CAPSTONE PROJECT - 2

Drive Your Way

DESCRIPTION

Design and develop an online car selling and buying portal.

Scenario:

Drive Your Way Ltd. is a company working in the business of selling and buying old cars. However, due to the pandemic and lockdown, their business took a hit. They were not able to achieve the decided targets. So, they have decided to go online to increase the revenue.

Code Snippets:

BACKEND

application.properties

```
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/capstone
spring.datasource.username=root
spring.datasource.password=root
spring.jpa.hibernate.ddl-auto=update
server.port=9090
```

MyApplication.java

```
package com;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.boot.autoconfigure.domain.EntityScan;

import org.springframework.data.jpa.repository.config.EnableJpaRepositories;


@SpringBootApplication(scanBasePackages = "com")

@EntityScan(basePackages = "com.onlineshop.bean")

@EnableJpaRepositories(basePackages = "com.onlineshop.repository")

public class MyAppApplication {

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

```
                SpringApplication.run(MyAppApplication.class, args);
                System.out.println("Server running on port number 9090");
        }
}
```

Login.java

```
package com.onlineshop.bean;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
@Entity
public class Login {
@Id
private String emailid;
private String password;
@Column(name = "typeofuser")
private String typeOfUser;
public String getEmailid() {
        return emailid;
}
public void setEmailid(String emailid) {
        this.emailid = emailid;
}
public String getPassword() {
        return password;
}
public void setPassword(String password) {
```

```
        this.password = password;

}

public String getTypeOfUser() {

        return typeOfUser;

}

public void setTypeOfUser(String typeOfUser) {

        this.typeOfUser = typeOfUser;

}

@Override

public String toString() {

        return "Login [emailid=" + emailid + ", password=" + password + ", typeOfUser=" +
typeOfUser + "]";

}

}
```

Product.java

```
package com.onlineshop.bean;

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

@Entity
public class Product {
@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)          //
auto_generate
private int pid;
private String pname;
private float price;
private String url;
public int getPid() {
      return pid;
}
public void setPid(int pid) {
      this.pid = pid;
}
public String getPname() {
      return pname;
}
```

```
public void setPname(String pname) {
	this.pname = pname;
}
public float getPrice() {
	return price;
}
public void setPrice(float price) {
	this.price = price;
}
public String getUrl() {
	return url;
}
public void setUrl(String url) {
	this.url = url;
}
@Override
public String toString() {
	return "Product [pid=" + pid + ", pname=" + pname + ", price=" +
price + ", url=" + url + "]";
}
```

}

LoginConroller.java

```
package com.onlineshop.controller;

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

import org.springframework.http.MediaType;

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

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.onlineshop.bean.Login;

import com.onlineshop.service.LoginService;

@RestController

@RequestMapping("login")

@CrossOrigin

public class LoginController {

	@Autowired

	LoginService loginService;
```

```
	@PostMapping(value = "signIn",consumes = MediaType.APPLICATION_JSON_VALUE)
	public String signIn(@RequestBody Login login) {
		System.out.println("Process executed");
		return loginService.signIn(login);
	}
	@PostMapping(value = "signUp",consumes = MediaType.APPLICATION_JSON_VALUE)
	public String signUp(@RequestBody Login login) {
		System.out.println(login);
		return loginService.signUp(login);
	}
}
```

ProductController.java

```
package com.onlineshop.controller;
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.MediaType;
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.PatchMapping;
import org.springframework.web.bind.annotation.PathVariable;
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.onlineshop.bean.Product;

import com.onlineshop.service.ProductService;

@RestController

@RequestMapping("product")

@CrossOrigin

public class ProductController {

	@Autowired

	ProductService productService;

	@PostMapping(value = "storeProduct",consumes = MediaType.APPLICATION_JSON_VALUE)

	public String storeProduct(@RequestBody Product product) {

		return productService.storeProduct(product);

	}

	@PatchMapping(value = "updateProduct",consumes = MediaType.APPLICATION_JSON_VALUE)

	public String updateProduct(@RequestBody Product product) {

		return productService.updateProduct(product);

	}

	@GetMapping(value="findAllProduct",produces = MediaType.APPLICATION_JSON_VALUE)

	public List<Product> getAllProduct() {

		return productService.getAllProducts();

	}

	@GetMapping(value="findProductByPrice/{price}",produces = MediaType.APPLICATION_JSON_VALUE)

	public List<Product> findProductByPrice(@PathVariable("price") float price) {

		return productService.findProductByPrice(price);
```

```
	}

	@GetMapping(value="findAllProduct/{pid}")
	public String findProductById(@PathVariable("pid") int pid) {
		return productService.findProductById(pid);
	}
	@DeleteMapping(value="deleteProduct/{pid}")
	public String deleteProductUsingId(@PathVariable("pid") int pid) {
		return productService.deleteProduct(pid);
	}
}
```

LoginRepository.java

```
package com.onlineshop.repository;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;
import com.onlineshop.bean.Login;
@Repository
public interface LoginRepository extends JpaRepository<Login, String>{
}
```

ProductRepository.java

```
package com.onlineshop.repository;
import java.util.List;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Repository;
```

```
import com.onlineshop.bean.Product;


@Repository
public interface ProductRepository extends JpaRepository<Product, Integer>{
	//JPQL
	@Query("select p from Product p where p.price > :price")
	public List<Product> findProductByPrice(@Param("price") float price);
}
```

LoginService.java

```
package com.onlineshop.service;
import java.util.Optional;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.onlineshop.bean.Login;
import com.onlineshop.repository.LoginRepository;
@Service
public class LoginService {
	@Autowired
	LoginRepository loginRepository;
	public String signIn(Login login) {
		Optional<Login> result = loginRepository.findById(login.getEmailid());
		if(result.isPresent()) {
			Login ll = result.get();
			if(ll.getPassword().equals(login.getPassword())) {
				if(login.getTypeOfUser().equals(ll.getTypeOfUser()) && login.getTypeOfUser().equals("admin")) {
```

```
                                        return "Admin sucessfully login";
                                }else   if(login.getTypeOfUser().equals(ll.getTypeOfUser())   &&
login.getTypeOfUser().equals("user")){
                                        return "User successfully login";
                                }else {
                                        return "Invalid details";
                                }
                        }else {
                                return "InValid password";
                        }
                }else {
                        return "InValid emailId";
                }
        }
        public String signUp(Login login) {
                Optional<Login> result = loginRepository.findById(login.getEmailid());
                if(result.isPresent()) {
                                return "Email Id already exists";
                }else {
                        if(login.getTypeOfUser().equals("admin")) {
                                return "You can't create admin account";
                        }else {
                                loginRepository.save(login);
                                return "Account created successfully";
                        }

                }
```

```
	}
}
```

ProductService.java

```
package com.onlineshop.service;

import java.util.List;

import java.util.Optional;


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

import org.springframework.stereotype.Service;


import com.onlineshop.bean.Product;

import com.onlineshop.repository.ProductRepository;


@Service

public class ProductService {


	@Autowired

	ProductRepository productRepository;


	public String storeProduct(Product product) {

		productRepository.save(product);

		return "Product details stored";

	}


	public List<Product> getAllProducts() {

		return productRepository.findAll();
```

```
}

public String findProductById(int pid) {
	Optional<Product> result  = productRepository.findById(pid);
	if(result.isPresent()) {
		Product p = result.get();
		return p.toString();
	}else {
		return "Product not present";
	}
}

public List<Product> findProductByPrice(float price){
	return productRepository.findProductByPrice(price);
}

public String deleteProduct(int pid) {
	Optional<Product> result  = productRepository.findById(pid);
	if(result.isPresent()) {
		Product p = result.get();
		productRepository.delete(p);
		return "Product deleted successfully";
	}else {
		return "Product not present";
	}
}
```

```
        public String updateProduct(Product product) {
                Optional<Product> result  = productRepository.findById(product.getPid());
                if(result.isPresent()) {
                        Product p = result.get();
                        p.setPrice(product.getPrice());
                        p.setUrl(product.getUrl());
                        productRepository.saveAndFlush(p);
                        return "Product updated successfully";
                }else {
                        return "Product not present";
                }
        }
}
```

FRONTEND

admindashboard.component.spec.ts

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

import { AdmindashboardComponent } from './admindashboard.component';

describe('AdmindashboardComponent', () => {
  let component: AdmindashboardComponent;
  let fixture: ComponentFixture<AdmindashboardComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ AdmindashboardComponent ]
    })
    .compileComponents();

    fixture = TestBed.createComponent(AdmindashboardComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
```

```
    expect(component).toBeTruthy();
  });
});
```

admindashboard.component.ts

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

@Component({
  selector: 'app-admindashboard',
  templateUrl: './admindashboard.component.html',
  styleUrls: ['./admindashboard.component.css']
})
export class AdmindashboardComponent implements OnInit {
  user:string ="";
  constructor(private router:Router) { }

  ngOnInit(): void {
    let obj = sessionStorage.getItem("userDetails");
    if(obj!=null){
      this.user=obj;
    }
  }


  logout() {
    sessionStorage.removeItem("userDetails");
    this.router.navigate(["login"]);
  }
}
```

userdashboard.component.specs.ts

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

import { UserdashboardComponent } from './userdashboard.component';

describe('UserdashboardComponent', () => {
  let component: UserdashboardComponent;
  let fixture: ComponentFixture<UserdashboardComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ UserdashboardComponent ]
```

```
    })
    .compileComponents();

    fixture = TestBed.createComponent(UserdashboardComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

userdashboard.component.ts

```
import { Component, OnInit } from '@angular/core';
import { Router } from '@angular/router';
import { Product } from '../product';
import { ProductService } from '../product.service';

@Component({
  selector: 'app-userdashboard',
  templateUrl: './userdashboard.component.html',
  styleUrls: ['./userdashboard.component.css']
})
export class UserdashboardComponent implements OnInit {
  flagos:boolean = false;

  user:string ="";
  products:Array<Product>=[];
  constructor(public router:Router,public ps:ProductService) { }

  ngOnInit(): void {
    this.findAllProduct();
    let obj = sessionStorage.getItem("userDetails");
    if(obj!=null){
      this.user=obj;
    }
  }


  logout() {
    sessionStorage.removeItem("userDetails");
    this.router.navigate(["login"]);
  }
```

```
flago(){
 this.flagos=true;

}

 flag:boolean = false;
 pid:number =0;
 price:number =0;
 url:string ="";
 findAllProduct() {
   this.ps.findAllProduct().subscribe({
     next:(result:any)=>this.products=result,
     error:(error:any)=>console.log(error),
     complete:()=>console.log("completed")
   })
 }

 deleteProduct(pid:number){
   //console.log(pid)
   this.ps.deleteProductById(pid).subscribe({
     next:(result:any)=>console.log(result),
     error:(error:any)=>console.log(error),
     complete:()=>{
         this.findAllProduct();
     }
   })
 }

 updateProduct(product:any){
     this.flag= true;
     this.pid=product.pid;
     this.price=product.price;
     this.url=product.url;
 }

 updateDataFromDb(){
   let product = {pid:this.pid,price:this.price,url:this.url};
   this.ps.updateProduct(product).subscribe({
     next:(result:any)=>console.log(result),
     error:(error:any)=>console.log(error),
     complete:()=>{
         this.findAllProduct();
     }
   })
   this.flag=false;
 }
}
```