

# **A Web eCommerce Application Based on Java Frameworks**

A Project submitted in partial fulfilment for the Degree of  
**Master of Computer Application** of  
Jadavpur University

By  
**ASIF ALI KHAN**

Registration No :- 154217 of 2020-2021

Exam Roll No :- MCA2360034  
Class Roll No :- 002010503009

Under the Guidance of  
**Dr. Sarmistha Neogy**  
Department of Computer Science and Engineering  
Jadavpur University, Kolkata-700032  
2023

# **ACULTY OF ENGINEERING AND TECHNOLOGY JADAVPUR UNIVERSITY**

## **To Whom It May Concern**

I hereby recommend that the project report entitled " **A Web eCommerce Application Based on Java Frameworks**" has been carried out by **ASIF ALI KHAN** (University Registration Number :- 154217 of 2020-2021, Exam Roll Number: MCA2360034, Class Roll Number:002010503009) under my guidance and supervision and be accepted in partial fulfilment of the requirement for the Degree of Master of Computer Application in Department of Computer Science and Engineering, Jadavpur University

-----  
**Dr. Sarmistha Neogy**  
Project Supervisor  
Jadavpur University

-----  
**Dr. Nandini Mukhopadhyay**  
Head Of Department  
Jadavpur University

-----  
**Prof. Ardhendu Ghoshal**  
Dean  
Faculty of Engineering and Technology, Jadavpur University

# FACULTY OF ENGINEERING AND TECHNOLOGY JADAVPUR UNIVERSITY

## Certificate of Approval

This is to certify that the project report entitled “**A Web eCommerce Application Based on Java Frameworks**” is a bona-fide record of work carried out by ASIF ALI KHAN in partial fulfilment of the requirements for the award of the degree of Master of Computer Application in the Department of Computer Science and Engineering, Jadavpur University. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein but approve the project report only for the purpose for which it has been submitted.

-----  
Dr. Sarmistha Neogy  
Project Supervisor  
Jadavpur University  
Dated,

# **FACULTY OF ENGINEERING AND TECHNOLOGY JADAVPUR UNIVERSITY**

## **Declaration of Originality and Compliance of Academic Ethics**

I hereby declare that this thesis entitled “**A Web eCommerce Application Based on Java Frameworks**” contains the original project work by the undersigned candidate, as part of his Master of Computer Application studies.

All information in this document has been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that, as required by these rules and conduct, I have fully cited and referenced all materials and results that are not original to this work.

**Name:** ASIF ALI KHAN

**Registration No :-** 154217 of 2020-2021

**Exam Roll No :-** MCA2360034

**Class Roll No :-** 002010503009

**Project Report Title:** A Web eCommerce Application Based on Java Frameworks

.....

Signature with Date

# **Acknowledgement**

I would sincerely like to thank my, **Department of Computer Science and Engineering**, for allowing me to explore this project. I humbly thank **Dr. Nandini Mukhopadhyay**, Professor and Head, Department of Computer Science and Engineering, for her continued support during the course of this Engagement, which eased the process and formalities involved.

I express my deep sense of gratitude to my respected and learned guide, **Dr. Sarmistha Neogy** or his valuable support and guidance. I am grateful to him for the support and constant guidance he has provided me for completing the project within the stipulated time.

-----  
**Name: Asif Ali Khan**

**Registration No :- 154217 of 2020-2021**

**Exam Roll No :- MCA2360034**

**Class Roll No :- 002010503009**

Department of Computer Science & Engineering  
Jadavpur University

# CONTENTS

**Certificate of Recommendation iii**

**Certificate of Approval iv**

**Declaration of Originality & Compliance of Academic Ethics v**

**Acknowledgements vi**

<b>1. Introduction .....</b>	<b>1</b>
1.1 Overview .....	1
1.2 Background Study... ..	1
1.3 Project Planning.....	2
1.4 purpose.....	2
<b>2.System Design .....</b>	<b>3</b>
2.1 Description .....	3
2.2 User Characteristics.....	3
2.3 DFD Diagram for The Project.....	4
<b>3.Implementation.....</b>	<b>8</b>
3.1 MVC Architecture.....	8
3.2 Front End or View section.....	8
3.3 Controller Section or Servlet.....	10
3.4 Connecting the Database.....	12
3.5 Implementing the Mail Sending Section.....	20
<b>4. Testing and Test Results.....</b>	<b>22</b>
<b>5. Conclusions and Future Scope .....</b>	<b>24</b>
<b>6. References .....</b>	<b>25</b>

# **Introduction**

## **1.1 Overview**

**myshop**, the E-commerce Web application, provide solutions to develop and transfer to sale products, manage inventory in an easy and efficient way in the digital age and help to reduces the human pressure and time. It supports to keep the shop collections and give users the digital experience with its feature.

“myshop” is a web application written for all operating systems, designed to help users maintain and organize shop virtually. This software is easy to use for both beginners and advanced users. It features a familiar and well-maintained UI thought- out the website, an attractive user interface, combined with product filtering property, product update in the store, interactive admin panel. The report generation facility of shop system helps to get a good idea of which are the various items brought by the members, makes users possible to get the product easily.

On the top of that user can update their data, see their previous orders and delete their account very easily. It has unique email registration and encoding for password so no one can access others account. Even an admin cannot access their users account and vice versa.

This web application has proper cart management so once one adds their product can easily decrease the quantity or remove the product from cart any time right before completing the order or increase the quantity. One can get the full description and detail of the product after clicking on the product.

It will send email for account verification, after complete the order and after the account deletion.

## **1.2 Background study**

E-commerce is fast gaining ground as an accepted and used business paradigm. More and more business houses are implementing web sites providing functionality for performing commercial transactions over the web. It is reasonable to say that the process of shopping on the web is becoming commonplace.

The objective of this project is to develop a general-purpose e-commerce store where any

product (such as books, CDs, computers, mobile phones, electronic items, and home appliances) can be bought from the comfort of home through the Internet. However, for implementation purposes, this paper will deal with an online ecommerce store. An online store is a virtual store on the Internet where customers can browse the catalogue and select products of interest. The selected items may be collected in a shopping cart. At

checkout time, the items in the shopping cart will be presented as an order. At that time, more information will be needed to complete the transaction.

Usually, the customer will be asked to fill or select a billing address, a shipping address, a shipping option, and payment information such as a credit card number. An email notification is sent to the customer as soon as the order is placed.

## **1.3 Project Planning**

Project planning is part of project management, which relates to the use of schedules such as Gantt charts to plan and subsequently report progress within the project environment. Initially, the project scope is defined and the appropriate methods for completing the project are determined. Following this step, the durations for the various tasks necessary to complete the work are listed and grouped into a work breakdown structure. The logical dependencies between tasks are defined using an activity network diagram that enables identification of the critical path. Float or slack time in the schedule can be calculated using project management software. Then the necessary resources can be estimated and costs for each activity can be allocated to each resource, giving the total project cost. At this stage, the project plan may be optimized to achieve the appropriate balance between resource usage and project duration to comply with the project objectives. Once established and agreed, the plan becomes what is known as the baseline. Progress will be measured against the baseline throughout the life of the project

## **1.4 Purposes**

The project is about to handle all the information of the shop regarding members. Also, it manages resources which were managed and handled by manpower previously. The main purpose of the project is to integrate distinct sections of the shop into consistent manner so that complex functions can be handled smoothly. The project aims at the following matters

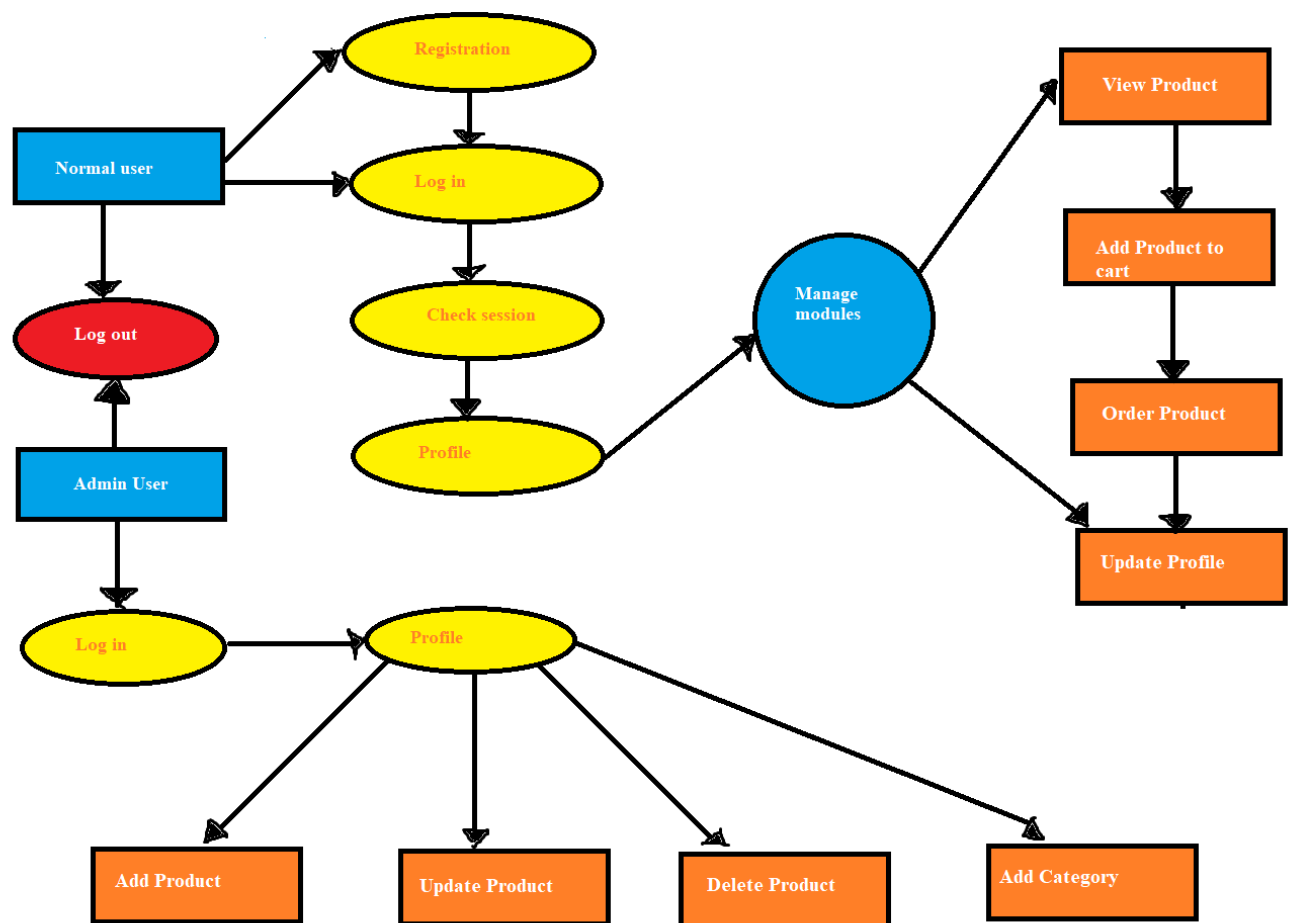
- Automation of product manipulation.
- Buying products.
- To manage information of different types of items.
- Consistently update information of all the item.
- Managing security by providing authorized email & password.
- Manages database efficiently.



# System Design

## 2.1 Description

The system is divided into some parts. These are Registration system, Login system, viewing system, order system and check out system with the database representation in server using **Servlet**, **MySQL** and **APACHE TOMCAT** server. System diagram is given below



## 2.2 User Characteristics

**Admin** The administrator has all the rights to access the system. He is the one who has all rights to view the members and product details, modify those details. He can add various product based on the category. He can also set the available quantity of a product and its reasonable price. Also, he can also set discount in various occasion. Admin can also view the details of a member. The



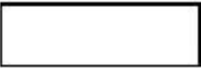

admin has the power to generate the scratch card so that users can also use the recharge card to buy various product.

**Users** The user can log in to the system by using his specific email and password. User can view the products and order the products according to their own needs. He can view his profile and update his details. He can update his personal information by logging into the system. User can find various product by using search option easily. update his details. He can update his personal information by logging into the system. User can find various product by using search option easily.

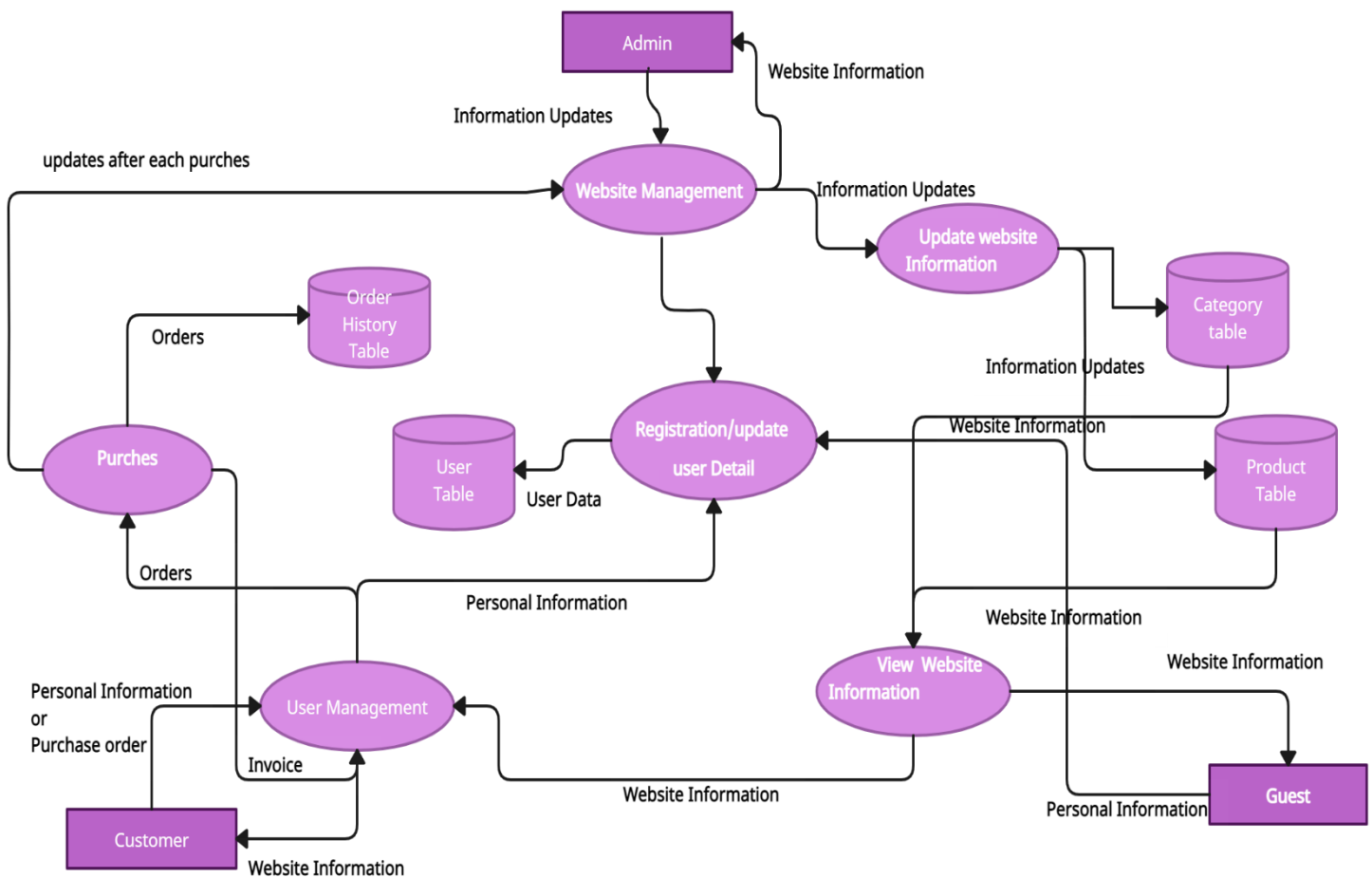
## 2.3 DFD Diagram for The Project

A data flow diagram is a graphical view of how data is processed in a system in terms of input and output.

In general, there are three levels of Data Flow Diagram. Level 0, Level 1 and Level 2.

Symbol	Description
	<b>Data Flow</b> – Data flow are pipelines through the packets of information flow.
	<b>Process:</b> A Process or task performed by the system.
	<b>Entity:</b> Entity are object of the system. A source or destination data of a system.
	<b>Data Store :</b> A place where data to be stored.

**Level 1 Data flow diagram is given below**

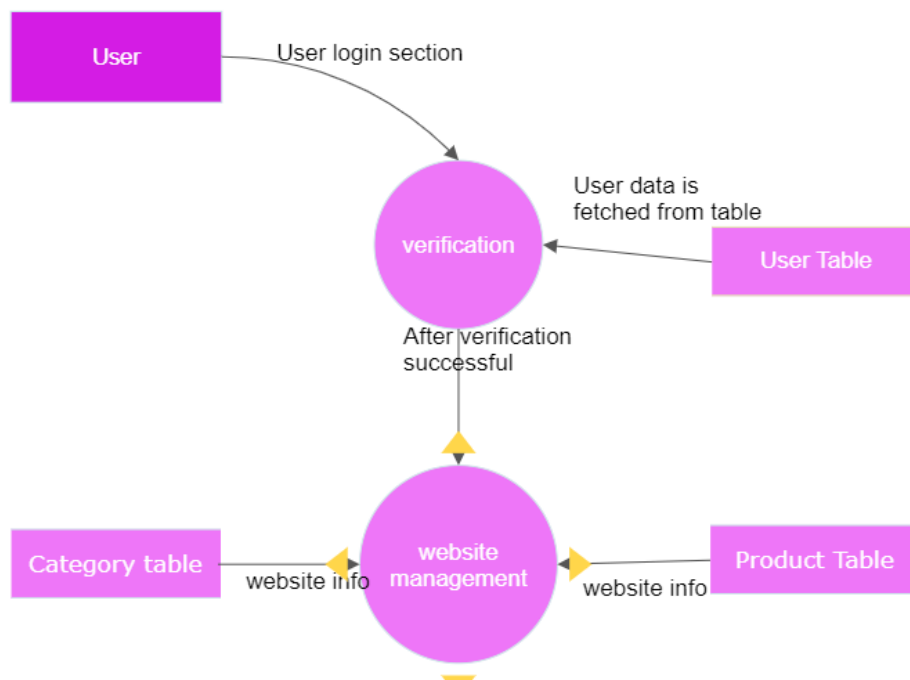
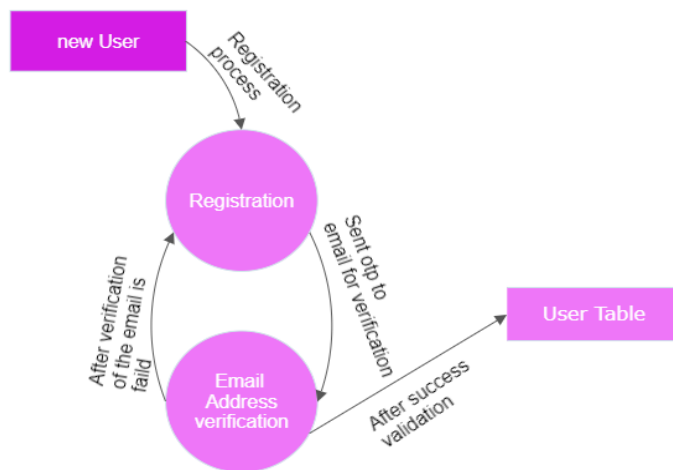


## Level 2 DFD diagrams

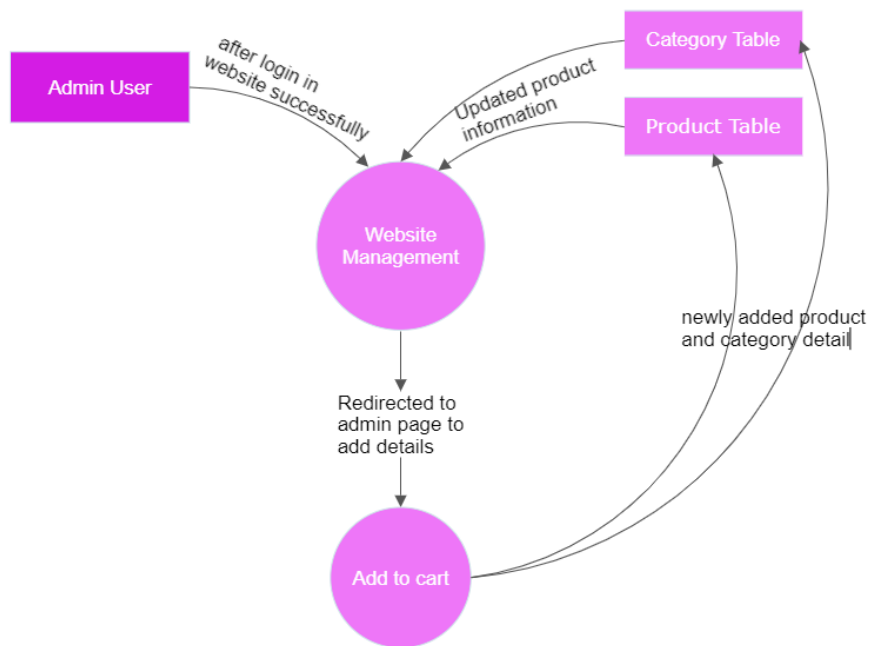
2-level DFD goes one process deeper into parts of 1-level DFD. It can be used to project or record the specific/necessary detail about the system's functioning.

### 1. User Registration

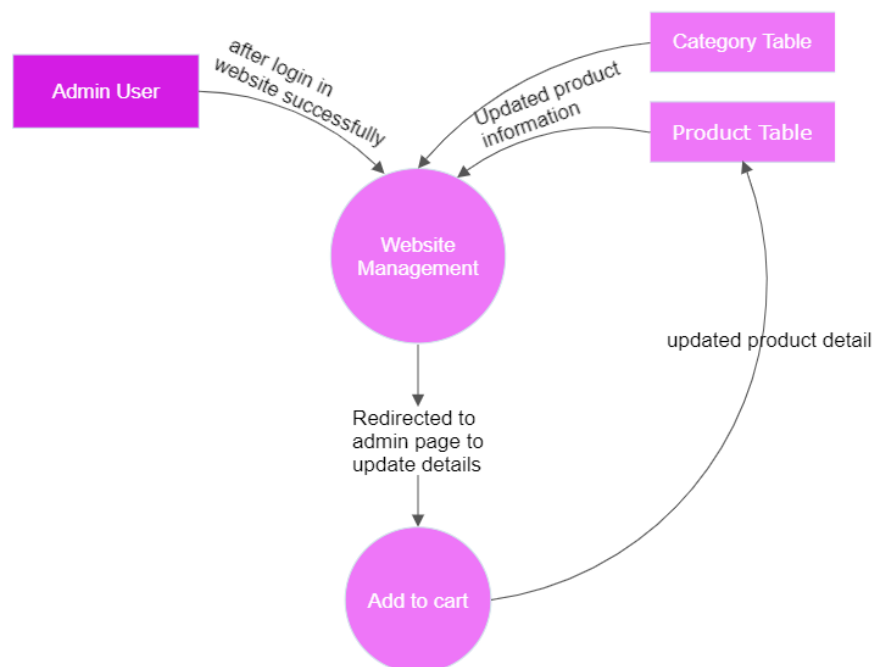
## 2. User Login



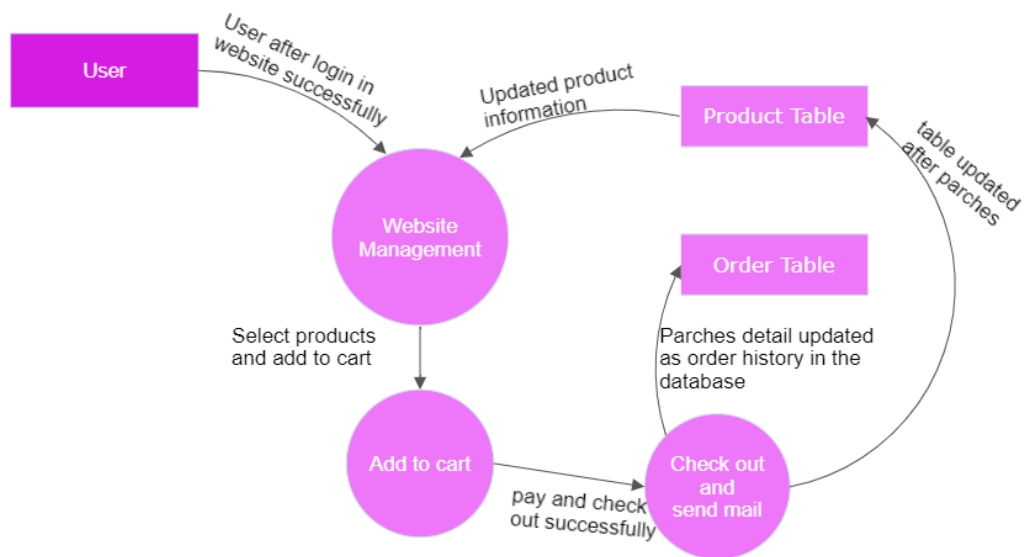
## 2. Product and Category add



### 3. Update Product details



### 4. Order & check out



Here system design ends and depending on this we will start designing our web application. Next, we will start the implementation part of the project.

# Implementation

As stated in the previous section here we will start the implementation of the project. Now ever web application is based on an architecture called MVC architecture. We will first describe and then try to maintain this overall the project.

## 3.1 MVC Architecture\*

**MVC** stands for Model View and Controller. It is a **design pattern** that separates the business logic, presentation logic and data.

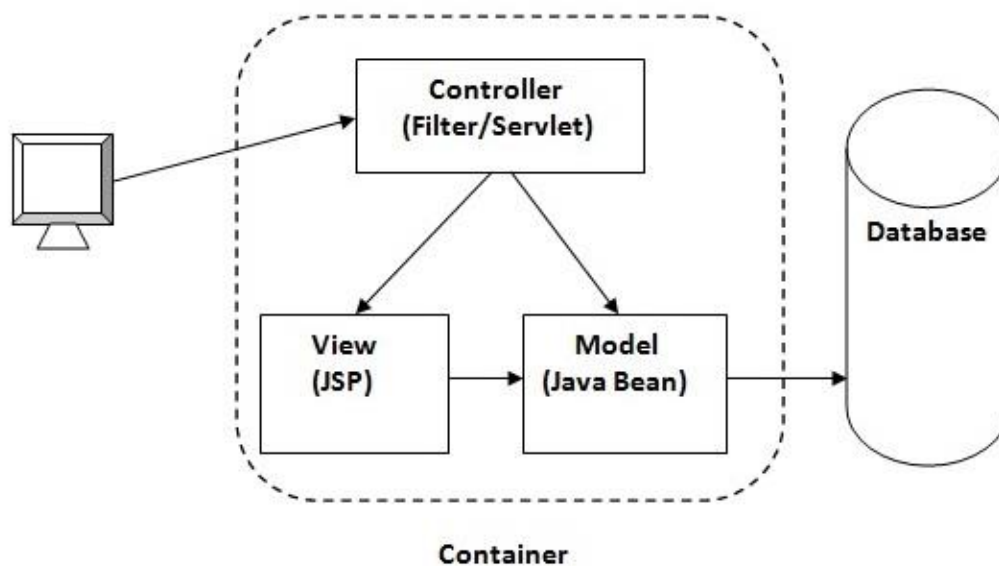
**Controller** acts as an interface between View and Model. Controller intercepts all the incoming requests and send them according to their need to Model or View. In this project Servlet acts a Controller

**Model** represents the state of the application i.e., data. It can also have business logic. In this project Dao classes are used for business and maintain the connectivity with database.

**View** represents the presentation i.e., UI (User Interface). Which is mainly done by JSP.

### Advantage of MVC Architecture

1. Navigation Control is centralized
2. Easy to maintain the large application



## 3.2 Front End or View section

Front end means the UI section of the project, which gives the look and fill of the project. There are various technologies available, among them I uses the followings.

### 3.2.1 JSP

It stands for Java Server Pages. It is a server-side technology. In this JSP tags are used to insert JAVA code into HTML pages. It is an advanced version of Servlet Technology. It is a Web based technology helps us to create dynamic and platform independent web pages. A JSP file is a server-generated web page and similar to a HTML web page we can add the CSS and JavaScript in the page to make it more responsive.

when a JSP page is requested for the first time, the JSP engine (often referred to as a "container") compiles the JSP page into a Java servlet. This compiled servlet is then loaded by the container and executed to generate the dynamic content of the web page.

There are four types of JSP tag. All are given as follows:

- i. Directive tag: - there are three types of directive tags
  - a. @page directive: - to import java packages.
  - b. @include directive: - to include jsp page to current jsp page.
  - c. @taglib directive: - we can work with different types of tags.
- ii. Declaration tag: - used to declaration java members within Servlet but outside service method. e.g., <%! variables, methods etc %>
- iii. Scriptlet tag: - used to process java code that is consider inside service method.
- iv. Expression tag: - It is used to access the value of a variable.

### 3.2.2 Use in the Project



```

<%@page import="java.util.ArrayList"%>
<%@page import="com.myshop.myshop.helper.Helper"%>
<%@page import="org.hibernate.cfg.annotations.HCANNHelper"%>
<%@page import="com.myshop.myshop.entites.Category"%>
<%@page import="com.myshop.myshop.dao.CategoryDao"%>
<%@page import="java.util.List"%>
<%@page import="com.myshop.myshop.entites.Product"%>
<%@page import="com.myshop.myshop.entites.User"%>
<%@page import="com.myshop.myshop.dao.ProductDao"%>
<%@page import="com.myshop.myshop.helper.FactoryProvider"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>myshop</title>
        <%@include file="components/comman-css-js.jsp"%>
    </head>

```

As shown above in the code snippet of my index.jsp file I used @page directive to import java classes and @include directive to include other jsp pages.

```

        <div class="col-md-2">
            <div class="list-group">
                <a href="index.jsp?category=all" class="list-group-item
list-group-item-action active" aria-current="true">
                    Categories
                </a>
                <%
                    for (Category category : catList) {%>

                        <a
href="index.jsp?category=<%=category.getCategoryId()%>&search=<%=reg%>"
class="list-group-item list-group-item-
action"><%=category.getCategoryTitle()%></a>
                        <%
                            }
                        <%
                    }
                </div>

```

In the above code snippet, I create anchor tag depending on the number of categories in the data. So, I use @Scriptlet tag to write the loop and the JSP will create HTML code dynamically. And I also use @expression tag to put data from the category object.

Similar kind of think I have done in all other pages for getting values and for dynamic generation of HTML code.

After successfully completing the View section we will move on to the Controller. This Controller will decide that for each request

### 3.3 Controller Section or Servlet

A servlet is a Java programming language class that is used to extend the capabilities of servers that host applications accessed by means of a request-response programming model. Although servlets can respond to any type of request, they are commonly used to extend the applications hosted by web servers. To make custom servlet it is mandatory to extend HTTP Servlet.

Servlet is an Interface containing 5 methods, 3 lifecycle method and 2 other methods. If we want to use Servlet interface directly, we need to implement all the methods. So, there are two Servlet classes that can be used to overcome the above problem.

- i. Generic Servlet: - It is used to handle any type of request and overriding service method is mandatory.
- ii. HttpServlet : It is used to handle only HTTP request and it is required to override doGet and doPost method.

In my project I used HttpServlet for server-side coding and uses HttpSession to communicate between two servlets. As shown below.

```
public class RegisterServlet extends HttpServlet {  
  
    protected void processRequest(HttpServletRequest request,  
HttpServletResponse response)  
        throws ServletException, IOException, Throwable {  
        response.setContentType("text/html;charset=UTF-8");  
        try (PrintWriter out = response.getWriter()) {  
            HttpSession httpSession = request.getSession();  
            /* TODO output your page here. You may use following sample code.  
*/  
            try {
```

#### 3.3.2 What is HttpSession?

HttpSession provides a way to identify a user across more than one page request or visit to a Web site and to store information about that user.

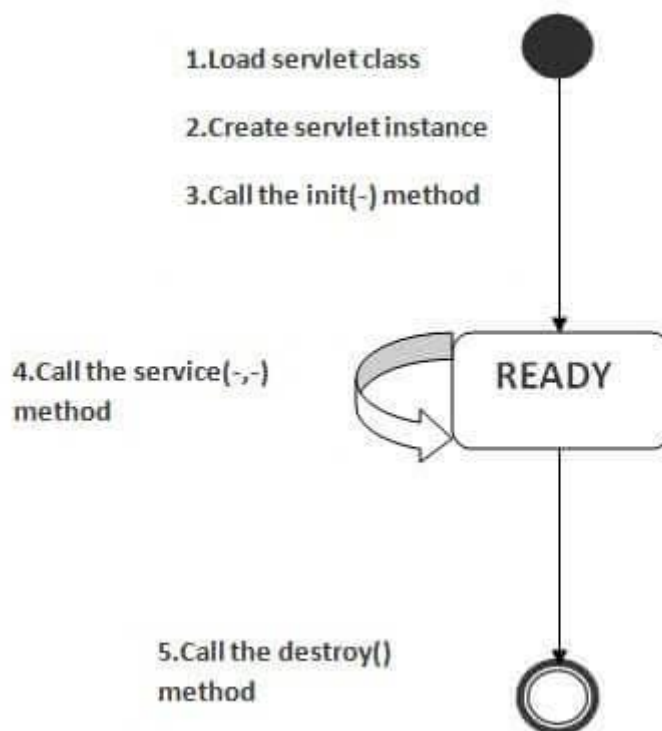
The servlet container uses this interface to create a session between an HTTP client and an HTTP server. The session persists for a specified time period, across more than one connection or page request from the user. A session usually corresponds to one user, who may visit a site many times. The server can maintain a session in many ways such as using cookies or rewriting URLs.

#### 3.3.3 Life Cycle of a Servlet (Servlet Life Cycle) \*\*

The web container maintains the life cycle of a servlet instance. Let's see the life cycle of the servlet:

- i. Servlet class is loaded.
- ii. Servlet instance is created.
- iii. Init () method is invoked.
- iv. service method is invoked.
- v. destroy method is invoked.

There are three states of a servlet: new, ready and end. The servlet is in new state if servlet instance is created. After invoking the init () method, Servlet comes in the ready state. In the ready state, servlet performs all the tasks. When the web container invokes the destroy () method, it shifts to the end state.



### 3.3.3 Request Dispatching

There are two ways to dispatch a request.

- i. By creating a web.xml file in WEB-INF folder which contains web container used to despatch request, contains URL-mapping for Servlet and Class.

- ii. Servlet Configuration Annotation: without writing code in web.xml we can write Annotation in Servlet to dispatch request.

In my project I used annotations to dispatch requests. E.g., there is a `registrationServlet.java` which is used for registration of new user and I use the annotation right above the class name. It's look like

```
@WebServlet(name = "com.myshop.myshop.servlets.RegisterServlet", urlPatterns = {"/RegisterServlet"})
```

Where the “name” is fully classified class name and “urlPattern” is the URL-mapping for the servlet.

I have used the same methods for all servlet class in my project for request dispatching. All the request and response are gone through servlet. It internally calls all the method to store data to data base or retrieve data from database. All authentication and data validation checks are done in the servlet.

## 3.4 Connecting the Database

As we completed the most of the implementation part now it is the time to connect the database with the project to make it fully functional. We will use JDBC and Hibernate as a ORM (object relation mapper). In this section we will discuss about the Hibernate and Database design.

### 3.4.1 Hibernate

Hibernate is an open-source object relational mapping (ORM) tool that provides a framework to map object-oriented domain models to relational databases for web applications.

Object relational mapping is based on the containerization of objects and the abstraction that provides that capacity. Abstraction makes it possible to address, access and manipulate objects without having to consider how they are related to their data sources.

The Hibernate ORM framework guides mapping Java classes to database tables and Java data types to SQL data types and provides querying and retrieval.

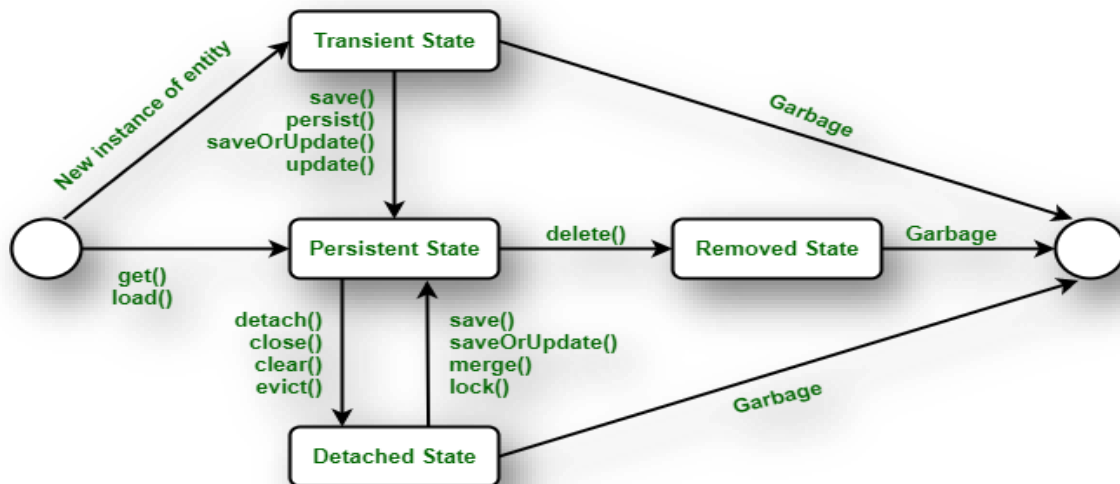
### 3.4.2 Hibernate object / Persistence lifecycle<sup>\*\*\*</sup>

In Hibernate, we can either create a new object of an entity and store it into the database, or we can fetch the existing data of an entity from the database. This entity is connected with the lifecycle and each object of entity passes through the various stages of the lifecycle.

There are mainly four states of the Hibernate Lifecycle:

- Transient State

- Persistent State
- Detached State
- Removed State



### State 1: Transient State

The transient state is the first state of an entity object. When we instantiate an object of a POJO class using the new operator then the object is in the transient state. This object is not connected with any hibernate session. As it is not connected to any Hibernate Session, so this state is not connected to any database table. So, if we make any changes in the data of the POJO Class then the database table is not altered.



There are two layouts in which transient state will occur as follows:

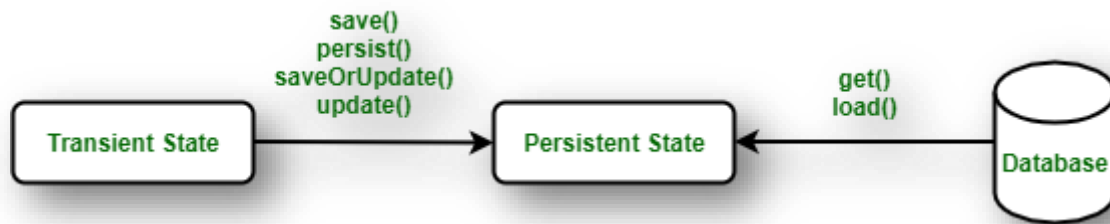
- When objects are generated by an application but are not connected to any session.
- The objects are generated by a closed session.

### State 2: Persistent State

Once the object is connected with the Hibernate Session then the object moves into the Persistent State. So, there are two ways to convert the Transient State to the Persistent State :

- Using the hibernated session, save the entity object into the database table.
- Using the hibernated session, load the entity object into the database table.

In this state, each object represents one row in the database table. Therefore, if we make any changes in the data then hibernate will detect these changes and make changes in the database table.



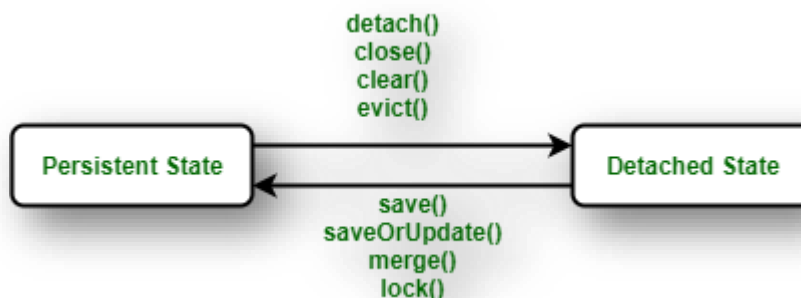
I used the Following two methods among all the methods for the persistent state:

- session.save(e);
- session.update(e);

### State 3: Detached State

For converting an object from Persistent State to Detached State, we either have to close the session or we have to clear its cache. object to a new hibernate session, we will use the following method as follow

- session.close()



### State 4: Removed State



In the hibernate lifecycle it is the last state. In the removed state, when the entity object is deleted from the database then the entity object is known to be in the removed state. It is done by calling the ***delete () operation***. As the entity object is in the removed state, if any change will be done in the data will not affect the database table.

```

public int saveUser(User user){
    Session openSession = factory.openSession();
    Transaction beginTransaction = openSession.beginTransaction();
    int userId = (int) openSession.save(user);
    beginTransaction.commit();
    openSession.close();
    return userId;
}

```

In the above example user is in transient state, after doing `openSession.save(user)` it goes to the persistence state, after doing the `openSession.close()` it goes to the detached state. There are 4 classes in the **com.myshop.myshop.dao** package `UserDao.java` for user, `CategoryDao.java` for category, `ProductDao.java` for product and `OrderDao.java` for order related CRUD operation.

The class below is used to create Session Factory which will again use to create a session to communicate with database. If a Session Factory is already created then it will not create another one. It will simply return the previous session factory.

```

public class FactoryProvider {
    private static SessionFactory factory;

    public static SessionFactory getFactory(){
        try{
            if(factory == null) {
                factory = new
Configuration().configure("hibernate.cfg.xml").buildSessionFactory();
            }

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

        return factory;
    }
}

```

### 3.4.3 Hibernate configuration

We have to inform hibernate which are the POJO classes in my project and how to map the classes to the required entities. In the configuration file there will be complete name of the classes and some other specifications is MySQL driver name, connect name, user id, password.

Behind the scene it will use JDBC to connect with database. Below I gave the configuration file details.

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE hibernate-configuration SYSTEM
"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
  <session-factory>
    <property name="connection.driver_class">com.mysql.jdbc.Driver</property>
    <property name="connection.url">jdbc:mysql://localhost:3306/myshop</property>
    <property name="connection.username">root</property>
    <property name="connection.password">SAURA</property>
    <property name="dialect">org.hibernate.dialect.MySQL8Dialect</property>
    <property name="hbm2ddl.auto">update</property>
    <property name="show_sql">>true</property>
    <mapping class="com.myshop.myshop.entites.User"/>
    <mapping class="com.myshop.myshop.entites.Category"/>
    <mapping class="com.myshop.myshop.entites.Product"/>
    <mapping class="com.myshop.myshop.entites.OrderHistory"/>
  </session-factory>
</hibernate-configuration>
```

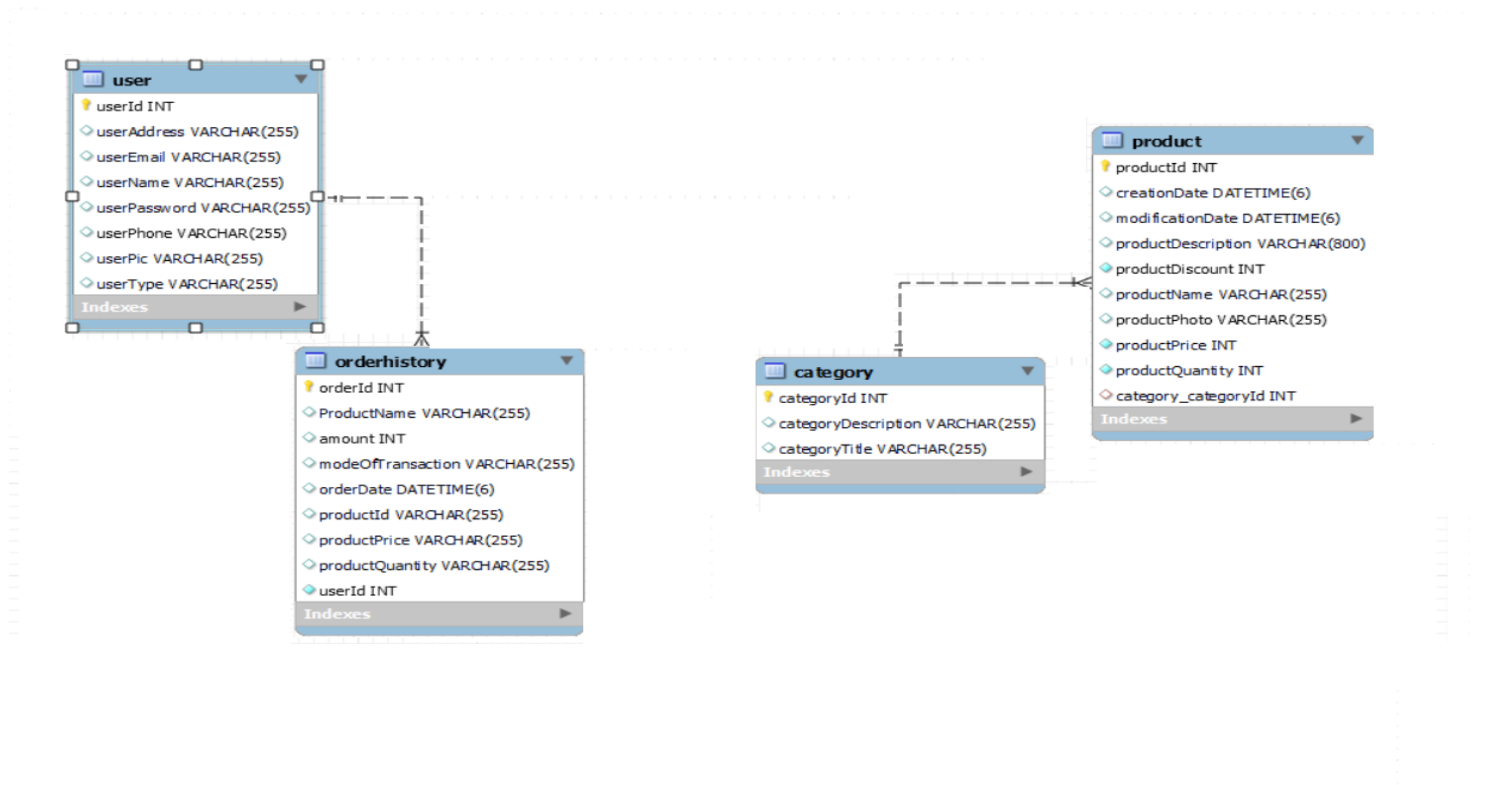
### 3.4.4 Database Design

In this project I made 4 tables User, Product, Category, OderHistory. User has the primary key userId which is again used as a foreign key in the OderHistory table to store the respective order of a user. In the Order table OrderId id the primary key.

Now in RDMS we cannot left a column blank or create Dynamic column so I gather all the selected product details, make a string separated by comma and store to the OrderHistory table. For example, all selected product names are concatenated together and separated by comma to create a string and store to the database. Same thing is done for productId, productPrice etc. That is why the productId in the OrderHistory table is not a foreign key. In the User table the userEmailAddress is a unique field. There is a many to one mapping between OrderHistory and User.

The Category table has a primary key which is a foreign key for the Product table as each product should belong to a particular category. The Product table has a primary key productId. A sample of the database is attached below. There is a many to one mapping between Product and Category.





## 3.4.5 Tables

### User Tables












Table Name:

Schema: **myshop**

Charset/Collation:

Engine:

Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
 <b>userId</b>	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
 <b>userAddress</b>	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 <b>userEmail</b>	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 <b>userName</b>	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 <b>userPassword</b>	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 <b>userPhone</b>	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 <b>userPic</b>	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 <b>userType</b>	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

## Category Table







Table Name:

Schema: **myshop**

Charset/Collation:

Engine:

Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
 <b>categoryId</b>	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
 <b>categoryDescription</b>	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 <b>categoryTitle</b>	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

## Product Table














Table Name:

Schema: **myshop**

Charset/Collation:

Engine:

Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
 productId	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
 creationDate	DATETIME(6)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 modificationDate	DATETIME(6)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 productDescription	VARCHAR(800)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 productDiscount	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 productName	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 productPhoto	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 productPrice	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 productQuantity	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 category_categoryId	INT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

## Order History Table













Table Name:

Schema: **myshop**

Charset/Collation:

Engine:

Comments:

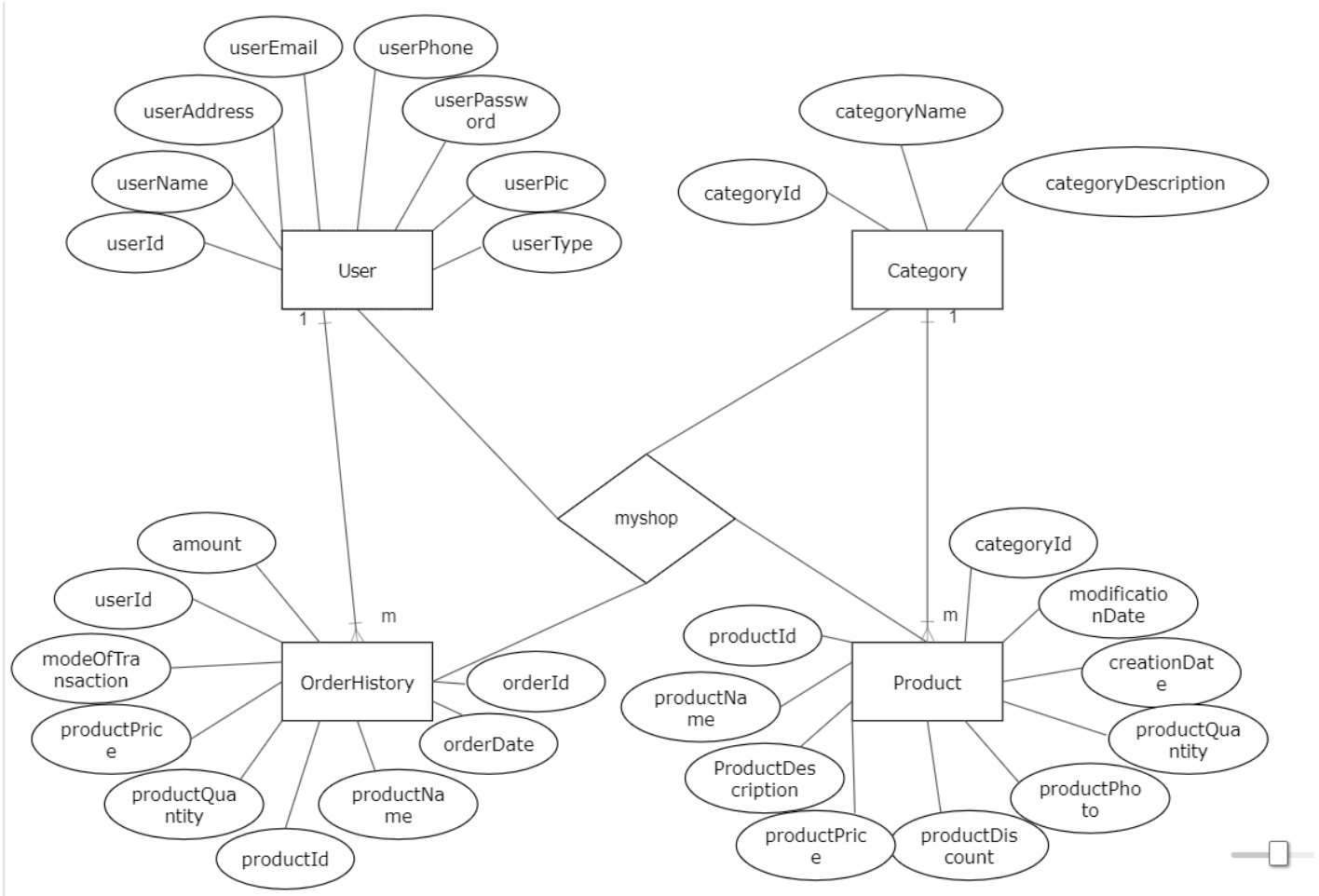
Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
 orderId	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
 ProductName	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 amount	INT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 modeOfTransaction	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 orderDate	DATETIME(6)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 productId	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 productPrice	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 productQuantity	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 userId	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

## 3.4.6 E-R Diagram

**ER Diagram** stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases. ER diagrams are created based on three basic concepts: entities, attributes and relationships.

ER Diagrams contain different symbols that use rectangles to represent entities, ovals to define attributes and diamond shapes to represent relationships.

At first look, an ER diagram looks very similar to the flowchart. However, ER Diagram includes many specialized symbols, and its meanings make this model unique. The purpose of ER Diagram is to represent the entity framework infrastructure.



### 3.5 Implementing the Mail Sending Section\*\*\*\*

This is the last section of the project here we will include the mail sending feature in the project. The main use of this section is to send notification to the user at the time of OTP verification, after successful log in, after successfully deleting the account and after successfully complete the order. I used the following APIs of java to send the mail

- jakarta.mail-api
- jakarta.mail

I also use Google secure App to let the web app send mail from my Gmail account.

```
Properties properties = new Properties();
    properties.put("mail.smtp.auth", true);
    properties.put("mail.smtp.starttls.enable", true);
    properties.put("mail.smtp.port", "587");
    properties.put("mail.smtp.host", "smtp.gmail.com");
```

The above section is used to set the smtp properties.

```
final String username = "user-name";
    final String password = "generated-password";
    //session
    Session session = Session.getInstance(properties, new Authenticator()
{
    @Override
    protected PasswordAuthentication getPasswordAuthentication() {
        return new PasswordAuthentication(username, password);
    }
});
```

The above section is for authenticate the mail.

```
Message message = new MimeMessage(session);
    message.setRecipient(Message.RecipientType.TO, new
InternetAddress(to));
    message.setFrom(new InternetAddress(from));
    message.setSubject(subject);
    message.setContent(text, "text/html");
    Transport.send(message);
    flag = true;
```

At last, the above section is used to send the message to the sender. Here we specify the subject of the mail, from and to mail addresses, body and the type of the body i.e., normal text or HTML.

# Testing and Test Results

E-Commerce testing is very important and crucial thing to test functionality of all buttons webpages and all the functions written. So, there are several parts of the testing functionality testing, usability testing, performance testing, Database testing. I will all the above thinks section by section.

## **Testing for Navigation and Search**

For the Navigation bar I tried all possible scenarios it works all the time and perfectly so functionality works fine. For the performance testing it took very less time to give the response for a particular action. There is no database related action available so we can't do database testing.

For the testing of the search bar, I used the following words to test that is it working or not and I put the corresponding result and the conclusion right below.

<b>Strings that are searched</b>	<b>result</b>
Samsung	success
Phone	success
Laptop	success
Dell	success
HP	success
Phone with 12' screen	fail
Cheapest laptop	fail
cloths	success
T shirts	success
12' screen	success
casual	success

So, this can perfectly do the single word search. For multiword it works for complete match.

## **Testing for Home page**

All functionality of the home page work perfectly but for the first time loading it took time more than 1 sec. for which we can se that its performance is not as good as the remaining page. All the products show in the proper order maintaining a row and column.

## **Testing for Registration page**

After performing the functionality test, I noticed that all functionality works fine but for performance test it fails as it takes more than 1.5 sec to register a user and if one fails it starts from very beginning. For database test I noticed that all success fully registered user details are stored in the database without any miss.

I tried to register with wrong email id but it gives an exception from the google that the mail id does not exist and the web site stops working. This is another problem that I face. So, I through an exception for that and catch the exception to return to some other page.

## Testing for Admin User page

Testing of this page is very impotent as all data bases are connected to this any error in this page will directly affect the database. For I tried to put the following data in the data base from the admin page and it works perfectly.

## Testing for Normal User Page

All functionality works fine. But I noticed a problem in database access for some time. While I try to delete an account, it shows **pessimistic lock for database** it occurs for 4-5 time out of 20 time. So, this page some issue. So, we can say that this page fails for 1/5 times. So, performance wise this page is not as good as other.

## Testing for User login Page

For this page my test cases are as follows

1. Correct email but wrong password
2. Correct email and correct password
3. Wrong email and wrong password

The correct userId was [@outlook.com](#) and password was **test@123**.

It fails to sign in for case 1 and case 3 but successfully sign in for the case 2.

## Test for the Password

- use at least 1 small case letter
- use at least 1 capital letter
- do not use spaces
- use at least 1 number
- use at least 1 special character

these all are the criteria for a password and I used the following test cases for the password word section test

Testcase1: test

Testcase2: Test

Testcase3: test@123

Testcase4: test123

Testcase5: test@

Testcase6: Test@123

For testcase1 to testcase5 it rejects to register as a user but for testcase6 it accepts as a user. So, all testcases pass in this section and worked perfectly.

# Conclusion and Future Scope

## Conclusion

This project is only a humble venture to satisfy the needs in a shop. Several user-friendly coding has also adopted. This package shall prove to be a powerful package in satisfying all the requirements of the organization. The objective of software planning is to provide a frame work that enables the manger to make reasonable estimates made within a limited time frame at the beginning of the software project and should be updated regularly as the project progresses.

This website provides a computerized version of shop manipulate system which will benefit the users as well as the visitor of the shop. It makes entire process online where users can search product, and buy various product. It also has a facility for common user by login into the system where user can login and can see status of ordered item as well request for items or give some suggestions. It provides the facility of admin's login where admins can add various item, review users' activity.

## Future Scope

The project has a very vast scope in future. The project can be implemented on intranet in future. Project can be updated in near future as and when requirement for the same arises, as it is very flexible in terms of expansion. With the proposed software of database Space Manager ready and fully functional the client is now able to manage and hence run the entire work in a much better, accurate and error free manner.

The following are the future scope for the project.

- Should be added payment gateway
- Can be added inventory management system
- Can be added multilingual to this site
- We can add **machine learning algorithms** to suggest similar products
- We can use **Elastic Search** as a search engine to search a product



# References

Following are the web sites that I have been used to create the project :-

- [1] Liferay Basic Training Course “Liferay Basic Training” by Brian Chan.
- [2] Liferay Module Project “Creating Web Portlet” – Liferay Documentation
- [3] Liferay Learn Front end, Backend, Business Studies by Liferay University
- [4] Chan, C. M., & Chan, S. C. (2016). Enhancing Organizational Performance with Enterprise Social Networking: A Case Study of Liferay. International Journal of Management, IT, and Engineering, 6(9), 243-260.
- [5] Liferay. (2021). Liferay DXP: Digital Experience Platform for Business. Retrieved from Liferay DXP Official Site
- [6] Liferay. (2021). Liferay Community
- [7] Liferay. (2021). Liferay Market Place

\* <https://developer.mozilla.org/en-US/docs/Glossary/MVC>

\*\* <https://www.javatpoint.com/life-cycle-of-a-servlet>

\*\*\* <https://www.javatpoint.com/hibernate-lifecycle>

\*\*\*\* <https://github.com/jakartaee/mail-api>