package com.foodbox;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class FoodboxApplication {

public static void main(String[] args) {

SpringApplication.run(FoodboxApplication.class, args);

}

}

package com.foodbox.model;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

@Data

@AllArgsConstructor

@NoArgsConstructor

@Entity

@Table(name = "admin\_foodbox")

public class Admin {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name="admin\_id")

private long id;

private String email;

private String password;

}

package com.foodbox.model;

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.ManyToOne;

import javax.persistence.OneToOne;

import javax.persistence.Table;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

@Data

@AllArgsConstructor

@NoArgsConstructor

@Entity

@Table(name = "cart\_foodbox")

public class Cart {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name="cart\_id")

private long id;

private int unit;

private double totalprice;

@OneToOne(targetEntity = Customer.class,fetch = FetchType.EAGER)

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

private Customer customer;

@ManyToOne

@JoinColumn(name = "item\_id")

private Item item;

public Cart(Customer customer, Item item) {

this.customer = customer;

this.item = item;

}

}

package com.foodbox.model;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

@Data

@AllArgsConstructor

@NoArgsConstructor

@Entity

@Table(name = "customer\_table")

public class Customer {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name="customer\_id")

private long id;

private String name;

private String email;

private String password;

private String location;

private long phone;

}

package com.foodbox.model;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

@Data

@AllArgsConstructor

@NoArgsConstructor

@Entity

@Table(name = "item\_table")

public class Item {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name="item\_id")

private long id;

private String name;

private String url;

private String catagory;

private double price;

private int quantity;

}

package com.foodbox.dto;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

@Data

@AllArgsConstructor

@NoArgsConstructor

public class AdminDto {

private long id;

private String email;

private String password;

}

package com.foodbox.dto;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

@Data

@AllArgsConstructor

@NoArgsConstructor

public class CustomerDto {

private long id;

private String name;

private String email;

private String password;

private String location;

private long phone;

}

package com.foodbox.dto;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

@Data

@AllArgsConstructor

@NoArgsConstructor

public class ItemDto {

private long id;

private String name;

private String url;

private String catagory;

private double price;

private int quantity;

}

package com.foodbox.repository;

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

import org.springframework.stereotype.Repository;

import com.foodbox.model.Admin;

@Repository

public interface AdminRepository extends JpaRepository<Admin, Long>{

}

package com.foodbox.repository;

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

import org.springframework.stereotype.Repository;

import com.foodbox.model.Customer;

@Repository

public interface CustomerRepository extends JpaRepository<Customer, Long>{

}

package com.foodbox.service;

import java.util.List;

import com.foodbox.dto.AdminDto;

import com.foodbox.model.Admin;

public interface AdminService {

public String addAdmin(AdminDto admindto);

List<Admin> getAllAdmin();

}

package com.foodbox.service;

import java.util.List;

import com.foodbox.dto.CustomerDto;

import com.foodbox.model.Customer;

public interface CustomerService {

public String addCustomer(CustomerDto customerdto);

List<Customer> getAllCustomer();

public Customer getCustomerById(long id);

public String updateCustomer(CustomerDto customerdto);

public String deleteCustomerById(long id);

}

package com.foodbox.service;

import java.util.List;

import org.springframework.web.multipart.MultipartFile;

import com.foodbox.dto.ItemDto;

import com.foodbox.model.Item;

public interface ItemService {

public String addItem(ItemDto itemdto);

List<Item> getAllItems();

public Item getItemById(long id);

public String updateItem(ItemDto itemdto);

public String deleteItemById(long id);

public void saveMultiple(MultipartFile file);

public List<Item> getSortedItem();

}

package com.foodbox.serviceImpl;

import java.util.List;

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

import org.springframework.stereotype.Service;

import com.foodbox.dto.AdminDto;

import com.foodbox.model.Admin;

import com.foodbox.repository.AdminRepository;

import com.foodbox.service.AdminService;

@Service

public class AdminServiceImpl implements AdminService{

@Autowired

private AdminRepository adminRepo;

@Override

public String addAdmin(AdminDto admindto) {

Admin admin = new Admin();

admin.setEmail(admindto.getEmail());

admin.setPassword(admindto.getPassword());

adminRepo.save(admin);

return "New Admin Record Saved";

}

@Override

public List<Admin> getAllAdmin() {

return adminRepo.findAll();

}

}

package com.foodbox.serviceImpl;

import java.util.List;

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

import org.springframework.stereotype.Service;

import com.foodbox.model.Cart;

import com.foodbox.repository.CartRepository;

import com.foodbox.service.CartService;

@Service

public class CartServiceImpl implements CartService{

@Autowired

private CartRepository cartRepo;

@Override

public Cart addToCart(Cart cart) {

return cartRepo.save(cart);

}

@Override

public List<Cart> findAllByCustomerCart(long id) {

return cartRepo.findAllByCustomerId(id);

}

@Override

public void deleteCart(long id) {

cartRepo.deleteById(id);

}

@Override

public void updateCart(Cart cart, long id) {

Cart updated = cartRepo.findById(id).get();

updated = cart;

cartRepo.save(updated);

}

}

package com.foodbox.serviceImpl;

import java.util.List;

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

import org.springframework.stereotype.Service;

import com.foodbox.dto.CustomerDto;

import com.foodbox.exception.ResourceNotFoundException;

import com.foodbox.model.Customer;

import com.foodbox.repository.CustomerRepository;

import com.foodbox.service.CustomerService;

@Service

public class CustomerServiceImpl implements CustomerService{

@Autowired

private CustomerRepository customerRepo;

@Override

public String addCustomer(CustomerDto customerdto) {

Customer customer = new Customer();

customer.setName(customerdto.getName());

customer.setEmail(customerdto.getEmail());

customer.setPassword(customerdto.getPassword());

customer.setLocation(customerdto.getLocation());

customer.setPhone(customerdto.getPhone());

customerRepo.save(customer);

return "New Customer Record Saved";

}

@Override

public List<Customer> getAllCustomer() {

return customerRepo.findAll();

}

@Override

public Customer getCustomerById(long id) {

return customerRepo.findById(id).orElseThrow(() -> new ResourceNotFoundException("Customer not exist with id :"+id));

}

@Override

public String updateCustomer(CustomerDto customerdto) {

Customer customerToUpdate = new Customer();

customerToUpdate.setName(customerdto.getName());

customerToUpdate.setEmail(customerdto.getEmail());

customerToUpdate.setPassword(customerdto.getPassword());

customerToUpdate.setLocation(customerdto.getLocation());

customerToUpdate.setPhone(customerdto.getPhone());

customerRepo.save(customerToUpdate);

return "Customer Details Record Updated";

}

@Override

public String deleteCustomerById(long id) {

customerRepo.deleteById(id);

return "Customer Records Deleted";

}

}

package com.foodbox.serviceImpl;

import java.util.List;

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

import org.springframework.data.domain.Sort;

import org.springframework.stereotype.Service;

import org.springframework.web.multipart.MultipartFile;

import com.foodbox.dto.ItemDto;

import com.foodbox.exception.ResourceNotFoundException;

import com.foodbox.helper.Helper;

import com.foodbox.model.Item;

import com.foodbox.repository.ItemRepository;

import com.foodbox.service.ItemService;

@Service

public class ItemServiceImpl implements ItemService{

@Autowired

private ItemRepository itemRepo;

@Override

public String addItem(ItemDto itemdto) {

Item item = new Item();

item.setName(itemdto.getName());

item.setUrl(itemdto.getUrl());

item.setCatagory(itemdto.getCatagory());

item.setPrice(itemdto.getPrice());

item.setQuantity(itemdto.getQuantity());

itemRepo.save(item);

return "New Item Record added";

}

@Override

public List<Item> getAllItems() {

return itemRepo.findAll();

}

@Override

public Item getItemById(long id) {

return itemRepo.findById(null).orElseThrow(() -> new ResourceNotFoundException("Item Not found: "+id));

}

@Override

public String updateItem(ItemDto itemdto) {

Item itemToUpdate = new Item();

itemToUpdate.setName(itemdto.getName());

itemToUpdate.setUrl(itemdto.getUrl());

itemToUpdate.setCatagory(itemdto.getCatagory());

itemToUpdate.setPrice(itemdto.getPrice());

itemToUpdate.setQuantity(itemdto.getQuantity());

itemRepo.save(itemToUpdate);

return "Record Updated";

}

@Override

public String deleteItemById(long id) {

itemRepo.deleteById(id);

return "Record Deleted";

}

@Override

public void saveMultiple(MultipartFile file) {

// TODO Auto-generated method stub

try {

List<Item> item= Helper.convertExcelToListOfProducts(file.getInputStream());

this.itemRepo.saveAll(item);

}

catch(Exception e) {

e.printStackTrace();

}

}

@Override

public List<Item> getSortedItem() {

// TODO Auto-generated method stub

return itemRepo.findAll(Sort.by("catagory"));

}

}

package com.foodbox.controller;

import java.util.List;

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

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

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.foodbox.dto.AdminDto;

import com.foodbox.model.Admin;

import com.foodbox.service.AdminService;

@RestController

@RequestMapping("/admin")

@CrossOrigin("http://localhost:4200/")

public class AdminController {

@Autowired

private AdminService adminService;

@PostMapping("/add-admin")

public String addAdmin(@RequestBody AdminDto dto) {

return adminService.addAdmin(dto);

}

@GetMapping("/getAll")

public List<Admin> getAlladmin() {

return adminService.getAllAdmin();

}

}

package com.foodbox.controller;

import java.util.List;

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

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

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.foodbox.dto.CustomerDto;

import com.foodbox.model.Customer;

import com.foodbox.service.CustomerService;

@RestController

@RequestMapping("/customer")

@CrossOrigin("http://localhost:4200/")

public class CustomerController {

@Autowired

private CustomerService customerService;

@PostMapping("/addCustomer")

public String addCustomer(@RequestBody CustomerDto dto) {

return customerService.addCustomer(dto);

}

@GetMapping("/getAll")

public List<Customer> getAllCustomer() {

return customerService.getAllCustomer();

}

@GetMapping("/getCustomer/{id}")

public Customer getCustomerById(@PathVariable long id) {

return customerService.getCustomerById(id);

}

@PutMapping("/updateCustomer")

public String updateCustomer(@RequestBody CustomerDto dto) {

return customerService.updateCustomer(dto);

}

@DeleteMapping("/deleteCustomer/{id}")

public String deleteCustomer(@PathVariable long id) {

return customerService.deleteCustomerById(id);

}

}

package com.foodbox.controller;

import java.util.List;

import java.util.Map;

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

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

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

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

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

import org.springframework.web.bind.annotation.RestController;

import org.springframework.web.multipart.MultipartFile;

import com.foodbox.dto.ItemDto;

import com.foodbox.helper.Helper;

import com.foodbox.model.Item;

import com.foodbox.service.ItemService;

@RestController

@RequestMapping("/item")

@CrossOrigin("http://localhost:4200/")

public class ItemController {

@Autowired

private ItemService itemService;

@PostMapping("/addItem")

public String addItem(@RequestBody ItemDto dto) {

return itemService.addItem(dto);

}

@GetMapping("/getAll")

public List<Item> getAllItems(){

return itemService.getAllItems();

}

@GetMapping("/getItem/{id}")

public Item getItemById(@PathVariable long id) {

return itemService.getItemById(id);

}

@PutMapping("/updateitem")

public String updateItem(@RequestBody ItemDto dto) {

return itemService.updateItem(dto);

}

@DeleteMapping("deleteItem/{id}")

public String deleteItem(@PathVariable long id) {

return itemService.deleteItemById(id);

}

@PostMapping("/add-bulk-items")

public ResponseEntity<?> upload(@RequestParam("file") MultipartFile file){

if(Helper.checkExelFormat(file))

{

this.itemService.saveMultiple(file);

return ResponseEntity.ok(Map.of("message","file is uploaded"));

}

return ResponseEntity.status(HttpStatus.BAD\_REQUEST).body("please upload excel file only");

}

@GetMapping("/get-sorted-item")

public List<Item> getSortedItem(){

return itemService.getSortedItem();

}

}

package com.foodbox.controller;

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

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

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

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.foodbox.model.Cart;

import com.foodbox.model.Customer;

import com.foodbox.model.Item;

import com.foodbox.service.CartService;

import com.foodbox.service.CustomerService;

import com.foodbox.service.ItemService;

@RestController

@RequestMapping("/cart")

@CrossOrigin("http://localhost:4200/")

public class CartController {

@Autowired

private CartService cartService;

@Autowired

private CustomerService customerService;

@Autowired

private ItemService itemService;

@GetMapping("/add/{itemId}/{customerId}")

public ResponseEntity<Cart> addCart(@PathVariable long itemId,@PathVariable long customerId) {

Item item = itemService.getItemById(itemId);

Customer customer = customerService.getCustomerById(customerId);

Cart cart = new Cart(customer, item);

cart.setUnit(1);

cart.setTotalprice(cart.getUnit()\*item.getPrice());

Cart addCart = cartService.addToCart(cart);

return new ResponseEntity<Cart>(addCart, HttpStatus.CREATED);

}

@DeleteMapping("/delete/{cartId}")

public ResponseEntity<?> deletCart(@PathVariable long cartId) {

cartService.deleteCart(cartId);

return new ResponseEntity<>(HttpStatus.OK);

}

@PutMapping("/update-cart/{cid}")

public void updateCart(@RequestBody Cart cart, @PathVariable long cid) {

cartService.updateCart(cart, cid);

}

}

package com.foodbox.exception;

import org.springframework.http.HttpStatus;

import org.springframework.web.bind.annotation.ResponseStatus;

@ResponseStatus(value = HttpStatus.NOT\_FOUND)

public class ResourceNotFoundException extends RuntimeException{

private static final long serialVersionUID = 1L;

public ResourceNotFoundException(String message) {

super(message);

}

}

package com.foodbox.helper;

import java.io.InputStream;

import java.util.ArrayList;

import java.util.Iterator;

import java.util.List;

import org.apache.poi.ss.usermodel.Cell;

import org.apache.poi.ss.usermodel.Row;

import org.apache.poi.xssf.usermodel.XSSFSheet;

import org.apache.poi.xssf.usermodel.XSSFWorkbook;

import org.springframework.web.multipart.MultipartFile;

import com.foodbox.model.Item;

public class Helper {

public static boolean checkExelFormat(MultipartFile file) {

String contentType = file.getContentType();

if(contentType=="application/vnd.openxmlformats-officedocument.spreadsheetml.sheet") {

return true;

}

else

return true;

}

//Converts excel to list of items

public static List<Item> convertExcelToListOfProducts(InputStream is){

List<Item> list = new ArrayList<Item>();

try {

@SuppressWarnings("resource")

XSSFWorkbook workbook= new XSSFWorkbook(is);

XSSFSheet sheet= workbook.getSheet("Sheet1");

int rowNumber=0;

Iterator<Row> iterator = sheet.iterator();

while(iterator.hasNext()) {

Row row = iterator.next();

if(rowNumber==0)

{

rowNumber++;

continue;

}

Iterator<Cell> cells = row.iterator();

int cid=0;

Item pdt =new Item();

while(cells.hasNext()) {

Cell cell = cells.next();

switch(cid)

{

case 0:

pdt.setName(cell.getStringCellValue());

break;

case 1:

pdt.setPrice(cell.getNumericCellValue());

break;

case 2:

pdt.setUrl(cell.getStringCellValue());

break;

case 3:

pdt.setQuantity((int)cell.getNumericCellValue());

break;

case 4:

pdt.setCatagory(cell.getStringCellValue());

break;

default :

break;

}

cid++;

}

list.add(pdt);

}

}catch(Exception e){

e.printStackTrace();

}

return list;

}

}spring.datasource.url=jdbc:mysql://localhost:3306/phase5

spring.datasource.username=root

spring.datasource.password=1234

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

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

spring.mvc.pathmatch.matching-strategy=ant-path-matcher

<?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>2.7.6</version>

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

</parent>

<groupId>com.foodbox</groupId>

<artifactId>Foodbox</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>Foodbox</name>

<description>Demo project for Spring Boot</description>

<properties>

<java.version>11</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-web</artifactId>

</dependency>

<dependency>

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

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

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<dependency>

<groupId>com.mysql</groupId>

<artifactId>mysql-connector-j</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>

</dependency>

<dependency>

<groupId>org.apache.poi</groupId>

<artifactId>poi-ooxml</artifactId>

<version>4.1.2</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

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

<configuration>

<excludes>

<exclude>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

</exclude>

</excludes>

</configuration>

</plugin>

</plugins>

</build>

</project>

<app-header></app-header>

<router-outlet></router-outlet>

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

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

import { AdminLoginComponent } from './admin-login/admin-login.component';

import { AdminPageComponent } from './admin-page/admin-page.component';

import { AdminComponent } from './admin/admin.component';

import { CartItemComponent } from './cart-item/cart-item.component';

import { CreateitemComponent } from './create-item/create-item.component';

import { DiscountComponent } from './discount/discount.component';

import { FileUploadComponent } from './file-upload/file-upload.component';

import { HomeComponent } from './home/home.component';

import { PaymentComponent } from './payment/payment.component';

import { PdpComponent } from './pdp/pdp.component';

import { AuthGuard } from './shared/auth.guard';

import { UserGuard } from './shared/user.guard';

import { UpdateitemComponent } from './update-item/update-item.component';

import { UpdateCustomerComponent } from './update-customer/update-customer.component';

import { CustomerLoginComponent } from './customer-login/customer-login.component';

import { CustomerPageComponent } from './customer-page/customer-page.component';

import { CustomerregisterComponent } from './customer-register/customer-register.component';

import { CustomersComponent } from './customers/customers.component';

const routes: Routes = [

  {path:"",component:HomeComponent},

  {path:"admin-page/itemList",component:PdpComponent,canActivate:[AuthGuard]},

  {path:"update-item/:id",component:UpdateitemComponent ,canActivate:[AuthGuard]},

  {path:"admin-page/itemList/create-item",component:CreateitemComponent,canActivate:[AuthGuard]},

  {path:"home",component:HomeComponent},

  {path:"admin-page/file-upload",component:FileUploadComponent ,canActivate:[AuthGuard]},

  {path:"customer-login/customer-register",component:CustomerregisterComponent},

  {path:"admin-page/userList",component:CustomersComponent ,canActivate:[AuthGuard]},

  {path:"admin-page/userList/create-customer",component:CustomerregisterComponent ,canActivate:[AuthGuard]},

  {path:"update-customer/:id",component:UpdateCustomerComponent ,canActivate:[AuthGuard]},

  {path:"customer-login",component:CustomerLoginComponent},

  {path:"admin-login",component:AdminLoginComponent},

  {path:"admin-page",component:AdminPageComponent,canActivate:[AuthGuard]},

  {path:"customer-page/:id",component:CustomerPageComponent, canActivate:[UserGuard]},

  {path:"bulk-upload",component:FileUploadComponent ,canActivate:[AuthGuard]},

  {path:"admin-page/reg-admin",component:AdminComponent, canActivate:[AuthGuard]},

  {path:"cart/:uid",component:CartItemComponent},

  {path:"discount",component:DiscountComponent,canActivate:[UserGuard]},

  {path:"payment/:total",component:PaymentComponent}

];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

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

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

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

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

import { HeaderComponent } from './header/header.component';

import { PdpComponent } from './pdp/pdp.component';

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

import { FormsModule } from '@angular/forms';

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

import { CustomersComponent } from './customers/customers.component';

import { CustomerregisterComponent } from './customer-register/customer-register.component';

import { UpdateitemComponent } from './update-item/update-item.component';

import { CreateitemComponent } from './create-item/create-item.component';

import { HomeComponent } from './home/home.component';

import { FileUploadComponent } from './file-upload/file-upload.component';

import { AdminComponent } from './admin/admin.component';

import { AdminLoginComponent } from './admin-login/admin-login.component';

import { CustomerLoginComponent } from './customer-login/customer-login.component';

import {ReactiveFormsModule} from '@angular/forms';

import { UpdateCustomerComponent } from './update-customer/update-customer.component';

import { CustomerPageComponent } from './customer-page/customer-page.component';

import { AdminPageComponent } from './admin-page/admin-page.component';

import { CartItemComponent } from './cart-item/cart-item.component';

import { DiscountComponent } from './discount/discount.component';

import { PaymentComponent } from './payment/payment.component';

@NgModule({

  declarations: [

    AppComponent,

    HeaderComponent,

    PdpComponent,

    CustomersComponent,

    CustomerregisterComponent,

    UpdateitemComponent,

    CreateitemComponent,

    HomeComponent,

    FileUploadComponent,

    AdminComponent,

    AdminLoginComponent,

    CustomerLoginComponent,

    UpdateCustomerComponent,

    AdminPageComponent,

    CartItemComponent,

    DiscountComponent,

    PaymentComponent

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    BrowserAnimationsModule,

    FormsModule,

    HttpClientModule,

    ReactiveFormsModule

  ],

  providers: [],

  bootstrap: [AppComponent]

})

export class AppModule { }

<nav class="navbar navbar-expand-lg navbar-dark bg-primary">

    <a class="navbar-brand" routerLink="home"><svg xmlns="http://www.w3.org/2000/svg" width="35" height="35" fill="currentColor" class="bi bi-shop" viewBox="0 0 16 16">

      <path d="M2.97 1.35A1 1 0 0 1 3.73 1h8.54a1 1 0 0 1 .76.35l2.609 3.044A1.5 1.5 0 0 1 16 5.37v.255a2.375 2.375 0 0 1-4.25 1.458A2.371 2.371 0 0 1 9.875 8 2.37 2.37 0 0 1 8 7.083 2.37 2.37 0 0 1 6.125 8a2.37 2.37 0 0 1-1.875-.917A2.375 2.375 0 0 1 0 5.625V5.37a1.5 1.5 0 0 1 .361-.976l2.61-3.045zm1.78 4.275a1.375 1.375 0 0 0 2.75 0 .5.5 0 0 1 1 0 1.375 1.375 0 0 0 2.75 0 .5.5 0 0 1 1 0 1.375 1.375 0 1 0 2.75 0V5.37a.5.5 0 0 0-.12-.325L12.27 2H3.73L1.12 5.045A.5.5 0 0 0 1 5.37v.255a1.375 1.375 0 0 0 2.75 0 .5.5 0 0 1 1 0zM1.5 8.5A.5.5 0 0 1 2 9v6h1v-5a1 1 0 0 1 1-1h3a1 1 0 0 1 1 1v5h6V9a.5.5 0 0 1 1 0v6h.5a.5.5 0 0 1 0 1H.5a.5.5 0 0 1 0-1H1V9a.5.5 0 0 1 .5-.5zM4 15h3v-5H4v5zm5-5a1 1 0 0 1 1-1h2a1 1 0 0 1 1 1v3a1 1 0 0 1-1 1h-2a1 1 0 0 1-1-1v-3zm3 0h-2v3h2v-3z"/>

    </svg> FoodBox</a>

    <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavAltMarkup" aria-controls="navbarNavAltMarkup" aria-expanded="false" aria-label="Toggle navigation">

      <span class="navbar-toggler-icon"></span>

    </button>

    <div class="collapse navbar-collapse" id="navbarNavAltMarkup">

      <div class="navbar-nav">

        <!-- <a class="nav-item nav-link active" routerLink="home">Home <span class="sr-only">(current)</span></a> -->

        <a style="margin-left: 5px; color:white;" class="nav-item nav-link" routerLink="customer-login"><svg xmlns="http://www.w3.org/2000/svg" width="22" height="22" fill="currentColor" class="bi bi-person" viewBox="0 0 16 16">

          <path d="M8 8a3 3 0 1 0 0-6 3 3 0 0 0 0 6zm2-3a2 2 0 1 1-4 0 2 2 0 0 1 4 0zm4 8c0 1-1 1-1 1H3s-1 0-1-1 1-4 6-4 6 3 6 4zm-1-.004c-.001-.246-.154-.986-.832-1.664C11.516 10.68 10.289 10 8 10c-2.29 0-3.516.68-4.168 1.332-.678.678-.83 1.418-.832 1.664h10z"/>

        </svg> customer</a>

        <a style="margin-left: 10px; color: white;" class="nav-item nav-link" routerLink="admin-login"><svg xmlns="http://www.w3.org/2000/svg" width="22" height="22" fill="currentColor" class="bi bi-gear" viewBox="0 0 16 16">

          <path d="M8 4.754a3.246 3.246 0 1 0 0 6.492 3.246 3.246 0 0 0 0-6.492zM5.754 8a2.246 2.246 0 1 1 4.492 0 2.246 2.246 0 0 1-4.492 0z"/>

          <path d="M9.796 1.343c-.527-1.79-3.065-1.79-3.592 0l-.094.319a.873.873 0 0 1-1.255.52l-.292-.16c-1.64-.892-3.433.902-2.54 2.541l.159.292a.873.873 0 0 1-.52 1.255l-.319.094c-1.79.527-1.79 3.065 0 3.592l.319.094a.873.873 0 0 1 .52 1.255l-.16.292c-.892 1.64.901 3.434 2.541 2.54l.292-.159a.873.873 0 0 1 1.255.52l.094.319c.527 1.79 3.065 1.79 3.592 0l.094-.319a.873.873 0 0 1 1.255-.52l.292.16c1.64.893 3.434-.902 2.54-2.541l-.159-.292a.873.873 0 0 1 .52-1.255l.319-.094c1.79-.527 1.79-3.065 0-3.592l-.319-.094a.873.873 0 0 1-.52-1.255l.16-.292c.893-1.64-.902-3.433-2.541-2.54l-.292.159a.873.873 0 0 1-1.255-.52l-.094-.319zm-2.633.283c.246-.835 1.428-.835 1.674 0l.094.319a1.873 1.873 0 0 0 2.693 1.115l.291-.16c.764-.415 1.6.42 1.184 1.185l-.159.292a1.873 1.873 0 0 0 1.116 2.692l.318.094c.835.246.835 1.428 0 1.674l-.319.094a1.873 1.873 0 0 0-1.115 2.693l.16.291c.415.764-.42 1.6-1.185 1.184l-.291-.159a1.873 1.873 0 0 0-2.693 1.116l-.094.318c-.246.835-1.428.835-1.674 0l-.094-.319a1.873 1.873 0 0 0-2.692-1.115l-.292.16c-.764.415-1.6-.42-1.184-1.185l.159-.291A1.873 1.873 0 0 0 1.945 8.93l-.319-.094c-.835-.246-.835-1.428 0-1.674l.319-.094A1.873 1.873 0 0 0 3.06 4.377l-.16-.292c-.415-.764.42-1.6 1.185-1.184l.292.159a1.873 1.873 0 0 0 2.692-1.115l.094-.319z"/>

        </svg> Admin</a>

      </div>

    </div>

  </nav>

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

@Component({

  selector: 'app-header',

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

  styleUrls: ['./header.component.css']

})

export class HeaderComponent implements OnInit {

  public searchItem:string='';

  constructor() { }

  ngOnInit(): void {

  }

  @Output()

  searchTextChanged : EventEmitter<string> =new  EventEmitter<string>();

  onSeachTextChanged(){

    this.searchTextChanged.emit(this.searchItem);

  }

}

<div class="wrapper" >

    <div style="align-self:center; text-align: center;" class="item" \*ngFor ="let item of items" >

        <img src="assets/images/{{item.url}}"><br><br>

        <div style="color:#556273;font-size: 1.5rem;

        margin-bottom: 3%; text-align: center;">

        {{item.name}}<br>

        </div>

        <div style="color:#ed2832; font-size: 1.25rem;">₹

        {{item.price}}.00

        </div>

            <br>

        <button style="margin-right:2%;background-color: #73665F;"  type="button" class="btn btn-primary" (click)="getData()">Add to Cart <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-cart" viewBox="0 0 16 16">

            <path d="M0 1.5A.5.5 0 0 1 .5 1H2a.5.5 0 0 1 .485.379L2.89 3H14.5a.5.5 0 0 1 .491.592l-1.5 8A.5.5 0 0 1 13 12H4a.5.5 0 0 1-.491-.408L2.01 3.607 1.61 2H.5a.5.5 0 0 1-.5-.5zM3.102 4l1.313 7h8.17l1.313-7H3.102zM5 12a2 2 0 1 0 0 4 2 2 0 0 0 0-4zm7 0a2 2 0 1 0 0 4 2 2 0 0 0 0-4zm-7 1a1 1 0 1 1 0 2 1 1 0 0 1 0-2zm7 0a1 1 0 1 1 0 2 1 1 0 0 1 0-2z"/>

          </svg></button>

        <button type="button"style="background-color:#F25270" class="btn btn-danger" (click)="getData()">Add to Wishlist <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-heart" viewBox="0 0 16 16">

            <path d="m8 2.748-.717-.737C5.6.281 2.514.878 1.4 3.053c-.523 1.023-.641 2.5.314 4.385.92 1.815 2.834 3.989 6.286 6.357 3.452-2.368 5.365-4.542 6.286-6.357.955-1.886.838-3.362.314-4.385C13.486.878 10.4.28 8.717 2.01L8 2.748zM8 15C-7.333 4.868 3.279-3.04 7.824 1.143c.06.055.119.112.176.171a3.12 3.12 0 0 1 .176-.17C12.72-3.042 23.333 4.867 8 15z"/>

          </svg></button>

        <!-- <button style="margin-right:20px"  mat-raised-button colour="primary" (click)="getData()">Add To cart</button> -->

        <!-- <button  mat-raised-button colour="primary" (click)="getData()">Add to Wishlist</button> -->

    </div>

</div>

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

import { Item } from '../item';

import { itemService } from '../item.service';

@Component({

  selector: 'app-home',

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

  styleUrls: ['./home.component.css']

})

export class HomeComponent implements OnInit {

  items: Item[]=[]

  searchText:any

  pid:any

  constructor(private pservice:itemService) { }

  ngOnInit(): void {

    this.pservice.getAllitems().subscribe(res=>{

      this.items=res

    })

  }

  getData(){

    alert("Login first")

  }

}

<div class="container" style="margin-top: 5%;;padding: 2%;">

  <h1 style="text-align: center; ">New Admin Registration</h1>

    <form #regForm="ngForm" (ngSubmit)="addAdmin(regForm.value)">

          <div class="form-group">

            <input type="email" class="form-control" id="inputEmail4" placeholder="Email" name="email" ngModel required>

            </div>

          <br>

          <div class="form-group">

              <input type="password" class="form-control" id="inputPassword4" placeholder="Password" name="password" ngModel required>

            </div>

            <br>

        <button style="text-align: center;font-size: 1.25rem; margin-left: 45%;" type="submit" class="btn btn-primary" >Register</button>

      </form>

</div>

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

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

import { Admin } from '../admin';

import { AdminService } from '../admin.service';

@Component({

  selector: 'app-admin',

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

  styleUrls: ['./admin.component.css']

})

export class AdminComponent implements OnInit {

  constructor(private aservice:AdminService, private route:Router) { }

  admin:Admin={

    id: 0,

    email: '',

    password: ''

  }

  ngOnInit(): void {

  }

  addAdmin(data:Admin){

    this.admin=data

    this.aservice.addAdmin(this.admin).subscribe(res=>{

     if(res) alert("admin added successfully")

    },(err)=>{

      alert("admin added successfully. Now Log in!!")

    })

  }

}

export interface Admin {

    id:number

    email:string

    password:string

}

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

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

import { Admin } from './admin';

@Injectable({

  providedIn: 'root'

})

export class AdminService {

  baseUrl:string="http://localhost:8080/admin"

  constructor(private http:HttpClient) { }

  addAdmin(adminData:Admin){

    console.log("service" +adminData)

     return    this.http.post(this.baseUrl+"/add-admin",adminData);

  }

getAllAdmin(){

  return this.http.get<Admin[]>(this.baseUrl+"/getAll");

}

}

<div class="container" style="margin-top: 5%;;padding: 2%;">

  <h1 style="text-align: center; ">Admin Login</h1>

<form  #loginForm="ngForm"  (ngSubmit)="adminLogin()">

    <div class="form-group">

      <!-- <label for="uname">CustomerName:</label> -->

      <input type="text" placeholder="Customername" class="form-control" name="uname"  [(ngModel)]="uname" required #name="ngModel" >

      <br><span \*ngIf="name.invalid && name.touched" style="color : red">Please Enter a valid input</span>

    </div>

    <div class="form-group">

      <!-- <label for="pwd">Password</label> -->

      <input type="password" class="form-control" name="pwd" [(ngModel)]="pwd" placeholder="Password" #pass="ngModel" required minlength="2">

      <br><span \*ngIf="pass.invalid && pass.touched" style="color : red">Please Enter a valid input</span>

    </div>

    <button style="text-align: center;font-size: 1.25rem; margin-left: 45%;" type="submit" class="btn btn-primary" [disabled]="loginForm.invalid">Login</button>

  </form>

</div>

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

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

import { Admin } from '../admin';

import { AdminService } from '../admin.service';

@Component({

  selector: 'app-admin-login',

  templateUrl: './admin-login.component.html',

  styleUrls: ['./admin-login.component.css']

})

export class AdminLoginComponent implements OnInit {

  constructor(private aservice:AdminService,private router:Router) { }

  uname:string=''

  pwd:string=''

  ngOnInit(): void {

  }

  adminLogin(){

    this.aservice.getAllAdmin().subscribe( adminList =>  {

      console.log(adminList);

        adminList.forEach( data => {

            console.log(data);

         console.log("uname "+this.uname+" password"+this.pwd)

      if(data.email == this.uname && data.password == this.pwd){

          alert("Login Successful!")

           console.log("login success")

           localStorage.setItem('token',"hgdgfjbvjh-dnjdbvjh-klkioubhawgf")

           this.uname=="admin@gmail.com" ? localStorage.setItem('userType','admin') : localStorage.setItem('userType','customer')

            this.router.navigate(['admin-page']);

         }

        else{

            console.log("Invalid Credentials, Enter valid data")

        }

      });

});

}

}

<div class="container">

  <button type="button" class="btn btn-warning"routerLink="reg-admin">Create New Admin</button>

  <button type="button" class="btn btn-info"routerLink="userList">Manage customers</button>

  <button type="button" class="btn btn-primary"routerLink="itemList">Manage items</button>

  <button type="button" class="btn btn-danger"(click)=adminLogout() routerLink="">Logout</button>

</div>

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

@Component({

  selector: 'app-admin-page',

  templateUrl: './admin-page.component.html',

  styleUrls: ['./admin-page.component.css']

})

export class AdminPageComponent implements OnInit {

  constructor() { }

  ngOnInit(): void {

  }

  adminLogout(){

    localStorage.setItem('token','')

  }

}

export interface Item {

    id:number

    name:string

    price:number

    url:string

    quantity:number

    catagory:string

}

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

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

import { Item } from './item';

@Injectable({

  providedIn: 'root'

})

export class itemService {

  baseUrl:string="http://localhost:8080/item"

  constructor(private http:HttpClient) { }

  getAllitems(){

    return this.http.get<Item[]>(this.baseUrl+"/getAll");

  }

  additem(itemData:Item){

    console.log("service" +itemData)

     return    this.http.post(this.baseUrl+"/add-item",itemData);

  }

  updateitem(item:Item){

    return this.http.put(this.baseUrl+"/updateitem",item);

  }

  getitemById(id:any){

    return this.http.get<any>(this.baseUrl+"/get-item/"+id)

  }

  deleteitemById(id:number){

    return this.http.delete(this.baseUrl+"/deleteitem/"+id)

  }

}

export interface Customer {

    id:number;

    name:String;

    email:String;

    password:String;

    location:String;

    phone:number;

}

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

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

import { Customer } from './customer';

@Injectable({

  providedIn: 'root'

})

export class CustomerService {

  baseUrl:string="http://localhost:8080/customer"

  constructor(private http:HttpClient) { }

  addCustomer(userData:Customer){

    console.log("service" +userData)

     return    this.http.post(this.baseUrl+"/add-customer",userData);

  }

  getAllCustomer(){

    return this.http.get<Customer[]>(this.baseUrl+"/getAll");

  }

   updateCustomer(customer:Customer) {

    return this.http.put<Customer>(this.baseUrl+"/updateCustomer",customer);

   }

   deleteCustomer(id:number){

    return this.http.delete(this.baseUrl+"/deleteCustomer/"+id)

   }

   getCustomerById(id:number){

    return this.http.get<any>(this.baseUrl+"/get-customer/"+id)

  }

}

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

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

@Injectable({

  providedIn: 'root'

})

export class CartService {

  constructor(private http:HttpClient) { }

  baseUrl:string="http://localhost:8080/cart"

  getCart(uid:any){

    return this.http.get<any>(this.baseUrl+"/getall/"+uid);

  }

  addToCart(pid:any,uid:any){

    return this.http.get(this.baseUrl+"/add/"+pid+"/"+uid)

  }

  updateCart(cart:any, cid:any){

    return this.http.put<any>(this.baseUrl+"/update-cart/"+cid,cart)

  }

  deleteCart(cid:any){

    return this.http.delete<any>(this.baseUrl+"/delete/"+cid)

  }

}

import { TestBed } from '@angular/core/testing';

import { CartService } from './cart.service';

describe('CartService', () => {

  let service: CartService;

  beforeEach(() => {

    TestBed.configureTestingModule({});

    service = TestBed.inject(CartService);

  });

  it('should be created', () => {

    expect(service).toBeTruthy();

  });

});

<html lang="en">

  <body>

    <div class="container">

      <div class="main-content">

 <H2  class="alert alert-success">  Pay Bill Amount : {{billAmount | currency:'INR'}} </H2>

        <h2 class="alert alert-primary"> Pay with Netbanking </h2>

      </div>

      <div class="card-details">

        <p>Pay using Credit or Debit card</p>

  <form   #myform="ngForm"  class="conatainer col-md-6" onsubmit="alert('Payment Success , Thank you')" >

        <div class="form-group">

          <label> Card Number </label>

          <input class="form-control"

            type="text"  required  name="cno" #cno="ngModel" ngModel

            placeholder="###-###-###"/>

            <span style="color:red" \*ngIf="cno.touched && cno.invalid">Card No is Mandatory </span>

        </div>

        <br />

        <div class="form-group">

          <label> Expiry Date </label>

          <input type="date"   class="form-control" name="date"  required #date="ngModel" ngModel

                 placeholder="DD-MM-YY" />

                 <span style="color:red" \*ngIf="date.touched && date.invalid">Date is Mandatory </span>

        </div>

        <div class="form-group">

          <label> CVV number </label>

          <input type="password" class="form-control" name="cvv"  required #cvv="ngModel" ngModel

                 placeholder="xxx" />

                 <span style="color:red" \*ngIf="cvv.touched && cvv.invalid">CVV No is Mandatory </span>

        </div>

        <div class="form-group">

          <label> Card Holder name </label>

          <input

            type="text"

            class="form-control" required name="name"  #name="ngModel" ngModel

            placeholder="Enter your Name"/>

            <span style="color:red" \*ngIf="name.touched && name.invalid">Card Holder Name is Mandatory </span>

        </div>

        <button type="submit"  [disabled]="myform.invalid"

      class="btn btn-primary"  >Pay</button>

      </form>

      </div>

    </div>

  </body>

</html>

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

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

@Component({

  selector: 'app-payment',

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

  styleUrls: ['./payment.component.css']

})

export class PaymentComponent implements OnInit {

  billAmount:number = 0;

  constructor(private route:ActivatedRoute) {

  }

  ngOnInit(): void {

      this.route.params.subscribe( (param:any) => {this.billAmount = param['total']} );

  }

}

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

import { ActivatedRouteSnapshot, CanActivate, Router, RouterStateSnapshot, UrlTree } from '@angular/router';

import { Observable } from 'rxjs';

import { AuthService } from './auth.service';

@Injectable({

  providedIn: 'root'

})

export class AuthGuard implements CanActivate {

  constructor(private auth : AuthService,private router:Router){}

  canActivate() {

 if(this.auth.isAdminLoggedIn()){

    return true;

 }

 alert("Please Sign in as Admin")

    this.router.navigate(['admin-login'])

    return false;

  }

}

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

@Injectable({

  providedIn: 'root'

})

export class AuthService {

  constructor() { }

  isAdminLoggedIn(){

    return !!localStorage.getItem('token');

  }

  isUserLoggedIn(){

    return !!localStorage.getItem('token');

  }

}

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

import { ActivatedRouteSnapshot, CanActivate, RouterStateSnapshot, UrlTree } from '@angular/router';

import { Observable } from 'rxjs';

@Injectable({

  providedIn: 'root'

})

export class RoleGuard implements CanActivate {

  canActivate() {

    let role=localStorage.getItem('userType')

    if(role=="admin"){

    return true;

    }

      alert("You dont have permission")

    return false

  }

}

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

import { ActivatedRouteSnapshot, CanActivate, Router, RouterStateSnapshot, UrlTree } from '@angular/router';

import { Observable } from 'rxjs';

import { AuthService } from './auth.service';

@Injectable({

  providedIn: 'root'

})

export class UserGuard implements CanActivate {

  constructor(private auth : AuthService,private router:Router){}

  canActivate() {

    if(this.auth.isUserLoggedIn()){

      return true;

   }

   alert("Please Sign in as User")

      this.router.navigate(['user-login'])

      return false;

  }

}