SHOPPING APP

MOHAMMED ANAS K P

**STS CODINGS(BACKEND)**

**package** org.dxc.shopping.app;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

**public** **class** ShoppingAppApplication {

**public** **static** **void** main(String[] args) {

SpringApplication.*run*(ShoppingAppApplication.**class**, args);

}

}

**//Configuration Class**

**package** org.dxc.shopping.app.config;

**import** java.util.ArrayList;

**import** java.util.List;

**import** java.util.Set;

**import** javax.persistence.EntityManager;

**import** javax.persistence.metamodel.EntityType;

**import** org.dxc.shopping.app.entity.Product;

**import** org.dxc.shopping.app.entity.ProductCategory;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.context.annotation.Configuration;

**import** org.springframework.data.rest.core.config.RepositoryRestConfiguration;

**import** org.springframework.data.rest.webmvc.config.RepositoryRestConfigurer;

**import** org.springframework.http.HttpMethod;

@Configuration

**public** **class** RestConfig **implements** RepositoryRestConfigurer {

**private** EntityManager entityManager;

@Autowired

**public** RestConfig(EntityManager entityManager) {

**this**.entityManager = entityManager;

}

@Override

**public** **void** configureRepositoryRestConfiguration(RepositoryRestConfiguration config) {

HttpMethod[] unsupportedActions= {HttpMethod.***POST***,HttpMethod.***PUT***,HttpMethod.***DELETE***,HttpMethod.***PATCH***};

config.getExposureConfiguration().forDomainType(Product.**class**).withItemExposure((metadata,httpMethods)->httpMethods.disable(unsupportedActions));

config.getExposureConfiguration().forDomainType(ProductCategory.**class**).withItemExposure((metadata,httpMethods)->httpMethods.disable(unsupportedActions));

exposeId(config);

}

**public** **void** exposeId(RepositoryRestConfiguration config) {

Set<EntityType<?>> entities = entityManager.getMetamodel().getEntities();

List<Class> entityClass = **new** ArrayList<>();

**for** (EntityType<?> entityType : entities) {

entityClass.add(entityType.getJavaType());

}

Class[] domainType = entityClass.toArray(**new** Class[0]);

config.exposeIdsFor(domainType);

}

}

// **Product Entity Class**

**package** org.dxc.shopping.app.entity;

**import** java.math.BigDecimal;

**import** java.sql.Date;

**import** javax.persistence.Column;

**import** javax.persistence.Entity;

**import** javax.persistence.GeneratedValue;

**import** javax.persistence.GenerationType;

**import** javax.persistence.Id;

**import** javax.persistence.JoinColumn;

**import** javax.persistence.ManyToOne;

**import** javax.persistence.Table;

**import** org.hibernate.annotations.CreationTimestamp;

**import** org.hibernate.annotations.UpdateTimestamp;

**import** lombok.Data;

@Entity

@Data

@Table(name = "product")

**public** **class** Product {

@Id

@GeneratedValue(strategy = GenerationType.***IDENTITY***)

**private** Long id;

@Column(name = "sku")

**private** String sku;

@ManyToOne

@JoinColumn(name="category\_id", nullable=**false**)

**private** ProductCategory category;

@Column(name = "name")

**private** String name;

@Column(name = "description")

**private** String description;

@Column(name = "unit\_price")

**private** BigDecimal unitPrice;

@Column(name = "image\_url")

**private** String imageUrl;

@Column(name = "active")

**private** Boolean active;

@Column(name = "units\_in\_stock")

**private** Integer unitInStock;

@Column(name = "date\_created")

@CreationTimestamp

**private** Date dateCreated;

@Column(name = "last\_updated")

@UpdateTimestamp

**private** Date lastModified;

}

**// Product-Category Class**

**package** org.dxc.shopping.app.entity;

**import** java.util.Set;

**import** javax.persistence.CascadeType;

**import** javax.persistence.Column;

**import** javax.persistence.Entity;

**import** javax.persistence.GeneratedValue;

**import** javax.persistence.GenerationType;

**import** javax.persistence.Id;

**import** javax.persistence.OneToMany;

**import** javax.persistence.Table;

**import** lombok.Data;

@Entity

@Data

@Table(name = "product\_category")

**public** **class** ProductCategory {

@Id

@GeneratedValue(strategy = GenerationType.***IDENTITY***)

@Column(name = "id")

**private** Long id;

@Column(name = "category\_name")

**private** String categoryName;

@OneToMany(cascade = CascadeType.***ALL***, mappedBy = "category")

**private** Set<Product> products;

}

// **Product-Category-Repository**

**package** org.dxc.shopping.app.repository;

**import** org.dxc.shopping.app.entity.ProductCategory;

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

**import** org.springframework.data.rest.core.annotation.RepositoryRestResource;

**import** org.springframework.web.bind.annotation.CrossOrigin;

@RepositoryRestResource(collectionResourceRel="productCategory",path="product\_category")

@CrossOrigin

**public** **interface** ProductCategoryRepository **extends** JpaRepository<ProductCategory,Long>{

}

// **Product-Repository**

**package** org.dxc.shopping.app.repository;

**import** org.dxc.shopping.app.entity.Product;

**import** org.springframework.data.domain.Page;

**import** org.springframework.data.domain.Pageable;

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

**import** org.springframework.data.rest.core.annotation.RepositoryRestResource;

**import** org.springframework.web.bind.annotation.CrossOrigin;

**import** org.springframework.web.bind.annotation.RequestParam;

@CrossOrigin

**public** **interface** ProductRepository **extends** JpaRepository<Product,Long>{

Page<Product> findByCategoryId(@RequestParam("id")Long id,Pageable pageable);

}

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.datasource.url=jdbc:mysql://localhost:3306/full-stack-eshop

spring.datasource.username=eshopapp

spring.datasource.password=eshopapp

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect

spring.data.rest.base-path=/api

**// POM.xml**

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

<project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.3.3.RELEASE</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<groupId>org.dxc.shopping.app</groupId>

<artifactId>shopping-app</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>Shopping-App</name>

<description>shopping-app</description>

<properties>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-rest</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<optional>true</optional>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

<exclusions>

<exclusion>

<groupId>org.junit.vintage</groupId>

<artifactId>junit-vintage-engine</artifactId>

</exclusion>

</exclusions>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

**ANGULAR CODINGS(FRONTEND)**

**product.ts**

export class Product {

    sku: string;

    name: string;

    description: string;

    unitPrice: number;

    imageUrl: string;

    active: boolean;

    unitsInStock: number;

    dateCreated: Date;

    dateUpdated: Date;

}

**product-category.ts**

export class ProductCategory {

    id: number;

    categoryName: string;

}

**product-category-menu.component.ts**

import { Component, OnInit } from '@angular/core';

import { ProductCategory } from 'src/app/common/product-category';

import { ProductService } from 'src/app/service/product.service';

@Component({

  selector: 'app-product-category-menu',

  templateUrl: './product-category-menu.component.html',

  styleUrls: ['./product-category-menu.component.css']

})

export class ProductCategoryMenuComponent implements OnInit {

 productCategories: ProductCategory[];

  constructor(private productService: ProductService) { }

  ngOnInit(): void {

    this.listProductCategory();

  }

listProductCategory(): void{

  this.productService.getProductCategories().subscribe(data => {

    console.log('product-categories' + JSON.stringify(data));

    this.productCategories = data;

  });

}

}

**product-category-menu.component.html**

<div class="menu-sidebar-content js-scrollbar1">

    <nav class="navbar-sidebar">

        <ul class="list-unstyled navbar-list">

            <li \*ngFor= "let productcategory of productCategories">

                <a routerLink="category/{{ productcategory.id }}"

                routerLinkActive="active-link">{{ productcategory.categoryName }}</a>

            </li>

        </ul>

    </nav>

</div>

**product-list.component.ts**

import { Component, OnInit } from '@angular/core';

import { ActivatedRoute } from '@angular/router';

import { Product } from 'src/app/common/product';

import { ProductService } from 'src/app/service/product.service';

@Component({

  selector: 'app-product-list',

  templateUrl: './product-list.component copy.html',

  styleUrls: ['./product-list.component.css']

})

export class ProductListComponent implements OnInit {

  products: Product[];

  currentCategoryId: number;

  constructor(private productService: ProductService, private route: ActivatedRoute) { }

  ngOnInit(): void {

    this.route.paramMap.subscribe(() => {this.listProducts(); } );

  }

  // tslint:disable-next-line: typedef

  listProducts(){

     // check if id parameter is available

     const hasCategoryId: boolean = this.route.snapshot.paramMap.has('id');

     if (hasCategoryId) {

       this.currentCategoryId = +this.route.snapshot.paramMap.get('id');

     } else{

       this.currentCategoryId = 2;

     }

     this.productService.getProductList(this.currentCategoryId).subscribe(

      data => {

        this.products = data;

      }

    );

}

}

**product-list.component copy.html**

<div class="main-content">

    <div class="section-content section-content-p30">

        <div class="container-fluid">

            <div class="row">

            <div \*ngFor="let tempProduct of products" class="col-md-3">

                <div class="product-box">

                        <img  src="{{ tempProduct.imageUrl}}" >

                    <h1>{{ tempProduct.name}}</h1>

                    <div class="price">{{ tempProduct.unitPrice | currency:'INR'}}</div>

                    <a href="#" class="primary-btn">Add to Cart</a>

                </div>

            </div>

            </div>

        </div>

    </div>

</div>

**product.service.ts**

import { Injectable } from '@angular/core';

import { HttpClient } from '@angular/common/http';

import { Observable } from 'rxjs';

import { Product } from '../common/product';

import { map } from 'rxjs/operators';

import { ProductCategory } from '../common/product-category';

@Injectable({

  providedIn: 'root'

})

export class ProductService {

  private baseUrl = 'http://localhost:8080/api/products';

  private categoryUrl = 'http://localhost:8080/api/product\_category';

  constructor(private httpClient: HttpClient) { }

  getProductList(theCategoryId: number): Observable<Product[]>{

  const searchUrl = `${this.baseUrl}/search/findByCategoryId?id=${theCategoryId}`;

  return this.httpClient.get<GetResponseProduct>(searchUrl)

    .pipe(map(response => response.\_embedded.products));

  }

getProductCategories(): Observable<ProductCategory[]>{

  return this.httpClient.get<GetResponseProductCategory>(this.categoryUrl)

    .pipe(map(response => response.\_embedded.productCategory));

}

}

interface GetResponseProduct{

  \_embedded: {

    products: Product[];

  };

}

interface GetResponseProductCategory{

  \_embedded: {

    productCategory: ProductCategory[];

  };

}

**app-routing.module.ts**

import { NgModule } from '@angular/core';

import { Routes, RouterModule } from '@angular/router';

import { ProductListComponent } from './component/product-list/product-list.component';

const routes: Routes = [

  { path: 'category/:id', component: ProductListComponent},

  { path: 'category', component: ProductListComponent},

  { path: 'product', component: ProductListComponent},

  { path: '', redirectTo: '/product', pathMatch: 'full'},

  { path: '\*\*', redirectTo: '/product', pathMatch: 'full'}

];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

**app.component.html**

<div class="page-wrapper">

  <!------------------ MENU SIDEBAR-------------------->

  <aside class="menu-sidebar d-none d-lg-block">

      <div class=logo>

          <a href="#">

              <img src="assets/logo.jpeg" alt="eShop" class="navbar-brand img-responsive">

          </a>

      </div>

      <app-product-category-menu></app-product-category-menu>

  </aside>

<!-----------------------END MENU SIDEBAR----------------------------->

<!-----------------------PAGE CONTAINER------------------------------->

<div class="page-container">

  <header class="header-desktop">

      <div class="section-content section-content-p30">

          <div class="container-fluid">

              <div class="header-wrap">

                  <!-- form data - to search product by keybord or text-->

                  <form class="form-header" onsubmit="return false;" method="GET">

                      <input  class="au-input au-input-xl" type="text" name="search" placeholder="Search for product"/>

                      <button class="au-btn-submit" type="submit">Search</button>

                  </form>

                   <div class="cart-area d-n">

                   <!-- add shopping cart-->

                   <a href="#">

                      <div class="total">19.22

                          <span>2</span>

                      </div>

                      <i class="fa fa-shopping-cart" aria-hidden="true"></i>

                  </a>

              </div>

          </div>

              <div class="account-wrap"> </div>

          </div>

      </div>

  </header>

  <router-outlet></router-outlet>

  <!--------------------------HEADER END---------------------------->

  <!------------------------------MAIN CONTENT---------------------->

</div>

</div>

<!----------------------------------ADD FOOTER------------------------>

<footer>

   <ul>

       <li>

          <a href="#">About Us</a>

       </li>

       <li>

          <a href="#">Contact US </a>

      </li>

      <li>

         <a href="#">Help</a>

      </li>

   </ul>

</footer>

**app.component.ts**

import { Component } from '@angular/core';

@Component({

  selector: 'app-root',

  templateUrl: './app.component.html',

  styleUrls: ['./app.component.css']

})

export class AppComponent {

  title = 'shoppingapp';

}

**app.module.ts**

import { BrowserModule } from '@angular/platform-browser';

import { NgModule } from '@angular/core';

import { HttpClientModule } from '@angular/common/http';

import { ProductService } from './service/product.service';

import { AppRoutingModule } from './app-routing.module';

import { AppComponent } from './app.component';

import { ProductListComponent } from './component/product-list/product-list.component';

import { ProductCategoryMenuComponent } from './component/product-category-menu/product-category-menu.component';

@NgModule({

  declarations: [

    AppComponent,

    ProductListComponent,

    ProductCategoryMenuComponent

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    HttpClientModule

  ],

  providers: [ProductService],

  bootstrap: [AppComponent]

})

export class AppModule { }

**index.html**

<!doctype html>

<html lang="en">

<head>

  <meta charset="utf-8">

  <title>Shoppingapp</title>

  <base href="/">

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">

  <link rel="icon" type="image/x-icon" href="favicon.ico">

</head>

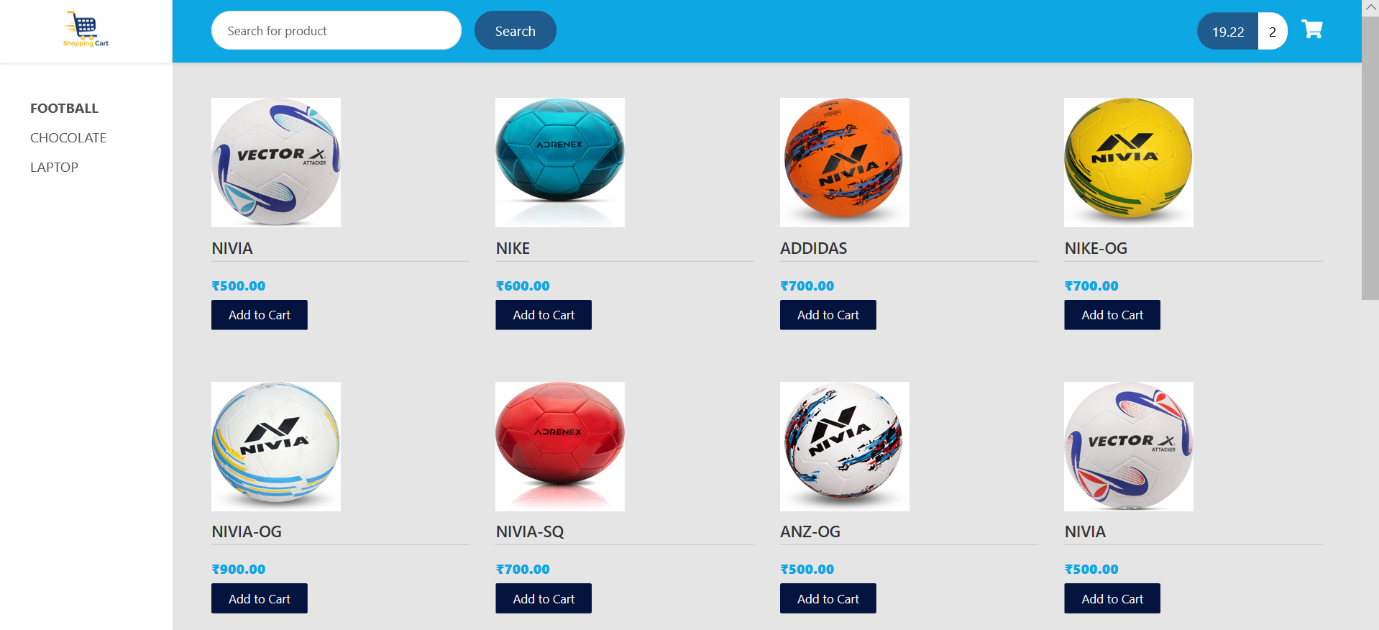
<body>

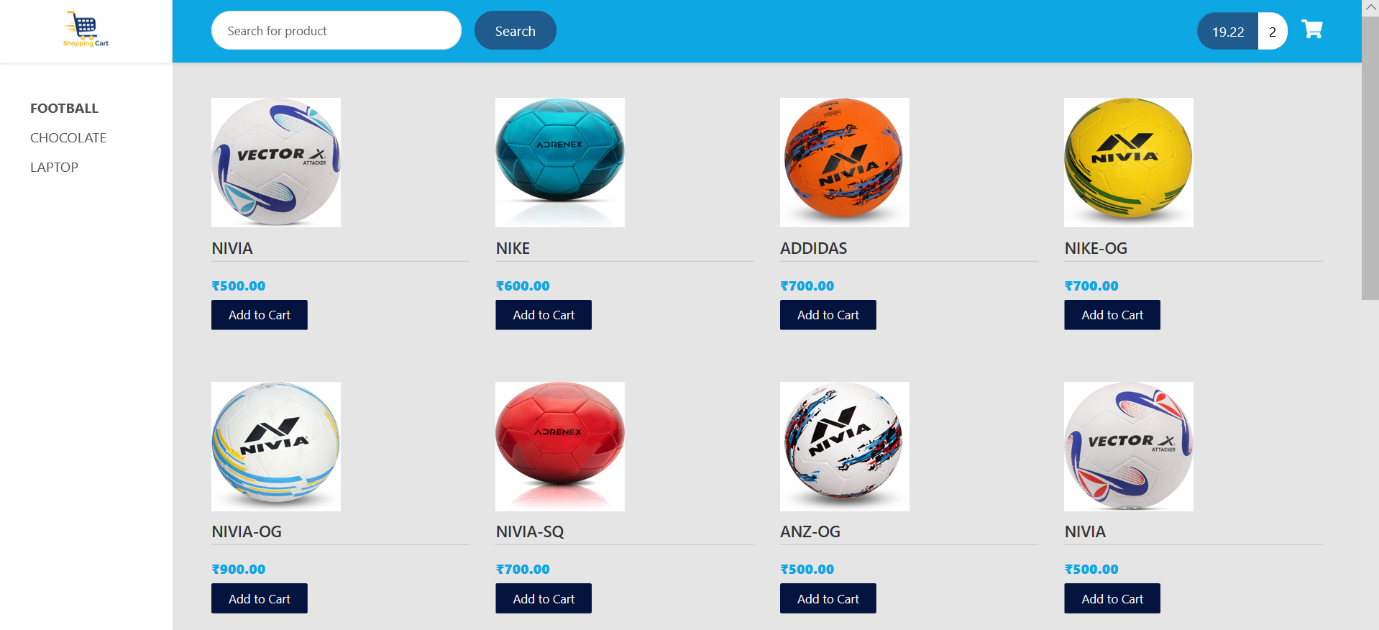
  <app-root></app-root>

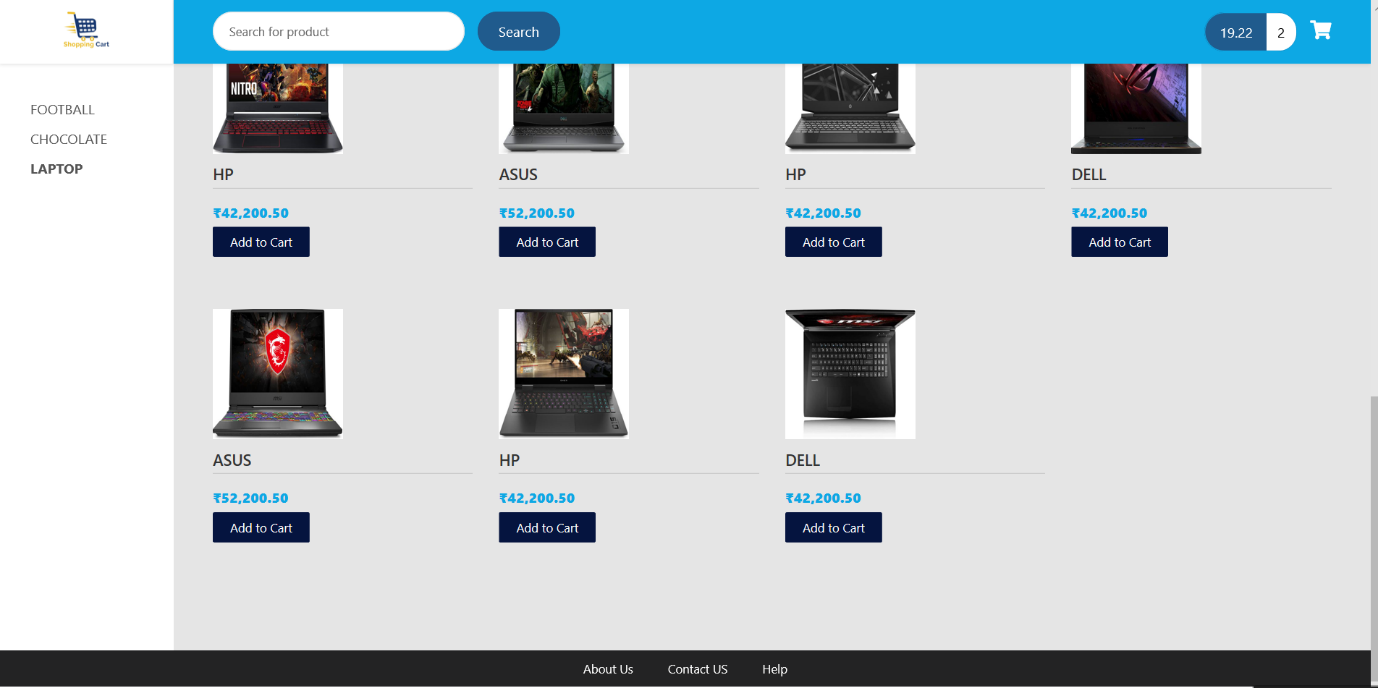
</body>

</html>

**PROJECT MODEL**







**MySQL COMMANDS**

* CREATE USER 'eshop'@'localhost' IDENTIFIED BY 'eshop';
* GRANT ALL PRIVILEGES ON \* . \* TO 'eshop'@'localhost';
* ALTER USER 'eshop'@'localhost' IDENTIFIED WITH mysql\_native\_password BY 'eshop';
* DROP SCHEMA IF EXISTS `full-stack-eshop`;
* CREATE SCHEMA `full-stack-eshop`;
* USE `full-stack-eshop`;
* CREATE TABLE IF NOT EXISTS `full-stack-eshop`.`Product\_category`(

`id`BIGINT(20) NOT NULL AUTO\_INCREMENT,

`category\_name` VARCHAR(255) NULL DEFAULT NULL,

PRIMARY KEY(`id`))

ENGINE=InnoDB

AUTO\_INCREMENT=1;

* CREATE TABLE IF NOT EXISTS `full-stack-eshop`.`product`(

`id` BIGINT(20) NOT NULL AUTO\_INCREMENT,

`sku` VARCHAR(255) DEFAULT NULL,

`name` VARCHAR(255) DEFAULT NULL,

`description` VARCHAR(255) DEFAULT NULL,

`unit\_price` DECIMAL(13,2) DEFAULT NULL,

`image\_url` VARCHAR(255) DEFAULT NULL,

`active` BIT DEFAULT 1,

`units\_in\_stock` INT(11) DEFAULT NULL,

`date\_created` DATETIME(6) DEFAULT NULL,

`last\_updated` DATETIME(6) DEFAULT NULL,

`category\_id` BIGINT(20) NOT NULL,

PRIMARY KEY (`id`),

KEY `fk\_category` (`category\_id`),

CONSTRAINT `fk\_category` FOREIGN KEY (`category\_id`) REFERENCES `product\_category` (`id`)

)

ENGINE=InnoDB

AUTO\_INCREMENT=1;

* insert into product\_category(category\_name) values('LAPTOP');
* insert into product (sku, name, description, unit\_price, image\_url, active,

units\_in\_stock, category\_id, date\_created)

values ('LAPTOP-1020', 'DELL', 'HIGH-SPEED-OFFICE-LAPTOP' , 42200.50, 'assets/images/laptop/laptop-1020', 1,

10, 4, now());

* insert into product (sku, name, description, unit\_price, image\_url, active,

units\_in\_stock, category\_id, date\_created)

values ('LAPTOP-1018', 'ASUS', 'HIGH-SPEED-OFFICE-LAPTOP' , 52200.50, 'assets/images/laptop/laptop-1018', 1,

10, 4, now());

* insert into product (sku, name, description, unit\_price, image\_url, active,

units\_in\_stock, category\_id, date\_created)

values ('LAPTOP-1019', 'HP', 'HIGH-SPEED-OFFICE-LAPTOP' , 42200.50, 'assets/images/laptop/laptop-1019', 1,

10, 4, now());

* insert into product\_category(category\_name) values('FOOTBALL');
* insert into product (sku, name, description, unit\_price, image\_url, active,

units\_in\_stock, category\_id, date\_created)

values ('FOOTBALL-1001', 'NIVIA', 'TURF-BALL' , 500, 'assets/images/football/football-1001.jpeg', 1,

10, 2, now());

* insert into product (sku, name, description, unit\_price, image\_url, active,

units\_in\_stock, category\_id, date\_created)

values ('FOOTBALL-1002', 'NIKE', 'TURF-BALL' , 600, 'assets/images/football/football-1002.jpeg', 1,

10, 2, now());

* insert into product (sku, name, description, unit\_price, image\_url, active,

units\_in\_stock, category\_id, date\_created)

values ('FOOTBALL-1003', 'ADDIDAS', 'TURF-BALL' , 700, 'assets/images/football/football-1003.jpeg', 1,

10, 2, now());

* insert into product (sku, name, description, unit\_price, image\_url, active,

units\_in\_stock, category\_id, date\_created)

values ('FOOTBALL-1004', 'NIKE-OG', 'PITCH-BALL' , 700, 'assets/images/football/football-1004.jpeg', 1,

10, 2, now());

* insert into product (sku, name, description, unit\_price, image\_url, active,

units\_in\_stock, category\_id, date\_created)

values ('FOOTBALL-1005', 'NIVIA-OG', 'PITCH-BALL' , 900, 'assets/images/football/football-1005.jpeg', 1,

10, 2, now());

* insert into product (sku, name, description, unit\_price, image\_url, active,

units\_in\_stock, category\_id, date\_created)

values ('FOOTBALL-1006', 'NIVIA-SQ', 'TURF-BALL' , 700, 'assets/images/football/football-1006.jpeg', 1,

10, 2, now());

* insert into product (sku, name, description, unit\_price, image\_url, active,

units\_in\_stock, category\_id, date\_created)

values ('FOOTBALL-1007', 'ANZ-OG', 'TURF-BALL' , 500, 'assets/images/football/football-1007.jpeg', 1,

10, 2, now());