a>
        </div>
        <table class="table table-dark">
            <thead>
                <tr>
                    <th scope="col">Name</th>
                    <th scope="col">Category</th>
                    <th scope="col">Price</th>
                    <th scope="col">Buy Now</th>
                    <th scope="col">Cancel</th>
                </tr>
            </thead>
            <tbody>
                <%
if (cart_list != null) {
for (Cart c : cartProduct) {
%>
<tr>
<td><%=c.getName()%></td>
<td><%=c.getCategory()%></td>
<td><%=dcf.format(c.getPrice())%></td>
<td>
<form action="order-now" method="post" class="form-inline">
<input type="hidden" name="id" value="<%=c.getId()%>"class="form-input">
<div class="form-group d-flex justify-content-between">
<a class="btn bnt-sm btn-incre"
href="quantity-inc-dec?action=inc&id=<%=c.getId()%>">
<i class="fas fa-plus-square"></i></a>
<input type="text"name="quantity" class="form-control"

```

```

value="<%=c.getQuantity()%>" readonly>
<a class="btn btn-sm btn-decre" href="quantity-inc-dec?action=dec&id=<%=c.getId()%>">
<i class="fas fa-minus-square"></i></a>
</div>
<button type="submit" class="btn btn-primary btn-sm">Buy</button>
</form>
</td>
<td><a href="remove-from-cart?id=<%=c.getId()%>"
class="btn btn-sm btn-danger">Remove</a></td>
</tr>

<%=
}
}
%>
</tbody>
</table>
</div>

<%=@include file="/footer.jsp"%>
</body>
</html>

```

2. footer.jsp

```

<script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>

```

3. Head.jsp

```

<head>
<meta name="viewport" content="width=device-width,initial-scale=1">
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.15.3/css/all.min.css" />
<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css">
</head>

```

4. index.jsp

```

<%@page import="java.util.List" %>
<%@page import="cn.techtutorial.dao.*" %>
<%@page import="cn.techtutorial.connection.DbCon"%>
<%@page import="cn.techtutorial.model.*"%>
<%@page import="java.util.*"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%
User auth = (User) request.getSession().getAttribute("auth");
if (auth != null) {
    request.setAttribute("person", auth);
}
ProductDao pd=new ProductDao(DbCon.getConnection());
List<Product> products=pd.getAllProducts();
ArrayList<Cart> cart_list = (ArrayList<Cart>) session.getAttribute("cart-list");
if (cart_list != null) {
    request.setAttribute("cart_list", cart_list);
}
%>
<%
ProductDao2 pd2=new ProductDao2(DbCon.getConnection());
List<Product2> products2=pd2.getAllProducts2();
if (cart_list != null) {
    request.setAttribute("cart_list", cart_list);
}
%>
<%
ProductDao3 pd3=new ProductDao3(DbCon.getConnection());
List<Product3> products3=pd3.getAllProducts3();
if (cart_list != null) {
    request.setAttribute("cart_list", cart_list);
}
%>
<%
ProductDao4 pd4=new ProductDao4(DbCon.getConnection());
List<Product4> products4=pd4.getAllProducts4();
if (cart_list != null) {
    request.setAttribute("cart_list", cart_list);
}
%>
<%
ProductDao5 pd5=new ProductDao5(DbCon.getConnection());
List<Product5> products5=pd5.getAllProducts5();

```

```

if (cart_list != null) {
    request.setAttribute("cart_list", cart_list);
}
%>
<%
ProductDao6 pd6=new ProductDao6(DbCon.getConnection());
List<Product6> products6=pd6.getAllProducts6();
if (cart_list != null) {
    request.setAttribute("cart_list", cart_list);
}
%>
<%
ProductDao7 pd7=new ProductDao7(DbCon.getConnection());
List<Product7> products7=pd7.getAllProducts7();
if (cart_list != null) {
    request.setAttribute("cart_list", cart_list);
}
%>
<%
ProductDao8 pd8=new ProductDao8(DbCon.getConnection());
List<Product8> products8=pd8.getAllProducts8();
if (cart_list != null) {
    request.setAttribute("cart_list", cart_list);
}
%>
<%
ProductDao9 pd9=new ProductDao9(DbCon.getConnection());
List<Product9> products9=pd9.getAllProducts9();
if (cart_list != null) {
    request.setAttribute("cart_list", cart_list);
}
%>
<!DOCTYPE html>
<html>
<head>
<title>Welcome to shopping cart!</title>
<%@include file="/head.jsp"%>
<style >
body{
    background-image: url("product-image/Background-1.jpg");
    background-repeat: repeat;
}
.card-header {
    color:red;
    background-color:skyblue;
}

```

```

</style>
</head>
<body>
    <%@include file="/navbar.jsp"%>

    <div class="container">
        <div class="card-header my-3"><h1>Top Products</h1></div>
        <div class="row">
            <%
            if(!products.isEmpty()){
                for(Product p:products){%>
                    <div class="col-md-3 my-3">
                        <div class="card w-100" style="width:18rem;">
                            
                            <div class="card-body">
                                <h5 class="card-title"><%= p.getName() %></h5>
                                <h6 class="price">Price: $<%=p.getPrice() %></h6>
                                <h6 class="category">Category:<%=p.getCategory() %></h6>
                                <div class="mt-3 d-flex justify-content-between">
                                    <a href="add-to-cart?id=<%=p.getId()%>" class="btn btn-dark">Add Cart</a>
                                    <a href="order-now?quantity=1&id=<%=p.getId()%>" class="btn btn-
                                primary">Buy Now</a>
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
            <% } } else { out.println("There is no proucts"); } %>

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

    <div class="container">
        <div class="card-header my-3"><h1>Laptops </h1></div>
        <div class="row">
            <%
            if(!products2.isEmpty()){
                for(Product2 p:products2){%>
                    <div class="col-md-3 my-3">
                        <div class="card w-100" style="width:18rem;">
                            
                            <div class="card-body">
                                <h5 class="card-title"><%= p.getName() %></h5>
                                <h6 class="price">Price: $<%=p.getPrice() %></h6>

```

```

<h6 class="category">Category:<%=p.getCategory() %></h6>
  <div class="mt-3 d-flex justify-content-between">
    <a href="add-to-cart?id=<%=p.getId()%>"
      class="btn btn-dark">Add to Cart</a>
    <a href="order-now?quantity=1&id=<%=p.getId()%>"
      class="btn btn-primary">Buy Now</a>
  </div>
</div>
</div>
<% } } else { out.println("There is no proucts"); } %>

</div>
</div>

<div class="container">
  <div class="card-header my-3"><h1>Ladies Bag </h1></div>
  <div class="row">
    <%
    if(!products3.isEmpty()){
      for(Product3 p:products3){%>
        <div class="col-md-3 my-3">
          <div class="card w-100" style="width:18rem;">
            
            <div class="card-body">
              <h5 class="card-title"><%= p.getName() %></h5>
              <h6 class="price">Price: $<%=p.getPrice() %></h6>
              <h6 class="category">Category:<%=p.getCategory() %></h6>
              <div class="mt-3 d-flex justify-content-between">
                <a href="add-to-cart?id=<%=p.getId()%>"
                  class="btn btn-dark">Add to Cart</a>
                <a href="order-now?quantity=1&id=<%=p.getId()%>"
                  class="btn btn-primary">Buy Now</a>
              </div>
            </div>
          </div>
        </div>
      }
    }
    <% } } else { out.println("There is no proucts"); } %>
  </div>
</div>

<div class="container">
  <div class="card-header my-3"><h1>Men Clothes </h1></div>
  <div class="row">
    <%
    if(!products4.isEmpty()){

```

```

for(Product4 p:products4){%>
    <div class="col-md-3 my-3">
        <div class="card w-100" style="width:18rem;">
            
            <div class="card-body">
                <h5 class="card-title"><%= p.getName() %></h5>
                <h6 class="price">Price: $<%=p.getPrice() %></h6>
                <h6 class="category">Category:<%=p.getCategory() %></h6>
                <div class="mt-3 d-flex justify-content-between">
                    <a href="add-to-cart?id=<%=p.getId()%>"
                    class="btn btn-dark">Add to Cart</a>
                    <a href="order-now?quantity=1&id=<%=p.getId()%>"
                    class="btn btn-primary">Buy Now</a>
                </div>
            </div>
        </div>
    </div>
</div>
<% } } else { out.println("There is no proucts"); } %>
</div>
</div>
<div class="container">
    <div class="card-header my-3"><h1>Men Watch </h1></div>
    <div class="row">
        <%
        if(!products5.isEmpty()){
            for(Product5 p:products5){%>
                <div class="col-md-3 my-3">
                    <div class="card w-100" style="width:18rem;">
                        
                        <div class="card-body">
                            <h5 class="card-title"><%= p.getName() %></h5>
                            <h6 class="price">Price: $<%=p.getPrice() %></h6>
                            <h6 class="category">Category:<%=p.getCategory() %></h6>
                            <div class="mt-3 d-flex justify-content-between">
                                <a href="add-to-cart?id=<%=p.getId()%>"
                                class="btn btn-dark">Add to Cart</a>
                                <a href="order-now?quantity=1&id=<%=p.getId()%>"
                                class="btn btn-primary">Buy Now</a>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        <% } } else { out.println("There is no proucts"); } %>
    </div>
</div>

```



```

</div>

<div class="container">
    <div class="card-header my-3"><h1>Mobiles </h1></div>
    <div class="row">
        <%
        if(!products6.isEmpty()){
            for(Product6 p:products6){%>
                <div class="col-md-3 my-3">
                    <div class="card w-100" style="width:18rem;">
                        
                        <div class="card-body">
                            <h5 class="card-title"><%= p.getName() %></h5>
                            <h6 class="price">Price: $<%=p.getPrice() %></h6>
                            <h6 class="category">Category:<%=p.getCategory() %></h6>
                            <div class="mt-3 d-flex justify-content-between">
                                <a href="add-to-cart?id=<%=p.getId()%>"
                                class="btn btn-dark">Add to Cart</a>
                                <a href="order-now?quantity=1&id=<%=p.getId()%>"
                                class="btn btn-primary">Buy Now</a>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
            <% } } else { out.println("There is no proucts"); } %>
        </div>
    </div>

<div class="container">
    <div class="card-header my-3"><h1>Headset </h1></div>
    <div class="row">
        <%
        if(!products7.isEmpty()){
            for(Product7 p:products7){%>
                <div class="col-md-3 my-3">
                    <div class="card w-100" style="width:18rem;">
                        
                        <div class="card-body">
                            <h5 class="card-title"><%= p.getName() %></h5>
                            <h6 class="price">Price: $<%=p.getPrice() %></h6>
                            <h6 class="category">Category:<%=p.getCategory() %></h6>
                            <div class="mt-3 d-flex justify-content-between">

```

```

<a href="add-to-cart?id=<%=p.getId()%>" class="btn btn-dark">Add to Cart</a><a
href="order-now?quantity=1&id=<%=p.getId()%>" class="btn btn-primary">Buy Now</a>

```

```

</div>
</div>
</div>
</div>
<% } } else { out.println("There is no proucts"); } %>

</div>
</div>

<div class="container">
  <div class="card-header my-3"><h1>Sport Shoes </h1></div>
  <div class="row">
    <%
    if(!products8.isEmpty()){
      for(Product8 p:products8){%>
        <div class="col-md-3 my-3">
          <div class="card w-100" style="width:18rem;">
            
            <div class="card-body">
              <h5 class="card-title"><%= p.getName() %></h5>
              <h6 class="price">Price: $<%=p.getPrice() %></h6>
              <h6 class="category">Category:<%=p.getCategory() %></h6>
              <div class="mt-3 d-flex justify-content-between">
                <a href="add-to-cart?id=<%=p.getId()%>" class="btn btn-dark">Add to Cart</a>
                <a href="order-now?quantity=1&id=<%=p.getId()%>"
                class="btn btn-primary">Buy Now</a>
              </div>
            </div>
          </div>
        </div>
      }
    }
    <% } } else { out.println("There is no proucts"); } %>
  </div>
</div>

```

```

</div>
</div>

<div class="container">
  <div class="card-header my-3"><h1>Home Appliances </h1></div>
  <div class="row">
    <%
    if(!products9.isEmpty()){

```

```

        for(Product9 p:products9){%>
            <div class="col-md-3 my-3">
                <div class="card w-100" style="width:18rem;">
                    
                    <div class="card-body">
                        <h5 class="card-title"><%= p.getName() %></h5>
                        <h6 class="price">Price: $<%=p.getPrice() %></h6>
                        <h6 class="category">Category:<%=p.getCategory() %></h6>
                        <div class="mt-3 d-flex justify-content-between">
                            <a href="add-to-cart?id=<%=p.getId()%>" class="btn btn-dark">Add to Cart</a>
                            <a href="order-now?quantity=1&id=<%=p.getId()%>"
                                class="btn btn-primary">Buy Now</a>
                        </div>
                    </div>
                </div>
            </div>
        </div>
        <%= %> } } else { out.println("There is no proucts"); } %>

    </div>
</div>
<%@include file="/footer.jsp"%>
</body>
</html>
</Cart>
</Cart>
</Product>

```

5. login.jsp

```

<%@page import="cn.techtutorial.model.*"%>
<%@page import="java.util.*"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%
    User auth = (User) request.getSession().getAttribute("auth");
    if (auth != null) {
        response.sendRedirect("index.jsp");
    }
    ArrayList<Cart> cart_list = (ArrayList<Cart>) session.getAttribute("cart-list");
    if (cart_list != null) {
        request.setAttribute("cart_list", cart_list);
    }
%>

```

```

%>
<!DOCTYPE html>
<html>
<head>
<%@include file="/head.jsp"%>
<title>E-Commerce Cart</title>
<style>
.card {
background-color:transparent;

margin-left:10px;
}
body{
background-image: url("product-image/loginbg3.jpg");
background-repeat: no-repeat;
background-size: cover;
}
h2{
color:blue;
background-color:transparent;
}
.email{
font-size:40px;
color:#E8BD0D}
.pwd{
font-size:40px;
color:#E8BD0D}
.container{
margin-left:0px;
}
.login{
background-color:red;
color:white;
border-radius:30px;
}
.btn{
padding:20px 40px;
border-radius:30px;
font-size:20px;}
.mt-3{
font-size:20px;
color:#02B290}
.form-control{
background-color:transparent;
}
</style>

```

```

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

    <div class="container">
        <div class="card w-50 mx-auto my-5">
            <table class="table">
                <div class="card-header text-center">
                    <h2 class="login">User Login</h2></div>
                    <div class="card-body">
                        <form action="user-login" method="post">
                            <div class="form-group">
                                <label class="email">Email address</label> <input type="email"
                                name="login-email" class="form-control" placeholder="Enter email">
                            </div>
                            <div class="form-group">
                                <label class="pwd">Password</label> <input type="password"
                                name="login-password" class="form-control" placeholder="Password">
                            </div>
                            <div class="text-center">
                                <button type="submit" class="btn btn-primary">Login</button>
                            </div>
                        </form>
                    </div>
                </table>
                <p class="mt-3">New User Registration <a href="Signup.html">SignUp</a></p>
            </div>
        </div>

        <%@include file="/footer.jsp"%>
    </body>
</html>

```

6. navbar.jsp

```

<%@page import="cn.techtutorial.connection.DbCon"%>
<%@page import="cn.techtutorial.model.*"%>
<%@page import="java.util.*"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
    <style>
        .navbar-brand{
            font-size:30px;

```

```

    }
    .nav-item{
    font-size:20px;}
</style>

<nav class="navbar navbar-expand-lg navbar-green bg-dark">
  <div class="container">
    <a class="navbar-brand" href="index.jsp">Online Shopping</a>
    <button class="navbar-toggler" type="button" data-toggle="collapse"
      data-target="#navbarSupportedContent"
      aria-controls="navbarSupportedContent" aria-expanded="false"
      aria-label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>
    </button>

    <div class="collapse navbar-collapse" id="navbarSupportedContent">
      <ul class="navbar-nav ml-auto">
        <li class="nav-item">
          <a class="nav-link" href="index.jsp">Home</a></li>
          <li class="nav-item"><a class="nav-link" href="cart.jsp">Cart
            <span class="badge badge-danger">${cart_list.size()}</span>
          </a></li>
          <%

            if (auth != null) {
              %>
              <li class="nav-item"><a class="nav-link" href="orders.jsp">Orders</a></li>
              <li class="nav-item"><a class="nav-link" href="log-out">Logout</a></li>
              <%
                } else {
                  %>
                  <li class="nav-item"><a class="nav-link" href="login.jsp">Login</a></li>
                  <%
                    }
                  %>
                %>
              </ul>
            </div>
          </div>
        </nav>

```

7. navbar.jsp

```

<%@page import="java.text.DecimalFormat"%>
<%@page import="cn.techtutorial.dao.OrderDao"%>

```

```

<%@page import="cn.techtutorial.connection.DbCon"%>
<%@page import="cn.techtutorial.dao.ProductDao"%>
<%@page import="cn.techtutorial.model.*"%>
<%@page import="java.util.*"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%
DecimalFormat dcf = new DecimalFormat("#.##");
request.setAttribute("dcf", dcf);
User auth = (User) request.getSession().getAttribute("auth");
List<Order> orders = null;
if (auth != null) {
    request.setAttribute("person", auth);
    OrderDao orderDao = new OrderDao(DbCon.getConnection());
    orders = orderDao.userOrders(auth.getId());
} else {
    response.sendRedirect("login.jsp");
}
ArrayList<Cart> cart_list = (ArrayList<Cart>) session.getAttribute("cart-list");
if (cart_list != null) {
    request.setAttribute("cart_list", cart_list);
}
%>
<!DOCTYPE html>
<html>
<head>
<%@include file="/head.jsp"%>
<title>E-Commerce Cart</title>
</head>
<body>
    <%@include file="/navbar.jsp"%>
    <div class="container">
        <div class="card-header my-3">All Orders</div>
        <table class="table table-light">
            <thead>
                <tr>
                    <th scope="col">Date</th>
                    <th scope="col">Name</th>
                    <th scope="col">Category</th>
                    <th scope="col">Quantity</th>
                    <th scope="col">Price</th>
                    <th scope="col">Cancel</th>
                </tr>
            </thead>
            <tbody>

```

```

<%
if (orders != null) {
    for (Order o : orders) {
%>
<tr>
    <td><%=o.getDate()%></td>
    <td><%=o.getName()%></td>
    <td><%=o.getCategory()%></td>
    <td><%=o.getQunatity()%></td>
    <td><%=dcf.format(o.getPrice())%></td>
    <td><a class="btn btn-sm btn-danger"
href="cancel-order?id=<%=o.getId()%>">Cancel Order</a></td>
    </tr>
<%
    }
}
%>

</tbody>
</table>
</div>
<%=@include file="/footer.jsp"%>
</body>
</html>

```


HTML

1. failure.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Registration Failed</title>
  <!-- Bootstrap CSS link -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
  <!-- Custom CSS -->
  <style>
    body {
      background-color: #f8f9fa;
    }
    .container {
      max-width: 400px;
      margin: auto;
      padding: 20px;
      margin-top: 100px;
    }
    .alert {
      text-align: center;
    }
  </style>
</head>
<body>
  <div class="container">
    <div class="alert alert-danger" role="alert">
      <h4 class="alert-heading">Registration Failed</h4>
      <p>There was an issue with your registration. Please try again later or contact
support.</p>
      <hr>
      <p class="mb-0">Need assistance? <a href="#">Contact Support</a></p>
    </div>
  </div>
</body>
</html>
```

2. Signup.html

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Sign Up - Online Shopping</title>
  <!-- Bootstrap CSS link -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
  <!-- Custom CSS -->
  <style>
    body {
      background-image: url("product-image/signupbg.jpg");
      background-repeat: no-repeat;
      background-size: cover;
    }
    .container {
      max-width: 500px;
      margin-left: 400px;
      padding: 30px;
      margin-top: 100px;
      border: 0px solid black;
    }
    label{
      font-size: 25px;
    }
    .form-control{
      background-color: transparent;
      border: 3px solid black;
    }
    .text-center{
      color:blue;}
  </style>
</head>
<body>

  <div class="container">
    <h2 class="text-center">Sign Up</h2>
    <form action="Signup" method="get">
      <div class="mb-3">
        <label for="id" class="form-label">Id</label>
        <input type="text" class="form-control" id="id" name="id" required>
      </div>

```

```

<div class="mb-3">
  <label for="name" class="form-label">Name</label>
  <input type="text" class="form-control" id="name" name="name" required>
</div>
<div class="mb-3">
  <label for="email" class="form-label">Email</label>
  <input type="email" class="form-control" id="email" name="email" required>
</div>
<div class="mb-3">
  <label for="password" class="form-label">Password</label>
  <input type="password" class="form-control" id="password" name="password"
required>
</div>
<div class="mb-3">
  <label for="confirmPassword" class="form-label">Confirm Password</label>
  <input type="password" class="form-control" id="confirmPassword"
name="confirmPassword" onkeyup="confirmPassword();" required>
</div>
<span id="msg"></span><br><br>
<button type="submit" class="btn btn-primary">Sign Up</button>
</form>
<div>
  <p class="mt-3">Already have an account? <a href="login.jsp">Log In</a></p>
</div>

<script>
function confirmPassword() {
  var pwd = document.getElementById("password").value;
  var cpwd = document.getElementById("confirmPassword").value;
  if (pwd != cpwd) {
    document.getElementById("msg").innerHTML =
      "<i style = 'color:red;'> password and confirm does not match</i>";
  } else {
    document.getElementById("msg").innerHTML =
      "<i style = 'color:green;'>password matched!</i>";
  }
}
</script>
</body>
</html>

```

3. successReg.html

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Registration Successful</title>
  <!-- Bootstrap CSS link -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
  <!-- Custom CSS -->
  <style>
    body {
      background-color: #f8f9fa;
    }
    .container {
      max-width: 400px;
      margin: auto;
      padding: 20px;
      margin-top: 100px;
    }
    .alert {
      text-align: center;
    }
  </style>
</head>
<body>
  <div class="container">
    <div class="alert alert-success" role="alert">
      <h4 class="alert-heading">Registration Successful!</h4>
      <p>Your account has been successfully registered. You can now log in and start
shopping.</p>
      <hr>
      <p>Click here to login <a href="login.jsp">Log In</a></p>
    </div>
  </div>
</body>
</html>

```

4. Welcome.html

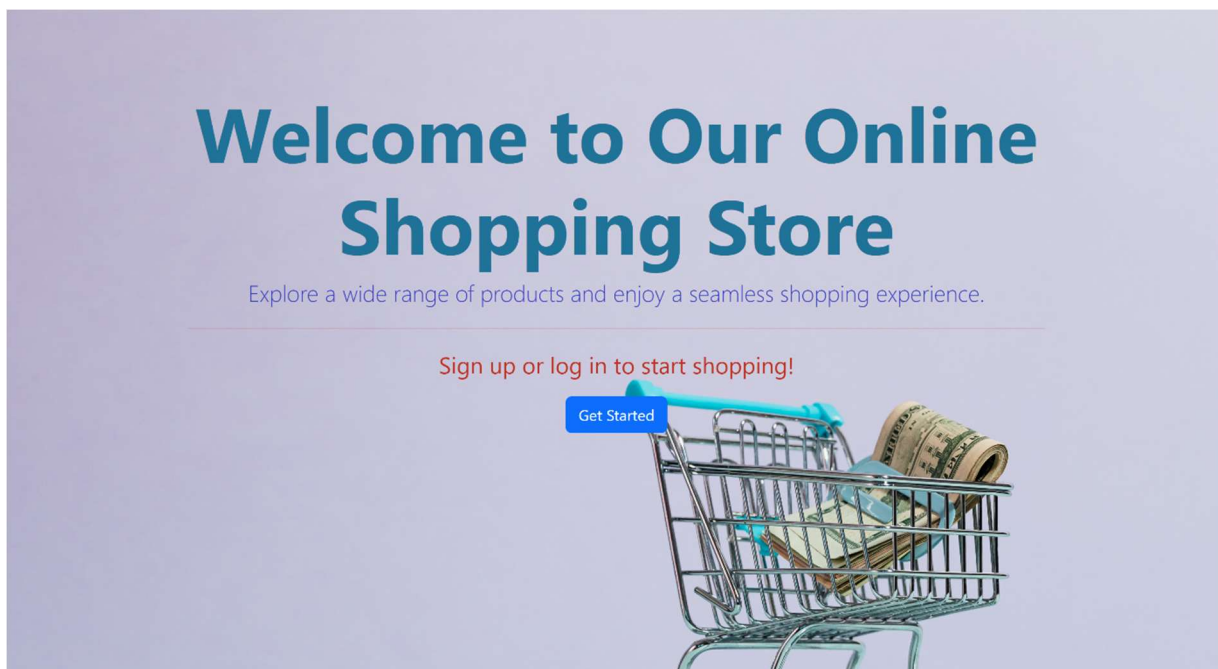
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Welcome to Online Shopping</title>
    <!-- Bootstrap CSS link -->
    <link
      href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
      rel="stylesheet"/>

    <style>
      body {
        background-image: url("product-image/welcomebg2.jpg");
        background-repeat: no-repeat;
        background-size: cover;
      }
      .container {
        text-align: center;
        padding: 100px;
      }
      .jumbotron {
        background-color: transparent;
        color: #BF3325;
        font-size: 30px;
      }
      .display-4 {
        font-weight: bolder;
        font-size: 100px;
        color: #207398;
      }
      .lead {
        font-size: 30px;
        color: #383cc1;
      }
      .p2 {
        background-color: transparent;
      }
    </style>
  </head>
  <body>
    <div class="container">
```

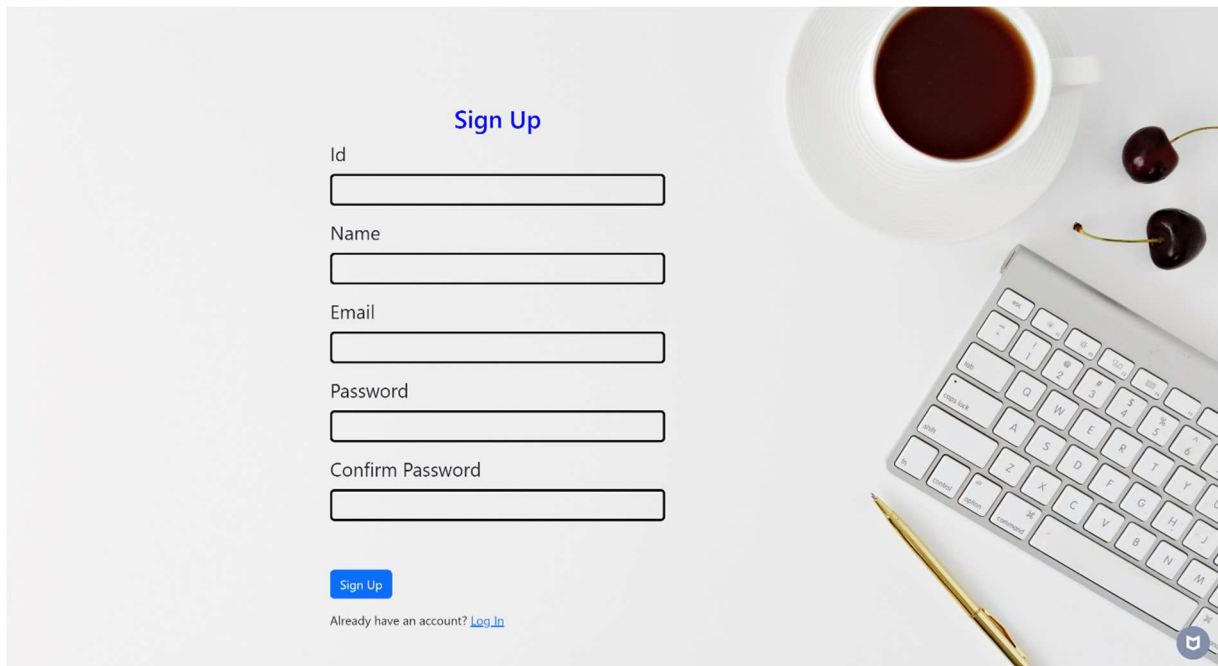
```
<div class="jumbotron">
  <h1 class="display-4">Welcome to Our Online Shopping Store</h1>
  <p class="lead">
    Explore a wide range of products and enjoy a seamless shopping
    experience.
  </p>
  <hr class="my-4" />
  <p class="p2">Sign up or log in to start shopping!</p>
  <a class="btn btn-primary btn-lg" href="Signup.html" role="button">Get Started</a>
</div>
</div>
</body>
</html>
```

OUTPUT

1. Welcome Page



2.Signup Page



The image shows a web application's signup page. The page has a light gray background. On the right side, there is a background image of a desk with a white cup of coffee, two cherries, and a silver keyboard. The signup form is located on the left side of the page. It has a title "Sign Up" in blue text. Below the title, there are five input fields: "Id", "Name", "Email", "Password", and "Confirm Password". Each field has a thin black border. Below the "Password" field, there is a blue button with the text "Sign Up". Below the button, there is a link that says "Already have an account? [Log In](#)".

Sign Up

Id

Name

Email

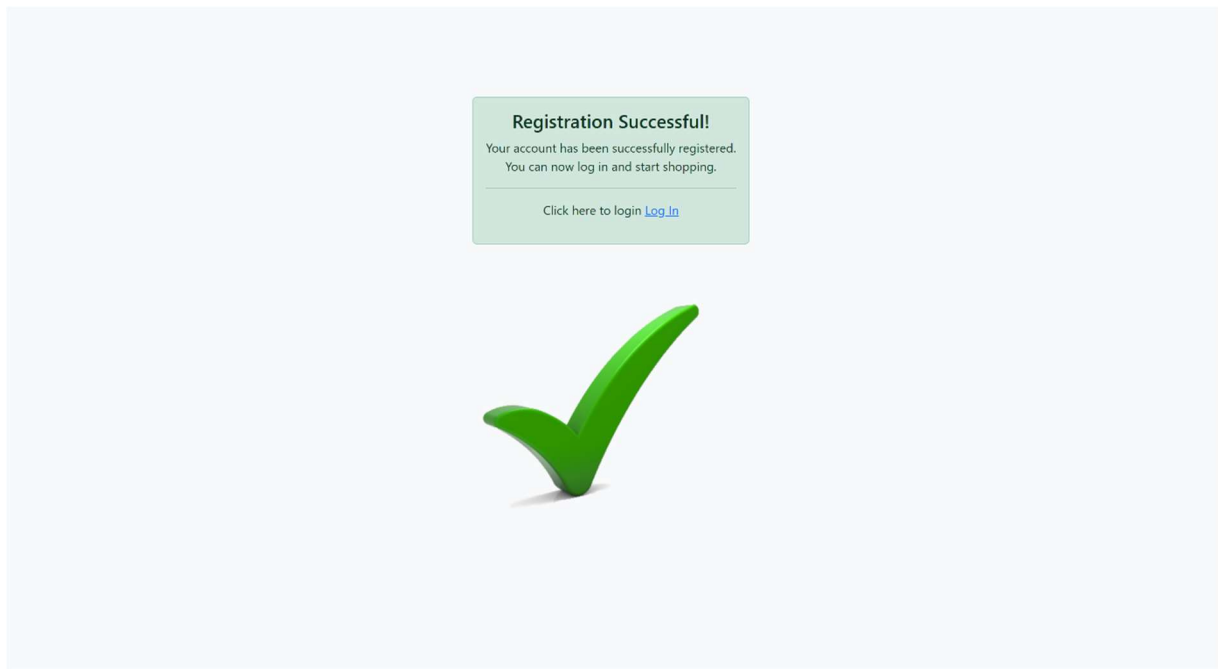
Password

Confirm Password

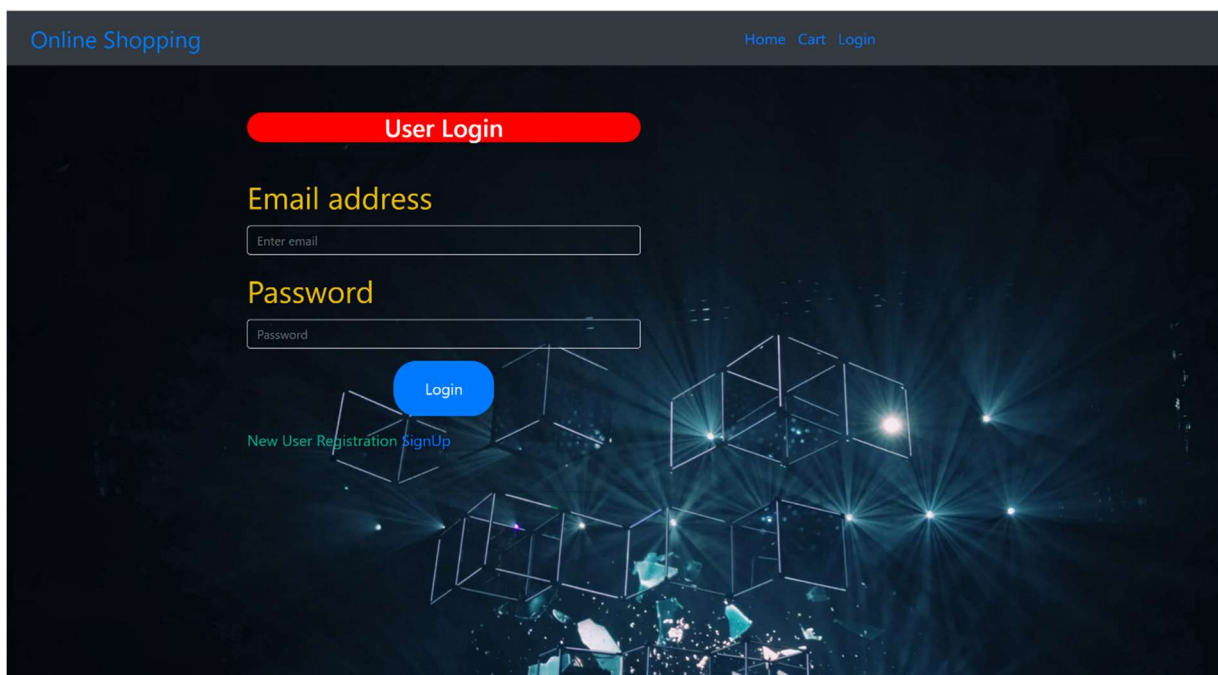
[Sign Up](#)

Already have an account? [Log In](#)

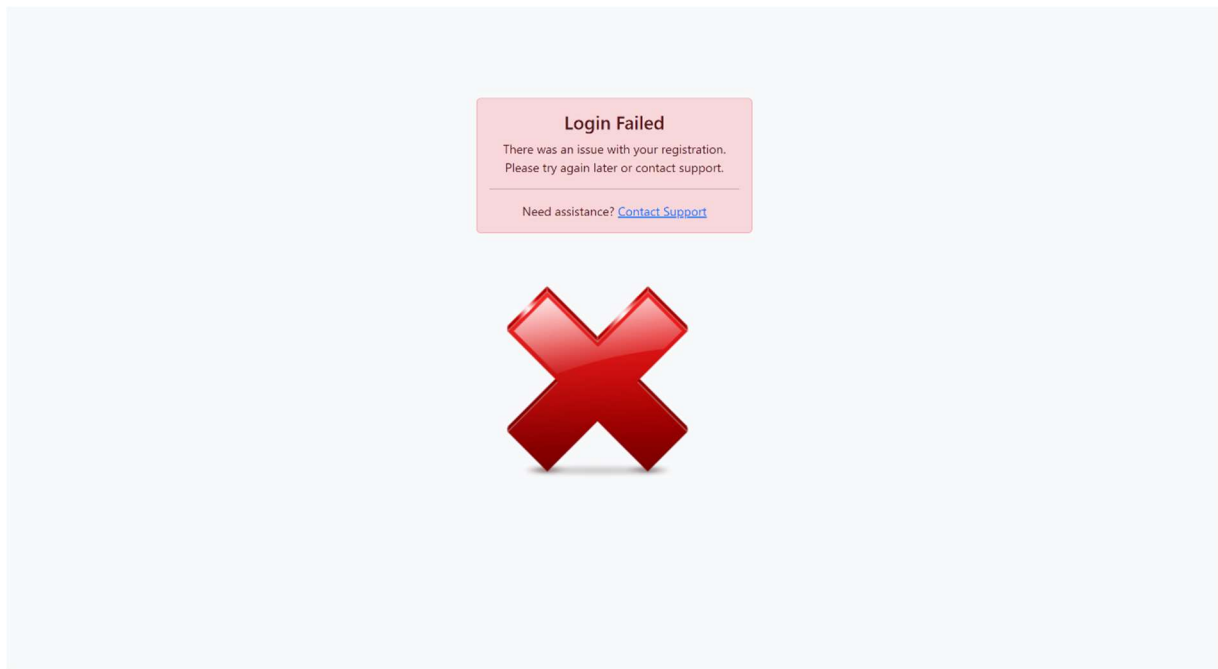
3.Successful Registration



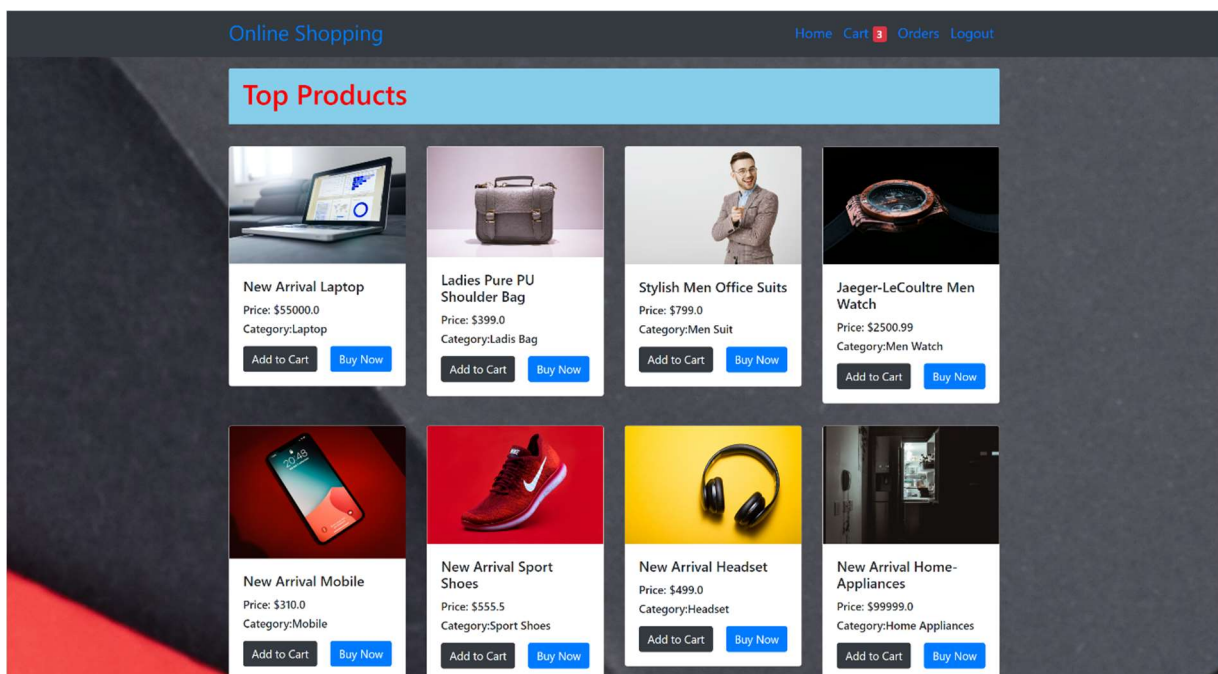
4.LoginPage



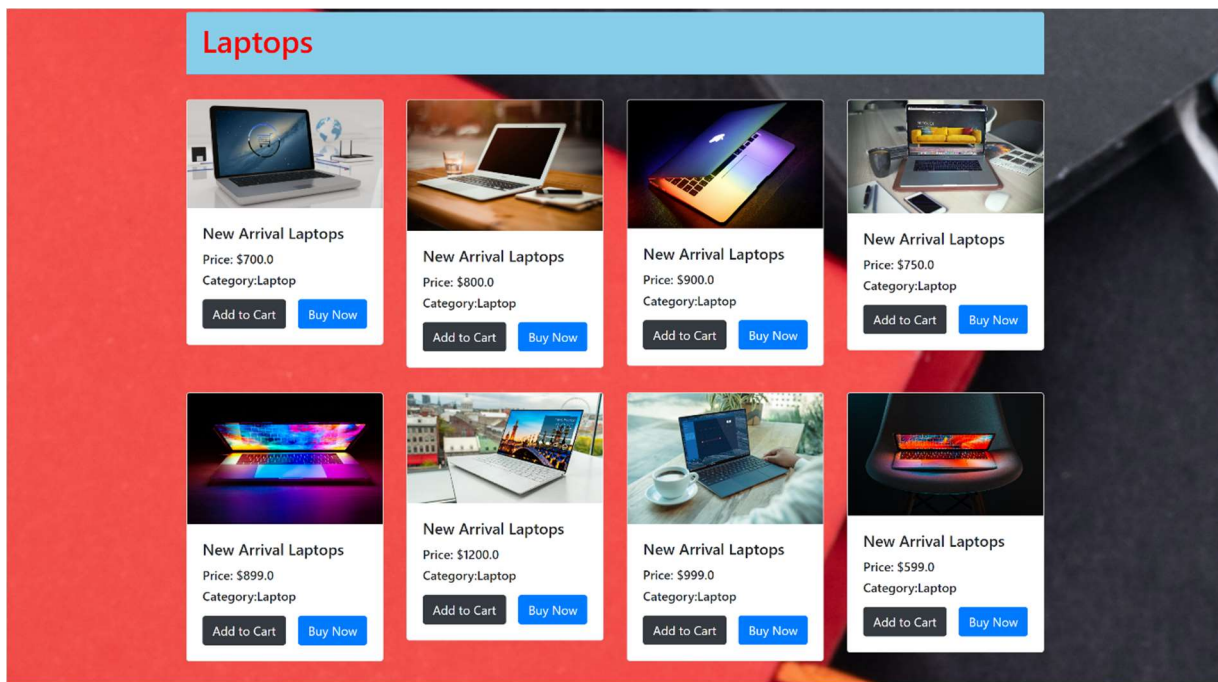
5.Login Failed Page



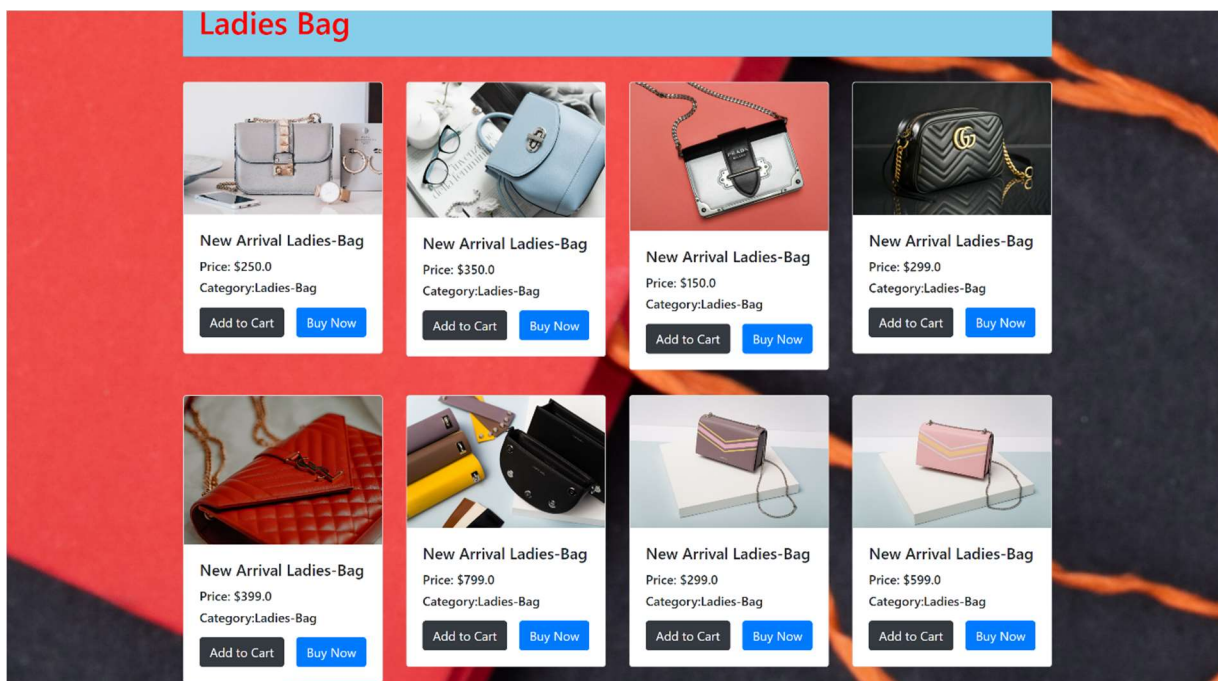
6.Home Page-1



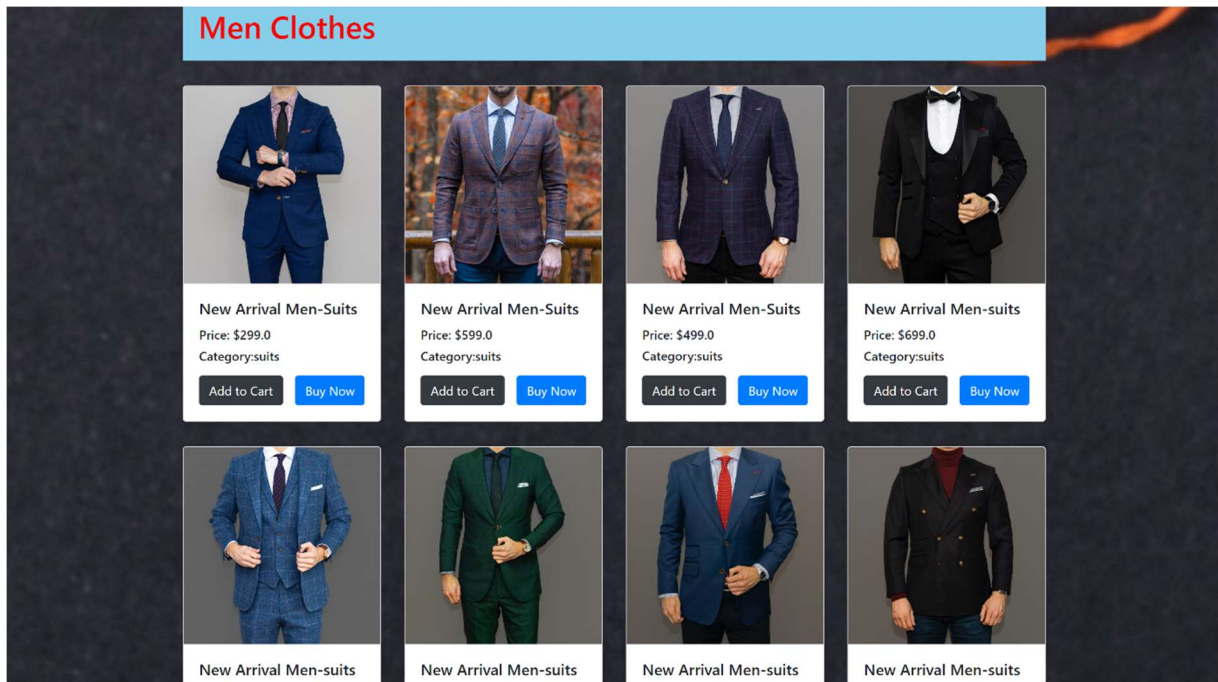
7. Home Page-2



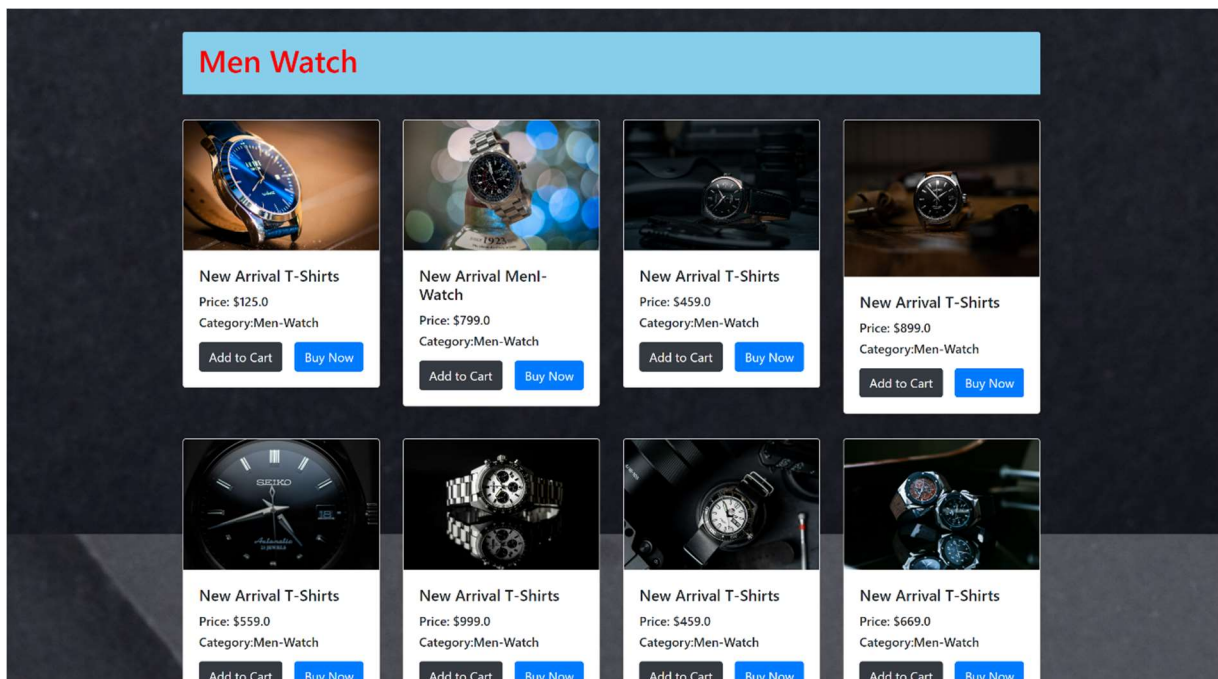
8.Home Page-3



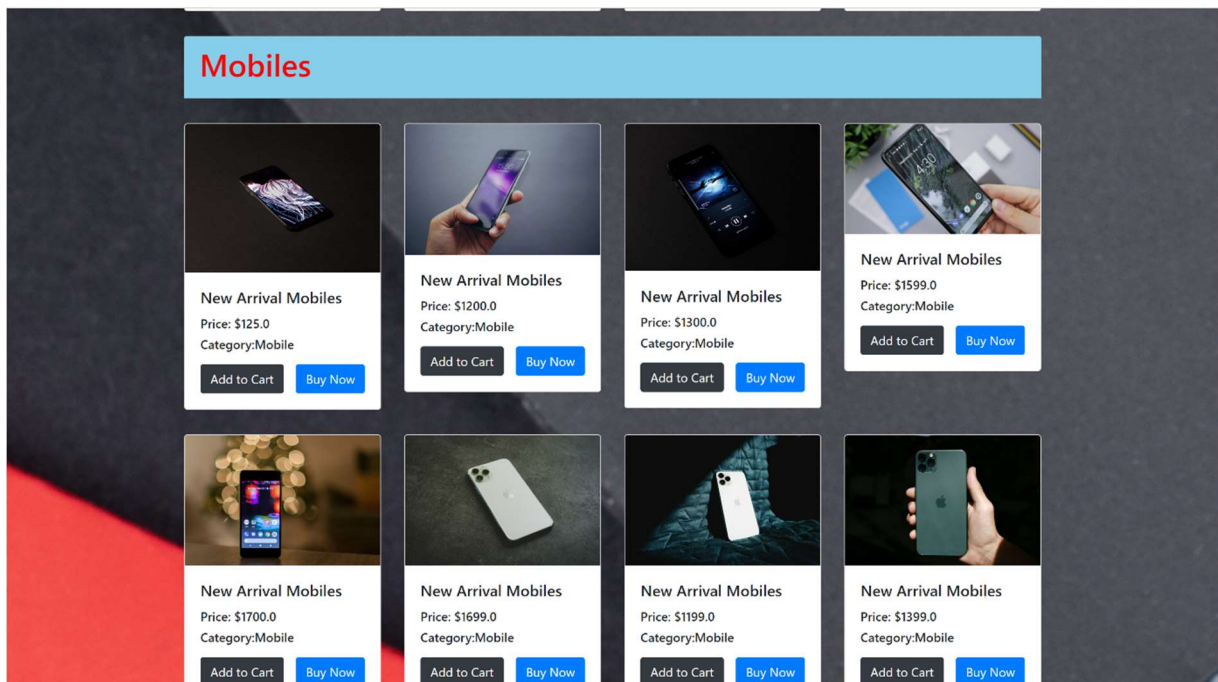
9.Home Page-4



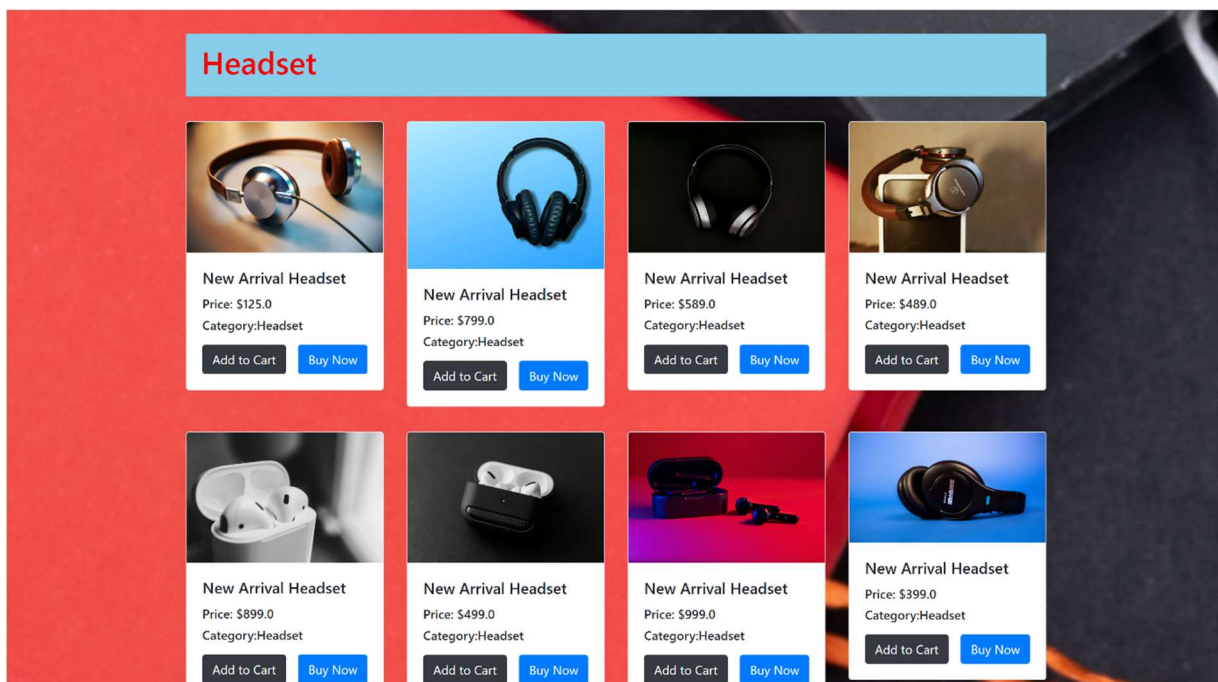
10.Home Page-5



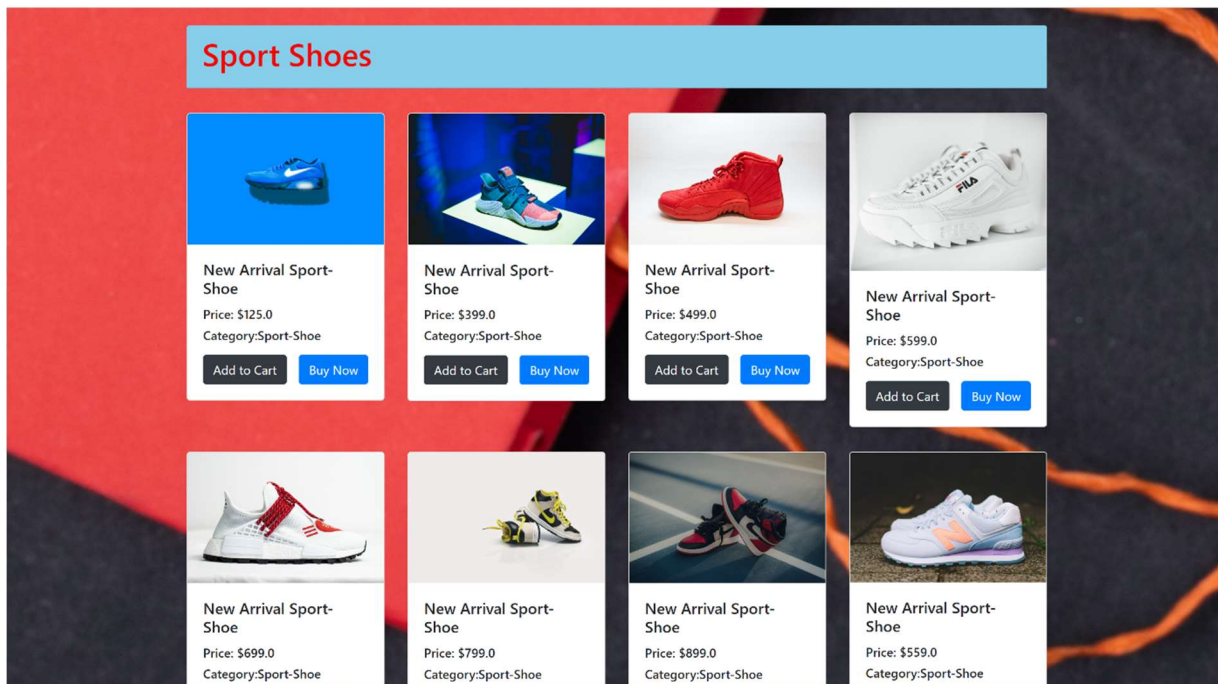
11.Home Page-6



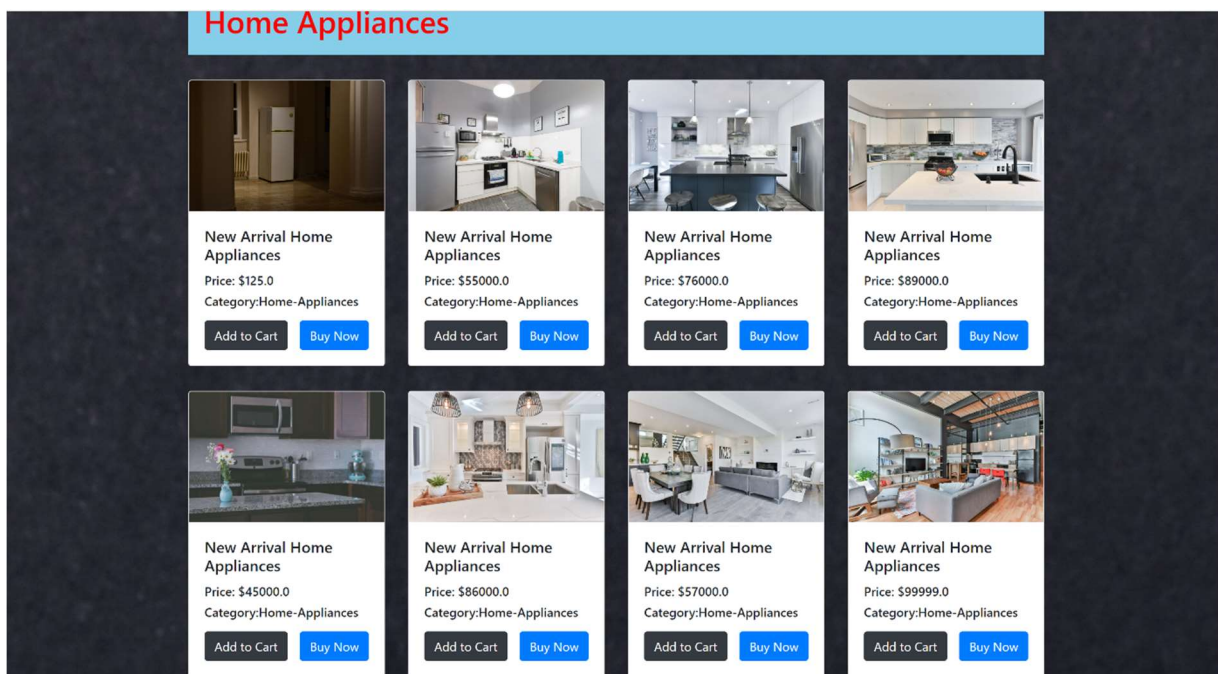
12.Home Page-7



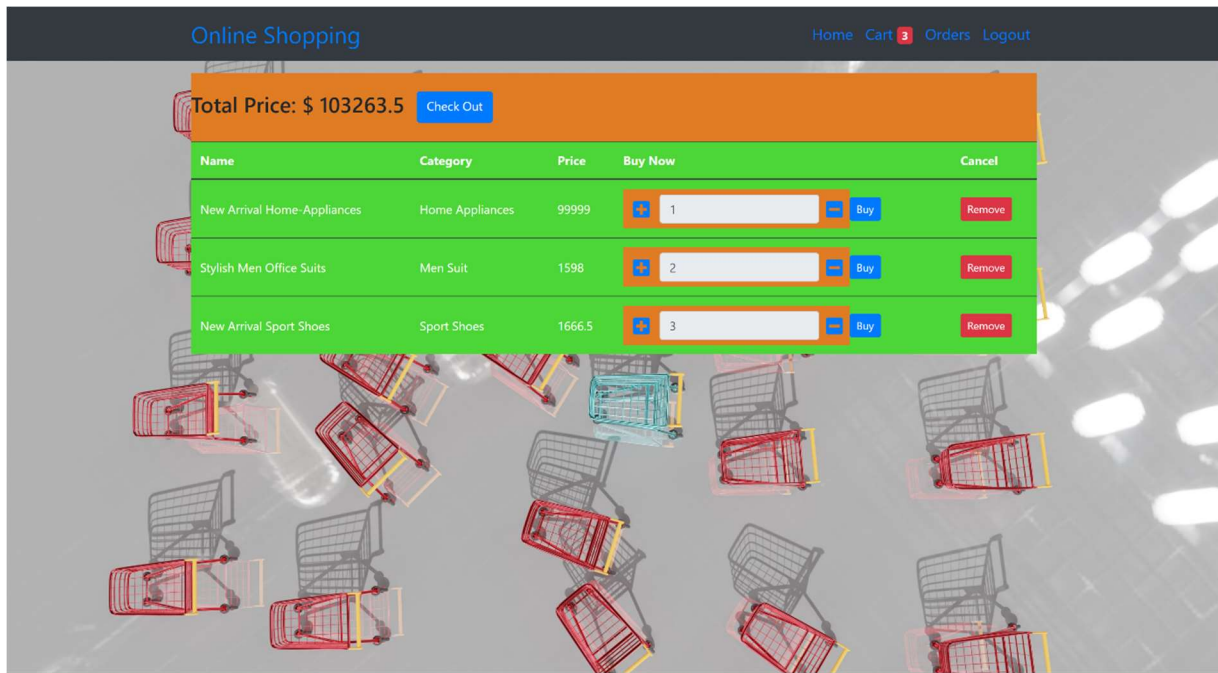
13.Home Page-8



14.Home Page-9



15.Cart Page



16.Order Page



16. SQL Tables

No nag screens on startup and shutdown : Reason #1 to upgrade

Query 1 Query 2 x Query 3 +

```

1  USE ecommerce_cart
2
3  CREATE TABLE `orders` (
4      `o_id` INT NOT NULL AUTO_INCREMENT,
5      `p_id` INT NOT NULL,
6      `u_id` INT NOT NULL,
7      `o_quantity` INT NOT NULL,
8      `o_date` VARCHAR(450) NOT NULL,
9      PRIMARY KEY (`o_id`)
10 )
11 SHOW TABLES
12 DESC orders
13
14 INSERT INTO `orders` VALUES (25,3,1,3,'2021-05-15'), (26,2,1,1,'2021-05-15');
15 SELECT * FROM orders
16
17 CREATE TABLE `products` (
18     `id` INT NOT NULL AUTO_INCREMENT,
19     `name` VARCHAR(450) NOT NULL,
20     `category` VARCHAR(450) NOT NULL,
21     `price` DOUBLE NOT NULL,
22     `image` VARCHAR(450) NOT NULL,
23     PRIMARY KEY (`id`)
24 )
25
26 SELECT * FROM products
  
```

1 Result 2 Profiler 3 Messages 4 Table Data 5 Info

(Read Only)

<input type="checkbox"/>	o_id	p_id	u_id	o_quantity	o_date
<input type="checkbox"/>	37	1	1	1	2023-08-12
<input type="checkbox"/>	38	3	1	1	2023-08-12
<input type="checkbox"/>	39	2	1	2	2023-08-12
<input type="checkbox"/>	42	4	1	1	2023-08-12
<input type="checkbox"/>	52	1	2	1	2023-08-12
<input type="checkbox"/>	53	3	2	1	2023-08-12
<input type="checkbox"/>	54	3	2	2	2023-08-12
<input type="checkbox"/>	56	2	2	2	2023-08-12
<input type="checkbox"/>	59	1	2	1	2023-08-12
<input type="checkbox"/>	60	2	2	1	2023-08-12

Limit rows First row 0 # of rows 1000

No nag screens on startup and shutdown : Reason #1 to upgrade

Query 1 Query 2 x Query 3 +

```

16
17 CREATE TABLE `products` (
18   `id` INT NOT NULL AUTO_INCREMENT,
19   `name` VARCHAR(450) NOT NULL,
20   `category` VARCHAR(450) NOT NULL,
21   `price` DOUBLE NOT NULL,
22   `image` VARCHAR(450) NOT NULL,
23   PRIMARY KEY (`id`)
24 )
25
26 SELECT * FROM products
27
28 CREATE TABLE `users` (
29   `id` INT NOT NULL AUTO_INCREMENT,
30   `name` VARCHAR(250) NOT NULL,
31   `email` VARCHAR(250) NOT NULL,
32   `password` VARCHAR(250) NOT NULL,
33   PRIMARY KEY (`id`),
34   UNIQUE KEY `email_UNIQUE` (`email`)
35 )
36 ENGINE=INNODB AUTO_INCREMENT=7 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
37
38 INSERT INTO `users` VALUES (1,'Almamun','almamun@mail.com','123456','123456');
39 DESC users
40 SELECT * FROM users
41 COMMIT

```

1 Result 2 Profiler 3 Messages 4 Table Data 5 Info

(Read Only) Limit rows First row 0 # of rows 1000

id	name	category	price	image
1	New Arrival Laptop	Laptop	55000	Laptop-1.jpg
2	Ladies Pure PU Shoulder Bag	Ladies Bag	399	Ladies-bag.jpg
3	Stylish Men Office Suits	Men Suit	799	men-suits.jpg
4	Jaeger-LeCoultre Men Watch	Men Watch	2500.99	men-watch.jpg
5	New Arrival Mobile	Mobile	310	mobile-1.jpg
6	New Arrival Sport Shoes	Sport Shoes	555.5	shoe-1.jpg
7	New Arrival Headset	Headset	499	headset-1.jpg
8	New Arrival Home-Appliances	Home Appliances	99999	refrigerator-1.jpg

No nag screens on startup and shutdown : Reason #1 to upgrade

Query 1 Query 2 x Query 3 +

```

26 SELECT * FROM products
27
28 CREATE TABLE `users` (
29   `id` INT NOT NULL AUTO_INCREMENT,
30   `name` VARCHAR(250) NOT NULL,
31   `email` VARCHAR(250) NOT NULL,
32   `password` VARCHAR(250) NOT NULL,
33   PRIMARY KEY (`id`),
34   UNIQUE KEY `email_UNIQUE` (`email`)
35 )
36 ENGINE=INNODB AUTO_INCREMENT=7 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
37
38 INSERT INTO `users` VALUES (1,'Almamun','almamun@mail.com','123456','123456');
39 DESC users
40 SELECT * FROM users
41 COMMIT
42 INSERT INTO users VALUES ('2','raveesh','raveesha404@gmail.com','1234','1234');
43
44 INSERT INTO `products` VALUES (1,'New Arrival Laptop','Laptop',999,'Laptop-1.jpg');
45
46 CREATE TABLE `products2` (
47   `id` INT NOT NULL AUTO_INCREMENT,
48   `name` VARCHAR(450) NOT NULL,
49   `category` VARCHAR(450) NOT NULL,
50   `price` DOUBLE NOT NULL,
51   `image` VARCHAR(450) NOT NULL

```

1 Result 2 Profiler 3 Messages 4 Table Data 5 Info

(Read Only) Limit rows First row 0 # of rows 1000

id	name	email	password	confirmpassword
1	Almamun	almamun@mail.com	123456	confirmpassword
102	smiley	smiley@gmail.com	smiley	smiley
110	venky	venky@gmail.com	6789	6789
2	raveesh	raveesha404@gmail.com	1234	1234
55	pavan	pavan@gmail.com	pavan	pann
78	rohan	rohan@gmail.com	rohan	rohan
8	surya	surya@gmail.com	123	123

Database Name:	Table Name
ecommerce_cart	orders
	users
	products
	products2
	products3
	products4
	products5

CONCLUSION:

As we can see, online shopping may or may not be greener than traditional shopping. There are simply way too many factors that we have to consider in such a model.

As we focus on the costs of online shopping, it seems that online shopping is really detrimental for the environment. Online shopping brings us great convenience, but it also encourages irresponsible consumption habits like exploiting the advantages of free returns and expedited shipping. These add on to the existing pool of environmental problems that we are dealing with – global warming, wastes and pollution. Therefore, we should change our attitude towards e-commerce – to be more responsible, less exploitative and more thoughtful for the environment.