**SOURCE CODE FOR ECOMMERCE SITE OF SORTY SHOES**

1. **src/main/java**
   1. **SportyshoesApplication.java**

**package** com.ecommerce.sportyshoes;

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

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

**import** springfox.documentation.swagger2.annotations.EnableSwagger2;

@SpringBootApplication

@EnableSwagger2

**public** **class** SportyshoesApplication {

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

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

}

}

* 1. **Controller**

**1.2.1 ExceptionController.java**

**package** com.ecommerce.sportyshoes.controller;

**import** org.springframework.http.HttpStatus;

**import** org.springframework.http.ResponseEntity;

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

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

**import** com.ecommerce.sportyshoes.exceptions.BusinessException;

@ControllerAdvice

**public** **class** ExceptionController {

@ExceptionHandler(value = BusinessException.**class**)

**public** ResponseEntity<Object> exception(BusinessException exception){

**return** **new** ResponseEntity<>(exception.getMessage(), HttpStatus.***EXPECTATION\_FAILED***);

}

}

**1.2.2 ProductController.java**

**package** com.ecommerce.sportyshoes.controller;

**import** java.util.List;

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

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

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

**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.ecommerce.sportyshoes.exceptions.BusinessException;

**import** com.ecommerce.sportyshoes.model.Product;

**import** com.ecommerce.sportyshoes.service.iface.ProductServiceIface;

@RestController

@RequestMapping("/product")

**public** **class** ProductController {

@Autowired

ProductServiceIface productServiceIface;

@PutMapping("/update")

**public** **int** updateProduct(@RequestBody Product prodDataInput) **throws** BusinessException {

**return** productServiceIface.updateProduct(prodDataInput);

}

@PostMapping("/create")

**public** **int** createProduct(@RequestBody Product prodDataInput) **throws** BusinessException {

**return** productServiceIface.createProduct(prodDataInput);

}

@DeleteMapping("/delete")

**public** **int** deleteProduct(@RequestBody Product prodDataInput) **throws** BusinessException {

**return** productServiceIface.deleteProduct(prodDataInput);

}

@GetMapping("/getByCategory")

**public** List<Product> getByCategory(@RequestBody Product prodDataInput) **throws** BusinessException{

**return** productServiceIface.getProdByCategory(prodDataInput);

}

}

**1.2.3 PurchaseController.java**

**package** com.ecommerce.sportyshoes.controller;

**import** java.util.List;

**import** java.util.Optional;

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

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

**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.ecommerce.sportyshoes.exceptions.BusinessException;

**import** com.ecommerce.sportyshoes.model.Purchase;

**import** com.ecommerce.sportyshoes.service.iface.PurchaseServiceIface;

@RestController

@RequestMapping ("/purchase")

**public** **class** PurchaseController {

@Autowired

PurchaseServiceIface purchaseServiceIface;

@PutMapping("/createsale")

**public** **int** createSale(@RequestBody Purchase purchaseDataInput) **throws** BusinessException

{

**return** purchaseServiceIface.createSale(purchaseDataInput);

}

@PutMapping("/deletesale")

**public** **int** deleteSale(@RequestBody Purchase purchaseDataInput) **throws** BusinessException

{

**return** purchaseServiceIface.deleteSale(purchaseDataInput);

}

@GetMapping("/getById")

**public** Optional<Purchase> getById(@RequestBody Purchase purchaseDataInput) **throws** BusinessException{

**return** purchaseServiceIface.getPurchaseById(purchaseDataInput);

}

@GetMapping("/getByDateCategory")

**public** List<Purchase> getByDateCategory(@RequestBody Purchase purchaseDataInput) **throws** BusinessException{

**return** purchaseServiceIface.getByDateCategory(purchaseDataInput);

}

}

**1.2.4 UserController.java**

**package** com.ecommerce.sportyshoes.controller;

**import** java.util.List;

**import** java.util.Optional;

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

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

**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.ecommerce.sportyshoes.exceptions.BusinessException;

**import** com.ecommerce.sportyshoes.model.User;

**import** com.ecommerce.sportyshoes.service.iface.UserServiceIface;

@RestController

@RequestMapping("/user")

**public** **class** UserController {

@Autowired

UserServiceIface userServiceIface;

@PutMapping(value = "updateUser")

**public** **int** updateUser(@RequestBody User userDataInput) **throws** BusinessException {

**return** userServiceIface.updateUser(userDataInput);

}

@GetMapping(value = "getAllUsers")

**public** List<User> getAllUsers() **throws** BusinessException{

**return** userServiceIface.getAllUsers();

}

@GetMapping(value = "/getUserByEmail")

**public** Optional<User> getUserByEmail(@RequestBody User userDataInput) **throws** BusinessException{

**return** userServiceIface.getUserByEmail(userDataInput);

}

@PutMapping(value = "createUser")

**public** **int** createUser(@RequestBody User userDataInput) **throws** BusinessException {

**return** userServiceIface.createUser(userDataInput);

}

@PutMapping(value = "deleteUser")

**public** **int** deleteUser(@RequestBody User userDataInput) **throws** BusinessException {

**return** userServiceIface.deleteUser(userDataInput);

}

}

* 1. **Enums**
     1. **Category.java**

**package** com.ecommerce.sportyshoes.enums;

**public** **enum** Category {

***SPORTS***,***FORMAL***,***KIDS***;

**public** **static** Category getCategory(String categoryString) {

**return** Category.*valueOf*(categoryString);

}

}

* + 1. **UserType.java**

**package** com.ecommerce.sportyshoes.enums;

**public** **enum** UserType {

***ADMIN***,***NORMAL***;

**public** **static** UserType getUserType(String userTypeString) {

**return** UserType.*valueOf*(userTypeString);

}

}

* 1. **Exception**
     1. **BusinessException.java**

**package** com.ecommerce.sportyshoes.exceptions;

**public** **class** BusinessException **extends** Exception{

**private** String message;

**public** BusinessException(String message) {

**this**.message=message;

}

@Override

**public** String getMessage() {

**return** message;

}

/\*\*

\*

\*/

**private** **static** **final** **long** ***serialVersionUID*** = -851150852212525045L;

}

* 1. **Model**
     1. **Product.java**

**package** com.ecommerce.sportyshoes.model;

**import** java.util.Date;

**import** org.hibernate.annotations.DynamicInsert;

**import** org.hibernate.annotations.DynamicUpdate;

**import** com.ecommerce.sportyshoes.enums.Category;

**import** lombok.AllArgsConstructor;

**import** lombok.Data;

**import** lombok.NoArgsConstructor;

**import** javax.persistence.\*;

@Data

@Entity

@Table(name = "products")

@DynamicUpdate

@DynamicInsert

@AllArgsConstructor

@NoArgsConstructor

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

@Id

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

**private** **long** id;

**private** String name;

**private** String description;

**private** **long** price;

**private** **long** availability;

@Enumerated(EnumType.***STRING***)

**private** Category category;

**private** Date createdAt;

**private** Date updatedAt;

}

* + 1. **Purchase.java**

**package** com.ecommerce.sportyshoes.model;

**import** java.util.Date;

**import** javax.persistence.Entity;

**import** javax.persistence.EnumType;

**import** javax.persistence.Enumerated;

**import** javax.persistence.GeneratedValue;

**import** javax.persistence.GenerationType;

**import** javax.persistence.Id;

**import** javax.persistence.Table;

**import** org.hibernate.annotations.DynamicInsert;

**import** org.hibernate.annotations.DynamicUpdate;

**import** com.ecommerce.sportyshoes.enums.Category;

**import** lombok.AllArgsConstructor;

**import** lombok.Data;

**import** lombok.Getter;

**import** lombok.NoArgsConstructor;

**import** lombok.Setter;

@Data

@Entity

@Table(name = "purchase")

@DynamicUpdate

@DynamicInsert

@AllArgsConstructor

@NoArgsConstructor

**public** **class** Purchase {

@Id

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

**private** **long** id;

**private** **long** user\_id;

**private** **long** product\_id;

@Enumerated(EnumType.***STRING***)

**private** Category category;

**private** **int** quantity;

**private** Date createdAt;

**private** Date updatedAt;

}

* + 1. **User.java**

**package** com.ecommerce.sportyshoes.model;

**import** java.util.Date;

**import** javax.persistence.Entity;

**import** javax.persistence.EnumType;

**import** javax.persistence.Enumerated;

**import** javax.persistence.GeneratedValue;

**import** javax.persistence.GenerationType;

**import** javax.persistence.Id;

**import** javax.persistence.Table;

**import** org.hibernate.annotations.DynamicInsert;

**import** org.hibernate.annotations.DynamicUpdate;

**import** com.ecommerce.sportyshoes.enums.Category;

**import** com.ecommerce.sportyshoes.enums.UserType;

**import** lombok.AllArgsConstructor;

**import** lombok.Data;

**import** lombok.Getter;

**import** lombok.NoArgsConstructor;

**import** lombok.Setter;

@Data

@Entity

@Table(name = "users")

@DynamicUpdate

@DynamicInsert

@AllArgsConstructor

@NoArgsConstructor

**public** **class** User {

@Id

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

**private** Long iD;

**private** String firstName;

**private** String lastName;

**private** String email;

**private** String mobile;

**private** String password;

@Enumerated(EnumType.***STRING***)

**private** UserType userType;

**private** Date createdAt;

**private** Date updatedAt;

}

* 1. **Repository**
     1. **ProductRepository.java**

**package** com.ecommerce.sportyshoes.repository;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**import** java.util.List;

**import** java.util.Optional;

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

**import** org.springframework.jdbc.core.BeanPropertyRowMapper;

**import** org.springframework.jdbc.core.JdbcTemplate;

**import** org.springframework.jdbc.core.RowMapper;

**import** org.springframework.stereotype.Repository;

**import** com.ecommerce.sportyshoes.enums.Category;

**import** com.ecommerce.sportyshoes.exceptions.BusinessException;

**import** com.ecommerce.sportyshoes.model.Product;

@Repository

**public** **class** ProductRepository {

@Autowired

JdbcTemplate jdbcTemplate;

**class** ProductRowMapper **implements** RowMapper<Product> {

@Override

**public** Product mapRow(ResultSet rs, **int** rowNum) **throws** SQLException {

Product product=**new** Product();

product.setId(rs.getLong("id"));

product.setName(rs.getString("name"));

product.setDescription(rs.getString("description"));

product.setAvailability(rs.getInt("availability"));

product.setPrice(rs.getLong("price"));

product.setCategory(Category.*getCategory*(rs.getString("category")));

product.setCreatedAt(rs.getDate("created\_at"));

product.setUpdatedAt(rs.getDate("updated\_at"));

**return** product;

}

}

**public** List<Product> findByCategory(String category) **throws** BusinessException{

**try** {

**return** jdbcTemplate.query("select \* from products where category=?", **new** Object[] {category},**new** ProductRowMapper());}

**catch** (Exception exception) {

**throw** **new** BusinessException("Internal Error Occured "+exception.getMessage());

}

}

**public** Optional<Product> findById(Long id) **throws** BusinessException{

**try** {

**return** Optional.*of*(jdbcTemplate.queryForObject("select \* from products where id=?",

**new** Object[] {id}, **new** BeanPropertyRowMapper<Product>(Product.**class**)));}

**catch** (Exception exception) {

**throw** **new** BusinessException("Internal Error Occured "+exception.getMessage());

}

}

**public** **int** createProduct(Product product) **throws** BusinessException {

**try** {

**return** jdbcTemplate.update("insert into products( name,description, availability,price, category) values(?,?,?,?,?)",

**new** Object[] {product.getName(), product.getDescription(),product.getAvailability(),product.getPrice(),

product.getCategory().toString()});

}

**catch** (Exception exception) {

**throw** **new** BusinessException("Internal Error Occured "+exception.getMessage());

}

}

**public** **int** deleteProduct(Long id) **throws** BusinessException {

**try** {

**return** jdbcTemplate.update("delete from products where id=?", **new** Object[] {id});}

**catch** (Exception exception) {

**throw** **new** BusinessException("Internal Error Occured "+exception.getMessage());

}

}

**public** **int** updateProduct(Product product) **throws** BusinessException {

**try** {

**return** jdbcTemplate.update("update products "

+ "set availability=ifnull(?,availability), "

+ "price=ifnull(?,price), "

+ " where id=?",

**new** Object[] {product.getAvailability(),product.getPrice(),product.getId()});

}

**catch** (Exception exception) {

**throw** **new** BusinessException("Internal Error Occured "+exception.getMessage());

}

}

}

* + 1. **PurchaseRepository.java**

**package** com.ecommerce.sportyshoes.repository;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**import** java.util.List;

**import** java.util.Optional;

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

**import** org.springframework.jdbc.core.BeanPropertyRowMapper;

**import** org.springframework.jdbc.core.JdbcTemplate;

**import** org.springframework.jdbc.core.RowMapper;

**import** org.springframework.stereotype.Repository;

**import** com.ecommerce.sportyshoes.enums.Category;

**import** com.ecommerce.sportyshoes.exceptions.BusinessException;

**import** com.ecommerce.sportyshoes.model.Purchase;

@Repository

**public** **class** PurchaseRepository {

@Autowired

JdbcTemplate jdbcTemplate;

**class** PurchaseRowMapper **implements** RowMapper<Purchase> {

@Override

**public** Purchase mapRow(ResultSet rs, **int** rowNum) **throws** SQLException {

Purchase purchase=**new** Purchase();

purchase.setId(rs.getLong("id"));

purchase.setUser\_id(rs.getLong("user\_id"));

purchase.setProduct\_id(rs.getLong("product\_id"));

purchase.setQuantity(rs.getInt("quantity"));

purchase.setCategory(Category.*getCategory*(rs.getString("category")));

purchase.setCreatedAt(rs.getDate("created\_at"));

purchase.setUpdatedAt(rs.getDate("updated\_at"));

**return** purchase;

}

}

**public** Optional<Purchase> findById(Long id) **throws** BusinessException{

**try** {

**return** Optional.*of*(jdbcTemplate.queryForObject("select \* from purchase\_order where id=?",

**new** Object[] {id}, **new** BeanPropertyRowMapper<Purchase>(Purchase.**class**)));}

**catch** (Exception exception) {

**throw** **new** BusinessException("Internal Error Occured "+exception.getMessage());

}

}

**public** List<Purchase> findByDateCategory(Purchase purchase) **throws** BusinessException{

**try** {

**return** jdbcTemplate.query("select \* from purchase\_order where category=? and created\_at=?", **new** Object[] {purchase.getCategory(),

purchase.getCreatedAt()},**new** PurchaseRowMapper());}

**catch** (Exception exception) {

**throw** **new** BusinessException("Internal Error Occured "+exception.getMessage());

}

}

**public** **int** deleteSale(Long id) **throws** BusinessException {

**try** {

**return** jdbcTemplate.update("delete from purchase\_order where id=?", **new** Object[] {id});}

**catch** (Exception exception) {

**throw** **new** BusinessException("Internal Error Occured "+exception.getMessage());

}

}

**public** **int** createSale(Purchase purchase) **throws** BusinessException {

**try** {

**return** jdbcTemplate.update("insert into purchase\_report( user\_id,product\_id, quantity, category) values(?,?,?,?)",

**new** Object[] {purchase.getUser\_id(),purchase.getProduct\_id(),purchase.getQuantity(),purchase.getCategory()});

}

**catch** (Exception exception) {

**throw** **new** BusinessException("Internal Error Occured "+exception.getMessage());

}

}

}

* + 1. **UserRepository.java**

**package** com.ecommerce.sportyshoes.repository;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**import** java.util.List;

**import** java.util.Optional;

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

**import** org.springframework.jdbc.core.BeanPropertyRowMapper;

**import** org.springframework.jdbc.core.JdbcTemplate;

**import** org.springframework.jdbc.core.RowMapper;

**import** org.springframework.stereotype.Repository;

**import** com.ecommerce.sportyshoes.enums.UserType;

**import** com.ecommerce.sportyshoes.exceptions.BusinessException;

**import** com.ecommerce.sportyshoes.model.User;

@Repository

**public** **class** UserRepository {

@Autowired

JdbcTemplate jdbcTemplate;

**class** UserRowMapper **implements** RowMapper<User> {

@Override

**public** User mapRow(ResultSet rs, **int** rowNum) **throws** SQLException {

User user=**new** User();

user.setID(rs.getLong("id"));

user.setFirstName(rs.getString("first\_name"));

user.setLastName(rs.getString("last\_name"));

user.setEmail(rs.getString("email"));

user.setMobile(rs.getString("mobile"));

user.setPassword(rs.getString("password"));

user.setUserType(UserType.*getUserType*(rs.getString("user\_type")));

user.setCreatedAt(rs.getDate("created\_at"));

user.setUpdatedAt(rs.getDate("updated\_at"));

**return** user;

}

}

**public** List<User> findAll() **throws** BusinessException{

**try** {

**return** jdbcTemplate.query("select \* from users",**new** UserRowMapper());}

**catch** (Exception exception) {

**throw** **new** BusinessException("Internal Error Occured "+exception.getMessage());

}

}

**public** Optional<User> findByEmail(String email) **throws** BusinessException{

**try** {

**return** Optional.*of*(jdbcTemplate.queryForObject("select \* from users where email=?",

**new** Object[] {email}, **new** BeanPropertyRowMapper<User>(User.**class**)));}

**catch** (Exception exception) {

**throw** **new** BusinessException("Internal Error Occured "+exception.getMessage());

}

}

**public** **int** createUser(User user) **throws** BusinessException {

**try** {

**return** jdbcTemplate.update("insert into users( first\_name,last\_name, email,password, mobile) values(?,?,?,?,?)",

**new** Object[] {user.getFirstName(),user.getLastName(),user.getEmail(),user.getPassword(),user.getMobile()});

}

**catch** (Exception exception) {

**throw** **new** BusinessException("Internal Error Occured "+exception.getMessage());

}

}

**public** **int** updateUser(User user) **throws** BusinessException {

**try** {

**return** jdbcTemplate.update("update users "

+ "set first\_name=ifnull(?,first\_name), "

+ "last\_name=ifnull(?,last\_name), "

+ "password=ifnull(?,password) where email=?",

**new** Object[] {user.getFirstName(),user.getLastName(),user.getPassword(),user.getEmail()});

}

**catch** (Exception exception) {

**throw** **new** BusinessException("Internal Error Occured "+exception.getMessage());

}

}

**public** **int** deleteUser(String email) **throws** BusinessException {

**try** {

**return** jdbcTemplate.update("delete from users where email=?", **new** Object[] {email});}

**catch** (Exception exception) {

**throw** **new** BusinessException("Internal Error Occured "+exception.getMessage());

}

}

}

* 1. **Service. Interface**
     1. **ProductServiceIface.java**

**package** com.ecommerce.sportyshoes.service.iface;

**import** java.util.List;

**import** java.util.Optional;

**import** com.ecommerce.sportyshoes.exceptions.BusinessException;

**import** com.ecommerce.sportyshoes.model.Product;

**public** **interface** ProductServiceIface {

**public** **int** updateProduct(Product prodDataInput) **throws** BusinessException;

**public** **int** createProduct(Product prodDataInput) **throws** BusinessException;

**public** **int** deleteProduct(Product prodDataInput) **throws** BusinessException;

**public** List<Product> getProdByCategory(Product prodDataInput) **throws** BusinessException;

**public** Optional<Product> getPodById(Product prodDataInput) **throws** BusinessException;

}

* + 1. **PurchaseServiceIface.java**

**package** com.ecommerce.sportyshoes.service.iface;

**import** java.util.List;

**import** java.util.Optional;

**import** com.ecommerce.sportyshoes.exceptions.BusinessException;

**import** com.ecommerce.sportyshoes.model.Purchase;

**public** **interface** PurchaseServiceIface {

**public** **int** createSale(Purchase purchaseDataInput) **throws** BusinessException;

**public** **int** deleteSale(Purchase purchaseDataInput) **throws** BusinessException;

**public** List<Purchase> getByDateCategory(Purchase purchaseDataInput) **throws** BusinessException;

**public** Optional<Purchase> getPurchaseById(Purchase purchaseDataInput) **throws** BusinessException;

}

* + 1. **UserServiceIface.java**

**package** com.ecommerce.sportyshoes.service.iface;

**import** java.util.List;

**import** java.util.Optional;

**import** com.ecommerce.sportyshoes.exceptions.BusinessException;

**import** com.ecommerce.sportyshoes.model.User;

**public** **interface** UserServiceIface {

**public** **int** updateUser(User userDataInput) **throws** BusinessException;

**public** List<User> getAllUsers() **throws** BusinessException;

**public** Optional<User> getUserByEmail(User userDataInput) **throws** BusinessException;

**public** **int** createUser(User userDataInput) **throws** BusinessException;

**public** **int** deleteUser(User userDataInput) **throws** BusinessException;

}

* 1. **Service.impl**
     1. **ProductServiceImpl.java**

**package** com.ecommerce.sportyshoes.service.impl;

**import** java.util.List;

**import** java.util.Optional;

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

**import** org.springframework.stereotype.Service;

**import** com.ecommerce.sportyshoes.exceptions.BusinessException;

**import** com.ecommerce.sportyshoes.model.Product;

**import** com.ecommerce.sportyshoes.repository.ProductRepository;

**import** com.ecommerce.sportyshoes.service.iface.ProductServiceIface;

@Service

**public** **class** ProductServiceImpl **implements** ProductServiceIface{

@Autowired

ProductRepository productRepository;

@Override

**public** **int** updateProduct(Product prodDataInput) **throws** BusinessException {

Product product = **new** Product();

product.setId(prodDataInput.getId());

product.setAvailability(prodDataInput.getAvailability());

product.setPrice(prodDataInput.getPrice());

**return** productRepository.updateProduct(product);

}

@Override

**public** **int** createProduct(Product prodDataInput) **throws** BusinessException {

// **TODO** Auto-generated method stub

**return** productRepository.createProduct(prodDataInput);

}

@Override

**public** **int** deleteProduct(Product prodDataInput) **throws** BusinessException {

// **TODO** Auto-generated method stub

**return** productRepository.deleteProduct(prodDataInput.getId());

}

@Override

**public** List<Product> getProdByCategory(Product prodDataInput) **throws** BusinessException {

// **TODO** Auto-generated method stub

**return** productRepository.findByCategory(prodDataInput.getCategory().toString());

}

@Override

**public** Optional<Product> getPodById(Product prodDataInput) **throws** BusinessException {

// **TODO** Auto-generated method stub

**return** productRepository.findById(prodDataInput.getId());

}

}

* + 1. **PurchaseServiceImpl.java**

**package** com.ecommerce.sportyshoes.service.impl;

**import** java.util.List;

**import** java.util.Optional;

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

**import** org.springframework.stereotype.Service;

**import** com.ecommerce.sportyshoes.exceptions.BusinessException;

**import** com.ecommerce.sportyshoes.model.Purchase;

**import** com.ecommerce.sportyshoes.repository.PurchaseRepository;

**import** com.ecommerce.sportyshoes.service.iface.PurchaseServiceIface;

@Service

**public** **class** PurchaseServiceImpl **implements** PurchaseServiceIface{

@Autowired

PurchaseRepository purchaseRepository;

@Override

**public** **int** createSale(Purchase purchaseDataInput) **throws** BusinessException {

// **TODO** Auto-generated method stub

**return** purchaseRepository.createSale(purchaseDataInput);

}

@Override

**public** **int** deleteSale(Purchase purchaseDataInput) **throws** BusinessException {

// **TODO** Auto-generated method stub

**return** purchaseRepository.deleteSale(purchaseDataInput.getId());

}

@Override

**public** List<Purchase> getByDateCategory(Purchase purchaseDataInput) **throws** BusinessException {

// **TODO** Auto-generated method stub

Purchase purchase=**new** Purchase();

purchase.setCategory(purchaseDataInput.getCategory());

purchase.setCreatedAt(purchaseDataInput.getCreatedAt());

**return** purchaseRepository.findByDateCategory(purchase);

}

@Override

**public** Optional<Purchase> getPurchaseById(Purchase purchaseDataInput) **throws** BusinessException {

// **TODO** Auto-generated method stub

**return** purchaseRepository.findById(purchaseDataInput.getId());

}

}

* + 1. **UserServiceImpl.java**

**package** com.ecommerce.sportyshoes.service.impl;

**import** java.util.List;

**import** java.util.Optional;

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

**import** org.springframework.stereotype.Service;

**import** com.ecommerce.sportyshoes.exceptions.BusinessException;

**import** com.ecommerce.sportyshoes.model.User;

**import** com.ecommerce.sportyshoes.repository.UserRepository;

**import** com.ecommerce.sportyshoes.service.iface.UserServiceIface;

@Service

**public** **class** UserServiceImpl **implements** UserServiceIface{

@Autowired

UserRepository userRepository;

**public** **int** updateUser(User userDataInput) **throws** BusinessException {

User user = **new** User();

user.setFirstName(userDataInput.getFirstName());

user.setLastName(userDataInput.getLastName());

user.setPassword(userDataInput.getPassword());

user.setEmail(userDataInput.getEmail());

**return** userRepository.updateUser(user);

}

**public** List<User> getAllUsers() **throws** BusinessException{

**return** userRepository.findAll();

}

**public** Optional<User> getUserByEmail(User userDataInput) **throws** BusinessException{

**return** userRepository.findByEmail(userDataInput.getEmail());

}

**public** **int** createUser(User userDataInput) **throws** BusinessException {

**return** userRepository.createUser(userDataInput);

}

**public** **int** deleteUser(User userDataInput) **throws** BusinessException {

**return** userRepository.deleteUser(userDataInput.getEmail());

}

}

* 1. **Swagger**
     1. **SwaggerConfiguration.java**

**package** com.ecommerce.sportyshoes.swagger;

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

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

**import** springfox.documentation.builders.PathSelectors;

**import** springfox.documentation.builders.RequestHandlerSelectors;

**import** springfox.documentation.spi.DocumentationType;

**import** springfox.documentation.spring.web.plugins.Docket;

**import** springfox.documentation.swagger2.annotations.EnableSwagger2;

**import** **static** springfox.documentation.builders.PathSelectors.*regex*;

@Configuration

@EnableSwagger2

**public** **class** SwaggerConfiguration {

@Bean

**public** Docket productAPi() {

**return** **new** Docket(DocumentationType.***SWAGGER\_2***)

.select().apis(RequestHandlerSelectors.

*basePackage*("com.ecommerce.sportyshoes.controller"))

.paths(PathSelectors.*any*()).build();}

}

1. **src/main/resource**
   1. **application.properties**

spring.datasource.url = jdbc:mysql://localhost:3306/sportyshoe

spring.datasource.username = root

spring.datasource.password = root

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5InnoDBDialect

spring.jpa.hibernate.ddl-auto=update

logging.level.org.hiberate.stat=debug

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.format\_sql=true

logging.level.org.hibernate.type=trace

server.port=8082

1. **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>2.3.2.RELEASE</version>

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

</parent>

<groupId>com.ecommerce</groupId>

<artifactId>sportyshoes</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>sportyshoes</name>

<description>Phase 3 live project sporty shoes</description>

<properties>

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

</properties>

<dependencies>

<dependency>

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

<artifactId>spring-boot-starter-data-jdbc</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.springframework.boot</groupId>

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

<version>2.6.4</version>

</dependency>

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<optional>true</optional>

</dependency>

<dependency>

<groupId>io.springfox</groupId>

<artifactId>springfox-swagger2</artifactId>

<version>3.0.0</version>

<scope>compile</scope>

</dependency>

<dependency>

<groupId>io.springfox</groupId>

<artifactId>springfox-swagger-ui</artifactId>

<version>3.0.0</version>

</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>