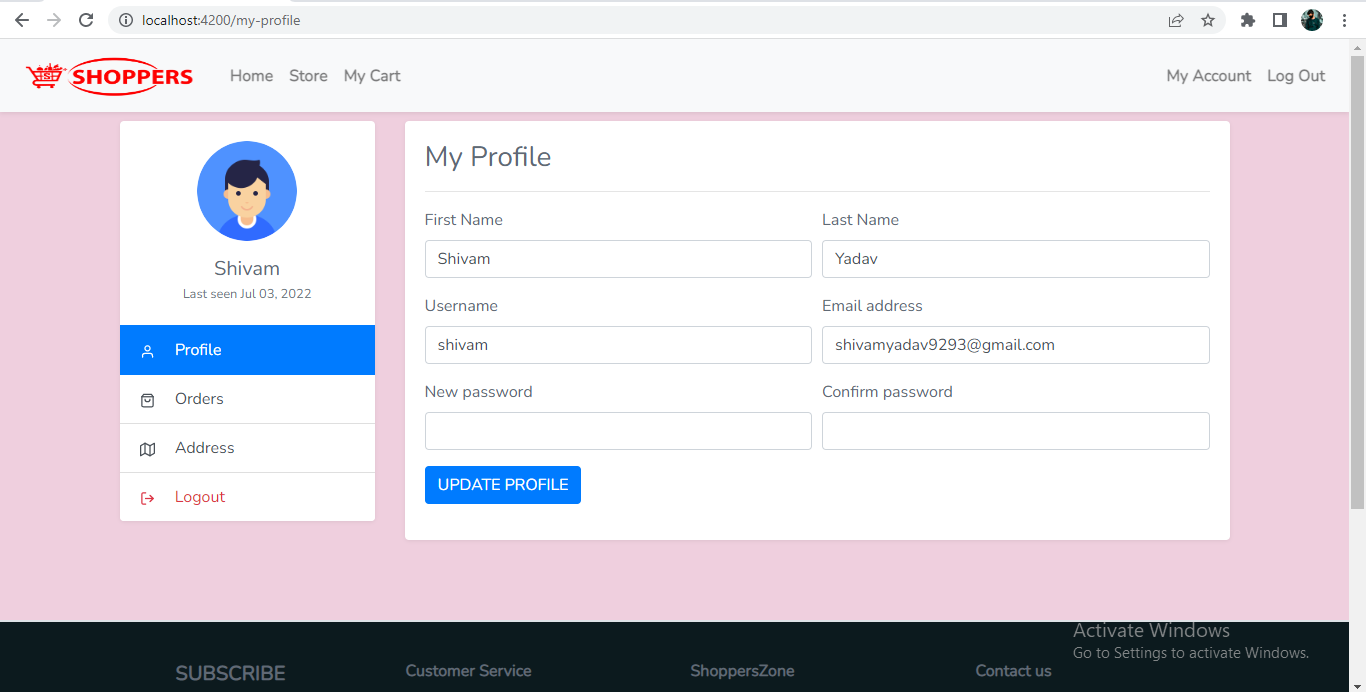
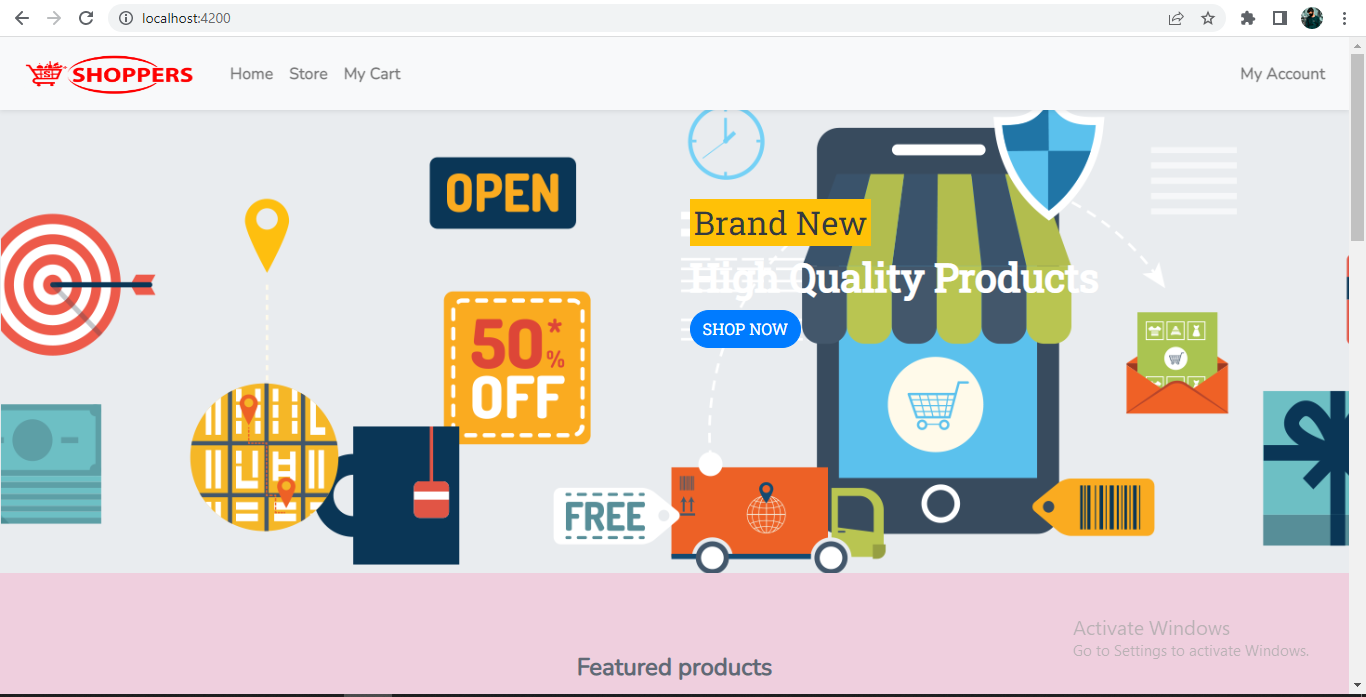
Software Requirement Specification(SRS) for

Online Shopping System(OSS)

**Documented by :**

**MUBASHIR P**

1. **Introduction**

**1 .1 Purpose:**

This document is meant to delineate the features of OSS, so as to serve as a guide to the developers on one hand and a software validation document for the prospective client on the other.

The Online Shopping System (OSS) for electronics item shop web application is intended to provide complete solutions for vendors as well as customers through a single get way using the internet. It will enable vendors to setup online shops, customer to browse through the shop and purchase them online without having to visit the shop physically. The administration module will enable a system administrator to approve and reject requests for new shops and maintain various lists of shop category.

**1 .2 Scope:**

This system allows the customers to maintain their cart for add or remove the product over the internet.

**1 .3 Definitions:**

OSS- Online Shopping System

SRS- Software Requirement Specification

GUI- Graphical User Interface

Stack holder- The person who will participate in system

Ex. Customer, Administrator, Visitor etc.

* 1. **Overview:**

This system provides an easy solution for customers to buy the product without going to the shop and also to shop owner to sale the product.

This proposed system can be used by any naïve users and it does not require any educational level, experience or technical expertise in computer field but it will be of good use if user has the good knowledge of how to operate a computer.

1. **Overall Description:**

The Online Shopping system (OSS) application enables customers to browse through the shops, and a system administrator to approve and reject requests for new shops and maintain lists of shop categories. Also the developer is designing an online shopping site to manage the items in the shop and also help customers to purchase them online without visiting the shop physically. The online shopping system will use the internet as the sole method for selling goods to its consumers.

**2.1 Product Perspective:**

This product aimed toward a person who don’t want to visit

the shop as he might don’t get time for that or might not interested in visiting there and dealing with lot of formalities.

**2.2 User Characteristics:**

User should be familiar with the terms like login, register, order system etc.

**2.3 Principle Actors:**

Principle actors are customer and administrator.

**2.4 General Constraints:**

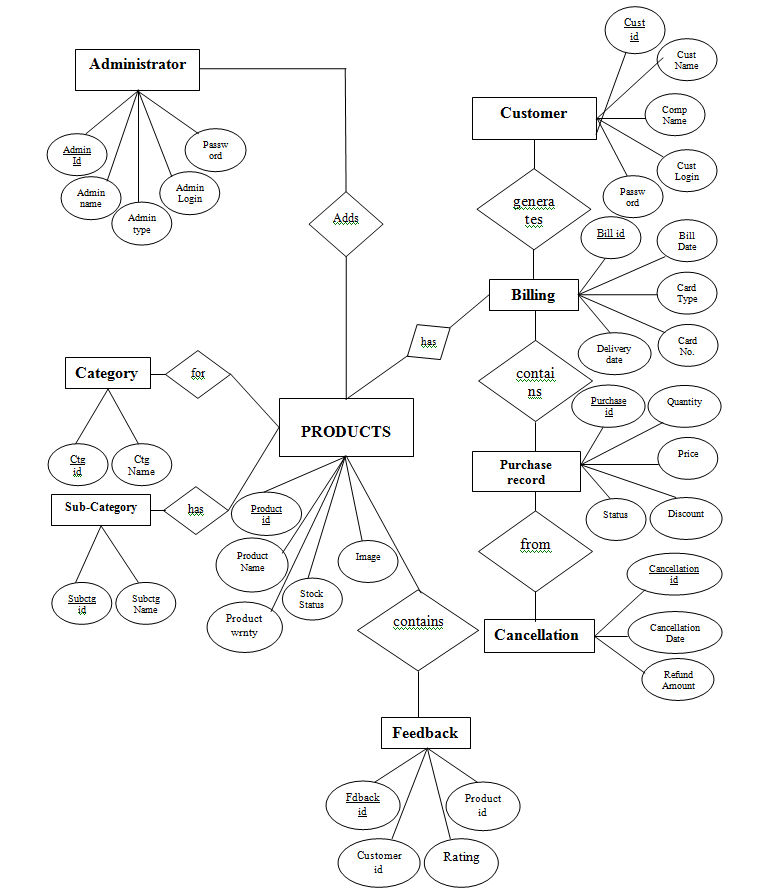
A full internet connection is required for OSS.

**2.5 Assumptions and Dependencies:**

Working of OSS need Internet Connection

**2.6**  **Product Functions:**

OSS supports this use case:



1. **Specific Requirements**:

* 1. **Functional Requirements:**

This section provides requirement overview of the system.

Various functional modules that can be implemented by the system will be –

**3.1 Description:**

* + 1. **Registration**

If customer wants to buy the product, then he/she must be

registered, unregistered user can’t go to the shopping cart.

* + 1. **Login**

Customer logins to the system by entering valid user id and password for the shopping.

* + 1. **Changes to Cart**

Changes to cart means the customer after login or registration can make order or cancel order of the product from the shopping cart.

* + 1. **Payment**

In this system we are dealing the mode of payment by Cash. We will

extend this to credit card, debit card etc. in the future.

* + 1. **Logout**

After ordering or surfing for the product customer has to logout.

* + 1. **Report Generation**

After ordering for the product, the system will generate invoice of the customer’s order with all the details.

**3.2 Non-Functional Requirements:**

Following Non-Functional Requirements will be there in the insurance to the internet:

* + - 1. Secure access to consumer’s confidential data.
      2. 24X7 availability**.**
      3. Better component design to get better performance at peak time.
      4. Flexible service based architecture will be highly desirable for future extension. Non-Functional Requirements define system properties and constraints.

Various other Non-Functional Requirements are:

* + - * **Security**
      * **Reliability**
      * **Maintainability**
      * **Portability**
      * **Extensibility**
      * **Reusability**
      * **Resource utilization**
      * **Compatibility**
  1. **Performance Requirements:**

In order to maintain an acceptable speed at maximum number of uploads allowed from a particular customer as any number of users can access to the system at any time.

Also the connections to the servers will be based on the attributes of the user like his location and server will be working 24X7 times.

* 1. **Technical Issues:**

This system will work on client-server architecture. It will require an internet server and which will be able to run spring application. The system should support some commonly used browser such as IE, Mozilla Firefox, chrome etc.

1. **Interface Requirement:**

Various interfaces for the product could be-

1). Login Page

2). Registration Form

3). There will be a screen displaying information about product that the shop having.

4). If the customers, select the buy button then another screen of shopping cart will be opened.

5). After ordering for the product, the system will have sent one copy of the bill to the customer’s Email address

**5. System Design Specification:**

**5.1 Architecture Design:**

5.1.1 Data Flow Diagram(DFD):

It is a way of representing system requirements in graphical form; this led to modular design. A DFD describes a data flow(logical) rather than how they are processed. So they do not depend upon

software, hardware, data structure or file organization. It is also known as ‘bubble sort’.

A DFD is a structured analysis and a design tool that can be used for flowcharting in place of, or in association with, information-oriented and process oriented system flowcharts.

A DFD is considered as an abstract of the logic of information-oriented or process-oriented system flowchart. The four basic symbols used to construct data flow diagrams are-

A rectangle represents a data source or destination.

A directed line represents flow of data.

An Oval represents a process that transforms into

streams.

An Open ended rectangle represents storage.

The points at which data is transformed are called as nodes. The principle processes that take place at nodes are:

1.Combining data streams

2.Splitting data streams

3.Modifiying data streams

**CONTEXT ANALYSIS DIAGRAM(CAD)**

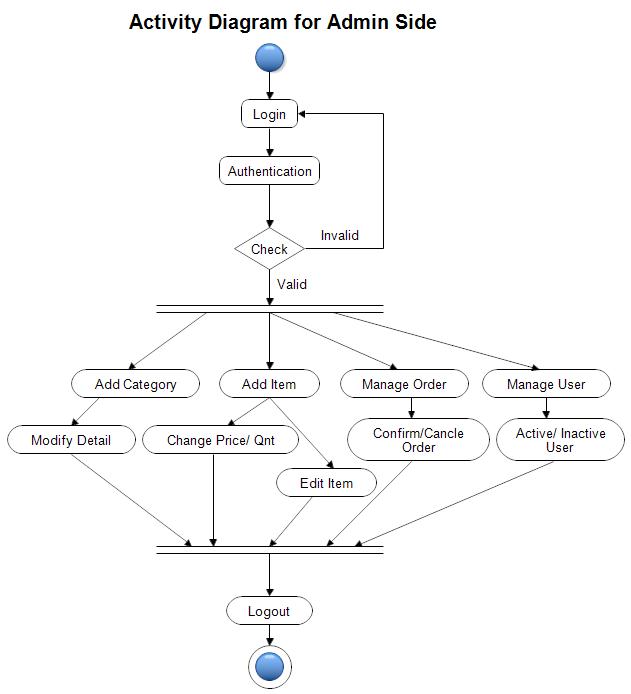
ONLINE

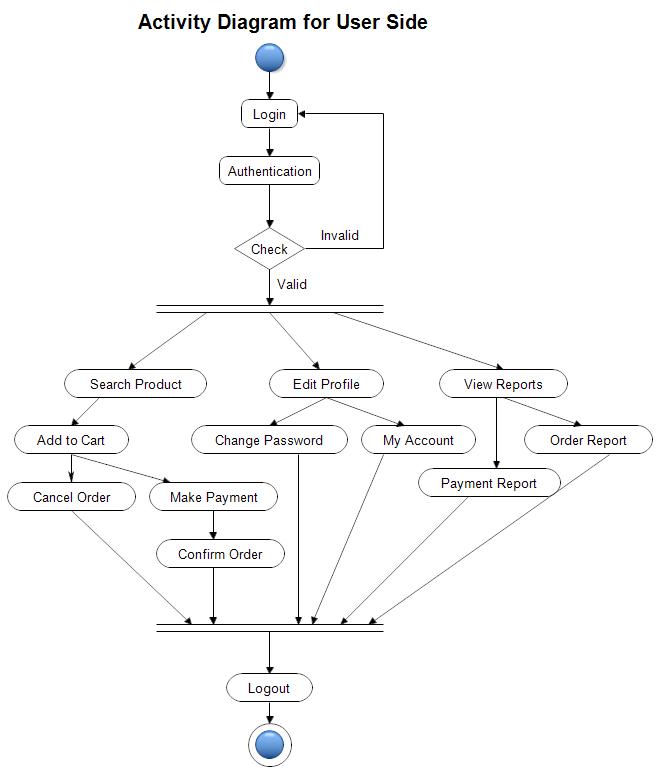
SHOPPING

SYSTEM

ADMINISTRATOR

CUSTOMER

[](https://www.youtube.com/coderbaba)

[](https://www.youtube.com/coderbaba)

**USE-CASE-DIAGRAM-CUSTOMER**

PROD\_NAME

PRICE

QUANTIT

Y

PRO\_ID

**PRODUCT**

**BUY PRODUCT**

**E-mail address**

**ADDRESS**

**PASSWORD**

**USER\_ID**

**USER\_NAME**

**PHONE\_NO**

**ORDERS**

**CUSTOMER**

**DETAILS**

**DELIVERY**

**PROFILEeE**

**MAKE PAYMENT**

[](https://www.youtube.com/coderbaba)

**SOFTWARE REQUIREMENTS:**

1. **Oracle/SQL DBMS-** It allows combination, extraction, manipulation and organization of data in the voters’ database. It is platform independent and therefore can be implemented and used across several such as Windows, Linux server and is compatible with various hardware mainframes. It is fast in performance, stable and provides business value at a low cost.
2. **Eclipse IDE-** Eclipse is an integrated development environment (IDE) used in computer programming. It contains a base workspace and an extensible plug-in system for customizing the environment. It is the second-most-popular IDE for Java development, and, until 2016, was the most popular. Eclipse is written mostly in Java and its primary use is for developing Java applications but it may also be used to develop applications in other programming languages via plug-ins, including Ada, ABAP, C, C++, C#,
3. **Web browsers**: Mozilla Firefox, Google chrome, Opera and Internet Explorer.
4. **FRONT – END AND BACK-END**
5. In their most general meanings, the terms front end and back end refer to the initial and the end stages of a process flow. In [software design](http://en.wikipedia.org/wiki/Software_design), the **front-end** is the part of a software system that deals with the user, and the **back-end** is the part that processes the input from the front-end. The separation of software systems into "front ends" and "back ends" is a kind of [abstraction](http://en.wikipedia.org/wiki/Abstraction_(computer_science)) that helps to keep different parts of the system separated. The general idea is that the front-end is responsible for collecting input from the user, which can be in a variety of forms, and processing it in such a way that it conforms to a specification that the back-end can use. The connection of the front-end to the back-end is a kind of [interface](http://en.wikipedia.org/wiki/Interface).
6. Front-end and back-end are terms used to characterize program interfaces and services relative to the initial user of these interfaces and services. (The "user" may be a human being or a program.) A ”front-end” [application](http://searchwebservices.techtarget.com/sDefinition/0,,sid26_gci211585,00.html) is one that application users interact with directly. A "back-end" application or program serves indirectly in support of the front-end services, usually by being closer to the required resource or having the capability to communicate with the required resource. The back-end application may interact directly with the front-end or, perhaps more typically, is a program called from an intermediate program that mediates front-end and back-end activities. These terms acquire more special meanings in particular areas: -
7. (1) For [software applications](http://www.webopedia.com/TERM/f/application.html), front end is the same as [user interface](http://www.webopedia.com/TERM/f/user_interface.html).
8. (2) In [client/server](http://www.webopedia.com/TERM/f/client_server_architecture.html) applications, the [client part](http://www.webopedia.com/TERM/f/client.html) of the [program](http://www.webopedia.com/TERM/f/program.html) is often called the front end and the server part is called the back end.
9. (3) [Compilers](http://www.webopedia.com/TERM/f/compiler.html), the programs that translate [source code](http://www.webopedia.com/TERM/f/source_code.html) into [object code](http://www.webopedia.com/TERM/f/object_code.html), are often composed of two parts: a front end and a back end. The front end is responsible for checking [syntax](http://www.webopedia.com/TERM/f/syntax.html) and detecting errors, whereas the back end performs the actual translation into object code.

**Technologies Used:**

**Front End:**

**HTML:**

HTML (Hypertext Markup Language) is the code that is used to structure a web page and its content. For example, content could be structured within a set of paragraphs, a list of bulleted points, or using images and data tables.

**CSS:**

Stands for "Cascading Style Sheet." Cascading style sheets are used to format the layout of Web pages. They can be used to define text styles, table sizes, and other aspects of Web pages that previously could only be defined in a page's HTML.

**JavaScript:**

JavaScript (JS) is a scripting language, primarily used on the Web. It is used to enhance HTML pages and is commonly found embedded in HTML code. JavaScript is an interpreted language. Thus, it doesn't need to be compiled. JavaScript renders web pages in an interactive and dynamic fashion. This allowing the pages to react to events, exhibit special effects, accept variable text, validate data, create cookies, detect a user’s browser, etc.

**Database:**

**Oracle SQL:**

SQL (pronounced sequel) is the **set-based, high-level declarative computer language** with which all programs and users access data in an Oracle database. Although some Oracle tools and applications mask SQL use, all database tasks are performed using SQL.

**Backend:**

***Spring Boot :-***

Spring Boot is an open source Java-based framework used to create a micro Service. It is developed by Pivotal Team and is used to build stand-alone and production ready spring applications. Spring Boot provides a good platform for Java developers to develop a stand-alone and production-grade spring application that you can **just run**. You can get started with minimum configurations without the need for an entire Spring configuration setup.

Spring Boot automatically configures your application based on the dependencies you have added to the project by using **@EnableAutoConfiguration** annotation. For example, if MySQL database is on your classpath, but you have not configured any database connection, then Spring Boot auto-configures an in-memory database.

The entry point of the spring boot application is the class contains **@SpringBootApplication** annotation and the main method.

Spring Boot automatically scans all the components included in the project by using **@ComponentScan** annotation.

Advantages

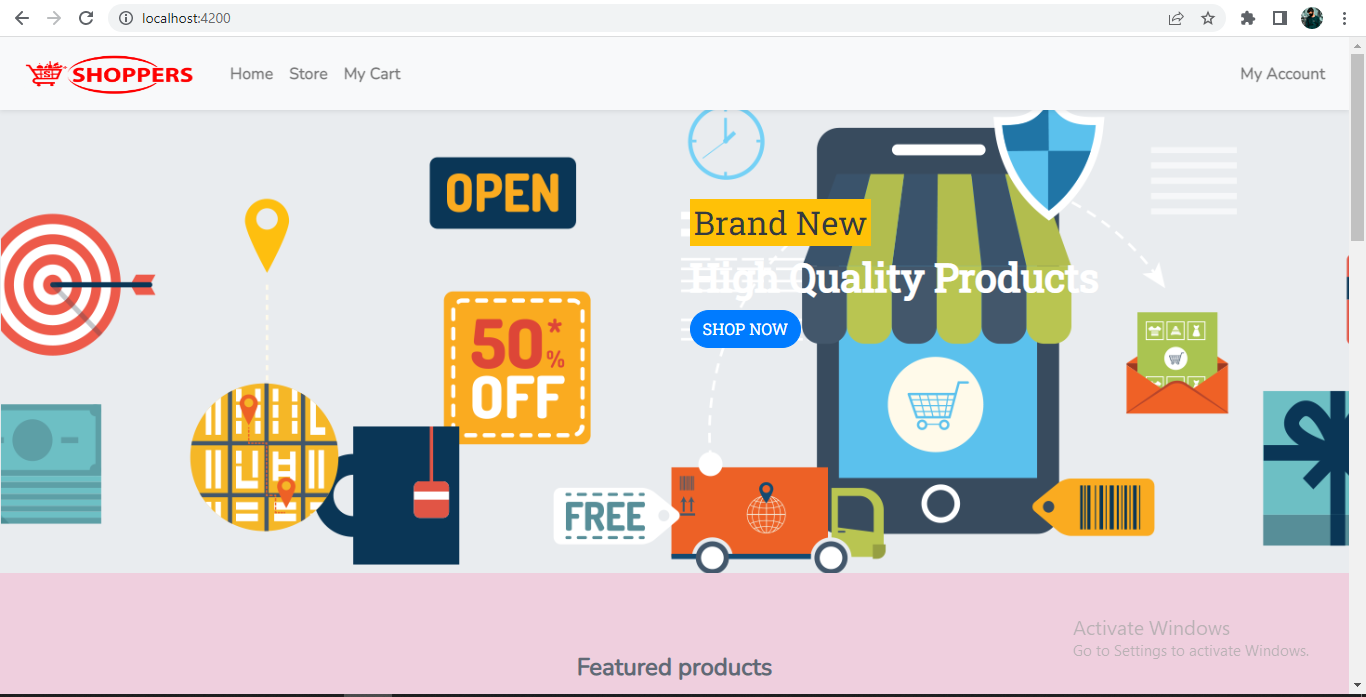
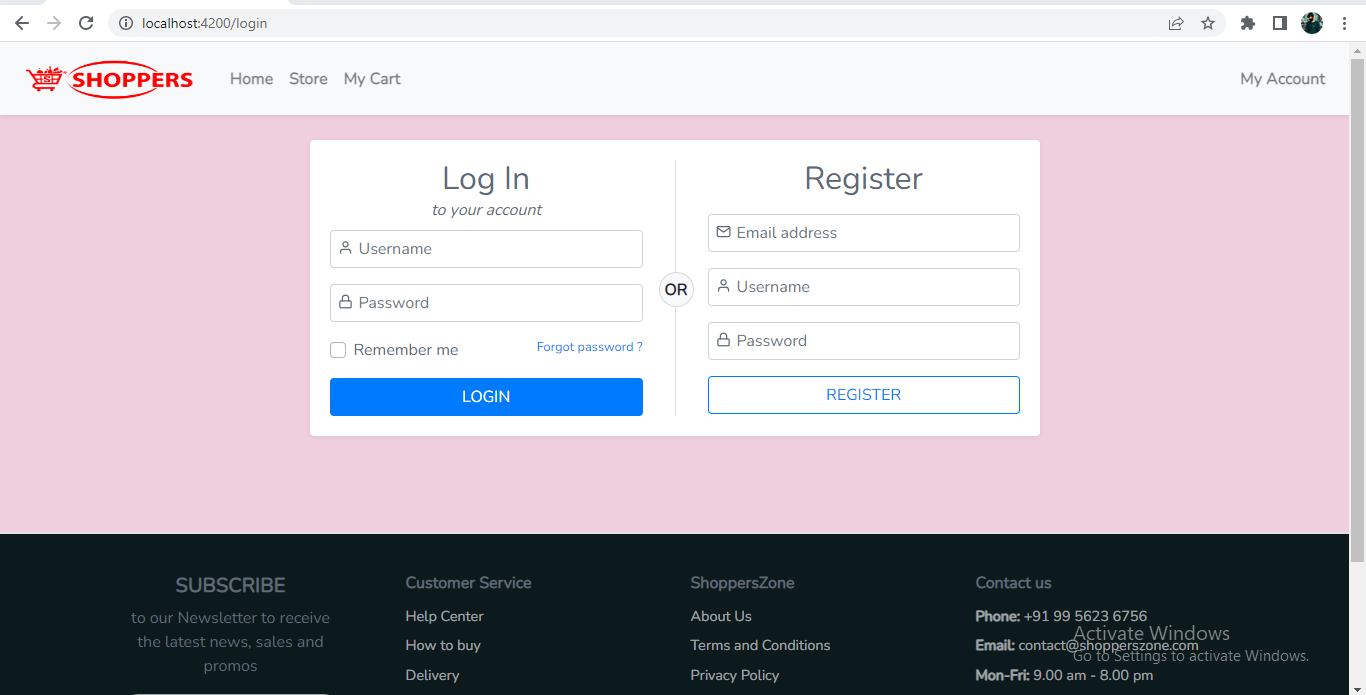
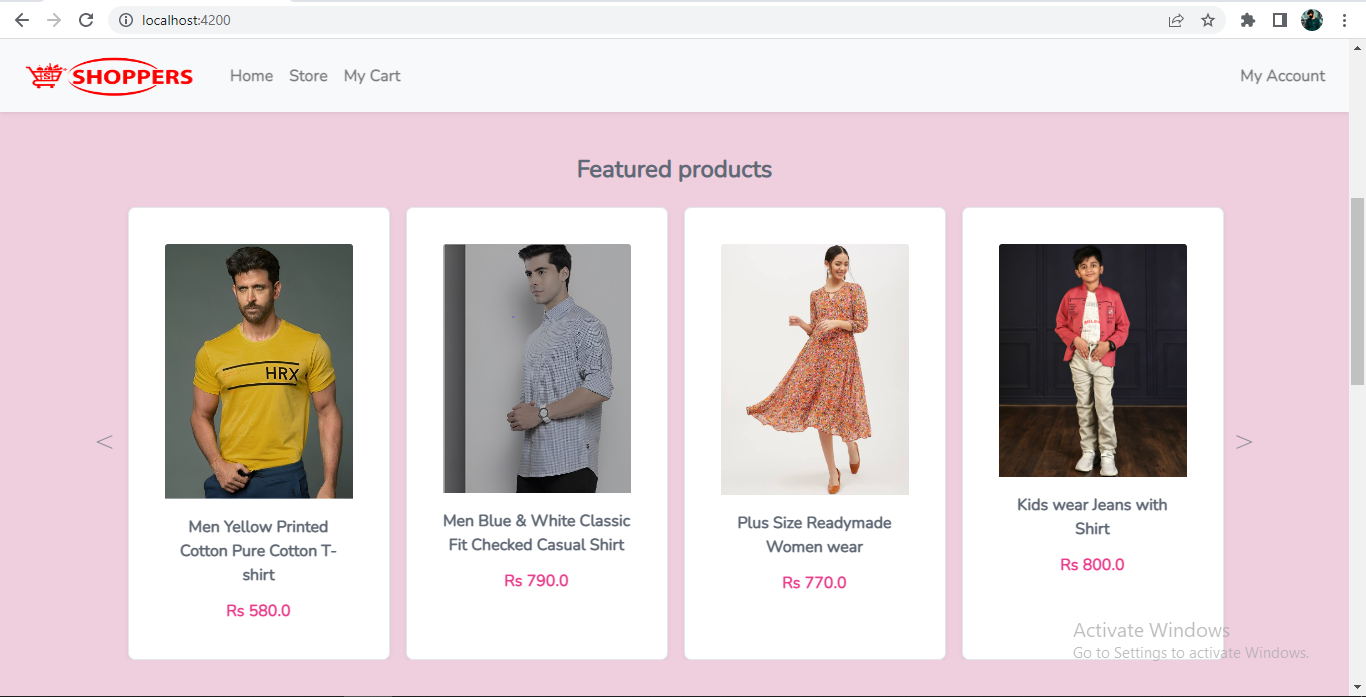
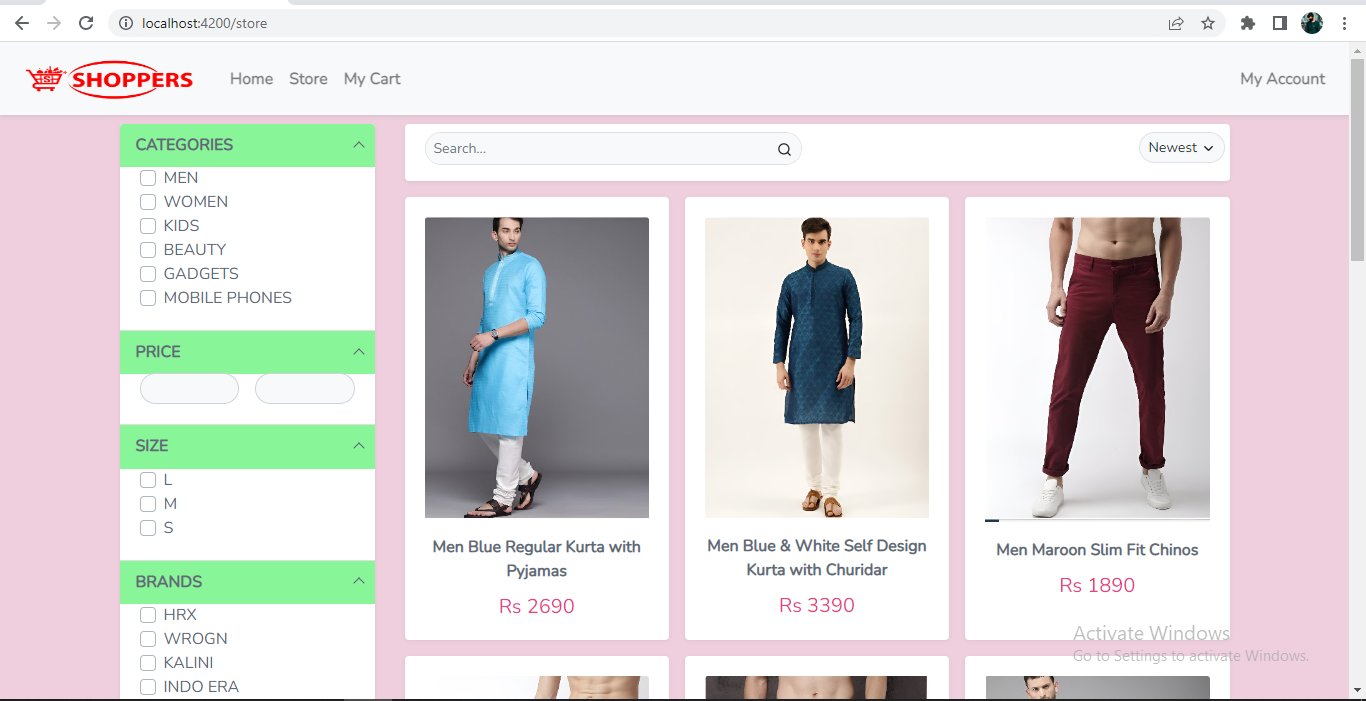
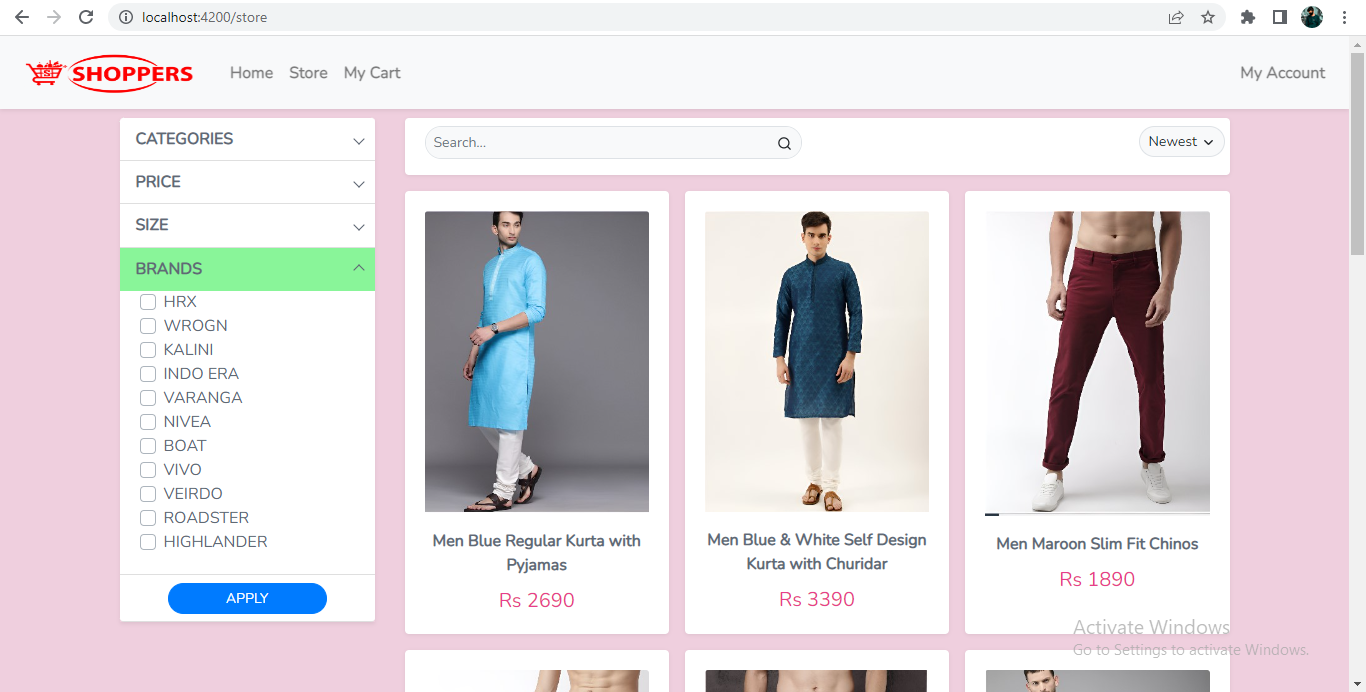
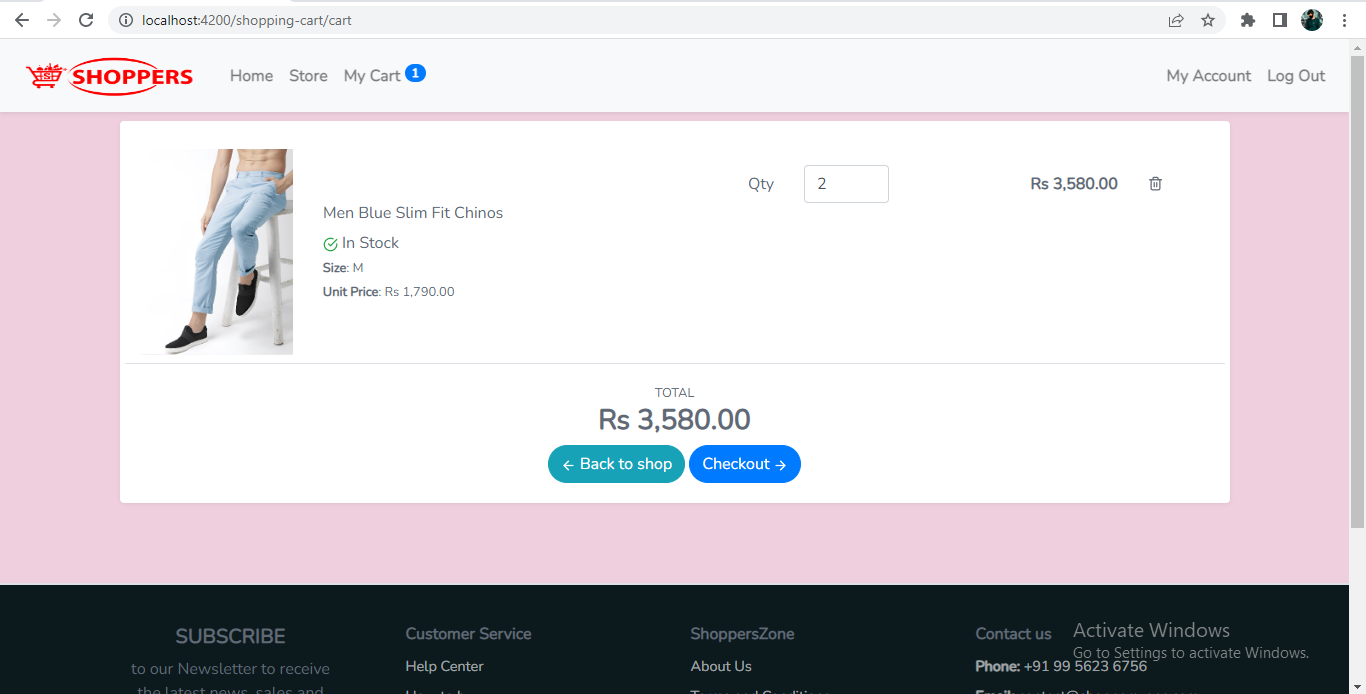
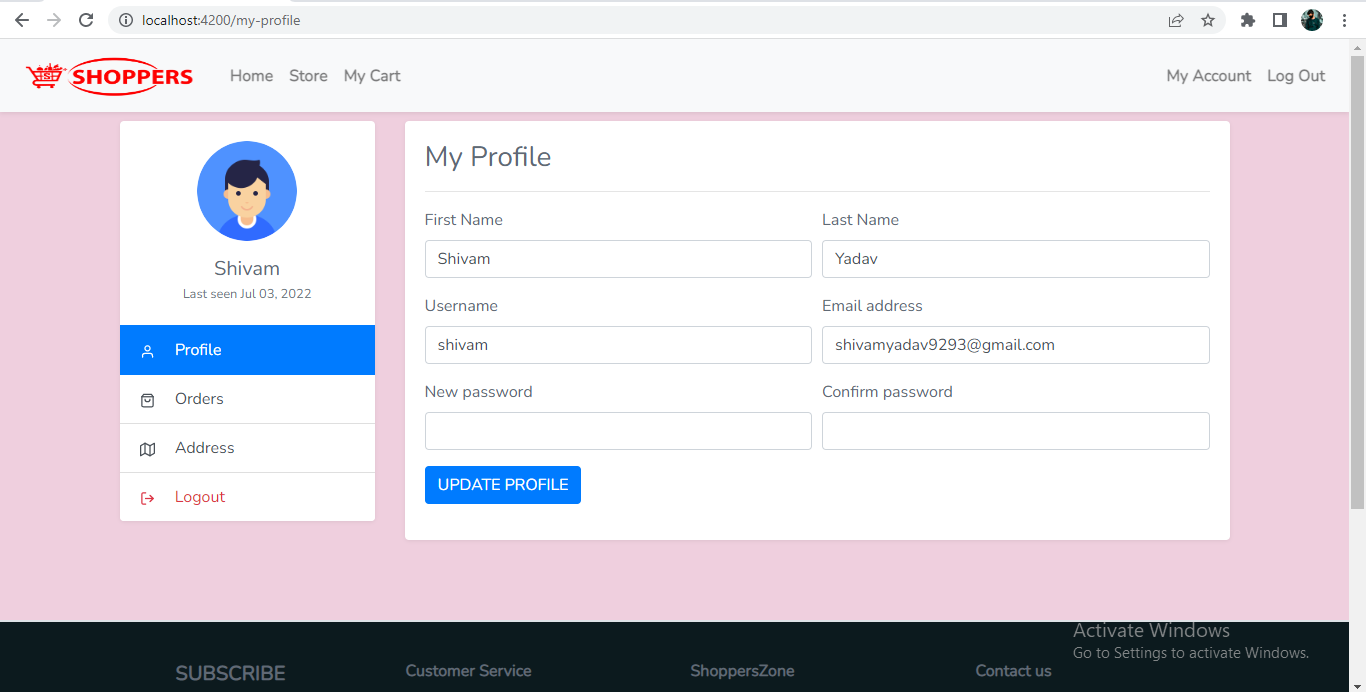
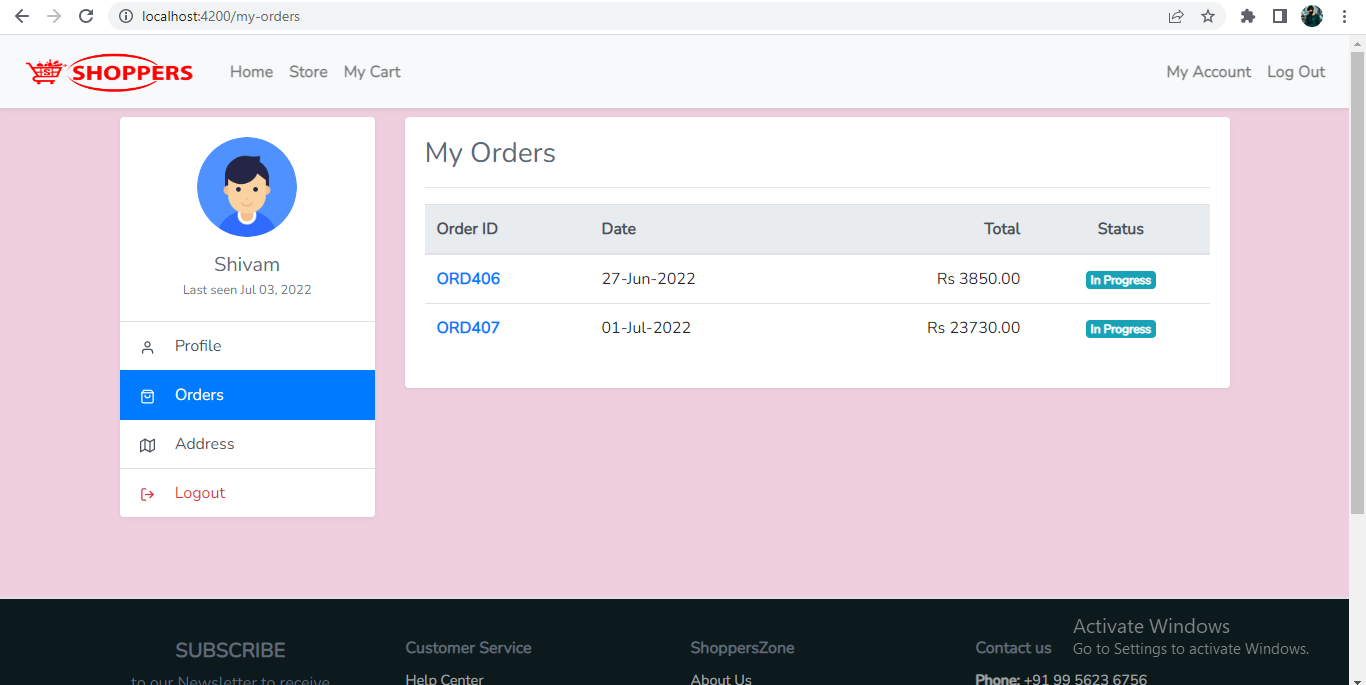
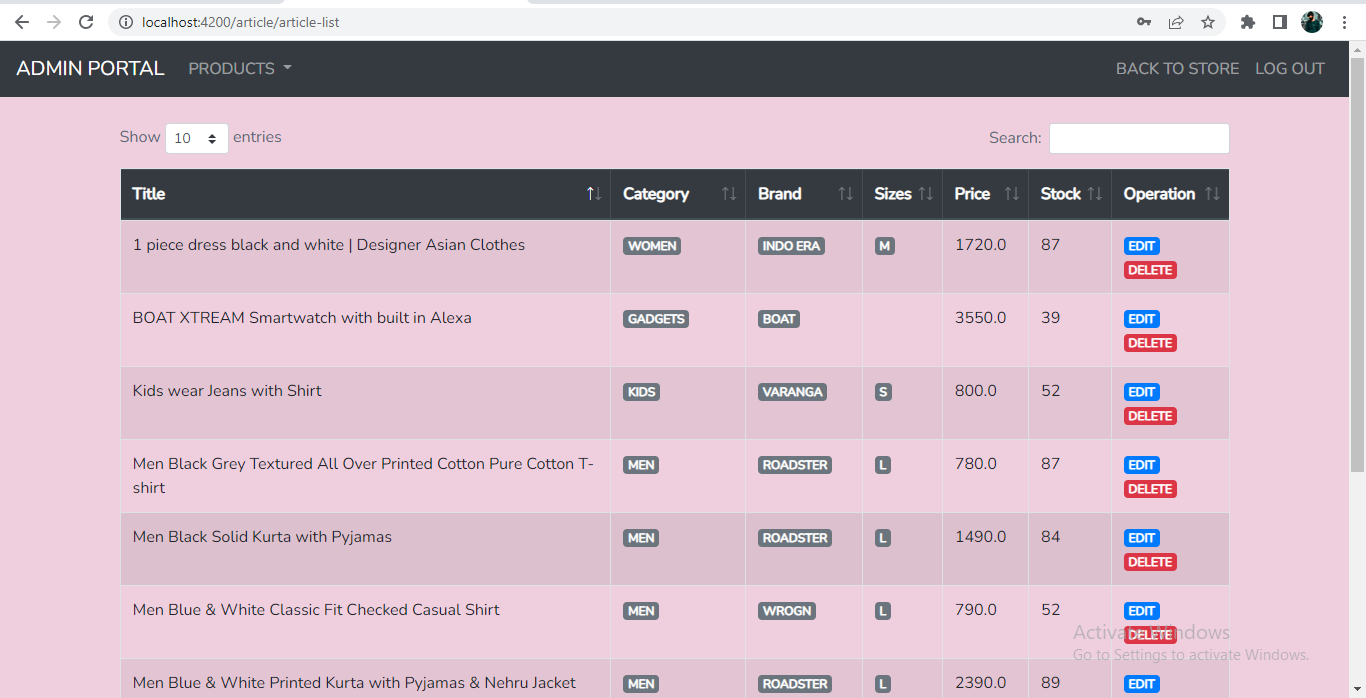
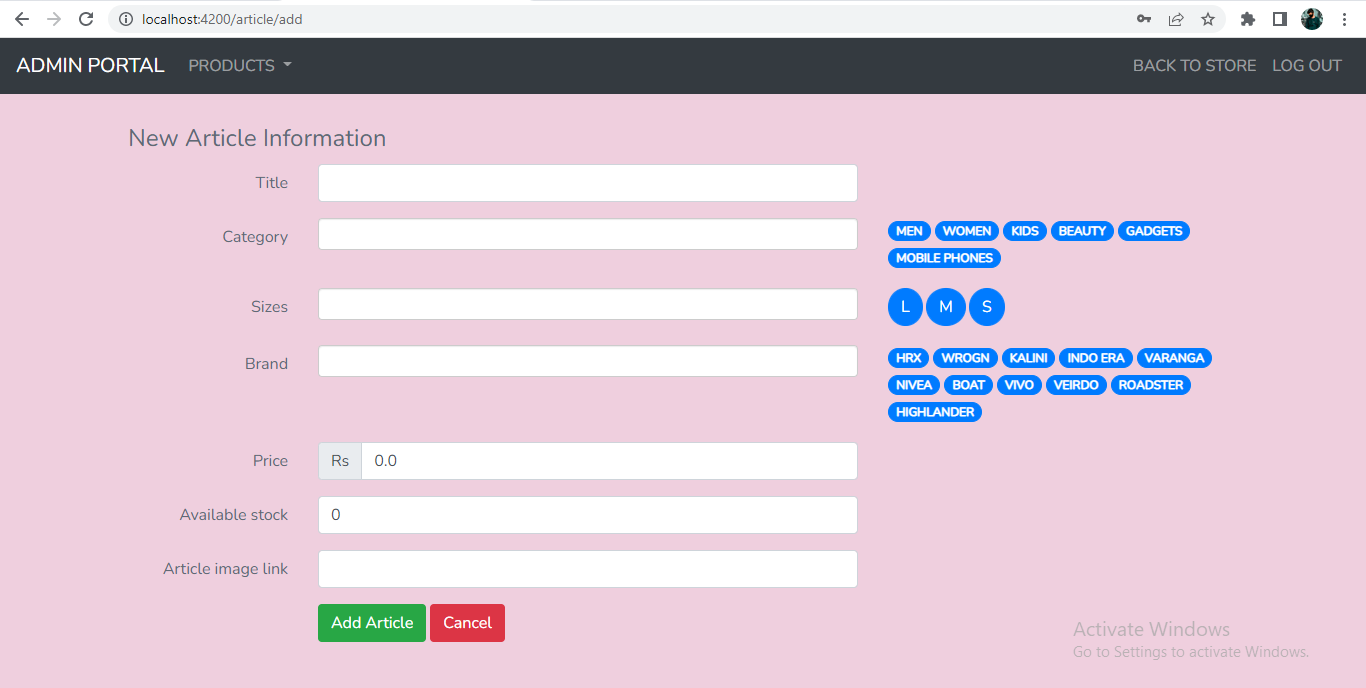
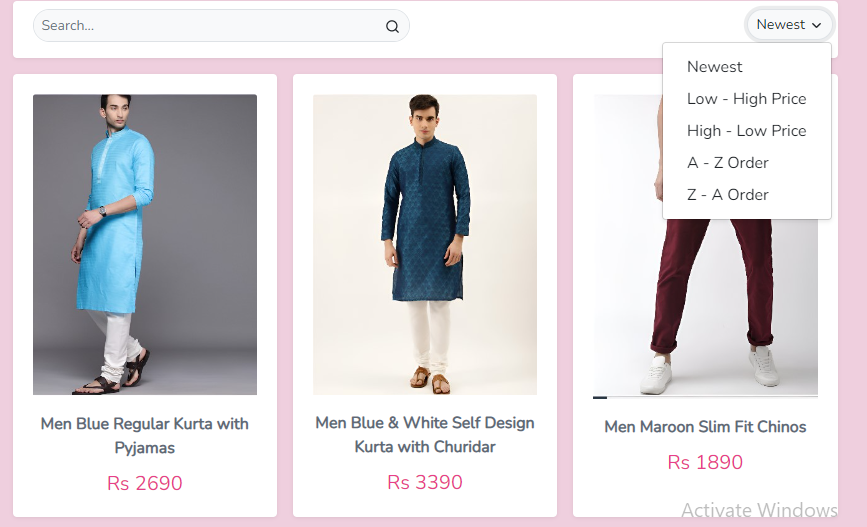
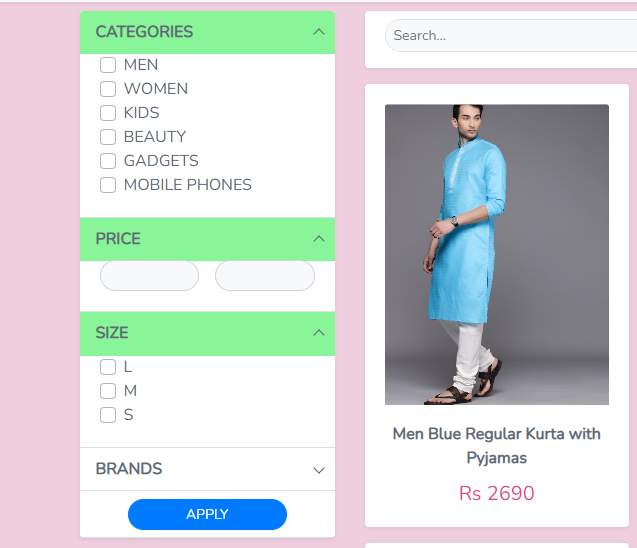
Spring Boot offers the following advantages to its developers −

* Easy to understand and develop spring applications
* Increases productivity
* Reduces the development time

***Spring Dependencies Used :***

* spring-boot-starter-web
* spring-boot-starter-jdbc
* spring-boot-starter-data-jpa
* mysql-connector-java
* spring-boot-starter-security
* spring-boot-devtools

**Snapshots**

1. ***Home Page***
2. ***Login & Register***
3. ***Featured Products***
4. ***Store***
5. ***Store\_Categories***
6. ***Cart-Page***
7. ***User-Profile***
8. ***User-Orders***
9. ***Admin\_Portal***
10. ***Admin\_Add\_Article***
11. ***Sorting\_Articles***
12. ***Sorting\_by\_price, Category***

