

Make an E-commerce Website for Sporty Shoes

This document contains the following.

- Project and developer details
- Sprint planning and tasks achieved
- Core concepts used in the project
- Flowchart of the application
- Links to the GitHub repository

Project objective:

As a Full Stack Developer, complete the features of the application by planning the development and pushing the source code to the GitHub repository.

Background of the problem statement:

Sporty Shoes is a company that manufactures and sells sports shoes. They have a walk-in store, and now, they wish to launch their e-commerce portal sportyshoes.com.

You're asked to develop a prototype of the application. It will be then presented to the relevant stakeholders for budget approval. Your manager has set up a meeting where you're asked to do the following:

- Presenting the specification document which has the product's capabilities, appearance, and user interactions
- Setting up Git and GitHub account to store and track your enhancements of the prototype
- Explaining the Java concepts used in the project
- Discussing the generic features of the product:
- There will be an admin to manage the website. An administrator login will be required to access the admin page.

The admin should be able to change his password if he wants, he should be able to:

- Manage the products in the store including categorizing them
- Browse the list of users who have signed up and be able to search users
- See purchase reports filtered by date and category

Sprint planning and tasks achieved

Sprint 1:

- I. Create a welcome page or home page displaying SportyShoes and name of developer.
- II. Take user input to select one of the options from the navbar.
- III. Clicking on Admin login tab will redirect to admin login page for asking input as username and password.

Release product for feedback over the frontend part of home page and admin login page.

Sprint 2:

After successful admin login -

- I. Add options to admin login page for further activities like
 - a. Change password and either continue on page or logout and login again with new credentials.
 - b. Managing products available in store, create/remove categories, create/remove products from existing categories.
 - c. Check list of signed up clients of website and search any particular client by name or id.
 - d. See purchase report filtered by date and category.
- II. Create webpage for the above-mentioned activities and interlink them as necessary.

Release product for feedback with all added webpages.

Sprint 3:

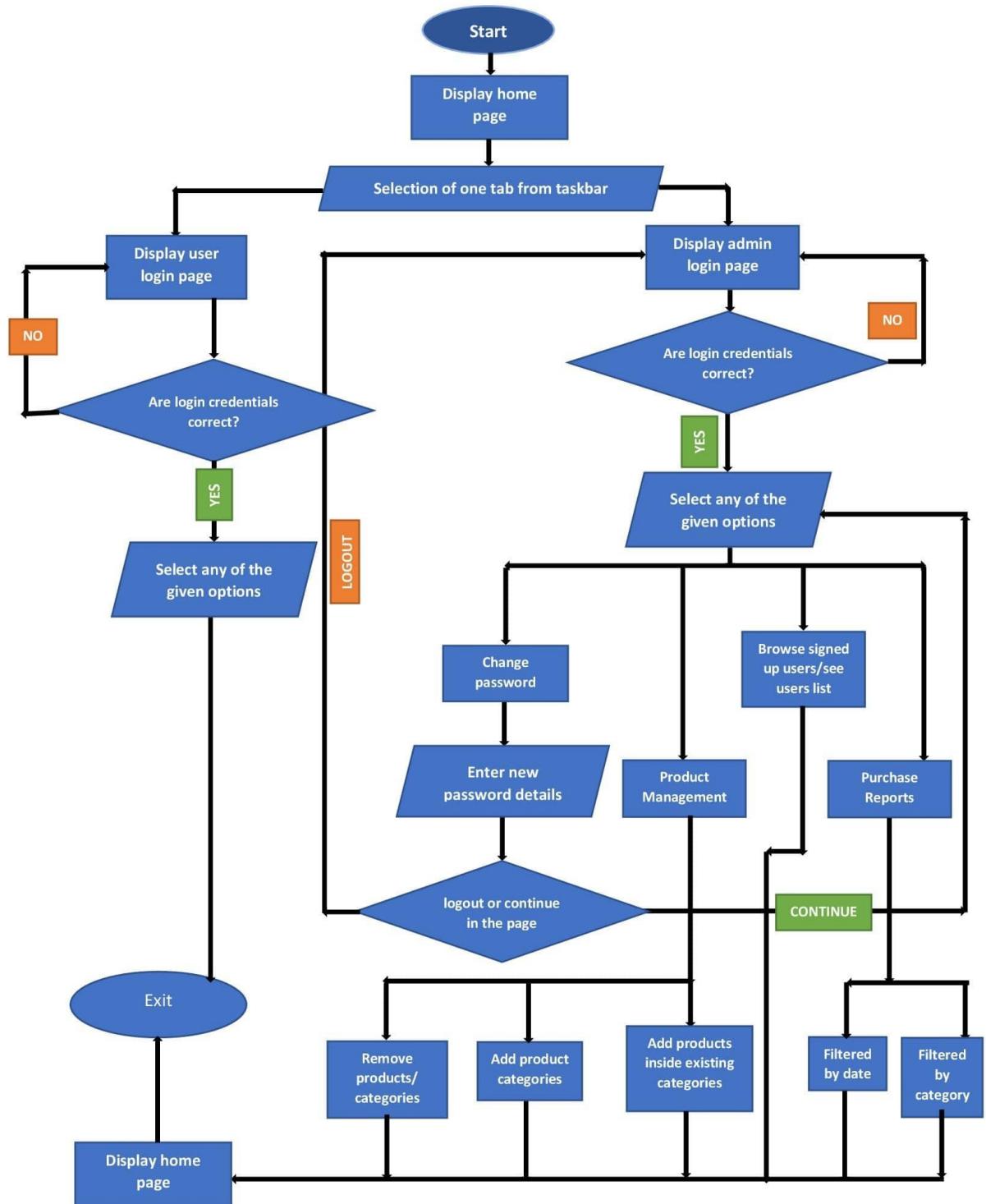
Create a program to store all the inputs given by the admin to a database (assigned to the SportyShoes). Develop relation between frontend developed in sprint2 with the database to store and retrieve data dynamically.

Release the product for feedback with all required database activities.

Flow of Application:

1. Welcome screen with name of website and developer.
2. Ask for user input to select admin login or user login.
3. For admin login, allow user to select one of the following options.
 - a. Change password and either continue on page or logout and login again with new credentials.
 - b. Managing products available in store, create/remove categories, create/remove products from existing categories.
 - c. Check list of signed up clients of website and search any particular client by name or id.
 - d. See purchase report filtered by date and category.
4. For client login, allow client to surf for available options.
 - a. Select options to choose categories/products and add them to cart.
 - b. Make checkout option to buy products added in cart.
5. Display home page or exit the website.

Flow Chart



Pushing the code to GitHub Repository:

- Open Git Bash and navigate to the folder where you have created your files.
`cd E-commerce \MahendraPhase3-SecondProject\E-commerce website`
- Initialize the repository using the below command `git init`
- Add all the files to your git repository using the below command `git add .`
- Commit the changes using the below command `git commit -m "Initial commit"`
- Add the URL for the remote repository where your local repository will be pushed `git remote add origin https://github.com/Mahendra1272/E-commerce.git`
- Push the files to the folder you initially created using below command `git push -u origin master`

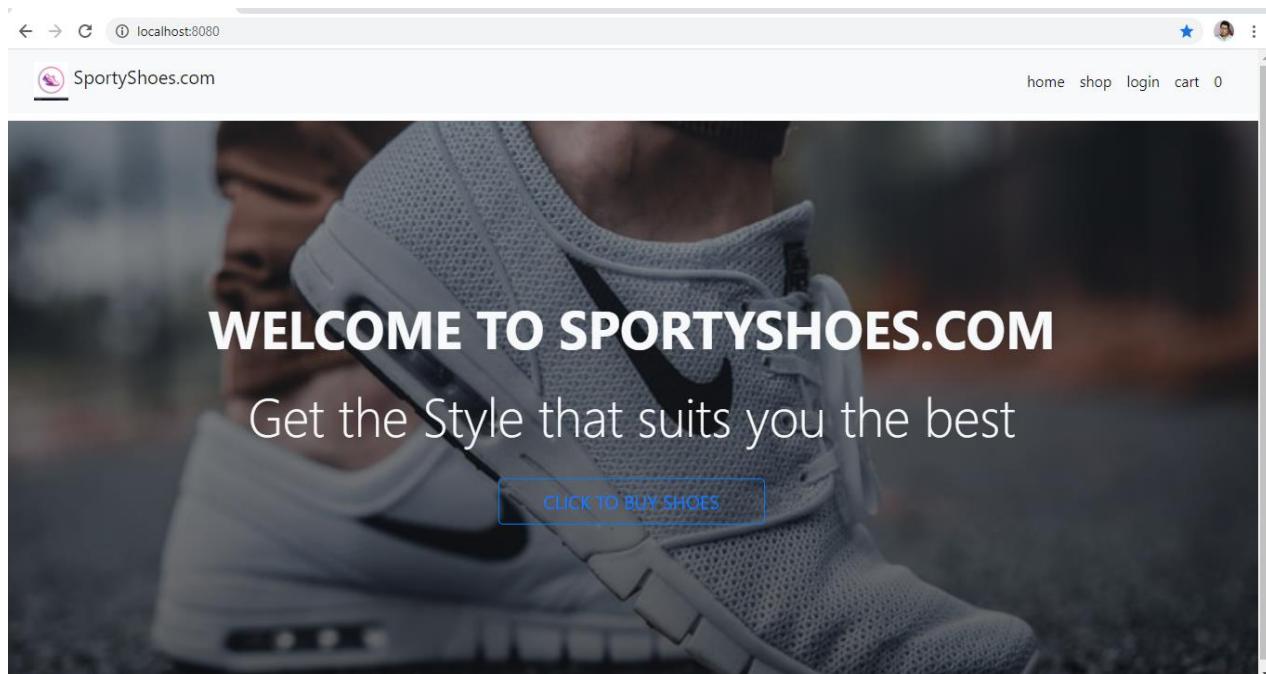
Links to the GitHub repository:

<https://github.com/Mahendra1272/E-commerce.git>

Screenshots of all Scenarios in the Application:

1. Customer Related Scenarios

Homepage displays options to users where they can login (if they already have an account or Sign Up if they are new users) , users also have an option to shop where they can shop for shoes and add to cart, they can also click on Click to Buy button which will redirect them to shop page to buy products.



In Shop page user will see list of All products if the category is not selected from the sidebar, User can also select individual categories and purchase products from there

The screenshot shows the 'shop' page of SportyShoes.com. At the top, there is a navigation bar with links for 'home', 'shop', 'login', and a 'cart' icon showing '0'. On the left side, there is a sidebar titled 'Categories' with a list of product types: 'All Products' (highlighted in yellow), 'Sports Shoe', 'Casual Shoe', 'Formal Shoe', and 'Sneakers'. The main content area displays three product cards. The first card features a red Nike running shoe, with the brand name 'Nike', price '₹ 900.0', and a description: 'Nike super light shoes Features: One year waranty with sweat and water proof'. A 'View Product' button is at the bottom. The second card features a black Reebok sports shoe, with the brand name 'Reebok', price '₹ 1300.0', and a description: 'Reebok Sports Shoe Features: one year warranty with sweat and water proof'. A 'View Product' button is at the bottom. The third card, which is partially visible, features a blue HRX shoe, with the brand name 'HRX'.

Sneakers



Reebok

₹ 1300.0

Reebok Sports Shoe Features: one year warranty with sweat and water proof

[View Product](#)



HRX

₹ 1900.0

HRX Sweat Proof Shoe Features: Features: one year warranty with sweat and water proof

[View Product](#)



WoodLand

₹ 3000.0

WoodLand Casual Shoe Features: one year warranty with sweat and water proof

localhost:8080/shop



Reebok Air
₹ 3000.0
Super Light Reebok Air Collection Features: one year warranty with sweat and water proof

[View Product](#)



Lakhani Office Shoe
₹ 800.0
Lakhani Office Collection Features: one year warranty with sweat and water proof

[View Product](#)



Puma Formal Shoe
₹ 1900.0
Puma Formal Collection Features: one year warranty with sweat and water proof

[View Product](#)



Puma Formal Shoe
₹ 1900.0
Puma Formal Collection Features: one year warranty with sweat and water proof

[View Product](#)



Reebok
₹ 2900.0
Reebok Formal Collection Features: one year warranty with sweat and water proof

[View Product](#)



Puma
₹ 1900.0
Puma Sneakers Super Light Features: one year warranty with sweat and water proof

[View Product](#)

[localhost:8080/shop](#)

	Puma ₹ 1900.0 Puma Sneakers Super Light Features: one year warranty with sweat and water proof View Product
	Reebok ₹ 4000.0 Reebok Air Sneakers Features: one year warranty with sweat and water proof View Product
	HRX Light ₹ 900.0 HRX Sneakers Features: one year warranty with sweat and water proof View Product

User can also select individual categories to view products only from that category (view products by category)

If user selects sports shoe category they can see products listed in that category

[localhost:8080/shop/category/154](#)

SportyShoes.com

home shop login cart 0

Categories

- All Products
- Sports Shoe**
- Casual Shoe
- Formal Shoe
- Sneakers

	Nike ₹ 900.0 Nike super light shoes Features: One year waranty with sweat and water proof View Product
	Reebok ₹ 1300.0 Reebok Sports Shoe Features: one year warranty with sweat and water proof View Product
	HRX ₹ 1900.0
	HRX ₹ 1900.0 HRX Sweat Proof Shoe Features: Features: one year warranty with sweat and water proof View Product

If user selects casual shoe category they can see products listed in that category

localhost:8080/shop/category/155

SportyShoes.com

home shop login cart 0

Categories

- All Products
- Sports Shoe
- Casual Shoe
- Formal Shoe
- Sneakers



WoodLand
₹ 3000.0
WoodLand Casual Shoe Features: one year warranty with sweat and water proof

[View Product](#)



Reebok Air
₹ 3000.0
Super Light Reebok Air Collection Features: one year warranty with sweat and water proof

[View Product](#)

If user selects Formal shoe category they can see products listed in that category

localhost:8080/shop/category/156

Categories

- All Products
- Sports Shoe
- Casual Shoe
- Formal Shoe
- Sneakers



Lakhani Office Shoe
₹ 800.0
Lakhani Office Collection Features: one year warranty with sweat and water proof

[View Product](#)



Puma Formal Shoe
₹ 1900.0
Puma Formal Collection Features: one year warranty with sweat and water proof

[View Product](#)



Reebok
₹ 2900.0
Reebok Formal Collection Features: one year warranty with sweat and water proof

If user selects Sneakers category they can see products listed in that category

localhost:8080/shop/category/168

Categories

- All Products
- Sports Shoe
- Casual Shoe
- Formal Shoe
- Sneakers



Puma
₹ 1900.0
Puma Sneakers Super Light Features: one year warranty with sweat and water proof
[View Product](#)



Reebok
₹ 4000.0
Reebok Air Sneakers Features: one year warranty with sweat and water proof
[View Product](#)



HRX Light
₹ 900.0
HRX Sneakers Features: one year warranty with sweat and water proof
[View Product](#)

When user clicks on View Product they are redirected to specific product description page

localhost:8080/shop/viewproduct/159

SportyShoes.com home shop login cart 0



Nike
Sports Shoe
₹ 900.0
Weight: 300.0 grams.
Nike super light shoes Features: One year warranty with sweat and water proof
[Add to cart](#)

In the product description page User have option to Add the product to cart, But In order to add the product to cart user must be signed in (implemented with Spring Security).

Users are redirected to Login page if they want to add products to cart

localhost:8080/login

SportyShoes.com

Login

Please fill out this to login

Email

Password

[Login](#)

Don't have an account [Register here](#)

After successful login we can see the product is added to users cart and also we can see logout option now in the navbar for logged in user

localhost:8080/shop

SportyShoes.com

home shop logout **cart 1**

Categories

- [All Products](#)
- [Sports Shoe](#)
- [Casual Shoe](#)
- [Formal Shoe](#)
- [Sneakers](#)



Nike
₹ 900.0
Nike super light shoes Features: One year waranty with sweat and water proof

[View Product](#)



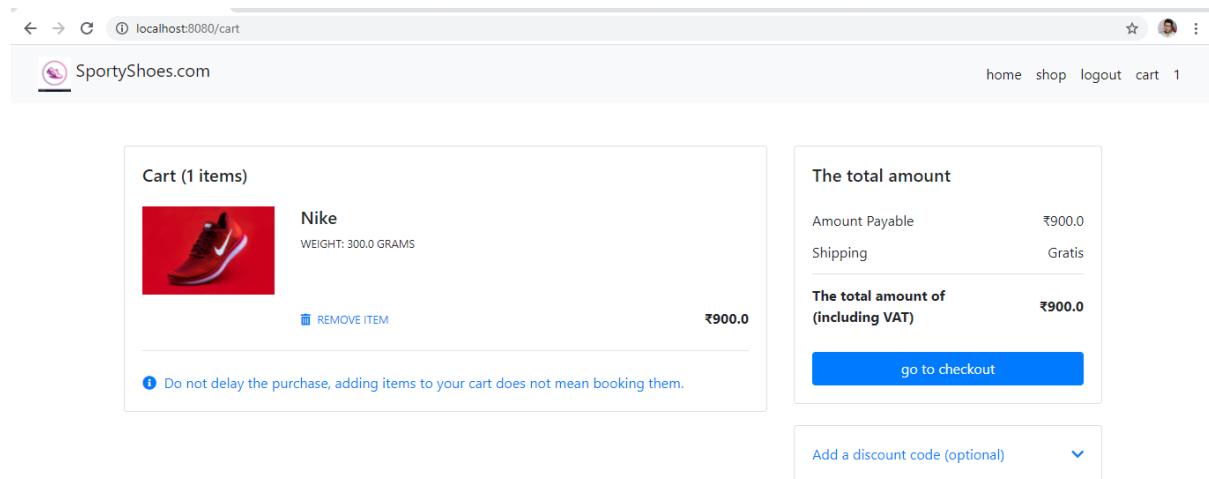
Reebok
₹ 1300.0
Reebok Sports Shoe Features: one year warranty with sweat and water proof

[View Product](#)



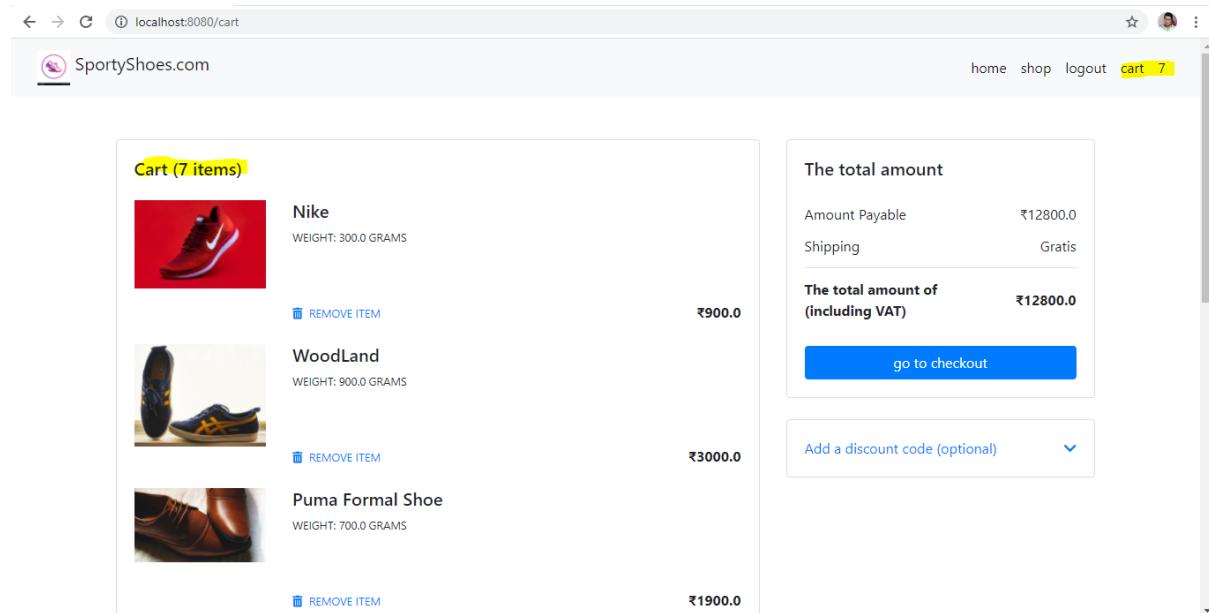
HRX

Users can view cart items by clicking on cart button in the navbar



A screenshot of a web browser displaying the SportyShoes.com website. The URL in the address bar is 'localhost:8080/cart'. The page shows a cart containing one item: a Nike running shoe. The product details are: Nike, WEIGHT: 300.0 GRAMS, and a price of ₹900.0. Below the product is a note: 'Do not delay the purchase, adding items to your cart does not mean booking them.' To the right of the cart area is a summary box titled 'The total amount' which lists: Amount Payable ₹900.0, Shipping Gratis, and The total amount of (including VAT) ₹900.0. A blue 'go to checkout' button is present. At the bottom right is a dropdown menu for 'Add a discount code (optional)'.

Users can also add multiple Items to their cart (Subtotal of each product, And the total price of all the products is shown)



A screenshot of a web browser displaying the SportyShoes.com website. The URL in the address bar is 'localhost:8080/cart'. The page shows a cart containing three items: 1. Nike running shoe: WEIGHT: 300.0 GRAMS, Price: ₹900.0. 2. WoodLand casual shoe: WEIGHT: 900.0 GRAMS, Price: ₹3000.0. 3. Puma Formal Shoe: WEIGHT: 700.0 GRAMS, Price: ₹1900.0. Each item has a 'REMOVE ITEM' link below it. To the right of the cart area is a summary box titled 'The total amount' which lists: Amount Payable ₹12800.0, Shipping Gratis, and The total amount of (including VAT) ₹12800.0. A blue 'go to checkout' button is present. At the bottom right is a dropdown menu for 'Add a discount code (optional)'.

localhost:8080/cart

	HRX WEIGHT: 600.0 GRAMS	₹1900.0
	Reebok WEIGHT: 500.0 GRAMS	₹1900.0
	Puma Formal Shoe WEIGHT: 700.0 GRAMS	₹1300.0
		₹1900.0

Do not delay the purchase, adding items to your cart does not mean booking them.

User can also remove products from cart by clicking on Remove Item link

localhost:8080/cart

SportyShoes.com

home shop logout cart 6

Cart (6 items)		
	WoodLand WEIGHT: 900.0 GRAMS	₹3000.0
	Puma Formal Shoe WEIGHT: 700.0 GRAMS	₹1900.0
	Puma WEIGHT: 800.0 GRAMS	₹1900.0
		₹1900.0

The total amount

Amount Payable	₹11900.0
Shipping	Gratis
The total amount of (including VAT)	₹11900.0

go to checkout

Add a discount code (optional)

(First item has been removed and now the total cart items are 6)

When user clicks Got to checkout button they will be redirected to Checkout Page where they need to fill Billing details and pay the total amount

The screenshot shows a web browser window with the URL `localhost:8080/checkout`. On the left, there is a form titled "Billing details" containing fields for First name (Shubham), Last name (Sondhiya), Country (INDIA), Address (Line 1: Singrauli), Address (Line 2: Apartment, suite, unit etc. (optional)), Postcode / ZIP (486689), Town / City (Singrauli), Phone (7898789878), and Email address. On the right, there is a section titled "The total amount" showing Amount Payable (₹11900.0), Shipping (Gratis), and The total amount of (including VAT) (₹11900.0). A blue "Pay Now" button is visible. Below it is a dropdown menu for "Add a discount code (optional)".

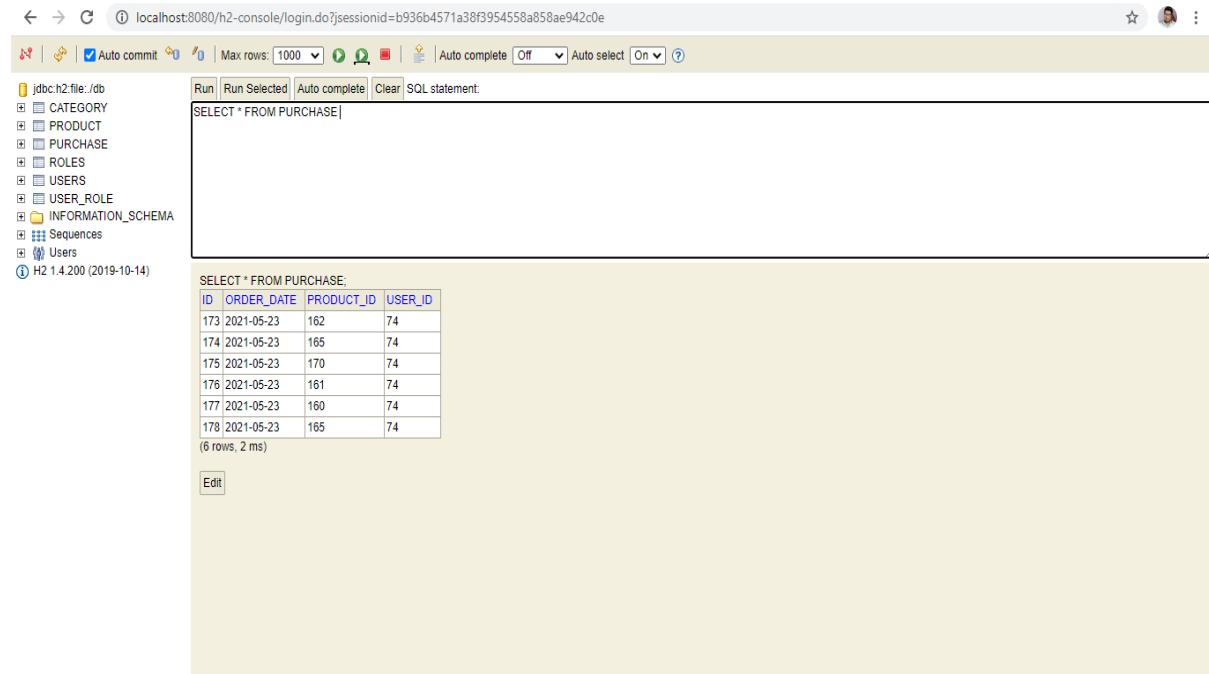
When user clicks on Pay now then they are redirected to Order Placed page where they can see the details of order placed and the products purchased by the user are persisted to the database

The screenshot shows a web browser window with the URL `localhost:8080/payNow`. At the top, there is a logo for SportyShoes.com, navigation links for home, shop, logout, and a yellow-highlighted cart icon. Below the header, the text "Receipt #: 78863" is displayed. The main content is a "Receipt" table with columns for SN, Product, and Price. The table lists six items: WoodLand (3000.0), Puma Formal Shoe (1900.0), Puma (1900.0), HRX (1900.0), Reebok (1300.0), and Puma Formal Shoe (1900.0). Below the table, the text "Total Amount Paid:" is followed by the amount "11900.0". The URL in the address bar is `localhost:8080/payNow#`.

SN	Product	Price
1	WoodLand	3000.0
2	Puma Formal Shoe	1900.0
3	Puma	1900.0
4	HRX	1900.0
5	Reebok	1300.0
6	Puma Formal Shoe	1900.0

(We can see now the cart is showing 0 items)

All the bought products by user are persisted in the database



The screenshot shows an H2 database console interface. The URL is `localhost:8080/h2-console/login.do?sessionid=b936b4571a38f3954558a858ae942c0e`. The left sidebar lists database objects: CATEGORY, PRODUCT, PURCHASE, ROLES, USERS, USER_ROLE, INFORMATION_SCHEMA, Sequences, and Users. The main area contains a SQL statement editor with the query `SELECT * FROM PURCHASE`. Below the editor is a table displaying the results of the query:

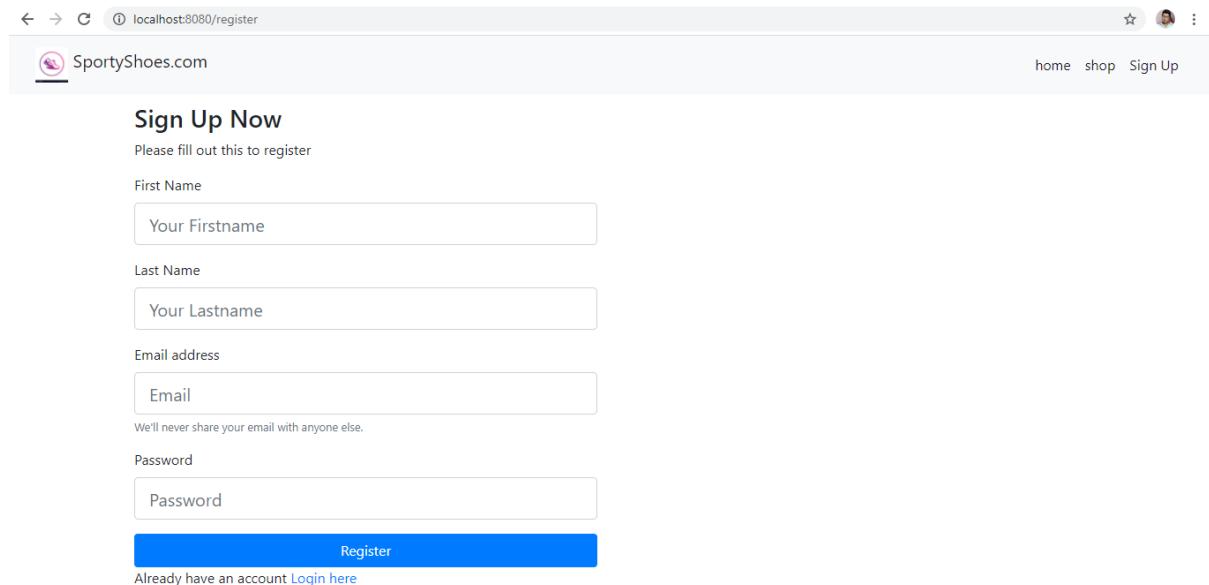
ID	ORDER_DATE	PRODUCT_ID	USER_ID
173	2021-05-23	162	74
174	2021-05-23	165	74
175	2021-05-23	170	74
176	2021-05-23	161	74
177	2021-05-23	160	74
178	2021-05-23	165	74

(6 rows, 2 ms)

Below the table is an [Edit](#) button.

2. User Sign In and Register

New users can register in the system by signing up



The screenshot shows a registration form for SportyShoes.com. The URL is `localhost:8080/register`. The page has a header with the SportyShoes logo and navigation links for home, shop, and Sign Up. The main section is titled "Sign Up Now" and includes fields for First Name, Last Name, Email address, and Password. A note below the email field states, "We'll never share your email with anyone else." At the bottom of the form is a blue "Register" button.

Please fill out this to register

First Name

Last Name

Email address

We'll never share your email with anyone else.

Password

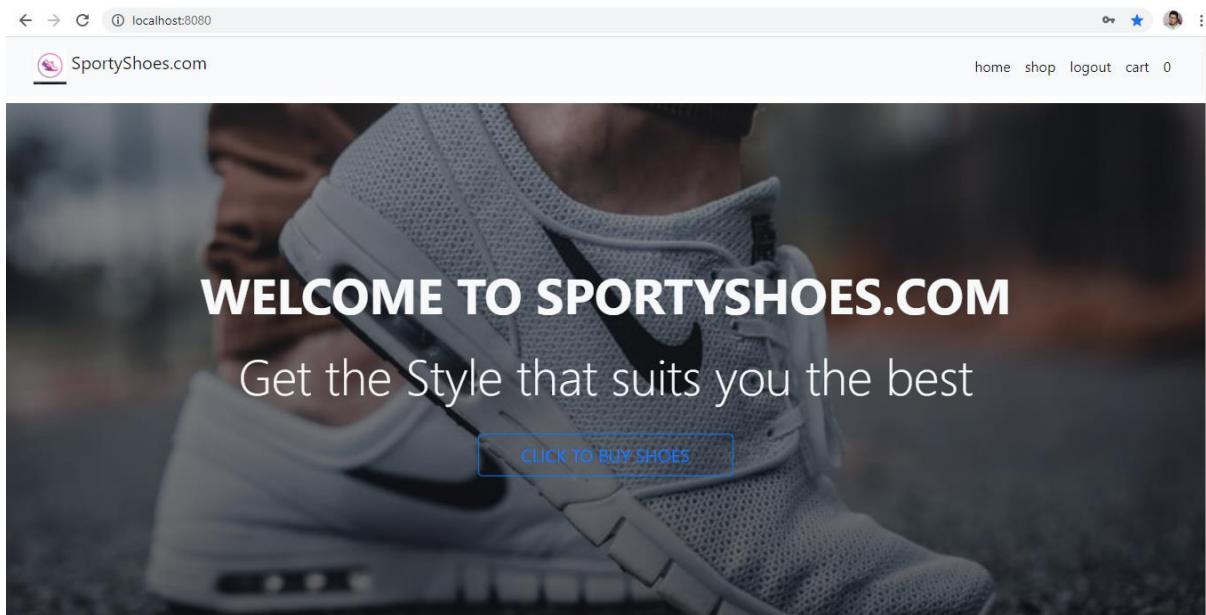
[Register](#)

Already have an account [Login here](#)

Filling up the details in the register form

The screenshot shows a web browser window with the URL localhost:8080/register. The page title is "SportyShoes.com". The main heading is "Sign Up Now" with the sub-instruction "Please fill out this to register". There are four input fields: "First Name" containing "Virat", "Last Name" containing "Kohli", "Email address" containing "virat@gmail.com", and a password field containing "*****". Below the email field is a small note: "We'll never share your email with anyone else." A blue "Register" button is at the bottom, and a link "Already have an account Login here" is just below it.

Once user gets registered they are automatically logged in and redirected to Home Page where they can buy products



We can see user registered user is added in the database

The screenshot shows the H2 Database Console interface. The left sidebar lists database objects: CATEGORY, PRODUCT, PURCHASE, ROLES, USERS, USER_ROLE, INFORMATION_SCHEMA, Sequences, and Users. The main area shows the results of a SQL query:

```
SELECT * FROM USERS;
```

ID	EMAIL	FIRST_NAME	LAST_NAME	PASSWORD
1	admin@gmail.com	Shubham	Sondhiya	\$2a\$10\$moTjfAVBjzNS32q7NPPVh.iAGDbzSkjZjBkdxT52LKQ3R1vhv9F
74	shubham@gmail.com	Shubham	Kumar	\$2a\$10\$MkaZeuofoxuhYEyy3EhIVuSmhEghhhYpUccggy4bToX1k1nVW.qrG
179	rahul@gmail.com	Rahul	Kumar	\$2a\$10\$O6Xowl50TAwvry6nYnI3uVpAvQcz4gwv2VFbg0beGHgtbJlFNv6
180	pradeep@gmail.com	Pradeep	Kumar	\$2a\$10\$G8Q2Z5yU2gpmB8VLvEvC0s.PBN7ipApIVYYw.D77qTces0WEHATvi
181	vira@gmail.com	Virat	Kohli	\$2a\$10\$jkq5ifPYx2RxhnpR7j5veKskMta6Fc08QEICR0JEqYxEzB4phBK

(5 rows, 3 ms)

[Edit](#)

Already registered users can login directly in the system



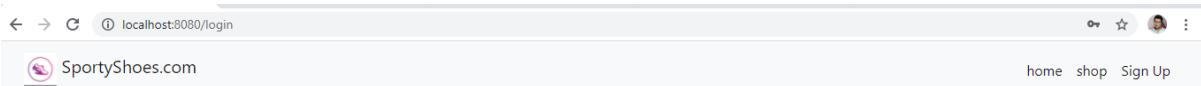
Login

Please fill out this to login

Email

Password

Don't have an account [Register here](#)



Login

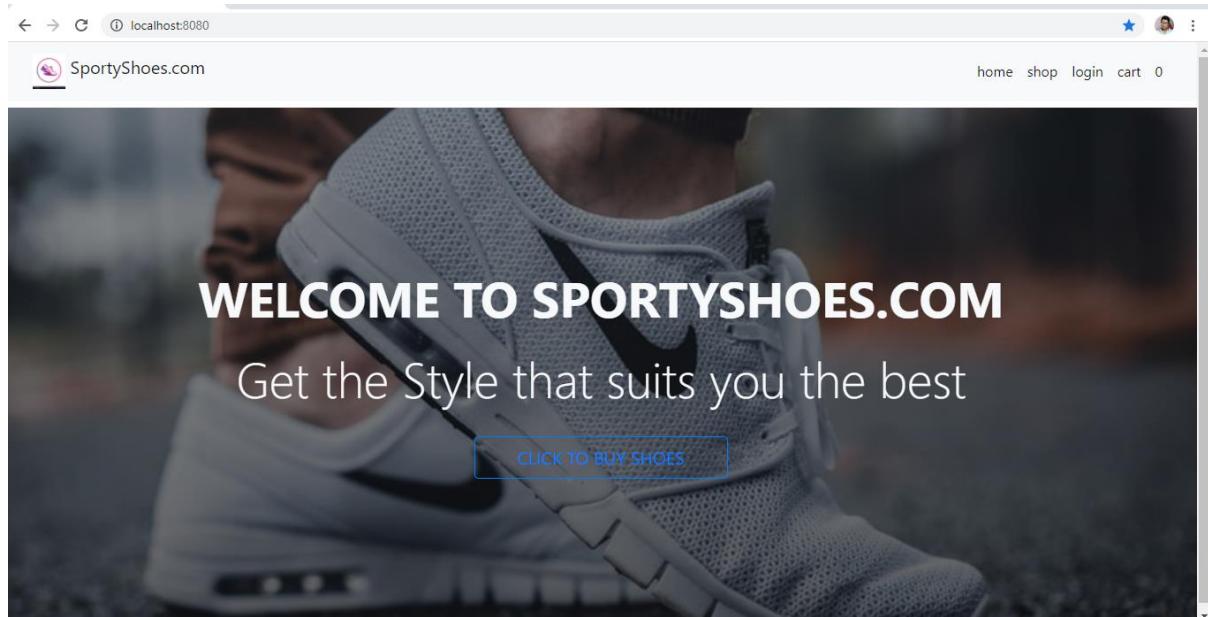
Please fill out this to login

Email

Password

Don't have an account [Register here](#)

After successful login they are redirected to Home page where they can buy products

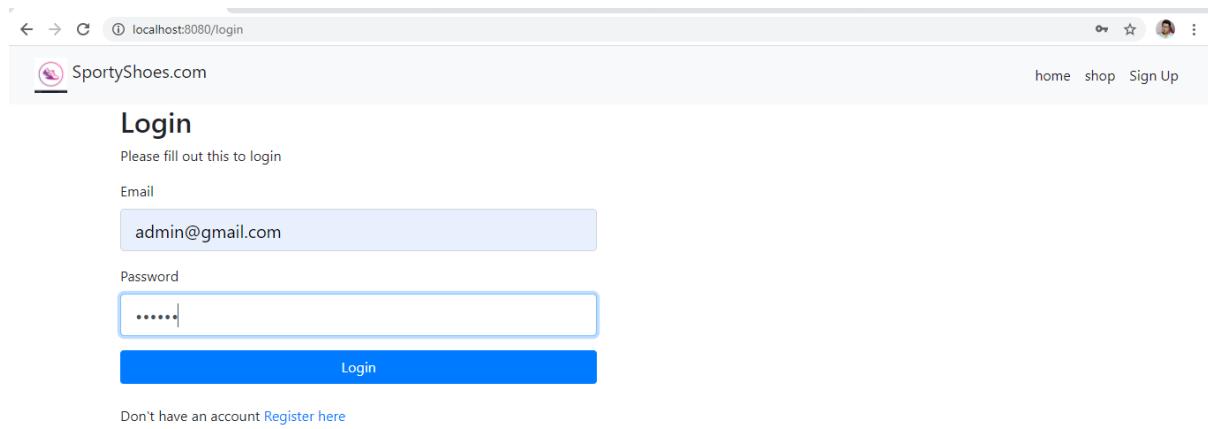


When user provided invalid details they are redirected to the same login page with error message as Invalid username or password

A screenshot of the login page at localhost:8080/login?error=true. The page has a header with the SportyShoes logo and navigation links for "home", "shop", and "Sign Up". The main section is titled "Login" and contains a message "Please fill out this to login". There are two input fields: "Email" (placeholder "Your Email") and "Password" (placeholder "Password"). Below the fields is an error message "Invalid username or password." A blue "Login" button is at the bottom. A link "Don't have an account Register here" is also present.

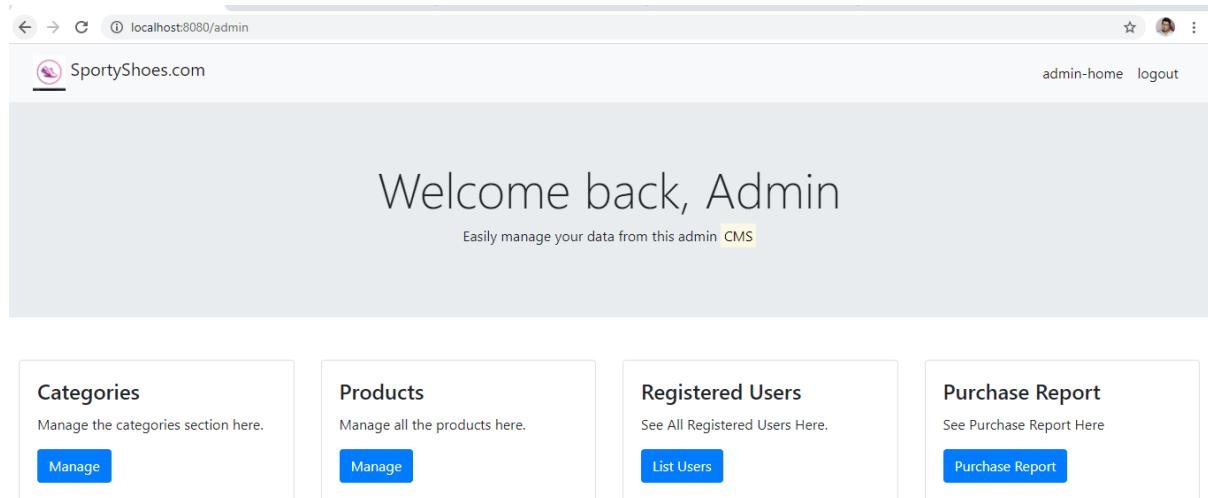
3. Admin Related Scenarios

Admin can login in the system by providing their credentials (Admin is added from the backend)



The screenshot shows a web browser window with the URL `localhost:8080/login`. The title bar says "SportyShoes.com". The main content is a "Login" form. It has a placeholder "Please fill out this to login". There are two input fields: "Email" containing "admin@gmail.com" and "Password" containing "*****". Below the password field is a blue "Login" button. At the bottom, there is a link "Don't have an account [Register here](#)".

Once admin is logged in they can see All the operations they can perform like , Manage Categories, Manage Products, See all the registered Users list in the system, See the purchase report



The screenshot shows a web browser window with the URL `localhost:8080/admin`. The title bar says "SportyShoes.com". The top right shows "admin-home" and "logout". The main area has a welcome message "Welcome back, Admin" and a sub-instruction "Easily manage your data from this admin CMS". Below this are four cards:

- Categories**: Manage the categories section here. [Manage](#)
- Products**: Manage all the products here. [Manage](#)
- Registered Users**: See All Registered Users Here. [List Users](#)
- Purchase Report**: See Purchase Report Here. [Purchase Report](#)

3.1. Admin can Manage Categories (All CRUD Operations)

Admin can see list of all categories added in the system

localhost:8080/admin/categories

SN	Category Name	Delete	Update
1	Sports Shoe	Delete	Update
2	Casual Shoe	Delete	Update
3	Formal Shoe	Delete	Update

Admin can add a new category by clicking on Add Category Button

localhost:8080/admin/categories/add

Name

Submit

(Here admin provide category name and adds a category)

We can see new category Sneakers is added

localhost:8080/admin/categories

SN	Category Name	Delete	Update
1	Sports Shoe	<button>Delete</button>	<button>Update</button>
2	Casual Shoe	<button>Delete</button>	<button>Update</button>
3	Formal Shoe	<button>Delete</button>	<button>Update</button>
4	Sneakers	<button>Delete</button>	<button>Update</button>

New added category is persisted in the database

localhost:8080/h2-console/login.do?jsessionid=a6d43398b7eedf98d5a185ef5badced9

CATEGORY_ID	NAME
154	Sports Shoe
155	Casual Shoe
156	Formal Shoe
167	Sneakers

Admin can update a category by clicking on update button next to each category (Form comes prefilled with current values)

The screenshot shows a web browser window with the following details:

- URL:** localhost:8080/admin/categories/update/167
- Title Bar:** SportyShoes.com
- User Information:** admin-home logout
- Form Fields:**
 - Name: Sneakers
- Buttons:** Submit

Sneakers category Updated to Boots

← → ⌛ ⓘ localhost:8080/admin/categories/update/167

SporyShoes.com admin-home logout

Name

Submit

← → ⌛ ⓘ localhost:8080/admin/categories

SporyShoes.com admin-home logout

Add Category

SN	Category Name	Delete	Update
1	Sports Shoe	<button>Delete</button>	<button>Update</button>
2	Casual Shoe	<button>Delete</button>	<button>Update</button>
3	Formal Shoe	<button>Delete</button>	<button>Update</button>
4	Boots	<button>Delete</button>	<button>Update</button>

Category updated in the database also

The screenshot shows the H2 Database Console interface at localhost:8080/h2-console/login.do?sessionid=a6d43398b7eedf98d5a185ef5badced9. The left sidebar lists database objects: CATEGORY, PRODUCT, PURCHASE, ROLES, USERS, USER_ROLE, INFORMATION_SCHEMA, Sequences, and Users. The right panel contains a SQL statement window with the query `SELECT * FROM CATEGORY`. Below it is a table with four rows of data:

CATEGORY_ID	NAME
154	Sports Shoe
155	Casual Shoe
156	Formal Shoe
167	Boots

(4 rows, 3 ms)

[Edit](#)

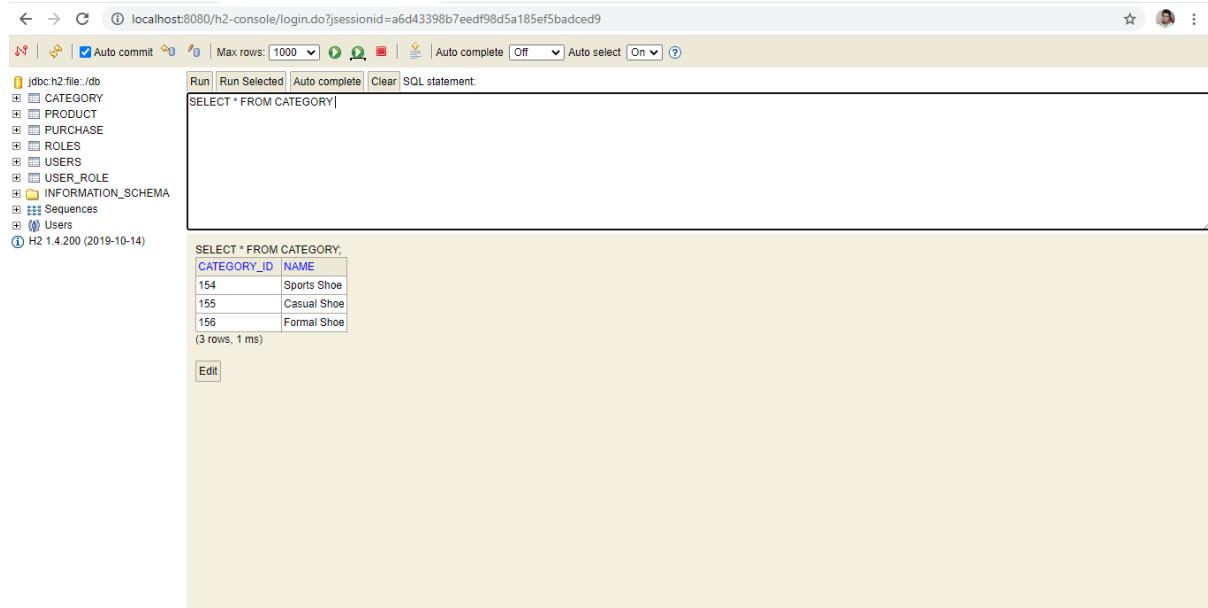
Admin can also delete a category by clicking on delete button next to each category

The screenshot shows the Admin Categories page at localhost:8080/admin/categories. The top navigation bar includes links for [admin-home](#) and [logout](#). The main content area has a header "Add Category". Below it is a table listing categories:

SN	Category Name	Delete	Update
1	Sports Shoe	Delete	Update
2	Casual Shoe	Delete	Update
3	Formal Shoe	Delete	Update

(Boots category Deleted)

Boots category deleted from database also



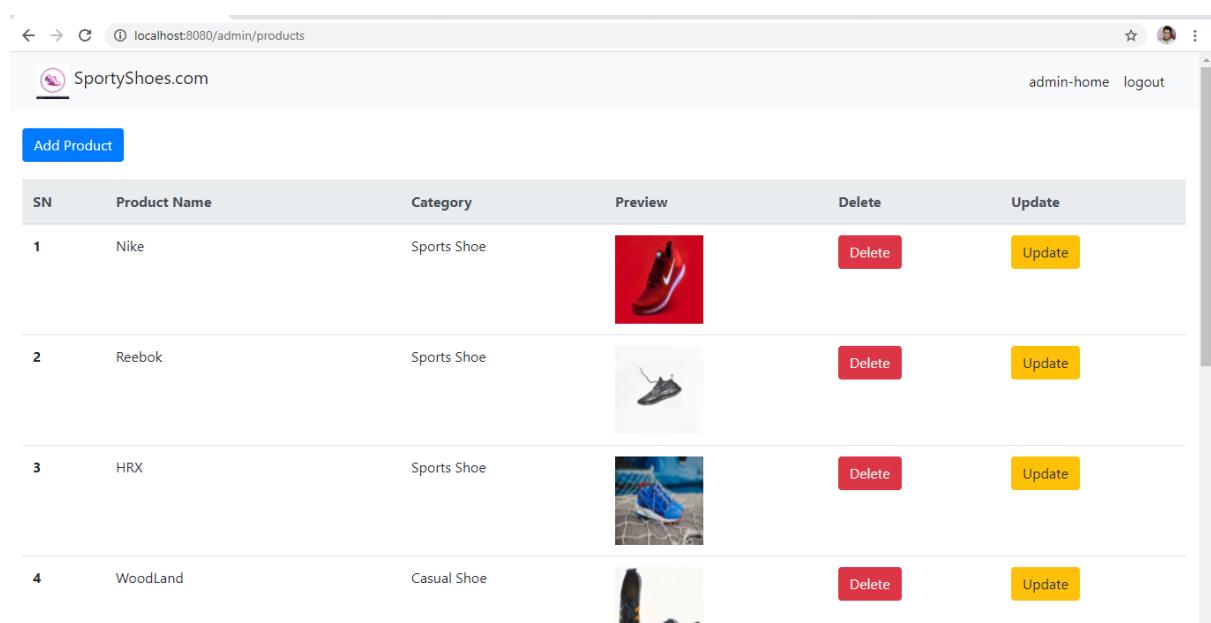
The screenshot shows the H2 Database Console interface. On the left, there's a tree view of the database schema with tables like CATEGORY, PRODUCT, PURCHASE, ROLES, USERS, and USER_ROLE. The main area has a SQL editor with the query "SELECT * FROM CATEGORY;" and a results table below it.

CATEGORY_ID	NAME
154	Sports Shoe
155	Casual Shoe
156	Formal Shoe

(3 rows, 1 ms)

3.2. Admin can Manage Products (All CRUD Operations)

Admin can see list of all products added in the system



The screenshot shows the Admin Product Management interface. At the top, there's a header with a logo, the site name "SportyShoes.com", and user links "admin-home" and "logout". Below the header is a button labeled "Add Product". The main area is a table listing products:

SN	Product Name	Category	Preview	Delete	Update
1	Nike	Sports Shoe		<button>Delete</button>	<button>Update</button>
2	Reebok	Sports Shoe		<button>Delete</button>	<button>Update</button>
3	HRX	Sports Shoe		<button>Delete</button>	<button>Update</button>
4	WoodLand	Casual Shoe		<button>Delete</button>	<button>Update</button>

Admin can add a new product by clicking on Add Product Button

SportyShoes.com

Add a new Product

Name

Select Category

Price

Weight in grams

Product Description

Product Image

 Submit

Admin has to fill all the details of the product and also provide an image from the system for the product (Adding PUMA Shoes of category Sneakers)

SportyShoes.com

Add a new Product

Name

Select Category

Price

Weight in grams

Product Description

Product Image



New product added in the product list after submission of details

ID	NAME	CATEGORY	IMAGE_NAME	DELETE	UPDATE
5	Reebok Air	Casual Shoe		<button>Delete</button>	<button>Update</button>
6	Lakhani Office Shoe	Formal Shoe		<button>Delete</button>	<button>Update</button>
7	Puma Formal Shoe	Formal Shoe		<button>Delete</button>	<button>Update</button>
8	Reebok	Formal Shoe		<button>Delete</button>	<button>Update</button>
9	Puma	Sneakers		<button>Delete</button>	<button>Update</button>

New Added product is also persisted in the database

ID	DESCRIPTION	IMAGE_NAME	NAME	PRICE	WEIGHT	CATEGORY_ID
159	Nike super light shoes Features: One year waranty with sweat and water proof	Nike1.jpg	Nike	900.0	300.0	154
160	Reebok Sports Shoe Features: one year warranty with sweat and water proof	Reebok.jpg	Reebok	1300.0	500.0	154
161	HRX Sweat Proof Shoe Features: Features: one year warranty with sweat and water proof	HRX.jpg	HRX	1900.0	600.0	154
162	WoodLand Casual Shoe Features: one year warranty with sweat and water proof	casual1.jpg	WoodLand	3000.0	900.0	155
163	Super Light Reebok Air Collection Features: one year warranty with sweat and water proof	casual2.jpg	Reebok Air	3000.0	400.0	155
164	Lakhani Office Collection Features: one year warranty with sweat and water proof	formal1.jpg	Lakhani Office Shoe	800.0	600.0	156
165	Puma Formal Collection Features: one year warranty with sweat and water proof	formal2.jpg	Puma Formal Shoe	1900.0	700.0	156
166	Reebok Formal Collection Features: one year warranty with sweat and water proof	formal3.jpg	Reebok	2900.0	800.0	156
169	Puma Sneakers Features: one year warranty with sweat and water proof	sneaker1.jpg	Puma	900.0	600.0	168

Admin can update a product by clicking on update button next to each product (Form comes pre-filled with current values)

← → ⌂ ⓘ localhost:8080/admin/product/update/169

SportyShoes.com admin-home logout

Add a new Product

Name

Product Image Choose file

Select Category

Price

Weight in grams

Submit

Product Description

Features: one year warranty with sweat and water proof

Updating to New Values

← → ⌂ ⓘ localhost:8080/admin/product/update/169

SportyShoes.com admin-home logout

Add a new Product

Name

Product Image Choose file

Select Category

Price

Weight in grams

Submit

Product Description

Features: one year warranty with sweat and water proof

Product details gets updated in the product list

5	Reebok Air	Casual Shoe		Delete	Update
6	Lakhani Office Shoe	Formal Shoe		Delete	Update
7	Puma Formal Shoe	Formal Shoe		Delete	Update
8	Reebok	Formal Shoe		Delete	Update
9	Puma Sneakers	Sneakers		Delete	Update

Product also gets updated in the database

localhost:8080/h2-console/login.do?sessionid=a767a0b209cb4001978b6a07e584f964

Auto commit | Max rows: 1000 | Auto complete | Off | Auto select | On | ?

jdbc:h2:file:/db

CATEGORY

PRODUCT

PURCHASE

ROLES

USERS

USER_ROLE

INFORMATION_SCHEMA

Sequences

Users

H2 1.4.200 (2019-10-14)

Run | Run Selected | Auto complete | Clear | SQL statement:

```
SELECT * FROM PRODUCT|
```

ID	DESCRIPTION	IMAGE_NAME	NAME	PRICE	WEIGHT	CATEGORY_ID
159	Nike super light shoes Features: One year waranty with sweat and water proof	Nike1.jpg	Nike	900.0	300.0	154
160	Reebok Sports Shoe Features: one year warranty with sweat and water proof	Reebok.jpg	Reebok	1300.0	500.0	154
161	HRX Sweat Proof Shoe Features: Features: one year warranty with sweat and water proof	Hrx.jpg	HRX	1900.0	600.0	154
162	WoodLand Casual Shoe Features: one year warranty with sweat and water proof	casual1.jpg	WoodLand	3000.0	900.0	155
163	Super Light Reebok Air Collection Features: one year warranty with sweat and water proof	casual2.jpg	Reebok Air	3000.0	400.0	155
164	Lakhani Office Collection Features: one year warranty with sweat and water proof	formal1.jpg	Lakhani Office Shoe	800.0	600.0	156
165	Puma Formal Collection Features: one year warranty with sweat and water proof	formal2.jpg	Puma Formal Shoe	1900.0	700.0	156
166	Reebok Formal Collection Features: one year warranty with sweat and water proof	formal3.jpg	Reebok	2900.0	800.0	156
169	Puma Sneakers Super Light Features: one year warranty with sweat and water proof	sneaker1.jpg	Puma Sneakers	1000.0	600.0	168

(9 rows, 4 ms)

localhost:8080/h2-console/tables.do?sessionid=a767a0b209cb4001978b6a07e584f964

Admin can also delete a product by clicking on delete button next to each product
(Deleted Puma Sneakers product)

4	WoodLand	Casual Shoe		Delete	Update
5	Reebok Air	Casual Shoe		Delete	Update
6	Lakhani Office Shoe	Formal Shoe		Delete	Update
7	Puma Formal Shoe	Formal Shoe		Delete	Update
8	Reebok	Formal Shoe		Delete	Update

Product is also deleted from the database

localhost:8080/h2-console/login.do?sessionid=a767a0b209cb4001978b6a07e584f964

Auto commit | Max rows: 1000 | Auto complete | Off | Auto select: On

Run | Run Selected | Auto complete | Clear | SQL statement:

```
SELECT * FROM PRODUCT|
```

SELECT * FROM PRODUCT;

ID	DESCRIPTION	IMAGE_NAME	NAME	PRICE	WEIGHT	CATEGORY_ID
159	Nike super light shoes Features: One year warranty with sweat and water proof	Nike1.jpg	Nike	900.0	300.0	154
160	Reebok Sports Shoe Features: one year warranty with sweat and water proof	Reebok.jpg	Reebok	1300.0	500.0	154
161	HRX Sweat Proof Shoe Features: Features: one year warranty with sweat and water proof	Hrx.jpg	HRX	1900.0	600.0	154
162	WoodLand Casual Shoe Features: one year warranty with sweat and water proof	casual1.jpg	WoodLand	3000.0	900.0	155
163	Super Light Reebok Air Collection Features: one year warranty with sweat and water proof	casual2.jpg	Reebok Air	3000.0	400.0	155
164	Lakhani Office Collection Features: one year warranty with sweat and water proof	formal1.jpg	Lakhani Office Shoe	800.0	600.0	156
165	Puma Formal Collection Features: one year warranty with sweat and water proof	formal2.jpg	Puma Formal Shoe	1900.0	700.0	156
166	Reebok Formal Collection Features: one year warranty with sweat and water proof	formal3.jpg	Reebok	2900.0	800.0	156

(8 rows, 2 ms)

Edit

3.3. Admin can see a list of all the registered users in the system (By going in Registered Users Tab)

Welcome back, Admin

Easily manage your data from this admin CMS

Categories
Manage the categories section here.
[Manage](#)

Products
Manage all the products here.
[Manage](#)

Registered Users
See All Registered Users Here.
[List Users](#)

Purchase Report
See Purchase Report Here
[Purchase Report](#)

Admin can see list all registered users in the system

Show 5 entries Search:

SN	User Id	Email	First Name	Last Name
1	1	admin@gmail.com	Shubham	Sondhiya
2	74	shubham@gmail.com	Shubham	Kumar
3	179	rahul@gmail.com	Rahul	Kumar
4	180	pradeep@gmail.com	Pradeep	Kumar
5	181	virat@gmail.com	Virat	Kohli

Showing 1 to 5 of 5 entries Previous [1](#) Next

Admin can also filter the results from the search bar

Filtering by First Name

SportyShoes.com					admin-home	logout
					Show <input type="button" value="5"/> entries	Search: <input type="text" value="Rahul"/> <input type="button" value="X"/>
SN	User Id	Email	First Name	Last Name		
3	179	rahul@gmail.com	Rahul	Kumar		

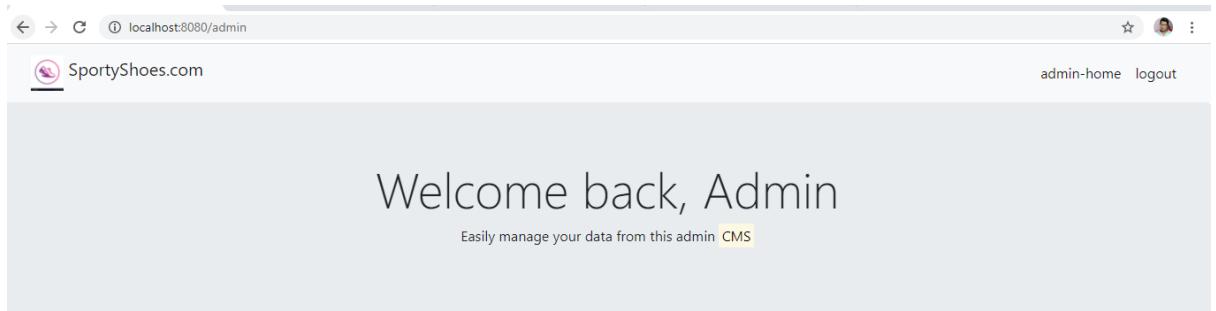
Showing 1 to 1 of 1 entries (filtered from 5 total entries) [Previous](#) [Next](#)

Results filtered by email

SportyShoes.com					admin-home	logout
					Show <input type="button" value="5"/> entries	Search: <input type="text" value="pradeep@gmail.com"/> <input type="button" value="X"/>
SN	User Id	Email	First Name	Last Name		
4	180	pradeep@gmail.com	Pradeep	Kumar		

Showing 1 to 1 of 1 entries (filtered from 5 total entries) [Previous](#) [Next](#)

3.4. Admin can see Purchase report in the system (By going in Purchase report Tab)



If nothing is present in the search bar then All purchases are returned

The screenshot shows the 'Purchase Report' page. The title 'Purchase Report' is at the top, followed by a search bar and a dropdown menu for selecting the number of entries to show (set to 7). A table below lists 13 purchase entries. The columns are: SN, User Email, User Name, Product Id, Category, Product Name, Product Price, and Order Date. The data is as follows:

SN	User Email	User Name	Product Id	Category	Product Name	Product Price	Order Date
1	shubham@gmail.com	Shubham	162	Casual Shoe	WoodLand	3000.0	2021-05-23
2	shubham@gmail.com	Shubham	165	Formal Shoe	Puma Formal Shoe	1900.0	2021-05-23
3	shubham@gmail.com	Shubham	170	Sneakers	Puma	1900.0	2021-05-23
4	shubham@gmail.com	Shubham	161	Sports Shoe	HRX	1900.0	2021-05-23
5	shubham@gmail.com	Shubham	160	Sports Shoe	Reebok	1300.0	2021-05-23
6	shubham@gmail.com	Shubham	165	Formal Shoe	Puma Formal Shoe	1900.0	2021-05-23
7	virat@gmail.com	Virat	159	Sports Shoe	Nike	900.0	2021-05-23
Showing 1 to 7 of 13 entries							
Previous				1	2	Next	

localhost:8080/admin/purchaseReport

SportyShoes.com admin-home logout

Purchase Report

Show 7 entries Search:

SN	User Email	User Name	Product Id	Category	Product Name	Product Price	Order Date
8	virat@gmail.com	Virat	165	Formal Shoe	Puma Formal Shoe	1900.0	2021-05-23
9	virat@gmail.com	Virat	160	Sports Shoe	Reebok	1300.0	2021-05-23
10	virat@gmail.com	Virat	171	Sneakers	Reebok	4000.0	2021-05-23
11	virat@gmail.com	Virat	172	Sneakers	HRX Light	900.0	2021-05-23
12	virat@gmail.com	Virat	165	Formal Shoe	Puma Formal Shoe	1900.0	2021-05-23
13	virat@gmail.com	Virat	163	Casual Shoe	Reebok Air	3000.0	2021-05-23

Showing 8 to 13 of 13 entries Previous 1 2 Next

Purchase report filtered by category (Formal Shoe)

localhost:8080/admin/purchaseReport

SportyShoes.com admin-home logout

Purchase Report

Show 7 entries Search:

SN	User Email	User Name	Product Id	Category	Product Name	Product Price	Order Date
2	shubham@gmail.com	Shubham	165	Formal Shoe	Puma Formal Shoe	1900.0	2021-05-23
6	shubham@gmail.com	Shubham	165	Formal Shoe	Puma Formal Shoe	1900.0	2021-05-23
8	virat@gmail.com	Virat	165	Formal Shoe	Puma Formal Shoe	1900.0	2021-05-23
12	virat@gmail.com	Virat	165	Formal Shoe	Puma Formal Shoe	1900.0	2021-05-23

Showing 1 to 4 of 4 entries (filtered from 13 total entries) Previous 1 Next

Purchase report filtered by Order date

localhost:8080/admin/purchaseReport

SportyShoes.com admin-home logout

Purchase Report

Show 7 entries Search: 2021-05-23

SN	User Email	User Name	Product Id	Category	Product Name	Product Price	Order Date
1	shubham@gmail.com	Shubham	162	Casual Shoe	WoodLand	3000.0	2021-05-23
2	shubham@gmail.com	Shubham	165	Formal Shoe	Puma Formal Shoe	1900.0	2021-05-23
3	shubham@gmail.com	Shubham	170	Sneakers	Puma	1900.0	2021-05-23
4	shubham@gmail.com	Shubham	161	Sports Shoe	HRX	1900.0	2021-05-23
5	shubham@gmail.com	Shubham	160	Sports Shoe	Reebok	1300.0	2021-05-23
6	shubham@gmail.com	Shubham	165	Formal Shoe	Puma Formal Shoe	1900.0	2021-05-23
7	virat@gmail.com	Virat	159	Sports Shoe	Nike	900.0	2021-05-23

Showing 1 to 7 of 13 entries Previous 1 2 Next

If there is no match present with respect to filter criteria then no records are returned

localhost:8080/admin/purchaseReport

SportyShoes.com admin-home logout

Purchase Report

Show 7 entries Search: 2021-05-22

SN	User Email	User Name	Product Id	Category	Product Name	Product Price	Order Date
No matching records found							

Showing 0 to 0 of 0 entries (filtered from 13 total entries) Previous Next

Purchase report filtered by User Name

localhost:8080/admin/purchaseReport

SportyShoes.com admin-home logout

Purchase Report

Show 7 entries Search: Virat

SN	User Email	User Name	Product Id	Category	Product Name	Product Price	Order Date
7	virat@gmail.com	Virat	159	Sports Shoe	Nike	900.0	2021-05-23
8	virat@gmail.com	Virat	165	Formal Shoe	Puma Formal Shoe	1900.0	2021-05-23
9	virat@gmail.com	Virat	160	Sports Shoe	Reebok	1300.0	2021-05-23
10	virat@gmail.com	Virat	171	Sneakers	Reebok	4000.0	2021-05-23
11	virat@gmail.com	Virat	172	Sneakers	HRX Light	900.0	2021-05-23
12	virat@gmail.com	Virat	165	Formal Shoe	Puma Formal Shoe	1900.0	2021-05-23
13	virat@gmail.com	Virat	163	Casual Shoe	Reebok Air	3000.0	2021-05-23

Showing 1 to 7 of 7 entries (filtered from 13 total entries) Previous 1 Next

When Admin Logouts of the system they are redirected to Home Page

localhost:8080

SportyShoes.com home shop login cart 0

WELCOME TO SPORTYSHOES.COM
Get the Style that suits you the best

CLICK TO BUY SHOES

AdminController

```
package com.project.sportyshoes.controller;

import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.ArrayList;
import java.util.List;
import java.util.Optional;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.multipart.MultipartFile;

import com.project.sportyshoes.dto.ProductDTO;
```

```
import com.project.sportyshoes.global.PurchaseReport;  
import com.project.sportyshoes.model.Category;  
import com.project.sportyshoes.model.Product;  
import com.project.sportyshoes.model.Purchase;  
import com.project.sportyshoes.model.User;  
import com.project.sportyshoes.repository.UserRepository;  
import com.project.sportyshoes.service.CategoryService;  
import com.project.sportyshoes.service.ProductService;  
import com.project.sportyshoes.service.PurchaseService;
```

```
@Controller
```

```
public class AdminController {
```

```
    public static String uploadDir = System.getProperty("user.dir") +  
        "/src/main/resources/static/productImages";
```

```
    @Autowired
```

```
    CategoryService categoryService;
```

```
    @Autowired
```

```
    ProductService productService;
```

```
@Autowired  
PurchaseService purchaseService;
```

```
@Autowired  
UserRepository userRepository;
```

```
@GetMapping("/admin")  
public String adminHome() {  
  
    return "adminHome";  
}
```

```
@GetMapping("/admin/categories")  
public String getCat(Model model) {  
  
    model.addAttribute("categories" ,  
categoryService.getAllCategories());  
  
    return "categories";  
}
```

```
@GetMapping("/admin/categories/add")  
public String getCatAdd(Model model) {  
  
    model.addAttribute("category" , new Category());
```

```
        return "categoriesAdd";  
    }  
  
  
    @PostMapping("/admin/categories/add")  
    public String postCatAdd(@ModelAttribute("category") Category  
category) {  
        categoryService.addCategory(category);  
        return "redirect:/admin/categories";  
    }  
  
  
    @GetMapping("/admin/categories/delete/{id}")  
    public String deleteCat(@PathVariable int id) {  
  
        categoryService.removeCategoryById(id);  
  
  
        return "redirect:/admin/categories";  
    }  
  
  
    @GetMapping("/admin/categories/update/{id}")  
    public String updateCat(@PathVariable int id , Model model) {
```

```
    Optional<Category> category =
categoryService.getCatById(id);

    if(category.isPresent()) {
        model.addAttribute("category" , category.get());
        return "categoriesAdd";
    }else {
        return "404";
    }
}
```

```
// Product Section
```

```
@GetMapping("/admin/products")
public String showProducts(Model model) {

    model.addAttribute("products" ,
productService.getAllProducts());

    return "products";
}
```

```
@GetMapping("/admin/products/add")
public String addProduct(Model model) {

    model.addAttribute("productDTO" , new ProductDTO());
    model.addAttribute("categories" ,
categoryService.getAllCategories());

    return "productsAdd";

}

@GetMapping("/admin/products/add")
public String productAddPost(@ModelAttribute ("productDTO")
ProductDTO productDTO ,
@RequestParam("productImage") MultipartFile file,
@RequestParam("imgName") String imgName) throws
IOException {

    Product product = new Product();

    product.setId(productDTO.getId());
```

```
        product.setName(productDTO.getName());  
  
        product.setCategory(categoryService.getCatById(productDTO.get  
CategoryId()).get());  
        product.setPrice(productDTO.getPrice());  
        product.setWeight(productDTO.getWeight());  
        product.setDescription(productDTO.getDescription());  
        String imageUUID;  
        if (!file.isEmpty()) {  
            imageUUID = file.getOriginalFilename();  
            Path fileNameAndPath = Paths.get(uploadDir,  
imageUUID);  
            Files.write(fileNameAndPath, file.getBytes());  
        }else {  
  
            imageUUID = imgName;  
        }  
  
        product.setImageName(imageUUID);  
        productService.addProduct(product);  
  
    return "redirect:/admin/products";
```

```
}

@GetMapping("/admin/product/delete/{id}")
public String deleteProd(@PathVariable long id) {

    productService.removeProductById(id);

    return "redirect:/admin/products";
}

@GetMapping("/admin/product/update/{id}")
public String updateProd(@PathVariable long id , Model model) {

    Product product = productService.getProductById(id).get();

    ProductDTO productDTO = new ProductDTO();
    productDTO.setId(product.getId());
    productDTO.setName(product.getName());
    productDTO.setCategoryId(product.getCategory().getId());
    productDTO.setPrice(product.getPrice());
    productDTO.setWeight(product.getWeight());
}
```

```
        productDTO.setDescription(product.getDescription());
        productDTO.setImageName(product.getImageName());

        model.addAttribute("productDTO" , productDTO);
        model.addAttribute("categories" ,
categoryService.getAllCategories());

        return "productsAdd";
    }

//Purchase Report
@GetMapping("/admin/purchaseReport")
public String purchaseReport(Model model) {

    List<Purchase>purchaseList =
purchaseService.getAllPurchases();

    List<PurchaseReport>purchaseReportList = new
ArrayList<PurchaseReport>();

    for(Purchase pur : purchaseList) {
```

```
Product product = new Product();

User user = new User();

Category category = new Category();

PurchaseReport purchaseReport = new
PurchaseReport();

Long productId = pur.getProductId();

int userId = pur.getUserId();

product =
productService.getProductById(productId).get();

user = userRepository.findById(userId).get();

category = product.getCategory();

purchaseReport.setEmail(user.getEmail());

purchaseReport.setName(user.getFirstName());

purchaseReport.setProductId(product.getId());

purchaseReport.setProductName(product.getName());

purchaseReport.setPrice(product.getPrice());

purchaseReport.setDate(pur.getOrderDate().toString());

purchaseReport.setCategory(category.getName());

//System.out.println(purchaseReport.getEmail() + " " +
purchaseReport.getProductId() + " " + purchaseReport.getDate());
```

```
        purchaseReportList.add(purchaseReport);

    }

    model.addAttribute("puchaseList" , purchaseReportList);

    return "purchaseReport";
}

@GetMapping("/admin/users")
public String listUsers(Model model) {

    List<User>userList = userRepository.findAll();
    model.addAttribute("userlist" , userList);

    return "userList";
}
```

```
}
```

CartController

```
package com.project.sportyshoes.controller;

import java.time.LocalDate;
import java.util.ArrayList;
import java.util.List;
import java.util.Random;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.core.Authentication;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;

import com.project.sportyshoes.global.GlobalData;
import com.project.sportyshoes.model.Product;
```

```
import com.project.sportyshoes.model.Purchase;
import com.project.sportyshoes.model.User;
import com.project.sportyshoes.repository.PurchaseRepository;
import com.project.sportyshoes.repository.UserRepository;
import com.project.sportyshoes.service.ProductService;

@Controller
public class CartController {

    @Autowired
    ProductService productService;

    @Autowired
    UserRepository userRepository;

    @Autowired
    PurchaseRepository purchaseRepository;

    @GetMapping("/addToCart/{id}")
    public String addToCart(@PathVariable int id) {
        GlobalData.cart.add(productService.getProductById(id).get());
        return "redirect:/shop";
    }
}
```

```
@GetMapping("/cart")
public String cartGet(Model model) {
    model.addAttribute("cartCount" , GlobalData.cart.size());
    model.addAttribute("total" ,
GlobalData.cart.stream().mapToDouble(Product::getPrice).sum());
    model.addAttribute("cart" , GlobalData.cart);
    return "cart";
}
```

```
@GetMapping("/cart/removeItem/{index}")
public String cartItemRemove(@PathVariable int index) {
    GlobalData.cart.remove(index);
    return "redirect:/cart";
}
```

```
@GetMapping("/checkout")
public String checkout(Model model) {
    model.addAttribute("total" ,
GlobalData.cart.stream().mapToDouble(Product::getPrice).sum());
    return "checkout";
}
```

```
@PostMapping("/payNow")
```

```
public String orderConfirmation(Model model) {  
    model.addAttribute("total" ,  
GlobalData.cart.stream().mapToDouble(Product::getPrice).sum());  
    Authentication auth =  
SecurityContextHolder.getContext().getAuthentication();  
    String currentPrincipalName = auth.getName();  
  
    List<Purchase>purchaseList = new ArrayList<Purchase>();  
    //System.out.println(currentPrincipalName);  
    User user =  
userRepository.findUserByEmail(currentPrincipalName).get();  
    for(Product product: GlobalData.cart) {  
        Purchase purchase = new Purchase();  
        //System.out.println(product.getId() + " " +  
product.getName());  
        purchase.setProductId(product.getId());  
        purchase.setUserId(user.getId());  
        purchase.setOrderDate(LocalDate.now());  
        purchaseList.add(purchase);  
    }  
  
    int n = 10000 + new Random().nextInt(90000);  
    model.addAttribute("Receipt" , n);  
    //System.out.println(purchaseList.toString());  
    model.addAttribute("products" , GlobalData.cart);  
    purchaseRepository.saveAll(purchaseList);
```

```
        return "orderPlaced";  
    }  
  
}
```

HomeController

```
package com.project.sportyshoes.controller;  
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Controller;  
import org.springframework.ui.Model;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.PathVariable;  
  
import com.project.sportyshoes.global.GlobalData;  
import com.project.sportyshoes.service.CategoryService;  
import com.project.sportyshoes.service.ProductService;  
  
@Controller  
public class HomeController {  
  
    @Autowired
```

```
CategoryService categoryService;

@Autowired

ProductService productService;

@GetMapping={"/" , "/home"})
public String home(Model model) {

    model.addAttribute("cartCount" , GlobalData.cart.size());

    return "index";

}

@GetMapping("/shop")
public String shop(Model model) {

    model.addAttribute("cartCount" , GlobalData.cart.size());

    model.addAttribute("categories" ,
categoryService.getAllCategories());

    model.addAttribute("products" , productService.getAllProducts());

    return "shop";

}

@GetMapping("/shop/category/{id}")
public String shopByCategory(Model model , @PathVariable int id) {

    model.addAttribute("cartCount" , GlobalData.cart.size());
```

```

        model.addAttribute("categories" ,
categoryService.getAllCategories());

        model.addAttribute("products" ,
productService.getAllProductsByCategoryId(id));

    return "shop";

}

@GetMapping("/shop/viewproduct/{id}")
public String viewProduct(Model model , @PathVariable long id) {

    model.addAttribute("product" ,
productService.getProductById(id).get());

    model.addAttribute("cartCount" , GlobalData.cart.size());

    return "viewProduct";

}

```

LoginController

```
package com.project.sportyshoes.controller;
```

```

import java.util.ArrayList;
import java.util.List;

import javax.servlet.ServletException;

```

```
import javax.servlet.http.HttpServletRequest;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;

import com.project.sportyshoes.global.GlobalData;
import com.project.sportyshoes.model.Role;
import com.project.sportyshoes.model.User;
import com.project.sportyshoes.repository.RoleRepository;
import com.project.sportyshoes.repository.UserRepository;

@Controller
public class LoginController {

    @Autowired
    private BCryptPasswordEncoder bCryptPasswordEncoder;

    @Autowired
    UserRepository userRepository;
```

```
@Autowired  
RoleRepository roleRepository;  
  
@GetMapping("/login")  
public String login() {  
    GlobalData.cart.clear();  
    return "login";  
}  
  
@GetMapping("/register")  
public String registerGet() {  
    return "register";  
}  
  
@PostMapping("/register")  
public String registerPost(@ModelAttribute("user") User user ,  
HttpServletRequest request) throws ServletException {  
  
    String password = user.getPassword();  
  
    user.setPassword(bCryptPasswordEncoder.encode(password));  
    List<Role>roles = new ArrayList<Role>();  
    roles.add(roleRepository.findById(2).get());  
    user.setRoles(roles);  
    userRepository.save(user);
```

```
    request.login(user.getEmail(), password);
    return "redirect:/";
}

}
```

SecurityConfig

```
package com.project.sportyshoes.configuration;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import
org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;
import
org.springframework.security.config.annotation.web.builders.HttpSecurity;
import
org.springframework.security.config.annotation.web.builders.WebSecurity;
import
org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import
org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
```

```
import org.springframework.security.web.util.matcher.AntPathRequestMatcher;

import com.project.sportyshoes.service.CustomUserDetailsService;

@Configuration
@EnableWebSecurity
public class SecurityConfig extends WebSecurityConfigurerAdapter {

    @Autowired
    CustomUserDetailsService customUserDetailsService;

    @Override
    protected void configure(HttpSecurity http) throws Exception {

        http
                .authorizeRequests()
                .antMatchers("/", "/shop/**" , "/forgotpassword" ,
"/register" , "/h2-console/**").permitAll()
                .antMatchers("/admin/**").hasRole("ADMIN")
                .anyRequest()
                .authenticated()
                .and()
                .formLogin()
                .loginPage("/login")
    }
}
```

```
.permitAll()  
.failureUrl("/login?error=true")  
.defaultSuccessUrl("/")  
.usernameParameter("email")  
.passwordParameter("password")  
.and()  
.logout()  
.logoutRequestMatcher(new  
AntPathRequestMatcher("/logout"))  
.logoutSuccessUrl("/login")  
.invalidateHttpSession(true)  
.deleteCookies("JSESSIONID")  
.and()  
.exceptionHandling()  
.and()  
.csrf()  
.disable();
```

```
http.headers().frameOptions().disable();  
}
```

```
@Bean  
public BCryptPasswordEncoder bCryptPasswordEncoder() {  
    return new BCryptPasswordEncoder();  
}
```

```
    @Override
    protected void configure(AuthenticationManagerBuilder auth) throws
Exception {
    auth.userDetailsService(customUserDetailsService);

}

@Override
public void configure(WebSecurity web) throws Exception {
    web.ignoring().antMatchers("/resources/**", "/static/**",
"/images/**", "/productImages/**", "/css/**", "/js/**")
    ;
}

}
```

ProductDTO

```
package com.project.sportyshoes.dto;
```

```
public class ProductDTO {
```

```
    private Long id;
```

```
    private String name;
```

```
    private int categoryId;
```

```
    private double price;
```

```
    private double weight;
```

```
    private String description;
```

```
    private String imageName;
```

```
    public Long getId() {
```

```
        return id;
```

```
    }
```

```
    public void setId(Long id) {
```

```
        this.id = id;
```

```
    }
```

```
    public String getName() {
```

```
        return name;
```

```
    }
```

```
public void setName(String name) {  
    this.name = name;  
}  
  
public int getCategorylD() {  
    return categorylD;  
}  
  
public void setCategorylD(int categorylD) {  
    this.categorylD = categorylD;  
}  
  
public double getPrice() {  
    return price;  
}  
  
public void setPrice(double price) {  
    this.price = price;  
}  
  
public double getWeight() {  
    return weight;  
}
```

```
public void setWeight(double weight) {  
    this.weight = weight;  
}  
  
public String getDescription() {  
    return description;  
}  
  
public void setDescription(String description) {  
    this.description = description;  
}  
  
public String getImageName() {  
    return imageName;  
}  
  
public void setImageName(String imageName) {  
    this.imageName = imageName;  
}  
}  
  
PurchaseReport  
package com.project.sportyshoes.global;
```

```
import org.springframework.stereotype.Component;

@Component
public class PurchaseReport {

    private String email;

    private String name;

    private Long productId;

    private String productName;

    private double price;

    private String date;

    private String category;

    public String getCategory() {
        return category;
    }
}
```

```
public void setCategory(String category) {  
    this.category = category;  
}  
  
public String getEmail() {  
    return email;  
}  
  
public void setEmail(String email) {  
    this.email = email;  
}  
  
public String getName() {  
    return name;  
}  
  
public void setName(String name) {  
    this.name = name;  
}  
  
public Long getProductId() {  
    return productId;  
}
```

```
public void setProductId(Long productId) {  
    this.productId = productId;  
}  
  
public String getProductName() {  
    return productName;  
}  
  
public void setProductName(String productName) {  
    this.productName = productName;  
}  
  
public double getPrice() {  
    return price;  
}  
  
public void setPrice(double price) {  
    this.price = price;  
}  
  
public String getDate() {  
    return date;  
}
```

```
public void setDate(String date) {  
    this.date = date;  
}  
  
}
```

Model:

Category

```
package com.project.sportyshoes.model;  
  
import javax.persistence.Column;  
import javax.persistence.Entity;  
import javax.persistence.GeneratedValue;  
import javax.persistence.GenerationType;  
import javax.persistence.Id;  
  
@Entity  
public class Category {  
  
    @Id  
    @GeneratedValue(strategy = GenerationType.AUTO)  
    @Column(name = "category_id")
```

```
private int id;  
  
private String name;  
  
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;  
}  
}
```

CustomUserDetail

```
package com.project.sportyshoes.model;

import java.util.ArrayList;
import java.util.Collection;
import java.util.List;

import org.springframework.security.core.GrantedAuthority;
import org.springframework.security.core.authority.SimpleGrantedAuthority;
import org.springframework.security.core.userdetails.UserDetails;

public class CustomUserDetail extends User implements UserDetails {

    public CustomUserDetail(User user) {
        super(user);
    }

    @Override
    public Collection<? extends GrantedAuthority> getAuthorities() {

        List<GrantedAuthority> authorityList = new
        ArrayList<GrantedAuthority>();

        super.getRoles().forEach(role -> {

            authorityList.add(new SimpleGrantedAuthority(role.getName())));
        })
    }
}
```

```
        authorityList.add(new
SimpleGrantedAuthority(role.getName())));
    });

    return authorityList;
}

@Override
public String getPassword() {
    // TODO Auto-generated method stub
    return super.getPassword();
}

@Override
public String getUsername() {
    // TODO Auto-generated method stub
    return super.getEmail();
}

@Override
public boolean isAccountNonExpired() {
    // TODO Auto-generated method stub
    return true;
}
```

```
@Override  
public boolean isAccountNonLocked() {  
    // TODO Auto-generated method stub  
    return true;  
}
```

```
@Override  
public boolean isCredentialsNonExpired() {  
    // TODO Auto-generated method stub  
    return true;  
}
```

```
@Override  
public boolean isEnabled() {  
    // TODO Auto-generated method stub  
    return true;  
}
```

```
}
```

Product

```
package com.project.sportyshoes.model;
```

```
import javax.persistence.Entity;  
import javax.persistence.FetchType;  
import javax.persistence.GeneratedValue;  
import javax.persistence.GenerationType;  
import javax.persistence.Id;  
import javax.persistence.JoinColumn;  
import javax.persistence.ManyToOne;
```

```
@Entity  
public class Product {  
  
    @Id  
    @GeneratedValue(strategy = GenerationType.AUTO)  
    private Long id;  
  
    private String name;  
  
    @ManyToOne(fetch = FetchType.LAZY)  
    @JoinColumn(name = "category_id" , referencedColumnName =  
    "category_id")  
    private Category category;  
  
    private double price;
```

```
private double weight;
```

```
private String description;
```

```
private String imageName;
```

```
public Long getId() {
```

```
    return id;
```

```
}
```

```
public void setId(Long id) {
```

```
    this.id = id;
```

```
}
```

```
public String getName() {
```

```
    return name;
```

```
}
```

```
public void setName(String name) {
```

```
    this.name = name;
```

```
}
```

```
public Category getCategory() {
```

```
        return category;  
    }  
  
    public void setCategory(Category category) {  
        this.category = category;  
    }  
  
    public double getPrice() {  
        return price;  
    }  
  
    public void setPrice(double price) {  
        this.price = price;  
    }  
  
    public double getWeight() {  
        return weight;  
    }  
  
    public void setWeight(double weight) {  
        this.weight = weight;  
    }  
  
    public String getDescription() {
```

```
        return description;  
    }  
  
    public void setDescription(String description) {  
        this.description = description;  
    }  
  
    public String getImageName() {  
        return imageName;  
    }  
  
    public void setImageName(String imageName) {  
        this.imageName = imageName;  
    }  
}
```

Purchase

```
package com.project.sportyshoes.model;
```

```
import java.time.LocalDate;
```

```
import javax.persistence.Column;
```

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

@Entity
public class Purchase {

    @Id
    @GeneratedValue(strategy = GenerationType.SEQUENCE)
    private long id;

    private int userId;

    private Long productId;

    @Column(name = "order_date")
    private LocalDate orderDate;

    public long getId() {
        return id;
    }

    public void setId(long id) {
```

```
        this.id = id;  
    }  
  
    public int getUserId() {  
        return userId;  
    }  
  
    public void setUserId(int userId) {  
        this.userId = userId;  
    }  
  
    public Long getProductId() {  
        return productId;  
    }  
  
    public void setProductId(Long productId) {  
        this.productId = productId;  
    }  
  
    public LocalDate getOrderDate() {  
        return orderDate;  
    }  
  
    public void setOrderDate(LocalDate orderDate) {
```

```
    this.orderDate = orderDate;  
}  
  
}
```

Role

```
package com.project.sportyshoes.model;  
  
import java.util.List;  
  
import javax.persistence.Column;  
import javax.persistence.Entity;  
import javax.persistence.GeneratedValue;  
import javax.persistence.GenerationType;  
import javax.persistence.Id;  
import javax.persistence.ManyToMany;  
import javax.persistence.Table;  
import javax.validation.constraints.NotEmpty;  
  
@Entity  
@Table(name = "roles")  
public class Role {
```

```
@Id  
@GeneratedValue(strategy = GenerationType.AUTO)  
private Integer id;  
  
@Column(nullable = false , unique = true)  
@NotEmpty  
private String name;  
  
@ManyToMany(mappedBy = "roles")  
private List<User>users;  
  
public Integer getId() {  
    return id;  
}  
  
public void setId(Integer id) {  
    this.id = id;  
}  
  
public String getName() {  
    return name;  
}  
  
public void setName(String name) {
```

```
        this.name = name;  
    }  
  
    public List<User> getUsers() {  
        return users;  
    }  
  
    public void setUsers(List<User> users) {  
        this.users = users;  
    }  
}
```

User

```
package com.project.sportyshoes.model;  
  
import java.util.List;  
  
import javax.persistence.CascadeType;  
import javax.persistence.Column;  
import javax.persistence.Entity;  
import javax.persistence.FetchType;  
import javax.persistence.GeneratedValue;  
import javax.persistence.GenerationType;  
import javax.persistence.Id;
```

```
import javax.persistence.JoinColumn;
import javax.persistence.JoinTable;
import javax.persistence.ManyToMany;
import javax.persistence.Table;
import javax.validation.constraints.Email;
import javax.validation.constraints.NotEmpty;

@Entity
@Table(name = "users")
public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Integer id;

    @NotEmpty
    @Column(nullable = false)
    private String firstName;

    private String lastName;

    @NotEmpty
    @Column(nullable = false , unique = true)
```

```
@Email(message = "{errors.invalid_email}")

private String email;

private String password;

@ManyToMany(cascade = CascadeType.MERGE , fetch = FetchType.EAGER)
@JoinTable(
    name = "user_role",
    joinColumns = {@JoinColumn( name = "USER_ID" ,
referencedColumnName = "ID" )},
    inverseJoinColumns = {@JoinColumn( name = "ROLE_ID" ,
referencedColumnName = "ID" )}
)

private List<Role>roles;

public Integer getId() {
    return id;
}

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

```
public String getFirstName() {  
    return firstName;  
}  
  
public void setFirstName(String firstName) {  
    this.firstName = firstName;  
}  
  
public String getLastname() {  
    return lastName;  
}  
  
public void setLastName(String lastName) {  
    this.lastName = lastName;  
}  
  
public String getEmail() {  
    return email;  
}  
  
public void setEmail(String email) {  
    this.email = email;  
}
```

```
public String getPassword() {  
    return password;  
}  
  
public void setPassword(String passsword) {  
    this.password = passsword;  
}  
  
public List<Role> getRoles() {  
    return roles;  
}  
  
public void setRoles(List<Role> roles) {  
    this.roles = roles;  
}  
  
public User(User user) {  
    super();  
    this.firstName = user.getFirstName();  
    this.lastName = user.getLastName();  
    this.email = user.getEmail();  
    this.password = user.getPassword();  
    this.roles = user.getRoles();  
}
```

```
public User() {}  
  
}  

```

Repository :

CategoryRepository

```
package com.project.sportyshoes.repository;  
  
import org.springframework.data.jpa.repository.JpaRepository;  
  
import com.project.sportyshoes.model.Category;  
  
public interface CategoryRepository extends JpaRepository<Category, Integer> {  
  
}  

```

ProductRepository

```
package com.project.sportyshoes.repository;
```

```
import java.util.List;

import org.springframework.data.jpa.repository.JpaRepository;

import com.project.sportyshoes.model.Product;

public interface ProductRepository extends JpaRepository<Product, Long> {

    List<Product> findAllByCategory_Id(int id);

}
```

PurchaseRepository

```
package com.project.sportyshoes.repository;

import org.springframework.data.jpa.repository.JpaRepository;

import com.project.sportyshoes.model.Purchase;

public interface PurchaseRepository extends JpaRepository<Purchase, Long> {

}
```

RoleRepository

```
package com.project.sportyshoes.repository;
```

```
import org.springframework.data.jpa.repository.JpaRepository;  
  
import com.project.sportyshoes.model.Role;  
  
public interface RoleRepository extends JpaRepository<Role, Integer> {  
}  

```

UserRepository

```
package com.project.sportyshoes.repository;  
  
import java.util.Optional;  
  
import org.springframework.data.jpa.repository.JpaRepository;  
  
import com.project.sportyshoes.model.User;  
  
public interface UserRepository extends JpaRepository<User, Integer> {  
    Optional<User> findUserByEmail(String email);  
}
```

