```
<?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>2.7.3</version>
		<relativePath /> <!-- lookup parent from repository -->
	</parent>
	<groupId>com.example</groupId>
	<artifactId>S13SpringBootREST</artifactId>
	<version>0.0.1-SNAPSHOT</version>
	<name>S13SpringBootREST</name>
	<description>Demo project for Spring Boot</description>
	<properties>
		<java.version>17</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>mysql</groupId>
			<artifactId>mysql-connector-java</artifactId>
			<scope>runtime</scope>
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-test</artifactId>
			<scope>test</scope>
		</dependency>
		<dependency>
			<groupId>javax.servlet</groupId>
			<artifactId>jstl</artifactId>
			<version>1.2</version>
		</dependency>
```

```
		<dependency>
			<groupId>taglibs</groupId>
			<artifactId>standard</artifactId>
			<version>1.1.2</version>
		</dependency>
		<dependency>
			<groupId>org.apache.tomcat.embed</groupId>
			<artifactId>tomcat-embed-jasper</artifactId>
			<version>9.0.44</version>
			<scope>provided</scope>
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-tomcat</artifactId>
			<version>2.4.4</version>
			<scope>provided</scope>
		</dependency>

	</dependencies>
	<build>
		<plugins>
			<plugin>
				<groupId>org.springframework.boot</groupId>
				<artifactId>spring-boot-maven-plugin</artifactId>
			</plugin>
		</plugins>
	</build>
</project>


package com.project.sportyshoes;

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

@SpringBootApplication
public class SportyShoesApplication {

	public static void main(String[] args) {
		SpringApplication.run(SportyShoesApplication.class, args);
	}

}

package com.project.sportyshoes.controller;

import java.util.List;

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

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.servlet.ModelAndView;

import com.project.sportyshoes.entity.Admin;
import com.project.sportyshoes.entity.Category;
import com.project.sportyshoes.entity.Product;
import com.project.sportyshoes.entity.Purchase;
import com.project.sportyshoes.entity.User;
import com.project.sportyshoes.exception.DataNotFoundException;
import com.project.sportyshoes.exception.DuplicateIdException;
import com.project.sportyshoes.service.AdminService;
import com.project.sportyshoes.service.CategoryService;
import com.project.sportyshoes.service.ProductService;
import com.project.sportyshoes.service.PurchaseService;
import com.project.sportyshoes.service.UserService;

@Controller
@RestController
@RequestMapping("admin")
public class AdminController {

	@Autowired
	AdminService adminService;

	@Autowired
	ProductService productService;

	@Autowired
	CategoryService categoryService;

	@Autowired
	UserService userService;

	@Autowired
	PurchaseService purchaseService;

	@RequestMapping(value = "create", method = RequestMethod.POST)
	public String create(@ModelAttribute Admin admin) {
		int res;
		try {
			res = adminService.create(admin);
```

```
                        return res + " Added";
                } catch (DuplicateIdException e) {
                        return e.getMessage();
                }
        }

        @RequestMapping(value = "{email}", method = RequestMethod.GET)
        public Object find(@PathVariable String email) {
                Admin admin;
                try {
                        admin = adminService.find(email);
                        return admin;
                } catch (DataNotFoundException e) {
                        return e.getMessage();
                }
        }

        @RequestMapping(value = "/authenticate", method = RequestMethod.POST)
        public ModelAndView find(@RequestParam("email") String email,
@RequestParam("password") String password,
                        HttpServletRequest request, ModelMap model) {
                Admin admin;
                ModelAndView mv = new ModelAndView();
                try {
                        admin = (Admin) adminService.find(email, password);
                        HttpSession session = request.getSession();
                        session.setAttribute("email", admin.getEmail());
                        model.addAttribute("admin", admin);
                        mv.setViewName("redirect:/admin/dashboard");
                        return mv;
                } catch (DataNotFoundException e) {
                        mv.addObject("message", "Invalid details");
                        mv.setViewName("adminLogin");
                        return mv;
                }
        }

        @RequestMapping("dashboard")
        public ModelAndView adminDashboard(ModelMap model, HttpServletRequest
request) {
                ModelAndView mv = new ModelAndView();
                HttpSession session = request.getSession(false);
                String email = (String) session.getAttribute("email");
                if (session == null || email == null ||
!request.isRequestedSessionIdValid()) {
                        mv.addObject("message", "Session is over please login
again");
                        mv.setViewName("adminLogin");
                } else {
                        Admin admin;
```

```
                        try {
                                List<Product> products = productService.loadAll();
                                List<Category> categories =
categoryService.findAll();
                                List<User> users = userService.loadAll();
                                List<Purchase> purchases =
purchaseService.findAll();
                                admin = adminService.find(email);
                                mv.addObject("purchases", purchases);
                                mv.addObject("products", products);
                                mv.addObject("categories", categories);
                                mv.addObject("admin", admin);
                                mv.addObject("users", users);
                                mv.setViewName("adminDashboard");
                        } catch (DataNotFoundException e) {
                                e.printStackTrace();
                        }
                }
                return mv;
        }

        @RequestMapping("changePassword")
        public ModelAndView changePassword(@RequestParam String oldPassword,
@RequestParam String newPassword,
                        HttpServletRequest request) {
                ModelAndView mv = new ModelAndView();
                HttpSession session = request.getSession(false);
                if (session == null || session.getAttribute("email") == null ||
!request.isRequestedSessionIdValid()) {
                        mv.addObject("message", "Session is over please login
again");
                        mv.setViewName("adminLogin");
                } else {
                        String email = (String) session.getAttribute("email");
                        try {
                                adminService.changePassword(email, oldPassword,
newPassword);
                                mv.setViewName("redirect:/admin/dashboard");
                        } catch (Exception e) {
                                mv.addObject("message", "Please enter correct
password");
                                mv.addObject("type", "danger");
                                mv.setViewName("forward:/admin/dashboard");
                        }
                }

                return mv;
        }

        @RequestMapping("filter/category")
```

```
        public ModelAndView filterCategory(@RequestParam("category_name") String
category_name) {
                ModelAndView mv = new ModelAndView();
                List<Purchase> purchases =
purchaseService.findByCategory(category_name);
                System.out.println(purchases);
                mv.setViewName("forward:/admin/dashboard");
                mv.addObject("filteredPurchases",purchases);
                return mv;
        }

        @RequestMapping("/logout")
        public ModelAndView logout(HttpServletRequest request) {
                ModelAndView mv = new ModelAndView();
                HttpSession session = request.getSession(false);
                session.invalidate();
                mv.addObject("message", "Logged out successfully");
                mv.addObject("type", "success");
                mv.setViewName("adminLogin");
                return mv;
        }
}
package com.project.sportyshoes.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.servlet.ModelAndView;

import com.project.sportyshoes.entity.Category;
import com.project.sportyshoes.exception.DataNotFoundException;
import com.project.sportyshoes.exception.DuplicateIdException;
import com.project.sportyshoes.service.CategoryService;

@Controller
@RestController
@RequestMapping("category")
public class CatogoryController {
        @Autowired
        CategoryService categoryService;

        @RequestMapping(value = "create", method = RequestMethod.POST)
        public ModelAndView create(@ModelAttribute Category catogory) {
                ModelAndView mv = new ModelAndView();
```

```
		try {
			int res = categoryService.create(catogory);
			mv.setViewName("redirect:/admin/dashboard");
		} catch (DuplicateIdException e) {
			mv.setViewName("forward:/admin/dashboard");
			mv.addObject("message", e.getMessage());
		}
		return mv;
	}

	@RequestMapping(value = "/", method = RequestMethod.GET)
	public List<Category> loadAll() {
		List<Category> catogories = categoryService.findAll();
		return catogories;
	}

	@RequestMapping(value = "{name}", method = RequestMethod.GET)
	public Object findByName(@PathVariable String name) {
		Category catogory;
		try {
			catogory = categoryService.find(name);
			return catogory;
		} catch (DataNotFoundException e) {
			return e.getMessage();
		}
	}

	@RequestMapping(value = "delete/{id}")
	public ModelAndView delete(@PathVariable int id) {
		ModelAndView mv = new ModelAndView();
		mv.setViewName("redirect:/admin/dashboard");
		try {
			categoryService.delete(id);
		} catch (Exception e) {
			mv.addObject("Message", "Error in deleteing");
		}
		return mv;
	}

}
package com.project.sportyshoes.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.servlet.ModelAndView;

import com.project.sportyshoes.service.AdminService;
```

```
@Controller
public class MainController {

	@Autowired
	AdminService adminService;

	@RequestMapping("/home")
	public ModelAndView home() {
		ModelAndView mv = new ModelAndView("index");
		return mv;
	}

	@RequestMapping("adminLogin")
	public String adminLogin() {
		return "adminLogin";
	}
}
package com.project.sportyshoes.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.servlet.ModelAndView;

import com.project.sportyshoes.entity.Category;
import com.project.sportyshoes.entity.Product;
import com.project.sportyshoes.exception.DataNotFoundException;
import com.project.sportyshoes.exception.DuplicateIdException;
import com.project.sportyshoes.service.CategoryService;
import com.project.sportyshoes.service.ProductService;

@Controller
@RestController
@RequestMapping("product")
public class ProductController {

	@Autowired
	ProductService productService;

	@Autowired
	CategoryService categoryService;

	@RequestMapping(value = "create", method = RequestMethod.POST)
```

```
	public ModelAndView create(@ModelAttribute("product") Product product,
@RequestParam String category_name) {
		ModelAndView mv = new ModelAndView();
		mv.setViewName("redirect:/admin/dashboard");
		try {
			try {
				productService.create(product, category_name);

			} catch (DataNotFoundException e) {
				e.getMessage();
			}
		} catch (DuplicateIdException e) {
			e.getMessage();
		}
		return mv;
	}

	@RequestMapping(value = "/", method = RequestMethod.GET)
	public List<Product> findAll() {
		List<Product> products = productService.loadAll();
		return products;
	}

	@RequestMapping(value = "{name}", method = RequestMethod.GET)
	public List<Product> findByName(@PathVariable String name) {
		List<Product> products = productService.findProduct(name);
		return products;
	}

	@RequestMapping(value = "category/{category_name}", method =
RequestMethod.GET)
	public List<Product> findByCatogory(@PathVariable String category_name) {
		List<Product> products = productService.findProduct(category_name,
true);
		return products;
	}

	@RequestMapping(value = "update", method = RequestMethod.POST)
	public ModelAndView update(@ModelAttribute("product") Product product) {
		ModelAndView mv = new ModelAndView();
		try {
			productService.update(product);
			mv.setViewName("redirect:/admin/dashboard");
		} catch (Exception e) {
			e.getMessage();
		}
		return mv;
	}

	@RequestMapping(value = "update/{id}", method = RequestMethod.GET)
```

```
	public ModelAndView update(@PathVariable int id) {
		ModelAndView mv = new ModelAndView();
		Product product = productService.findProduct(id);
		mv.addObject("product", product);
		System.out.println(product);
		List<Category> categories = categoryService.findAll();
		mv.addObject("categories", categories);
		mv.setViewName("updateProduct");
		return mv;
	}

	@RequestMapping(value = "delete/{id}")
	public ModelAndView delete(@PathVariable int id) {
		ModelAndView mv = new ModelAndView();
		mv.setViewName("redirect:/admin/dashboard");
		try {
			productService.delete(id);
		} catch (Exception e) {
			mv.addObject("Message", "Error in deleteing");
		}
		return mv;
	}
}
package com.project.sportyshoes.controller;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.servlet.ModelAndView;

import com.project.sportyshoes.service.PurchaseService;

@RestController
@RequestMapping("/purchase")
public class PurchaseController {

	@Autowired
	PurchaseService purchaseService;

	@RequestMapping("buy/{productId}")
	public ModelAndView create(@PathVariable int productId, HttpServletRequest
request) {
		ModelAndView mv = new ModelAndView();
		HttpSession session = request.getSession(false);
		if (session != null && session.getAttribute("UserEmail") != null) {
			String userEmail = (String)
```

```
session.getAttribute("UserEmail");
                        try {
                                purchaseService.create(productId,userEmail);
                                mv.setViewName("redirect:/user/dashboard");
                        } catch (Exception e) {
                                mv.setViewName("forward:/user/dashboard");
                                mv.addObject("message", "Error in buying product");
                        }
                }
                return mv;
        }
}
package com.project.sportyshoes.controller;

import java.util.List;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.ui.ModelMap;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.servlet.ModelAndView;

import com.project.sportyshoes.entity.Admin;
import com.project.sportyshoes.entity.Category;
import com.project.sportyshoes.entity.Product;
import com.project.sportyshoes.entity.Purchase;
import com.project.sportyshoes.entity.User;
import com.project.sportyshoes.exception.DataNotFoundException;
import com.project.sportyshoes.exception.DuplicateIdException;
import com.project.sportyshoes.service.ProductService;
import com.project.sportyshoes.service.PurchaseService;
import com.project.sportyshoes.service.UserService;

@RestController
@RequestMapping("/user")
public class UserController {

        @Autowired
        UserService userService;

        @Autowired
        ProductService productService;

        @Autowired
```

```
	PurchaseService purchaseService;

	@RequestMapping(value = "create", method = RequestMethod.POST)
	public ModelAndView create(@ModelAttribute("user") User user) {
		ModelAndView mv = new ModelAndView();
		try {
			userService.create(user);
			mv.setViewName("userLogin");
			mv.addObject("message", "Youre accout created login
below");
			mv.addObject("type", "success");
			mv.addObject("message", "Your account is successfully
created please login");
		} catch (DuplicateIdException e) {
			mv.setViewName("signUpUser");
			mv.addObject("message", "Account already exist please
login");
		}
		return mv;
	}

	@RequestMapping(value = "/", method = RequestMethod.GET)
	public List<User> loadAll() {
		return userService.loadAll();
	}

	@RequestMapping("login")
	public ModelAndView userLogin() {
		return new ModelAndView("userLogin");
	}

	@RequestMapping("signUp")
	public ModelAndView userSiginUp() {
		return new ModelAndView("signUpUser");
	}

	@RequestMapping(value = "/authenticate", method = RequestMethod.POST)
	public ModelAndView authenticate(@RequestParam("email") String email,
@RequestParam("password") String password,
			HttpServletRequest request, ModelMap model) {
		User user;
		ModelAndView mv = new ModelAndView();
		try {
			user = (User) userService.getUser(email, password);
			HttpSession session = request.getSession();
			session.setAttribute("UserEmail", user.getEmail());
			model.addAttribute("user", user);
			mv.setViewName("redirect:/user/dashboard");
		} catch (DataNotFoundException e) {
			mv.addObject("message", "Invalid details");
```

```
                        mv.addObject("type", "danger");
                        mv.setViewName("userLogin");
                }
                return mv;
        }

        @RequestMapping("dashboard")
        public ModelAndView adminDashboard(ModelMap model, HttpServletRequest
request) {
                ModelAndView mv = new ModelAndView();
                HttpSession session = request.getSession(false);
                String email = (String) session.getAttribute("UserEmail");
                if (session == null || email == null ||
!request.isRequestedSessionIdValid()) {
                        mv.addObject("message", "Session is over please login
again");
                        mv.addObject("type", "danger");
                        mv.setViewName("userLogin");
                } else {
                        User user;
                        List<Product> products = productService.loadAll();
                        user = userService.find(email);
                        List<Purchase> purchases =
purchaseService.findByUser(user.getId());
                        mv.addObject("products", products);
                        mv.addObject("purchases", purchases);
                        mv.addObject("user", user);
                        mv.setViewName("userDashboard");
                }
                return mv;
        }

        @RequestMapping("/logout")
        public ModelAndView logout(HttpServletRequest request) {
                ModelAndView mv = new ModelAndView();
                HttpSession session = request.getSession(false);
                session.invalidate();
                mv.addObject("message", "Logged out successfully");
                mv.addObject("type", "success");
                mv.setViewName("userLogin");
                return mv;
        }

}
package com.project.sportyshoes.entity;

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

```
@Entity
public class Admin {

	@Id
	@GeneratedValue(strategy = GenerationType.AUTO)
	private int id;
	private String name;
	private String email;
	private String password;

	public Admin() {
		super();
	}

	public Admin(String name, String email, String password) {
		super();
		this.name = name;
		this.email = email;
		this.password = password;
	}

	public int getId() {
		return id;
	}

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

	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
	}

	public String getEmail() {
		return email;
	}

	public void setEmail(String email) {
		this.email = email;
	}

	public String getPassword() {
		return password;
	}
```

```
	public void setPassword(String password) {
		this.password = password;
	}

	@Override
	public String toString() {
		return "Admin [id=" + id + ", name=" + name + ", email=" + email +
", password=" + password + "]";
	}

}
package com.project.sportyshoes.entity;

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

@Entity
public class Category {

	@Id
	@GeneratedValue(strategy = GenerationType.AUTO)
	private int id;
	private String name;
	private String discription;

	public Category() {
	}

	public Category(String name, String discription) {
		super();
		this.name = name;
		this.discription = discription;
	}

	public int getId() {
		return id;
	}

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

	public String getName() {
		return name;
	}

	public void setName(String name) {
```

```
		this.name = name;
	}

	public String getDiscription() {
		return discription;
	}

	public void setDiscription(String discription) {
		this.discription = discription;
	}

	@Override
	public String toString() {
		return "Category [id=" + id + ", name=" + name + ", discription=" +
discription + "]";
	}

}
package com.project.sportyshoes.entity;

import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.Table;

@Entity
@Table(name = "Shoe")
public class Product {
	@Id
	@GeneratedValue(strategy = GenerationType.AUTO)
	private int id;
	private String name;
	private String discription;
	private double price;
	@ManyToOne(cascade = { CascadeType.PERSIST })
	@JoinColumn(name = "category_id")
	private Category category;

	public Product() {
		super();
	}

	public Product(String name, String discription, double price, Category
category) {
		this.name = name;
		this.discription = discription;
```

```
                this.price = price;
                this.category = category;
        }

        public int getId() {
                return id;
        }

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

        public String getName() {
                return name;
        }

        public void setName(String name) {
                this.name = name;
        }

        public String getDiscription() {
                return discription;
        }

        public void setDiscription(String discription) {
                this.discription = discription;
        }

        public double getPrice() {
                return price;
        }

        public void setPrice(double price) {
                this.price = price;
        }

        public Category getCategory() {
                return category;
        }

        public void setCategory(Category category) {
                this.category = category;
        }

        @Override
        public String toString() {
                return "Product [id=" + id + ", name=" + name + ", discription=" +
discription + ", catogory=" + category + "]";
        }
```

```
}
package com.project.sportyshoes.entity;

import java.sql.Timestamp;

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

import org.hibernate.annotations.CreationTimestamp;

@Entity
public class Purchase {
	@Id
	@GeneratedValue(strategy = GenerationType.AUTO)
	private int purchase_id;

	@ManyToOne()
	@JoinColumn(name = "user_id")
	private User user;

	@ManyToOne()
	@JoinColumn(name = "product_id")
	private Product product;

	@CreationTimestamp
	private Timestamp orderedDate;

	public Purchase() {
	}

	public Purchase(User user, Product product) {
		this.user = user;
		this.product = product;
	}

	public int getPurchase_id() {
		return purchase_id;
	}

	public void setPurchase_id(int purchase_id) {
		this.purchase_id = purchase_id;
	}

	public Timestamp getOrderedDate() {
		return orderedDate;
	}
```

```
	public void setOrderedDate(Timestamp orderedDate) {
		this.orderedDate = orderedDate;
	}

	public User getUser() {
		return user;
	}

	public void setUser(User user) {
		this.user = user;
	}

	public Product getProduct() {
		return product;
	}

	public void setProduct(Product product) {
		this.product = product;
	}

	public Timestamp getOrdered_date() {
		return orderedDate;
	}

	public void setOrdered_date(Timestamp ordered_date) {
		this.orderedDate = ordered_date;
	}

	@Override
	public String toString() {
		return "Purchase [id=" + purchase_id + ", user=" + user + ",
product=" + product + ", ordered_date="
				+ orderedDate + "]";
	}

}
package com.project.sportyshoes.entity;

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

@Entity
public class User {

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

```
	private String name;
	private String email;
	private String password;

	public User() {
	}

	public User(String name, String email, String password) {
		this.name = name;
		this.email = email;
		this.password = password;
	}

	public int getId() {
		return id;
	}

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

	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
	}

	public String getEmail() {
		return email;
	}

	public void setEmail(String email) {
		this.email = email;
	}

	public String getPassword() {
		return password;
	}

	public void setPassword(String password) {
		this.password = password;
	}

	@Override
	public String toString() {
		return "User [id=" + id + ", name=" + name + ", email=" + email +
", password=" + password + "]";
	}
```

```
}
package com.project.sportyshoes.exception;

public class DataNotFoundException extends Exception{
	public DataNotFoundException(){
		super("Requested data not found");
	}
}
package com.project.sportyshoes.exception;

public class DuplicateIdException extends Exception {

	public DuplicateIdException(String email) {
		super(email+" Already exists");
	}

}
package com.project.sportyshoes.repository;

import org.springframework.data.repository.CrudRepository;
import org.springframework.stereotype.Repository;

import com.project.sportyshoes.entity.Admin;

@Repository
public interface AdminRepository extends CrudRepository<Admin, Integer> {
	Admin findByEmail(String Email);

	boolean existsByEmail(String email);

	Admin findByEmailAndPassword(String email, String password);

}
package com.project.sportyshoes.repository;

import org.springframework.data.repository.CrudRepository;

import org.springframework.stereotype.Repository;

import com.project.sportyshoes.entity.Category;

@Repository
public interface CategoryRepository extends CrudRepository<Category, Integer> {
	Category findByName(String Name);

	boolean existsByName(String Name);

	void deleteById(int id);
}
```

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

import java.util.List;

import org.springframework.data.repository.CrudRepository;

import com.project.sportyshoes.entity.Category;
import com.project.sportyshoes.entity.Product;

public interface ProductRepository extends CrudRepository<Product, Integer> {
        List<Product> findByCategory(Category category);

        boolean existsByName(String name);

        List<Product> findByName(String name);

        Product findById(int id);

        List<Product> findByCategoryName(String name);

        boolean existsById(int id);

        void deleteById(int id) throws Exception;
}
package com.project.sportyshoes.repository;

import java.util.Date;
import java.util.List;

import org.springframework.data.repository.CrudRepository;
import org.springframework.stereotype.Repository;

import com.project.sportyshoes.entity.Purchase;

@Repository
public interface PurchaseRepository extends CrudRepository<Purchase, Integer> {
        List<Purchase> findByProductCategoryName(String id);

        List<Purchase> findByorderedDate(Date date);

        List<Purchase> findByUserId(int id);
}
package com.project.sportyshoes.repository;

import org.springframework.data.repository.CrudRepository;

import com.project.sportyshoes.entity.User;

public interface UserRepository extends CrudRepository<User, Integer> {
        User findByEmailAndPassword(String email, String password);
```

```
        boolean existsByEmail(String emai);

        User findByEmail(String email);
}
package com.project.sportyshoes.service;

import javax.transaction.Transactional;

import com.project.sportyshoes.entity.Admin;
import com.project.sportyshoes.exception.DataNotFoundException;
import com.project.sportyshoes.exception.DuplicateIdException;

public interface AdminService {
        @Transactional
        int create(Admin admin) throws DuplicateIdException;

        Admin find(String email) throws DataNotFoundException;

        boolean exists(String email);

        Object find(String email, String password) throws DataNotFoundException;

        void changePassword(String email, String oldPassword, String newPassword)
throws Exception;
}
package com.project.sportyshoes.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import com.project.sportyshoes.entity.Admin;
import com.project.sportyshoes.exception.DataNotFoundException;
import com.project.sportyshoes.exception.DuplicateIdException;
import com.project.sportyshoes.repository.AdminRepository;

@Service
public class AdminServiceImpl implements AdminService {

        @Autowired
        AdminRepository adminRepository;

        @Override
        @Transactional
        public int create(Admin admin) throws DuplicateIdException {

                if (adminRepository.existsByEmail(admin.getEmail())) {
                        throw new DuplicateIdException(admin.getEmail());
                } else {
```

```
                        int id = adminRepository.save(admin).getId();
                        return id;
                }
        }

        @Override
        public Admin find(String email) throws DataNotFoundException {
                Admin admin = adminRepository.findByEmail(email);
                if (admin == null) {
                        throw new DataNotFoundException();
                }
                return admin;
        }

        @Override
        public boolean exists(String email) {

                return false;
        }

        @Override
        public Admin find(String email, String password) throws
DataNotFoundException {
                System.out.println(email + " " + password);
                Admin admin = adminRepository.findByEmailAndPassword(email,
password);
                if (admin == null) {
                        throw new DataNotFoundException();
                }
                return admin;
        }

        @Override
        public void changePassword(String email, String oldPassword, String
newPassword) throws Exception {

                Admin admin = adminRepository.findByEmail(email);
                if (oldPassword.equals(admin.getPassword())) {
                        admin.setPassword(newPassword);
                        adminRepository.save(admin);
                } else {
                        throw new Exception("Invalid Password");
                }

        }
}
package com.project.sportyshoes.service;

import java.util.List;
```

```
import javax.transaction.Transactional;

import com.project.sportyshoes.entity.Category;
import com.project.sportyshoes.exception.DataNotFoundException;
import com.project.sportyshoes.exception.DuplicateIdException;

public interface CategoryService {
	@Transactional
	int create(Category catogory) throws DuplicateIdException;

	List<Category> findAll();

	Category find(String Name) throws DataNotFoundException;

	void delete(int id);
}
package com.project.sportyshoes.service;

import java.util.List;

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

import com.project.sportyshoes.entity.Category;
import com.project.sportyshoes.exception.DataNotFoundException;
import com.project.sportyshoes.exception.DuplicateIdException;
import com.project.sportyshoes.repository.CategoryRepository;

@Service
public class CategoryServiceImpl implements CategoryService {

	@Autowired
	CategoryRepository catogoryRepository;

	@Override
	public int create(Category catogory) throws DuplicateIdException {
		if (catogoryRepository.existsByName(catogory.getName())) {
			throw new DuplicateIdException(catogory.getName());
		} else {
			return catogoryRepository.save(catogory).getId();
		}
	}

	@Override
	public List<Category> findAll() {
		List<Category> catogories = (List<Category>)
catogoryRepository.findAll();
		return catogories;
	}
```

```
	@Override
	public Category find(String Name) throws DataNotFoundException {
		Category catogory = catogoryRepository.findByName(Name);
		if (catogory == null) {
			throw new DataNotFoundException();
		} else {
			return catogory;
		}
	}

	@Override
	public void delete(int id) {
		catogoryRepository.deleteById(id);
	}

}
package com.project.sportyshoes.service;

import java.util.List;

import javax.transaction.Transactional;

import com.project.sportyshoes.entity.Product;
import com.project.sportyshoes.exception.DataNotFoundException;
import com.project.sportyshoes.exception.DuplicateIdException;

public interface ProductService {
	@Transactional
	String create(Product product, String catogory_name) throws
DuplicateIdException, DataNotFoundException;

	List<Product> loadAll();

	List<Product> findProduct(String name);

	Product findProduct(int name);

	List<Product> findProduct(String catogory_name, Boolean catogory);

	String update(Product product) throws Exception;

	void delete(int id) throws Exception;

}
package com.project.sportyshoes.service;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
```

```
import com.project.sportyshoes.entity.Category;
import com.project.sportyshoes.entity.Product;
import com.project.sportyshoes.exception.DataNotFoundException;
import com.project.sportyshoes.exception.DuplicateIdException;
import com.project.sportyshoes.repository.ProductRepository;

@Service
public class ProductServiceImpl implements ProductService {

	@Autowired
	ProductRepository productRepository;

	@Autowired
	CategoryService categoryService;

	@Override
	public String create(Product product, String category_name) throws
DuplicateIdException, DataNotFoundException {
		if (productRepository.existsByName(product.getName())) {
			return "Product with name " + product.getName() + "
exists";
		}
		try {
			Category category = categoryService.find(category_name);
			product.setCategory(category);
			String id = productRepository.save(product).getName();
			return id;
		} catch (DataNotFoundException e) {
			throw new DataNotFoundException();
		}
	}

	@Override
	public List<Product> loadAll() {
		List<Product> products = (List<Product>)
productRepository.findAll();
		return products;
	}

	@Override
	public List<Product> findProduct(String name) {
		List<Product> products = productRepository.findByName(name);
		return products;
	}

	@Override
	public List<Product> findProduct(String name, Boolean catogory) {
		List<Product> products =
productRepository.findByCategoryName(name);
```

```
                return products;
        }

        @Override
        public String update(Product product) throws Exception {

                if (!productRepository.existsById(product.getId())) {
                        throw new Exception("Invalid data please update with unique
name or data not found for request id");
                } else {
                        try {
                                String name =
productRepository.save(product).getName();
                                return name;
                        } catch (Exception e) {
                                throw new Exception();
                        }
                }
        }

        @Override
        public void delete(int id) throws Exception {

                if (!productRepository.existsById(id))
                        throw new Exception("Product not found");

                else {

                        productRepository.deleteById(id);
                }
        }

        @Override
        public Product findProduct(int id) {
                Product product = productRepository.findById(id);
                return product;
        }

}
package com.project.sportyshoes.service;

import java.text.ParseException;
import java.util.List;

import com.project.sportyshoes.entity.Purchase;

public interface PurchaseService {
        void create(int id,String userEmail);

        List<Purchase> findByUser(int id);
```

```
	List<Purchase> findAll();

	List<Purchase> findByCategory(String name);

	List<Purchase> findByDate(String date) throws ParseException;

}
package com.project.sportyshoes.service;

import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.List;

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

import com.project.sportyshoes.entity.Purchase;
import com.project.sportyshoes.repository.PurchaseRepository;

@Service
public class PurchaseServiceImpl implements PurchaseService {

	@Autowired
	PurchaseRepository purchaseRepository;

	@Autowired
	UserService userService;

	@Autowired
	ProductService productService;

	@Override
	public void create(int id, String email) {
		Purchase purchase = new Purchase(userService.find(email),
productService.findProduct(id));
		purchaseRepository.save(purchase);
	}

	@Override
	public List<Purchase> findByUser(int id) {
		return purchaseRepository.findByUserId(id);
	}

	@Override
	public List<Purchase> findByCategory(String name) {
		return purchaseRepository.findByProductCategoryName(name);
	}
```

```
	@Override
	public List<Purchase> findByDate(String date) throws ParseException {
		SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");
		Date formatedDate;
		try {
			formatedDate = dateFormat.parse(date);
			List<Purchase> purchases =
purchaseRepository.findByorderedDate(formatedDate);
			return purchases;
		} catch (ParseException e) {
			throw new ParseException(date, 0);
		}
	}

	@Override
	public List<Purchase> findAll() {

		return (List<Purchase>) purchaseRepository.findAll();
	}

}
package com.project.sportyshoes.service;

import java.util.List;

import javax.transaction.Transactional;

import com.project.sportyshoes.entity.User;
import com.project.sportyshoes.exception.DataNotFoundException;
import com.project.sportyshoes.exception.DuplicateIdException;

public interface UserService {

	@Transactional
	String create(User user) throws DuplicateIdException;

	User getUser(String email, String password) throws DataNotFoundException;

	List<User> loadAll();

	User find(String email);

}
package com.project.sportyshoes.service;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
```

```
import com.project.sportyshoes.entity.User;
import com.project.sportyshoes.exception.DataNotFoundException;
import com.project.sportyshoes.exception.DuplicateIdException;
import com.project.sportyshoes.repository.UserRepository;

@Service
public class UserServiceImpl implements UserService {

	@Autowired
	UserRepository userRepository;

	@Override
	public String create(User user) throws DuplicateIdException {
		if (userRepository.existsByEmail(user.getEmail())) {
			throw new DuplicateIdException(user.getEmail());
		} else {
			String name = userRepository.save(user).getName();
			return name;
		}
	}

	@Override
	public User getUser(String email, String password) throws
DataNotFoundException {
		User user = userRepository.findByEmailAndPassword(email, password);
		if (user == null) {
			throw new DataNotFoundException();
		} else {
			return user;
		}
	}

	@Override
	public List<User> loadAll() {
		List<User> users = (List<User>) userRepository.findAll();
		return users;
	}

	@Override
	public User find(String email) {
		return userRepository.findByEmail(email);
	}

}
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
	pageEncoding="ISO-8859-1"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
<head>
```

```
<meta charset="ISO-8859-1">
<title>User Dashboard</title>
<link rel="stylesheet"

href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css"

integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
        crossorigin="anonymous" />
</head>
<body>

        <div class="card ml-auto text-center fixed-top" data-toggle="modal"
                data-target="#exampleModalLong" style="width: 10rem">
                <div class="card-body">
                        <h6 class="card-title">${user.name }</h6>
                </div>
        </div>

        <div class="modal fade" id="exampleModalLong" tabindex="-1"
                role="dialog" aria-labelledby="exampleModalLongTitle"
                aria-hidden="true">
                <div class="modal-dialog" role="document">
                        <div class="modal-content">
                                <div class="modal-header">
                                        <h5 class="modal-title"
id="exampleModalLongTitle">User
                                                Details</h5>
                                        <button type="button" class="close"
data-dismiss="modal"
                                                aria-label="Close">
                                                <span
aria-hidden="true">&times;</span>
                                        </button>
                                </div>
                                <div class="model-body" style="padding: 10px">
                                        <h6>Name:${user.name}</h6>
                                        <h6>Name:${user.email}</h6>
                                </div>
                                <div class="modal-footer">
                                        <button type="button" class="btn
btn-secondary"
                                                data-dismiss="modal">Close</button>
                                        <a href="/user/logout" class="btn
btn-danger">Logout</a>
                                </div>
                        </div>
                </div>
        </div>

        <h2 class="m-2">Hii ${user.name} welcome to Sporty shoe</h2>
```

```
        <h3 class="m-3">Products</h3>
        <div class="d-flex p-2">
                <c:forEach var="product" items="${products}">
                        <div class="card m-2" style="width: 18rem;">
                                <div class="card-body">
                                        <h5 class="card-title">${product.name}</h5>
                                        <h6 class="card-subtitle mb-2
text-muted">${product.category.name}</h6>
                                        <p
class="card-text">${product.discription}</p>
                                        <a href="/purchase/buy/${product.id}"
class="card-link btn btn-primary btn-sm">Buy Product</a>
                                </div>
                        </div>
                </c:forEach>
        </div>

        <h3 class="m-3">Orders</h3>
        <div class="container mt-10 table-responsive">
      <table class="table table-hover">
        <thead>
          <tr>
            <th>Order Id</th>
            <th>Product name</th>
            <th>Product Category</th>
            <th>Product Price</th>
            <th>Purchased Date</th>
          </tr>
        </thead>
        <tbody>
         <c:forEach var="purchase" items="${purchases}">
                <tr>
                        <td>${purchase.purchase_id }</td>
                        <td>${purchase.product.id}</td>
                        <td>${purchase.product.category.name}</td>
                        <td>${purchase.product.price}</td>
                        <td>${purchase.ordered_date}</td>
                </tr>
         </c:forEach>
        </tbody>
      </table>
    </div>

        <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"

integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
                crossorigin="anonymous"></script>
        <script
```

```
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"

integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPIPm49"
                crossorigin="anonymous"></script>
        <script

src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js"

integrity="sha384-smHYKdLADwkXOn1EmN1qk/HfnUcbVRZyYmZ4qpPea6sjB/pTJ0euyQp0Mk8ck+5T"
                crossorigin="anonymous"></script>
</body>
</html>
<%@page import="java.util.List"%>
<%@page import="com.project.sportyshoes.entity.Product"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
        pageEncoding="ISO-8859-1" isELIgnored="false"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Admin Dashboad</title>
<link rel="stylesheet"

href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css"

integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
        crossorigin="anonymous" />
</head>
<body>
        <h5>${message }</h5>
        <h2 class="m-2">Hii ${admin.name} welcome to Sporty shoe</h2>

        <div class="card ml-auto fixed-top" style="width: 15rem">
                <div class="card-body">
                        <h6 class="card-title">${admin.name }</h6>
                        <h6 class="card-title">${admin.email }</h6>
                        <a href="#" data-toggle="modal"
data-target="#exampleModalLong"
                                class="text-primary">Change Password</a> <a
href="/admin/logout"
                                class="text-danger">Logout</a>
                </div>
        </div>

        <div class="modal fade" id="exampleModalLong" tabindex="-1"
                role="dialog" aria-labelledby="exampleModalLongTitle"
                aria-hidden="true">
```

```
                <div class="modal-dialog" role="document">
                        <div class="modal-content">
                                <div class="modal-header">
                                        <h5 class="modal-title"
id="exampleModalLongTitle">Admin
                                                Details</h5>
                                        <button type="button" class="close"
data-dismiss="modal"
                                                aria-label="Close">
                                                <span
aria-hidden="true">&times;</span>
                                        </button>
                                </div>
                                <div class="model-body" style="padding: 10px">
                                        <form action="/admin/changePassword"
method="post">
                                                <div class="form-group">
                                                        <label
for="oldPassword">Old Password</label> <input type="text"
                                                                name="oldPassword"
required="true"
                                                                placeholder="Enter
Old Password" id="oldPassword"

class="form-control" />
                                                </div>
                                                <div class="form-group">
                                                        <label for="password">New
Password</label> <input type="password"
                                                                name="newPassword"
required="true"
                                                                placeholder="Enter
new password" id="password"

class="form-control" />
                                                </div>
                                                <div class="form-group">
                                                        <input type="submit"
value="Change Password"
                                                                class="btn
btn-primary btn-block" />
                                                </div>
                                                <div class="form-group">
                                                        <input type="button"
value="Close"
                                                                class="btn
btn-secondary btn-block" data-dismiss="modal" />
                                                </div>
                                        </form>
                                </div>
```

```
                        </div>
                </div>
        </div>

        <h2 class="m-2 mt-5 mb-3 d-inline-block">Category</h2>
        <button type="button" class="btn btn-primary btn-sm mb-2"
                data-toggle="modal" data-target="#addCategory">Add Category
                &plus;</button>

        <div class="modal fade" id="addCategory" tabindex="-1" role="dialog"
                aria-labelledby="exampleModalLongTitle" aria-hidden="true">
                <div class="modal-dialog" role="document">
                        <div class="modal-content">
                                <div class="modal-header">
                                        <h5 class="modal-title"
id="exampleModalLongTitle">Add
                                                Category</h5>
                                        <button type="button" class="close"
data-dismiss="modal"
                                                aria-label="Close">
                                                <span
aria-hidden="true">&times;</span>
                                        </button>
                                </div>
                                <div class="model-body" style="padding: 10px">
                                        <form action="/category/create"
method="post">
                                                <div class="form-group">
                                                        <label
for="category_name">Name</label> <input type="text"
                                                                required="true"
name="name" id="category_name"
                                                                placeholder="Enter
Name" class="form-control" />
                                                </div>
                                                <div class="form-group">
                                                        <label
for="desc">Description</label> <input type="text"
                                                                name="discription"
required="true"
                                                                placeholder="Enter
description" id="desc" class="form-control" />
                                                </div>
                                                <div class="form-group">
                                                        <input type="submit"
value="Add Category"
                                                                class="btn
btn-primary btn-block" />
                                                </div>
                                                <div class="form-group">
```

```
                                                        <input type="button"
value="Close"
                                                                class="btn
btn-secondary btn-block" data-dismiss="modal" />
                                                </div>
                                        </form>
                                </div>
                        </div>
                </div>
        </div>
        <div class="container mt-10 table-responsive">
                <table class="table table-hover">
                        <tr>
                                <th>Category Id</th>
                                <th>Category Name</th>
                                <th>Category Description</th>
                                <th>Action</th>
                        </tr>
                        <c:forEach var="category" items="${categories}">
                                <tr>
                                        <td>${category.id}</td>
                                        <td>${category.name}</td>
                                        <td>${category.discription}</td>
                                        <td><a
href="/category/delete/${category.id}" class="text-danger">Delete</a></td>
                                </tr>
                        </c:forEach>
                </table>
        </div>

        <h2 class="m-2 mb-3 d-inline-block">Products</h2>
        <button type="button" class="btn btn-primary btn-sm mb-2"
                data-toggle="modal" data-target="#addProduct">Add Products
                &plus;</button>

        <div class="modal fade" id="addProduct" tabindex="-1" role="dialog"
                aria-labelledby="exampleModalLongTitle" aria-hidden="true">
                <div class="modal-dialog" role="document">
                        <div class="modal-content">
                                <div class="modal-header">
                                        <h5 class="modal-title"
id="exampleModalLongTitle">Add
                                                Category</h5>
                                        <button type="button" class="close"
data-dismiss="modal"
                                                aria-label="Close">
                                                <span
aria-hidden="true">&times;</span>
                                        </button>
                                </div>
```

```
                                <div class="model-body" style="padding: 10px">
                                        <form action="/product/create"
method="post">
                                                <div class="form-group">
                                                        <label
for="product_name">Name</label> <input type="text"
                                                                required="true"
name="name" id="product_name"
                                                                placeholder="Enter
Name" class="form-control" />
                                                </div>
                                                <div class="form-group">
                                                        <label
for="desc">Description</label> <input type="text"
                                                                name="discription"
required="true"
                                                                placeholder="Enter
description" id="desc" class="form-control" />
                                                </div>
                                                <div class="form-group">
                                                        <label
for="category">Category</label> <select required="true"

name="category_name" class="form-control">
                                                                <option
value="">Choose the category</option>
                                                                <c:forEach
var="category" items="${categories}">
                                                                        <option
value="${category.name}">${category.name}</option>
                                                                </c:forEach>
                                                        </select>
                                                </div>
                                                <div class="form-group">
                                                        <label
for="price">Price</label> <input type="number"
                                                                name="price"
required="true" placeholder="Enter price"
                                                                id="price"
class="form-control" />
                                                </div>

                                                <div class="form-group">
                                                        <input type="submit"
value="Add Product"
                                                                class="btn
btn-primary btn-block" />
                                                </div>
                                                <div class="form-group">
                                                        <input type="button"
```

```
value="Close"
                                                                    class="btn
btn-secondary btn-block" data-dismiss="modal" />
                                                  </div>
                                          </form>
                                  </div>
                          </div>
                  </div>
          </div>

          <div class="container mt-10 table-responsive">
                  <table class="table table-hover">
                          <tr>
                                  <th>Product Id</th>
                                  <th>Shoe Name</th>
                                  <th>Shoe Description</th>
                                  <th>Shoe price</th>
                                  <th colspan="2">Action</th>
                          </tr>
                          <c:forEach var="product" items="${products}">
                                  <tr>
                                          <td>${product.id}</td>
                                          <td>${product.name}</td>
                                          <td>${product.discription}</td>
                                          <td>${product.price}</td>
                                          <td><a
href="/product/update/${product.id}">Update</a></td>
                                          <td><a href="/product/delete/${product.id}"
class="text-danger">Delete</a></td>
                                  </tr>
                          </c:forEach>
                  </table>
          </div>


          <h2 class="m-2 mb-3">Users</h2>
          <div class="container mt-10 table-responsive">

                  <table class="table table-hover">
                          <tr>
                                  <th>User Id</th>
                                  <th>User Name</th>
                                  <th>User Email</th>
                          </tr>
                          <c:forEach var="user" items="${users}">
                                  <tr>
                                          <td>${user.id}</td>
                                          <td>${user.name}</td>
                                          <td>${user.email}</td>
                                  </tr>
```

```
                        </c:forEach>
                </table>
        </div>
        <h2 class="m-2 mb-3">Purchases</h2>
        <div class="container mt-10 d-flex justify-content-between">
                <form action="/admin/filter/category">
                        <div class="form-inline m-2 ml-10">
                                <label for="category" class="mr-2">Filter by
Category : </label> <select
                                        required="true" name="category_name"
class="form-control">
                                        <option value="">Choose the
category</option>
                                        <c:forEach var="category"
items="${categories}">
                                                <option
value="${category.name}">${category.name}</option>
                                        </c:forEach>
                                </select> <input type="submit" value="Filter
Category"
                                        class="btn btn-primary ml-2" />
                                <c:if test="${filteredPurchases != null }">
                                        <a href="/admin/dashboard" class="btn
btn-danger ml-2">Clear Filter</a>
                                </c:if>
                        </div>
                </form>
        </div>
        <div class="container mt-10 table-responsive">
                <table class="table table-hover">
                        <thead>
                                <tr>
                                        <th>Order Id</th>
                                        <th>User name</th>
                                        <th>Product name</th>
                                        <th>Product Category</th>
                                        <th>Product Price</th>
                                        <th>Purchased Date</th>
                                </tr>
                        </thead>
                        <tbody>
                                <c:if test="${filteredPurchases == null }">
                                        <c:forEach var="purchase"
items="${purchases}">
                                                <tr>
                                                        <td>${purchase.purchase_id
}</td>
                                                        <td>${purchase.user.name
}</td>
```

```
<td>${purchase.product.id}</td>

<td>${purchase.product.category.name}</td>

<td>${purchase.product.price}</td>

<td>${purchase.ordered_date}</td>
                                                </tr>
                                        </c:forEach>
                                </c:if>
                                <c:if test="${filteredPurchases != null }">
                                        <c:forEach var="purchase"
items="${filteredPurchases}">
                                                <tr>
                                                        <td>${purchase.purchase_id
}</td>
                                                        <td>${purchase.user.name
}</td>

<td>${purchase.product.id}</td>

<td>${purchase.product.category.name}</td>

<td>${purchase.product.price}</td>

<td>${purchase.ordered_date}</td>
                                                </tr>
                                        </c:forEach>
                                </c:if>
                        </tbody>
                </table>
        </div>

        <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"

integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
                crossorigin="anonymous"></script>
        <script

src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"

integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPIPm49"
                crossorigin="anonymous"></script>
        <script

src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js"

integrity="sha384-smHYKdLADwkXOn1EmN1qk/HfnUcbVRZyYmZ4qpPea6sjB/pTJ0euyQp0Mk8ck+5T"
                crossorigin="anonymous"></script>
</body>
```

```
</html>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
	pageEncoding="ISO-8859-1" isELIgnored="false"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Admin Login</title>
<link rel="stylesheet"

href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css"

integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
	crossorigin="anonymous" />
</head>
<body>
	<c:if test="${message != null}">
		<div class="alert alert-${type } alert-dismissible">
			<button class="close" type="button"
data-dismiss="alert">&times;</button>
			${message }
		</div>
	</c:if>
	<div
		class="mt-5 d-flex flex-column align-items-center
justify-content-conter">
		<h2>Admin Login</h2>
		<div class="container w-25">
			<form action="/admin/authenticate" method="post">
				<div class="form-group">
					<label for="email">Email</label> <input
type="email" name="email"
						required="true" placeholder="Enter
email" id="email"
						class="form-control" />
				</div>
				<div class="form-group">
					<label for="password">Password</label>
<input type="password"
						name="password" required="true"
placeholder="Enter password"
						id="password" class="form-control"
/>
				</div>
				<div class="form-group">
					<input type="submit" value="Login"
						class="btn btn-primary btn-block"
/>
```

```
                                </div>
                                <p class="text-center">Or</p>
                                <div class="form-group">
                                        <a href="/home" type="submit" value="Login"
                                                class="btn btn-secondary btn-block"
> Home</a>
                                </div>
                        </form>
                </div>
        </div>



        <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"

integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
                crossorigin="anonymous"></script>
        <script

src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"

integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPIPm49"
                crossorigin="anonymous"></script>
        <script

src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js"

integrity="sha384-smHYKdLADwkXOn1EmN1qk/HfnUcbVRZyYmZ4qpPea6sjB/pTJ0euyQp0Mk8ck+5T"
                crossorigin="anonymous"></script>
</body>
</html>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
        pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Home</title>
<link rel="stylesheet"

href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css"

integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
        crossorigin="anonymous" />
</head>
<body>

        <div class="d-flex align-items-center justify-content-center text-center"
style="height: 100vh; width: 100vw;">
```

```
                <div>
                        <h1 class="display-2 mb-5">Welcome to Sporty Shoe</h1>
                        <a href="/adminLogin" class="btn btn-primary">Admin
Login</a>
                        <a href="/user/login" class="btn btn-primary">User
Login</a>
                        <a href="/user/signUp" class="btn btn-primary">User
SignUp</a>
                </div>
        </div>


        <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"

integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
                crossorigin="anonymous"></script>
        <script

src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"

integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPIPm49"
                crossorigin="anonymous"></script>
        <script

src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js"

integrity="sha384-smHYKdLADwkXOn1EmN1qk/HfnUcbVRZyYmZ4qpPea6sjB/pTJ0euyQp0Mk8ck+5T"
                crossorigin="anonymous"></script>
</body>
</html>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
        pageEncoding="ISO-8859-1"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Sign Up user</title>
<link rel="stylesheet"

href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css"

integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
        crossorigin="anonymous" />
</head>
<body>
        <c:if test="${message != null}">
                <div class="alert alert-danger alert-dismissible text-center">
                        <button class="close" type="button"
data-dismiss="alert">&times;</button>
```

```
                        ${message } <strong><a href="/user/login">here</a></strong>
                </div>
        </c:if>
        <div
                class="mt-5 d-flex flex-column align-items-center
justify-content-conter mt-20"
                style="height: 100vh">
                <h2>SignUp User</h2>
                <div class="container w-25">
                        <form action="/user/create" method="post">
                                <div class="form-group">
                                        <label for="name">Name</label> <input
type="text" name="name"
                                                required="true" placeholder="Enter
name" id="name"
                                                class="form-control" />
                                </div>
                                <div class="form-group">
                                        <label for="email">Email</label> <input
type="email" name="email"
                                                required="true" placeholder="Enter
email" id="email"
                                                class="form-control" />
                                </div>
                                <div class="form-group">
                                        <label for="password">Password</label>
<input type="password"
                                                name="password" required="true"
placeholder="Enter password"
                                                id="password" class="form-control"
/>
                                </div>
                                <div class="form-group">
                                        <input type="submit" value="SignUp"
                                                class="btn btn-primary btn-block"
/>
                                </div>
                                <p class="text-center">Or</p>
                                <div class="form-group">
                                        <a href="/home" type="submit" value="Login"
                                                class="btn btn-secondary
btn-block"> Home</a>
                                </div>
                        </form>
                </div>
        </div>
```

```
        <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"

integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
                crossorigin="anonymous"></script>
        <script

src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"

integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPIPm49"
                crossorigin="anonymous"></script>
        <script

src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js"

integrity="sha384-smHYKdLADwkXOn1EmN1qk/HfnUcbVRZyYmZ4qpPea6sjB/pTJ0euyQp0Mk8ck+5T"
                crossorigin="anonymous"></script>
</body>
</html>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
        pageEncoding="ISO-8859-1"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Update Product</title>
<link rel="stylesheet"

href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css"

integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
        crossorigin="anonymous" />
</head>
<body>
        <h2 class="m-4 text-center">Update Product</h2>

        <div class="container w-50">
                <form action="/product/update" method="post">
                        <div class="form-group">
                                <label for="id">Id</label> <input type="text"
value="${product.id}"
                                        name="id" readonly="readonly" id="id"
class="form-control" />
                                <small id="filehelp" class="form-text
text-muted">You can't Edit</small>
                        </div>
                        <div class="form-group">
                                <label for="name">Name</label> <input type="text"
                                        value="${product.name}" required="required"
name="name" id="name"
```

```
                                        class="form-control" />
                        </div>
                        <div class="form-group">
                                <label for="discription">Description</label> <input
type="text"
                                        value="${product.discription}"
name="discription" id="discription"
                                        class="form-control" />
                        </div>
                        <div class="form-group">
                                <label for="category">Category</label> <select
required="required"
                                        name="category" id="category"
class="form-control">
                                        <option value="${product.category.id
}">${product.category.name}</option>
                                        <c:forEach var="category"
items="${categories}">
                                                <c:if test="${category.id !=
product.category.id }">
                                                        <option
value="${category.id }">${category.name}</option>
                                                </c:if>
                                        </c:forEach>
                                </select>
                        </div>
                        <div class="form-group">
                                <label for="price">Price</label> <input
type="number"
                                        name="price" value=${product.price }
required="required"
                                        id="price" class="form-control" />
                        </div>
                        <div class="form-group">
                                <input type="submit" value="Update"
                                        class="btn btn-primary btn-block" />
                        </div>
                </form>
        </div>
        <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"

integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
                crossorigin="anonymous"></script>
        <script

src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"

integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPIPm49"
                crossorigin="anonymous"></script>
        <script
```

```
src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js"

integrity="sha384-smHYKdLADwkXOn1EmN1qk/HfnUcbVRZyYmZ4qpPea6sjB/pTJ0euyQp0Mk8ck+5T"
                crossorigin="anonymous"></script>
</body>
</html>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
        pageEncoding="ISO-8859-1"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Sign Up user</title>
<link rel="stylesheet"

href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css"

integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
        crossorigin="anonymous" />
</head>
<body>
        <c:if test="${message != null}">
                <div class="alert alert-${type } alert-dismissible">
                        <button class="close" type="button"
data-dismiss="alert">&times;</button>
                        ${message }
                </div>
        </c:if>
        <div
                class="mt-5 d-flex flex-column align-items-center
justify-content-conter">
                <h2>User Login</h2>
                <div class="container w-25">
                        <form action="/user/authenticate" method="post">
                                <div class="form-group">
                                        <label for="email">Email</label> <input
type="email" name="email"
                                                required="true" placeholder="Enter
email" id="email"
                                                class="form-control" />
                                </div>
                                <div class="form-group">
                                        <label for="password">Password</label>
<input type="password"
                                                name="password" required="true"
placeholder="Enter password"
                                                id="password" class="form-control"
/>
```

```
                                </div>
                                <div class="form-group">
                                        <input type="submit" value="Login"
                                                class="btn btn-primary btn-block"
/>
                                </div>
                                <p class="text-center">Or</p>
                                <div class="form-group">
                                        <a href="/home" type="submit" value="Login"
                                                class="btn btn-secondary btn-block"
> Home</a>
                                </div>
                        </form>
                </div>
        </div>



        <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"

integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
                crossorigin="anonymous"></script>
        <script

src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"

integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPIPm49"
                crossorigin="anonymous"></script>
        <script

src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js"

integrity="sha384-smHYKdLADwkXOn1EmN1qk/HfnUcbVRZyYmZ4qpPea6sjB/pTJ0euyQp0Mk8ck+5T"
                crossorigin="anonymous"></script>
</body>
</html>
```