**Source Code:**

**Package name: com.simplilearn.controller**

**Class name: AdminController.java**

package com.simplilearn.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.Model;

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

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

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

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

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

import com.simplilearn.dao.OrderDAO;

import com.simplilearn.dao.ProductDAO;

import com.simplilearn.dao.UserDAO;

import com.simplilearn.entity.Order;

import com.simplilearn.entity.Product;

import com.simplilearn.entity.User;

@Controller

@RequestMapping("/admin")

public class AdminController {

@Autowired

private ProductDAO productDAO;

// inject

@Autowired

private UserDAO userDAO;

// inject

@Autowired

private OrderDAO orderDAO;

@GetMapping("/products")

public String products(Model theModel, HttpServletRequest request) {

HttpSession session = request.getSession(true);

User currentUser = (User) session.getAttribute("currentUser");

// get all products from DAO

List<Product> products = productDAO.getProducts();

// add the products to the model

theModel.addAttribute("currentUser", currentUser);

// add the products to the model

theModel.addAttribute("products", products);

return "mange-products";

}

@GetMapping("/deleteProduct")

public String deleteProduct(@RequestParam("productId") int productId, Model theModel) {

productDAO.deleteProduct(productId);

return "redirect:/admin/products";

}

@GetMapping("/deleteUser")

public String deleteUser(@RequestParam("userId") int userId, Model theModel) {

userDAO.deleteUser(userId);

return "redirect:/admin/manageUsers";

}

@GetMapping("/deleteOrder")

public String deleteOrder(@RequestParam("orderId") int ordereId, Model theModel) {

orderDAO.deleteOrder(ordereId);

return "redirect:/admin/manageOrders";

}

@GetMapping("/manageUsers")

public String manageUsers(Model theModel) {

// get all users from DAO

List<User> users = userDAO.getUsers();

// add the users to the model

theModel.addAttribute("users", users);

return "manageUsers";

}

@GetMapping("/manageOrders")

public String manageOrders(Model theModel) {

// get all users from DAO

List<Order> orders = orderDAO.getOrder();

int totalPrice = 0;

for (Order order : orders) {

totalPrice = totalPrice + order.getProduct().getPrice();

}

// add the users to the model

theModel.addAttribute("orders", orders);

// add the users to the model

theModel.addAttribute("totalPrice", totalPrice);

return "manageOrders";

}

@GetMapping("/addProdcut")

public String addProdcut() {

return "addproduct";

}

@PostMapping("/addProductProcess")

public String addProductProcess(HttpServletRequest request,

@RequestParam("name") String name,

@RequestParam("company") String company,

@RequestParam("size") int size,

@RequestParam("price") int price,

@RequestParam("image") String image

) {

Product product = new Product(name, company, size, price, image);

productDAO.saveProduct(product);

return "redirect:/admin/products";

}

@GetMapping("/updateProduct")

public String updateProduct(@RequestParam("productId") int productId , Model theModel) {

// get all products from DAO

Product product = productDAO.getProduct(productId);

// add the users to the model

theModel.addAttribute("product", product);

return "update-product";

}

@PostMapping("/updateProductProcess")

public String updateProductProcess(HttpServletRequest request,

@RequestParam("productId") int productId,

@RequestParam("name") String name,

@RequestParam("company") String company,

@RequestParam("size") int size,

@RequestParam("price") int price,

@RequestParam("image") String image

) {

Product product = productDAO.getProduct(productId);

product.setCompany(company);

product.setSize(size);

product.setName(name);

product.setPrice(price);

product.setImage\_link(image);

productDAO.saveProduct(product);

return "redirect:/admin/products";

}

}

**Class name: HomepageController.java**

package com.simplilearn.controller;

import java.util.ArrayList;

import java.util.List;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

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

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

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

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

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

import com.simplilearn.dao.OrderDAO;

import com.simplilearn.dao.ProductDAO;

import com.simplilearn.entity.Order;

import com.simplilearn.entity.Product;

import com.simplilearn.entity.User;

@Controller

@RequestMapping("/homepage")

public class HomePageController {

// need to inject

@Autowired

private ProductDAO productDAO;

@Autowired

private OrderDAO orderDAO;

@GetMapping("/products")

public String products(Model theModel, HttpServletRequest request) {

HttpSession session = request.getSession(true);

User currentUser = (User) session.getAttribute("currentUser");

// get all products from DAO

List<Product> products = productDAO.getProducts();

List<Order> userOrders = orderDAO.getUserOrders(currentUser.getId());

List<Integer> userProducts = new ArrayList<Integer>();

for (Order order : userOrders) {

userProducts.add(order.getProduct().getId());

}

// add the products to the model

theModel.addAttribute("products", products);

theModel.addAttribute("userProducts", userProducts);

theModel.addAttribute("currentUser", currentUser);

return "user-home";

}

@GetMapping("/orderProcess")

public String orderProcess(Model theModel,@RequestParam("productId") int productId, HttpServletRequest request) {

HttpSession session = request.getSession(true);

User currentUser = (User) session.getAttribute("currentUser");

Product product = productDAO.getProduct(productId);

Order order = new Order(currentUser, product);

orderDAO.saveOrder(order);

theModel.addAttribute("currentUser", currentUser);

return "redirect:/homepage/products";

}

@GetMapping("/mycart")

public String showMyCart(Model theModel, HttpServletRequest request) {

HttpSession session = request.getSession(true);

User currentUser = (User) session.getAttribute("currentUser");

// get all products from DAO

List<Order> userOrders = orderDAO.getUserOrders(currentUser.getId());

List<Product> userProducts = new ArrayList<Product>();

for (Order order : userOrders) {

userProducts.add(order.getProduct());

}

int total\_price = 0;

for (Product product : userProducts) {

total\_price = total\_price + product.getPrice();

}

theModel.addAttribute("userProducts", userProducts);

theModel.addAttribute("total\_price", total\_price);

theModel.addAttribute("currentUser", currentUser);

return "mycart";

}

@PostMapping("/searchProducts")

public String searchProducts(HttpServletRequest request, Model theModel, @RequestParam("keySearch") String key) {

HttpSession session = request.getSession(true);

User currentUser = (User) session.getAttribute("currentUser");

System.out.println(key);

// get all products from DAO

List<Product> products = productDAO.searchProducts(key);

if (currentUser.getType() == 0) {

// get all products from DAO

List<Order> userOrders = orderDAO.getUserOrders(currentUser.getId());

List<Integer> userProducts = new ArrayList<Integer>();

for (Order order : userOrders) {

userProducts.add(order.getProduct().getId());

}

theModel.addAttribute("userProducts", userProducts);

theModel.addAttribute("currentUser", currentUser);

}

// add the products to the model

theModel.addAttribute("products", products);

if (currentUser.getType() == 0) {

return "user-home";

}

else {

return "mange-products";

}

}

}

**Class name: LoginController.java**

package com.simplilearn.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.Model;

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

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

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

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

import com.simplilearn.dao.ProductDAO;

import com.simplilearn.dao.UserDAO;

import com.simplilearn.entity.Product;

import com.simplilearn.entity.User;

@Controller

public class LoginController {

// inject

@Autowired

private UserDAO userDAO;

@RequestMapping("/login")

public String login(Model theModel) {

return "login";

}

@RequestMapping("/register")

public String register(Model theModel) {

return "register";

}

@PostMapping("/loginProcess")

public String loginProcess(HttpServletRequest request, @RequestParam("username") String username,

@RequestParam("password") String password) {

HttpSession session = request.getSession(true);

// get all users from DAO

List<User> users = userDAO.getUsers();

boolean isusesr = false;

User tempUser = null;

for (User user : users) {

if (user.getUsername().equals(username) && user.getPassword().equals(password)) {

isusesr = true;

tempUser = user;

break;

}

}

if (isusesr && tempUser.getType() == 0) {

session.setAttribute("currentUser", tempUser);

return "redirect:/homepage/products";

}

else if (isusesr && tempUser.getType() == 1) {

session.setAttribute("currentUser", tempUser);

return "redirect:/admin/products";

}

else

return "login";

}

@PostMapping("/registerProcess")

public String registerProcess(@RequestParam("username") String username,

@RequestParam("age") String age,

@RequestParam("password") String password) {

User user = new User(username, password, Integer.parseInt(age));

userDAO.saveUser(user);

return "login";

}

}

**Class name: UserController.java**

package com.simplilearn.controller;

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.Model;

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

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

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

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

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

import com.simplilearn.dao.ProductDAO;

import com.simplilearn.dao.UserDAO;

import com.simplilearn.entity.Product;

import com.simplilearn.entity.User;

@Controller

@RequestMapping("/user")

public class UserController {

@Autowired

private ProductDAO productDAO;

// inject

@Autowired

private UserDAO userDAO;

@GetMapping("/myaccount")

public String myaccount(Model theModel, HttpServletRequest request) {

HttpSession session = request.getSession(true);

User currentUser = (User) session.getAttribute("currentUser");

// add the products to the model

theModel.addAttribute("currentUser", currentUser);

return "myaccount";

}

@PostMapping("/updateAcount")

public String updateAcount(HttpServletRequest request,

@RequestParam("username") String username,

@RequestParam("age") String age,

@RequestParam("password") String password , Model theModel

) {

HttpSession session = request.getSession(true);

User currentUser = (User) session.getAttribute("currentUser");

System.out.println(currentUser.toString());

currentUser.setUsername(username);

currentUser.setAge(Integer.parseInt(age));

currentUser.setPassword(password);

theModel.addAttribute("currentUser", currentUser);

userDAO.saveUser(currentUser);

return "myaccount";

}

}

**Package name: com.simplilearn.dao**

**Class name: OrderDAO.java**

package com.simplilearn.dao;

import java.util.List;

import com.simplilearn.entity.Order;

import com.simplilearn.entity.Product;

import com.simplilearn.entity.User;

public interface OrderDAO {

public List<Order> getOrder();

public Order getOrder(int orderId);

public void saveOrder(Order order);

public void deleteOrder(int orderId);

public List<Order> getUserOrders(int userId);

}

**Class name: OrderDAOlmpl.java**

package com.simplilearn.dao;

import java.util.List;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.query.Query;

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

import org.springframework.stereotype.Repository;

import org.springframework.transaction.annotation.Transactional;

import com.simplilearn.entity.Order;

import com.simplilearn.entity.User;

@Repository

public class OrderDAOImpl implements OrderDAO {

// need to inject the session factory

@Autowired

private SessionFactory sessionFactory;

@Override

@Transactional

public List<Order> getOrder() {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// create a query ... sort by last name

Query<Order> theQuery = currentSession.createQuery("from Order", Order.class);

// execute query and get result list

List<Order> orders = theQuery.getResultList();

// return the results

return orders;

}

@Override

@Transactional

public void saveOrder(Order order) {

// get current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// save/update the customer

currentSession.saveOrUpdate(order);

}

@Override

@Transactional

public Order getOrder(int orderId) {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// now retrieve/read from database using the primary key

Order order = currentSession.get(Order.class, orderId);

return order;

}

@Override

@Transactional

public void deleteOrder(int orderId) {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// delete object with primary key

Query theQuery = currentSession.createQuery("delete from Order where id=:orderId");

theQuery.setParameter("orderId", orderId);

theQuery.executeUpdate();

}

@Override

@Transactional

public List<Order> getUserOrders(int userId) {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// create a query ... sort by last name

Query<Order> theQuery = currentSession.createQuery("from Order where user\_id =: userId", Order.class);

theQuery.setParameter("userId", userId);

// execute query and get result list

List<Order> orders = theQuery.getResultList();

// return the results

return orders;

}

}

**Class name: ProductDAO.java**

package com.simplilearn.dao;

import java.util.List;

import com.simplilearn.entity.Order;

import com.simplilearn.entity.Product;

public interface ProductDAO {

public List<Product> getProducts();

public Product getProduct(int productId);

public void saveProduct(Product product);

public void deleteProduct(int productId);

public List<Product> searchProducts(String key);

}

**Class name: ProductDAOlmpl.java**

package com.simplilearn.dao;

import java.util.List;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.query.Query;

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

import org.springframework.stereotype.Repository;

import org.springframework.transaction.annotation.Transactional;

import com.simplilearn.entity.Order;

import com.simplilearn.entity.Product;

@Repository

public class ProductDAOImpl implements ProductDAO {

// need to inject the session factory

@Autowired

private SessionFactory sessionFactory;

@Override

@Transactional

public List<Product> getProducts() {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// create a query

Query<Product> theQuery = currentSession.createQuery("from Product", Product.class);

// execute query and get result list

List<Product> products = theQuery.getResultList();

// return the results

return products;

}

@Override

@Transactional

public Product getProduct(int productId) {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// now retrieve/read from database using the primary key

Product prod = currentSession.get(Product.class, productId);

return prod;

}

@Override

@Transactional

public void saveProduct(Product product) {

// get current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// save/update

currentSession.saveOrUpdate(product);

}

@Override

@Transactional

public void deleteProduct(int productId) {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// delete object with primary key

Query theQuery = currentSession.createQuery("delete from Product where id=:productId");

theQuery.setParameter("productId", productId);

theQuery.executeUpdate();

}

@Override

@Transactional

public List<Product> searchProducts(String key) {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// create a query

Query<Product> theQuery = currentSession.createQuery("from Product where name like: key", Product.class);

theQuery.setParameter("key", "%" + key + "%");

// execute query and get result list

List<Product> products = theQuery.getResultList();

System.out.println();

// return the results

return products;

}

}

**Class name: UserDAO.java**

package com.simplilearn.dao;

import java.util.List;

import com.simplilearn.entity.User;

public interface UserDAO {

public List<User> getUsers();

public User getUser(int user);

public void saveUser(User user);

public void deleteUser(int user);

}

**Class name: UserDAOlmpl.java**

package com.simplilearn.dao;

import java.util.List;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.query.Query;

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

import org.springframework.stereotype.Repository;

import org.springframework.transaction.annotation.Transactional;

import com.simplilearn.entity.User;

@Repository

public class UserDAOImpl implements UserDAO {

// need to inject the session factory

@Autowired

private SessionFactory sessionFactory;

@Override

@Transactional

public List<User> getUsers() {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// create a query ... sort by last name

Query<User> theQuery = currentSession.createQuery("from User", User.class);

// execute query and get result list

List<User> users = theQuery.getResultList();

// return the results

return users;

}

@Override

@Transactional

public User getUser(int userId) {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// now retrieve/read from database using the primary key

User user = currentSession.get(User.class, userId);

return user;

}

@Override

@Transactional

public void saveUser(User user) {

// get current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// save/update the customer

currentSession.saveOrUpdate(user);

}

@Override

@Transactional

public void deleteUser(int userId) {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// delete object with primary key

Query theQuery =

currentSession.createQuery("delete from User where id=:userId");

theQuery.setParameter("userId", userId);

theQuery.executeUpdate();

}

}

**Package name: com.simplilearn.entity**

**Class name: Order.java**

package com.simplilearn.entity;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.FetchType;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.JoinColumn;

import javax.persistence.ManyToOne;

import javax.persistence.Table;

@Entity

@Table(name="orders")

public class Order {

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

@Column(name="id")

private int id;

@ManyToOne

@JoinColumn(name="user\_id")

private User user;

@ManyToOne

@JoinColumn(name="product\_id")

private Product product;

public Order() {

}

public Order(User user, Product product) {

super();

this.user = user;

this.product = product;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

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;

}

@Override

public String toString() {

return "Order [id=" + id + ", user=" + user + ", product=" + product + "]";

}

}

**Class name: Product.java**

package com.simplilearn.entity;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

@Entity

@Table(name="products")

public class Product {

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

@Column(name="id")

private int id;

@Column(name="name")

private String name;

@Column(name="company")

private String company;

@Column(name="size")

private int size;

@Column(name="price")

private int price;

@Column(name="image\_link")

private String image\_link;

public Product() {

}

public Product(String name, String company, int size, int price, String image\_link) {

super();

this.name = name;

this.company = company;

this.size = size;

this.price = price;

this.image\_link = image\_link;

}

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 getCompany() {

return company;

}

public void setCompany(String company) {

this.company = company;

}

public int getSize() {

return size;

}

public void setSize(int size) {

this.size = size;

}

public int getPrice() {

return price;

}

public void setPrice(int price) {

this.price = price;

}

public String getImage\_link() {

return image\_link;

}

public void setImage\_link(String image\_link) {

this.image\_link = image\_link;

}

@Override

public String toString() {

return "Product [id=" + id + ", name=" + name + ", company=" + company + ", size=" + size + ", price=" + price

+ ", image\_link=" + image\_link + "]";

}

}

**Class name: User.java**

package com.simplilearn.entity;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

import javax.validation.constraints.Max;

import javax.validation.constraints.NotNull;

import javax.validation.constraints.Pattern;

import org.hibernate.validator.constraints.Range;

@Entity

@Table(name="users")

public class User {

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

@Column(name="id")

private int id;

@Column(name="type")

private int type;

@Column(name="username")

private String username;

@Column(name="password")

private String password;

@Range(min=1, max=120,message = "Invalied Age")

@Column(name="age")

private int age;

public User() {

}

public User(String username, String password, int age) {

super();

this.username = username;

this.password = password;

this.age = age;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public int getType() {

return type;

}

public void setType(int type) {

this.type = type;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

@Override

public String toString() {

return "User [id=" + id + ", type=" + type + ", username=" + username + ", password=" + password + ", age="

+ age + "]";

}

}

**Folder name:**

**Src->main->java->com->simpilearn->controller**

**Class name: AdminController.java**

package com.simplilearn.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.Model;

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

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

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

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

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

import com.simplilearn.dao.OrderDAO;

import com.simplilearn.dao.ProductDAO;

import com.simplilearn.dao.UserDAO;

import com.simplilearn.entity.Order;

import com.simplilearn.entity.Product;

import com.simplilearn.entity.User;

@Controller

@RequestMapping("/admin")

public class AdminController {

@Autowired

private ProductDAO productDAO;

// inject

@Autowired

private UserDAO userDAO;

// inject

@Autowired

private OrderDAO orderDAO;

@GetMapping("/products")

public String products(Model theModel, HttpServletRequest request) {

HttpSession session = request.getSession(true);

User currentUser = (User) session.getAttribute("currentUser");

// get all products from DAO

List<Product> products = productDAO.getProducts();

// add the products to the model

theModel.addAttribute("currentUser", currentUser);

// add the products to the model

theModel.addAttribute("products", products);

return "mange-products";

}

@GetMapping("/deleteProduct")

public String deleteProduct(@RequestParam("productId") int productId, Model theModel) {

productDAO.deleteProduct(productId);

return "redirect:/admin/products";

}

@GetMapping("/deleteUser")

public String deleteUser(@RequestParam("userId") int userId, Model theModel) {

userDAO.deleteUser(userId);

return "redirect:/admin/manageUsers";

}

@GetMapping("/deleteOrder")

public String deleteOrder(@RequestParam("orderId") int ordereId, Model theModel) {

orderDAO.deleteOrder(ordereId);

return "redirect:/admin/manageOrders";

}

@GetMapping("/manageUsers")

public String manageUsers(Model theModel) {

// get all users from DAO

List<User> users = userDAO.getUsers();

// add the users to the model

theModel.addAttribute("users", users);

return "manageUsers";

}

@GetMapping("/manageOrders")

public String manageOrders(Model theModel) {

// get all users from DAO

List<Order> orders = orderDAO.getOrder();

int totalPrice = 0;

for (Order order : orders) {

totalPrice = totalPrice + order.getProduct().getPrice();

}

// add the users to the model

theModel.addAttribute("orders", orders);

// add the users to the model

theModel.addAttribute("totalPrice", totalPrice);

return "manageOrders";

}

@GetMapping("/addProdcut")

public String addProdcut() {

return "addproduct";

}

@PostMapping("/addProductProcess")

public String addProductProcess(HttpServletRequest request,

@RequestParam("name") String name,

@RequestParam("company") String company,

@RequestParam("size") int size,

@RequestParam("price") int price,

@RequestParam("image") String image

) {

Product product = new Product(name, company, size, price, image);

productDAO.saveProduct(product);

return "redirect:/admin/products";

}

@GetMapping("/updateProduct")

public String updateProduct(@RequestParam("productId") int productId , Model theModel) {

// get all products from DAO

Product product = productDAO.getProduct(productId);

// add the users to the model

theModel.addAttribute("product", product);

return "update-product";

}

@PostMapping("/updateProductProcess")

public String updateProductProcess(HttpServletRequest request,

@RequestParam("productId") int productId,

@RequestParam("name") String name,

@RequestParam("company") String company,

@RequestParam("size") int size,

@RequestParam("price") int price,

@RequestParam("image") String image

) {

Product product = productDAO.getProduct(productId);

product.setCompany(company);

product.setSize(size);

product.setName(name);

product.setPrice(price);

product.setImage\_link(image);

productDAO.saveProduct(product);

return "redirect:/admin/products";

}

}

**Class name: HomePageController.java**

package com.simplilearn.controller;

import java.util.ArrayList;

import java.util.List;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

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

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

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

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

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

import com.simplilearn.dao.OrderDAO;

import com.simplilearn.dao.ProductDAO;

import com.simplilearn.entity.Order;

import com.simplilearn.entity.Product;

import com.simplilearn.entity.User;

@Controller

@RequestMapping("/homepage")

public class HomePageController {

// need to inject

@Autowired

private ProductDAO productDAO;

@Autowired

private OrderDAO orderDAO;

@GetMapping("/products")

public String products(Model theModel, HttpServletRequest request) {

HttpSession session = request.getSession(true);

User currentUser = (User) session.getAttribute("currentUser");

// get all products from DAO

List<Product> products = productDAO.getProducts();

List<Order> userOrders = orderDAO.getUserOrders(currentUser.getId());

List<Integer> userProducts = new ArrayList<Integer>();

for (Order order : userOrders) {

userProducts.add(order.getProduct().getId());

}

// add the products to the model

theModel.addAttribute("products", products);

theModel.addAttribute("userProducts", userProducts);

theModel.addAttribute("currentUser", currentUser);

return "user-home";

}

@GetMapping("/orderProcess")

public String orderProcess(Model theModel,@RequestParam("productId") int productId, HttpServletRequest request) {

HttpSession session = request.getSession(true);

User currentUser = (User) session.getAttribute("currentUser");

Product product = productDAO.getProduct(productId);

Order order = new Order(currentUser, product);

orderDAO.saveOrder(order);

theModel.addAttribute("currentUser", currentUser);

return "redirect:/homepage/products";

}

@GetMapping("/mycart")

public String showMyCart(Model theModel, HttpServletRequest request) {

HttpSession session = request.getSession(true);

User currentUser = (User) session.getAttribute("currentUser");

// get all products from DAO

List<Order> userOrders = orderDAO.getUserOrders(currentUser.getId());

List<Product> userProducts = new ArrayList<Product>();

for (Order order : userOrders) {

userProducts.add(order.getProduct());

}

int total\_price = 0;

for (Product product : userProducts) {

total\_price = total\_price + product.getPrice();

}

theModel.addAttribute("userProducts", userProducts);

theModel.addAttribute("total\_price", total\_price);

theModel.addAttribute("currentUser", currentUser);

return "mycart";

}

@PostMapping("/searchProducts")

public String searchProducts(HttpServletRequest request, Model theModel, @RequestParam("keySearch") String key) {

HttpSession session = request.getSession(true);

User currentUser = (User) session.getAttribute("currentUser");

System.out.println(key);

// get all products from DAO

List<Product> products = productDAO.searchProducts(key);

if (currentUser.getType() == 0) {

// get all products from DAO

List<Order> userOrders = orderDAO.getUserOrders(currentUser.getId());

List<Integer> userProducts = new ArrayList<Integer>();

for (Order order : userOrders) {

userProducts.add(order.getProduct().getId());

}

theModel.addAttribute("userProducts", userProducts);

theModel.addAttribute("currentUser", currentUser);

}

// add the products to the model

theModel.addAttribute("products", products);

if (currentUser.getType() == 0) {

return "user-home";

}

else {

return "mange-products";

}

}

}

**Class name: LoginController.java**

package com.simplilearn.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.Model;

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

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

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

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

import com.simplilearn.dao.ProductDAO;

import com.simplilearn.dao.UserDAO;

import com.simplilearn.entity.Product;

import com.simplilearn.entity.User;

@Controller

public class LoginController {

// inject

@Autowired

private UserDAO userDAO;

@RequestMapping("/login")

public String login(Model theModel) {

return "login";

}

@RequestMapping("/register")

public String register(Model theModel) {

return "register";

}

@PostMapping("/loginProcess")

public String loginProcess(HttpServletRequest request, @RequestParam("username") String username,

@RequestParam("password") String password) {

HttpSession session = request.getSession(true);

// get all users from DAO

List<User> users = userDAO.getUsers();

boolean isusesr = false;

User tempUser = null;

for (User user : users) {

if (user.getUsername().equals(username) && user.getPassword().equals(password)) {

isusesr = true;

tempUser = user;

break;

}

}

if (isusesr && tempUser.getType() == 0) {

session.setAttribute("currentUser", tempUser);

return "redirect:/homepage/products";

}

else if (isusesr && tempUser.getType() == 1) {

session.setAttribute("currentUser", tempUser);

return "redirect:/admin/products";

}

else

return "login";

}

@PostMapping("/registerProcess")

public String registerProcess(@RequestParam("username") String username,

@RequestParam("age") String age,

@RequestParam("password") String password) {

User user = new User(username, password, Integer.parseInt(age));

userDAO.saveUser(user);

return "login";

}

}

**Class name: UserController.java**

package com.simplilearn.controller;

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.Model;

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

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

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

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

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

import com.simplilearn.dao.ProductDAO;

import com.simplilearn.dao.UserDAO;

import com.simplilearn.entity.Product;

import com.simplilearn.entity.User;

@Controller

@RequestMapping("/user")

public class UserController {

@Autowired

private ProductDAO productDAO;

// inject

@Autowired

private UserDAO userDAO;

@GetMapping("/myaccount")

public String myaccount(Model theModel, HttpServletRequest request) {

HttpSession session = request.getSession(true);

User currentUser = (User) session.getAttribute("currentUser");

// add the products to the model

theModel.addAttribute("currentUser", currentUser);

return "myaccount";

}

@PostMapping("/updateAcount")

public String updateAcount(HttpServletRequest request,

@RequestParam("username") String username,

@RequestParam("age") String age,

@RequestParam("password") String password , Model theModel

) {

HttpSession session = request.getSession(true);

User currentUser = (User) session.getAttribute("currentUser");

System.out.println(currentUser.toString());

currentUser.setUsername(username);

currentUser.setAge(Integer.parseInt(age));

currentUser.setPassword(password);

theModel.addAttribute("currentUser", currentUser);

userDAO.saveUser(currentUser);

return "myaccount";

}

}

**Folder name: dao**

**Class name:OrderDAO.java**

package com.simplilearn.dao;

import java.util.List;

import com.simplilearn.entity.Order;

import com.simplilearn.entity.Product;

import com.simplilearn.entity.User;

public interface OrderDAO {

public List<Order> getOrder();

public Order getOrder(int orderId);

public void saveOrder(Order order);

public void deleteOrder(int orderId);

public List<Order> getUserOrders(int userId);

}

**Class name:OrderDAOlmpl.java**

package com.simplilearn.dao;

import java.util.List;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.query.Query;

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

import org.springframework.stereotype.Repository;

import org.springframework.transaction.annotation.Transactional;

import com.simplilearn.entity.Order;

import com.simplilearn.entity.User;

@Repository

public class OrderDAOImpl implements OrderDAO {

// need to inject the session factory

@Autowired

private SessionFactory sessionFactory;

@Override

@Transactional

public List<Order> getOrder() {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// create a query ... sort by last name

Query<Order> theQuery = currentSession.createQuery("from Order", Order.class);

// execute query and get result list

List<Order> orders = theQuery.getResultList();

// return the results

return orders;

}

@Override

@Transactional

public void saveOrder(Order order) {

// get current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// save/update the customer

currentSession.saveOrUpdate(order);

}

@Override

@Transactional

public Order getOrder(int orderId) {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// now retrieve/read from database using the primary key

Order order = currentSession.get(Order.class, orderId);

return order;

}

@Override

@Transactional

public void deleteOrder(int orderId) {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// delete object with primary key

Query theQuery = currentSession.createQuery("delete from Order where id=:orderId");

theQuery.setParameter("orderId", orderId);

theQuery.executeUpdate();

}

@Override

@Transactional

public List<Order> getUserOrders(int userId) {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// create a query ... sort by last name

Query<Order> theQuery = currentSession.createQuery("from Order where user\_id =: userId", Order.class);

theQuery.setParameter("userId", userId);

// execute query and get result list

List<Order> orders = theQuery.getResultList();

// return the results

return orders;

}

}

**Class name:ProductDAO.java**

package com.simplilearn.dao;

import java.util.List;

import com.simplilearn.entity.Order;

import com.simplilearn.entity.Product;

public interface ProductDAO {

public List<Product> getProducts();

public Product getProduct(int productId);

public void saveProduct(Product product);

public void deleteProduct(int productId);

public List<Product> searchProducts(String key);

}

**Class name:ProductDAOlmpl.java**

package com.simplilearn.dao;

import java.util.List;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.query.Query;

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

import org.springframework.stereotype.Repository;

import org.springframework.transaction.annotation.Transactional;

import com.simplilearn.entity.Order;

import com.simplilearn.entity.Product;

@Repository

public class ProductDAOImpl implements ProductDAO {

// need to inject the session factory

@Autowired

private SessionFactory sessionFactory;

@Override

@Transactional

public List<Product> getProducts() {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// create a query

Query<Product> theQuery = currentSession.createQuery("from Product", Product.class);

// execute query and get result list

List<Product> products = theQuery.getResultList();

// return the results

return products;

}

@Override

@Transactional

public Product getProduct(int productId) {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// now retrieve/read from database using the primary key

Product prod = currentSession.get(Product.class, productId);

return prod;

}

@Override

@Transactional

public void saveProduct(Product product) {

// get current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// save/update

currentSession.saveOrUpdate(product);

}

@Override

@Transactional

public void deleteProduct(int productId) {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// delete object with primary key

Query theQuery = currentSession.createQuery("delete from Product where id=:productId");

theQuery.setParameter("productId", productId);

theQuery.executeUpdate();

}

@Override

@Transactional

public List<Product> searchProducts(String key) {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// create a query

Query<Product> theQuery = currentSession.createQuery("from Product where name like: key", Product.class);

theQuery.setParameter("key", "%" + key + "%");

// execute query and get result list

List<Product> products = theQuery.getResultList();

System.out.println();

// return the results

return products;

}

}

**Class name:UserDAO.java**

package com.simplilearn.dao;

import java.util.List;

import com.simplilearn.entity.User;

public interface UserDAO {

public List<User> getUsers();

public User getUser(int user);

public void saveUser(User user);

public void deleteUser(int user);

}

**Class name:UserDAOlmpl.java**

package com.simplilearn.dao;

import java.util.List;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.query.Query;

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

import org.springframework.stereotype.Repository;

import org.springframework.transaction.annotation.Transactional;

import com.simplilearn.entity.User;

@Repository

public class UserDAOImpl implements UserDAO {

// need to inject the session factory

@Autowired

private SessionFactory sessionFactory;

@Override

@Transactional

public List<User> getUsers() {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// create a query ... sort by last name

Query<User> theQuery = currentSession.createQuery("from User", User.class);

// execute query and get result list

List<User> users = theQuery.getResultList();

// return the results

return users;

}

@Override

@Transactional

public User getUser(int userId) {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// now retrieve/read from database using the primary key

User user = currentSession.get(User.class, userId);

return user;

}

@Override

@Transactional

public void saveUser(User user) {

// get current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// save/update the customer

currentSession.saveOrUpdate(user);

}

@Override

@Transactional

public void deleteUser(int userId) {

// get the current hibernate session

Session currentSession = sessionFactory.getCurrentSession();

// delete object with primary key

Query theQuery =

currentSession.createQuery("delete from User where id=:userId");

theQuery.setParameter("userId", userId);

theQuery.executeUpdate();

}

}

**Folder name: entity**

**Class name: Order.java**

package com.simplilearn.entity;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.FetchType;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.JoinColumn;

import javax.persistence.ManyToOne;

import javax.persistence.Table;

@Entity

@Table(name="orders")

public class Order {

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

@Column(name="id")

private int id;

@ManyToOne

@JoinColumn(name="user\_id")

private User user;

@ManyToOne

@JoinColumn(name="product\_id")

private Product product;

public Order() {

}

public Order(User user, Product product) {

super();

this.user = user;

this.product = product;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

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;

}

@Override

public String toString() {

return "Order [id=" + id + ", user=" + user + ", product=" + product + "]";

}

}

**Class name: Product.java**

package com.simplilearn.entity;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

@Entity

@Table(name="products")

public class Product {

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

@Column(name="id")

private int id;

@Column(name="name")

private String name;

@Column(name="company")

private String company;

@Column(name="size")

private int size;

@Column(name="price")

private int price;

@Column(name="image\_link")

private String image\_link;

public Product() {

}

public Product(String name, String company, int size, int price, String image\_link) {

super();

this.name = name;

this.company = company;

this.size = size;

this.price = price;

this.image\_link = image\_link;

}

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 getCompany() {

return company;

}

public void setCompany(String company) {

this.company = company;

}

public int getSize() {

return size;

}

public void setSize(int size) {

this.size = size;

}

public int getPrice() {

return price;

}

public void setPrice(int price) {

this.price = price;

}

public String getImage\_link() {

return image\_link;

}

public void setImage\_link(String image\_link) {

this.image\_link = image\_link;

}

@Override

public String toString() {

return "Product [id=" + id + ", name=" + name + ", company=" + company + ", size=" + size + ", price=" + price

+ ", image\_link=" + image\_link + "]";

}

}

**Class name: User.java**

package com.simplilearn.entity;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

import javax.validation.constraints.Max;

import javax.validation.constraints.NotNull;

import javax.validation.constraints.Pattern;

import org.hibernate.validator.constraints.Range;

@Entity

@Table(name="users")

public class User {

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

@Column(name="id")

private int id;

@Column(name="type")

private int type;

@Column(name="username")

private String username;

@Column(name="password")

private String password;

@Range(min=1, max=120,message = "Invalied Age")

@Column(name="age")

private int age;

public User() {

}

public User(String username, String password, int age) {

super();

this.username = username;

this.password = password;

this.age = age;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public int getType() {

return type;

}

public void setType(int type) {

this.type = type;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

@Override

public String toString() {

return "User [id=" + id + ", type=" + type + ", username=" + username + ", password=" + password + ", age="

+ age + "]";

}

}

**Webapp->META-INF->resource->bootstrap->css->login.css**

Body {

font-family: Calibri, Helvetica, sans-serif;

background-image: url('../images/background.jpg');

}

button {

justify-content: center;

background-color: #4CAF50;

width: 100%;

color: white;

padding: 15px;

margin: 10px 0px;

border: none;

cursor: pointer;

}

form {

border: 1.4px solid black;

width: 45%;

margin: 0 auto;

}

input[type=text], input[type=password] {

justify-content: center;

width: 100%;

margin: 8px 0;

padding: 12px 20px;

display: inline-block;

border: 2px solid green;

box-sizing: border-box;

}

button:hover {

opacity: 0.7;

}

.container {

justify-content: center;

padding: 15px;

background-color: #FFF8DC;

}

**Style.css**

html, body{

padding:0px;

font-family:Verdana, Arial, Helvetica, sans-serif;

min-height: 100%;

height: 100%;

}

.flex-fill {

flex:1;

}

.sidenav {

border-color: #FFFFFF;

position: fixed;

z-index: 1;

top: 0;

left: 0;

background-color: #000080;

overflow-x: hidden;

padding-top: 20px;

min-height: 100%;

height: 100%;

}

.sidenav a {

padding: 6px 6px 6px 32px;

text-decoration: none;

font-size: 25px;

color: white;

display: block;

}

.sidenav a:hover {

color: blue;

}

#page{

height: 100%;

margin-left: 13%;

}

#logo{

font-family: 'Trebuchet MS', sans-serif;

text-align: center;

color: white;

}

.bar-item{

border-color: #FFFFFF;

border-width: 3px;

border-bottom: 1.5px solid rgba(255, 255, 255, 0.247);

}

table {

border-collapse:collapse;

border:1px solid gray;

font-family: Tahoma,Verdana,Segoe,sans-serif;

width:63%;

table-layout: fixed;

}

th {

border-bottom:1px solid gray;

background:none repeat scroll 0 0 #0775d3;

padding:10px;

color: #FFFFFF;

}

tr {

border-top:1px solid gray;

text-align:center;

}

tr:nth-child(even) {background: #FFFFFF}

tr:nth-child(odd) {background: #BBBBBB}

**Database:**

**Admindb.sql:**

-- phpMyAdmin SQL Dump

-- version 5.1.0

-- <https://www.phpmyadmin.net/>

--

-- Host: 127.0.0.1:3307

-- Generation Time: Jun 26, 2021 at 03:43 AM

-- Server version: 10.4.18-MariaDB

-- PHP Version: 8.0.3

SET SQL\_MODE = "NO\_AUTO\_VALUE\_ON\_ZERO";

START TRANSACTION;

SET time\_zone = "+00:00";

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

/\*!40101 SET NAMES utf8mb4 \*/;

--

-- Database: `Admindb`

--

-- --------------------------------------------------------

--

-- Table structure for table `orders`

--

CREATE TABLE `orders` (

`id` int(11) NOT NULL,

`user\_id` int(11) NOT NULL,

`product\_id` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Dumping data for table `orders`

--

INSERT INTO `orders` (`id`, `user\_id`, `product\_id`) VALUES

(7, 3, 1),

(10, 20, 3),

(11, 20, 1),

(12, 20, 4);

drop table orders;

-- --------------------------------------------------------

--

-- Table structure for table `products`

--

CREATE TABLE `products` (

`id` int(11) NOT NULL,

`name` varchar(50) NOT NULL,

`company` varchar(50) NOT NULL,

`size` int(11) DEFAULT NULL,

`price` double NOT NULL,

`image\_link` varchar(2555) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Dumping data for table `products`

--

INSERT INTO `products` (`id`, `name`, `company`, `size`, `price`, `image\_link`) VALUES

(1, 'columbia sportwear co', 'Columbia ', 45, 55, '<https://i.imgur.com/R4WLP4G.jpg>'),

(2, 'Timerland originals', 'Timberland', 36, 131, '<https://i.imgur.com/Q9IPIid.jpg>'),

(3, 'Saucony 765', 'Saucony', 46, 157, '<https://i.imgur.com/gbvL7MN.jpg>'),

(4, 'Anta sports', 'Anta', 45, 211, '<https://i.imgur.com/0Nsm0Os.jpg>');

drop table products;

-- --------------------------------------------------------

--

-- Table structure for table `users`

--

CREATE TABLE `users` (

`id` int(11) NOT NULL,

`type` int(10) NOT NULL DEFAULT 0,

`username` varchar(50) NOT NULL,

`password` varchar(50) NOT NULL,

`age` int(11) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Dumping data for table `users`

--

INSERT INTO `users` (`id`, `type`, `username`, `password`, `age`) VALUES

(1, 0, 'Bharat', '1234', 24),

(2, 0, 'Manikanta', '1234', 34),

(3, 0, 'roshi', '1234', 1),

(4, 0, 'kamal', '1234', 23),

(16, 1, 'admin', 'admin', 0),

(17, 0, 'sanju', '4242', 24),

(20, 0, 'ram', '12345', 33);

drop table users;

--

-- Indexes for dumped tables

--

--

-- Indexes for table `orders`

--

ALTER TABLE `orders`

ADD PRIMARY KEY (`id`),

ADD KEY `product\_id` (`product\_id`),

ADD KEY `user\_id` (`user\_id`);

--

-- Indexes for table `products`

--

ALTER TABLE `products`

ADD PRIMARY KEY (`id`);

--

-- Indexes for table `users`

--

ALTER TABLE `users`

ADD PRIMARY KEY (`id`);

--

-- AUTO\_INCREMENT for dumped tables

--

--

-- AUTO\_INCREMENT for table `orders`

--

ALTER TABLE `orders`

MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=13;

--

-- AUTO\_INCREMENT for table `products`

--

ALTER TABLE `products`

MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=8;

--

-- AUTO\_INCREMENT for table `users`

--

ALTER TABLE `users`

MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=21;

--

-- Constraints for dumped tables

--

--

-- Constraints for table `orders`

--

ALTER TABLE `orders`

ADD CONSTRAINT `product\_id` FOREIGN KEY (`product\_id`) REFERENCES `products` (`id`) ON DELETE CASCADE ON UPDATE CASCADE,

ADD CONSTRAINT `user\_id` FOREIGN KEY (`user\_id`) REFERENCES `users` (`id`) ON DELETE CASCADE ON UPDATE CASCADE;

COMMIT;

/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/;

**WEB-INF/VIEW/SPORTY-SHOES-SERVLET.xml:**

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

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:context="http://www.springframework.org/schema/context"

xmlns:tx="http://www.springframework.org/schema/tx"

xmlns:mvc="http://www.springframework.org/schema/mvc"

xsi:schemaLocation="

<http://www.springframework.org/schema/beans>

<http://www.springframework.org/schema/beans/spring-beans.xsd>

<http://www.springframework.org/schema/context>

<http://www.springframework.org/schema/context/spring-context.xsd>

<http://www.springframework.org/schema/mvc>

<http://www.springframework.org/schema/mvc/spring-mvc.xsd>

<http://www.springframework.org/schema/tx>

<http://www.springframework.org/schema/tx/spring-tx.xsd>">

<!-- Add support for component scanning -->

<context:component-scan base-package="com.simplilearn" />

<!-- Add support for conversion, formatting and validation support -->

<mvc:annotation-driven/>

<!-- Define Spring MVC view resolver -->

<bean

class="org.springframework.web.servlet.view.InternalResourceViewResolver">

<property name="prefix" value="/WEB-INF/view/" />

<property name="suffix" value=".jsp" />

</bean>

<!-- Step 1: Define Database DataSource / connection pool -->

<bean id="myDataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource"

destroy-method="close">

<property name="driverClass" value="com.mysql.cj.jdbc.Driver" />

<property name="jdbcUrl" value="jdbc:mysql://localhost:3306/admindb" />

<property name="user" value="root" />

<property name="password" value="India@2018" />

<!-- these are connection pool properties for C3P0 -->

<property name="minPoolSize" value="5" />

<property name="maxPoolSize" value="20" />

<property name="maxIdleTime" value="30000" />

</bean>

<!-- Step 2: Setup Hibernate session factory -->

<bean id="sessionFactory"

class="org.springframework.orm.hibernate5.LocalSessionFactoryBean">

<property name="dataSource" ref="myDataSource" />

<property name="packagesToScan" value="com.simplilearn.entity" />

<property name="hibernateProperties">

<props>

<prop key="hibernate.dialect">org.hibernate.dialect.MySQLDialect</prop>

<prop key="hibernate.show\_sql">true</prop>

</props>

</property>

</bean>

<!-- Step 3: Setup Hibernate transaction manager -->

<bean id="myTransactionManager"

class="org.springframework.orm.hibernate5.HibernateTransactionManager">

<property name="sessionFactory" ref="sessionFactory"/>

</bean>

<!-- Step 4: Enable configuration of transactional behavior based on annotations -->

<tx:annotation-driven transaction-manager="myTransactionManager" />

<!-- Add support for reading web resources: css, images, js, etc ... -->

<mvc:resources location="/resources/" mapping="/resources/\*\*"></mvc:resources>

</beans>

**Web.xml:**

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

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee <http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd>" id="WebApp\_ID" version="3.1">

<display-name>Sporty-Shoes-Store</display-name>

<absolute-ordering />

<welcome-file-list>

<welcome-file>index.jsp</welcome-file>

<welcome-file>index.html</welcome-file>

</welcome-file-list>

<servlet>

<servlet-name>dispatcher</servlet-name>

<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

<init-param>

<param-name>contextConfigLocation</param-name>

<param-value>/WEB-INF/sporty-shoes-servlet.xml</param-value>

</init-param>

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>dispatcher</servlet-name>

<url-pattern>/</url-pattern>

</servlet-mapping>

</web-app>

**Index.jsp:**

<% response.sendRedirect("login"); %>

**Folder:**

**Target/m2e-wtp/web-resources/META-INF/maven/com.simplilearn/Sporty-Shoes-Store:**

**File name: pom.properties:**

#Generated by Maven Integration for Eclipse

#Sat Jun 26 03:34:25 AST 2021

m2e.projectLocation=C\:\\Users\\User\\eclipse-workspace\\Sporty-Shoes-Store

m2e.projectName=Sporty-Shoes-Store

groupId=com.simplilearn

artifactId=Sporty-Shoes-Store

version=1.0

**File name: 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 <http://maven.apache.org/xsd/maven-4.0.0.xsd>">

<modelVersion>4.0.0</modelVersion>

<groupId>com.simplilearn</groupId>

<artifactId>Sporty-Shoes-Store</artifactId>

<version>1.0</version>

<packaging>war</packaging>

<name>Sporty-Shoes-Store Maven Webapp</name>

<!-- FIXME change it to the project's website -->

<url>http://www.example.com</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

</properties>

<dependencies>

<!-- <https://mvnrepository.com/artifact/org.springframework/spring-webmvc> -->

<!-- <https://mvnrepository.com/artifact/org.springframework/spring-web> -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-web</artifactId>

<version>5.2.15.RELEASE</version>

</dependency>

<!-- <https://mvnrepository.com/artifact/javax.servlet/javax.servlet-api> -->

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>javax.servlet-api</artifactId>

<version>3.1.0</version>

<scope>provided</scope>

</dependency>

<!-- <https://mvnrepository.com/artifact/org.hibernate/hibernate-core> -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>5.5.2.Final</version>

</dependency>

<!-- <https://mvnrepository.com/artifact/org.springframework/spring-tx> -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-tx</artifactId>

<version>5.3.3</version>

</dependency>

<!-- <https://mvnrepository.com/artifact/mysql/mysql-connector-java> -->

<dependency>

<groupId>mysql</groupId>

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

<version>8.0.25</version>

</dependency>

<!-- <https://mvnrepository.com/artifact/org.hibernate/hibernate-validator> -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-validator</artifactId>

<version>6.1.0.Final</version>

</dependency>

<!-- <https://mvnrepository.com/artifact/org.springframework/spring-core> -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>5.3.4</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>5.2.15.RELEASE</version>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.11</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<finalName>Sporty-Shoes-Store</finalName>

<pluginManagement><!-- lock down plugins versions to avoid using Maven

defaults (may be moved to parent pom) -->

<plugins>

<plugin>

<artifactId>maven-clean-plugin</artifactId>

<version>3.1.0</version>

</plugin>

<!-- see <http://maven.apache.org/ref/current/maven-core/default-bindings.html#Plugin_bindings_for_war_packaging> -->

<plugin>

<artifactId>maven-resources-plugin</artifactId>

<version>3.0.2</version>

</plugin>

<plugin>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.8.0</version>

</plugin>

<plugin>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.22.1</version>

</plugin>

<plugin>

<artifactId>maven-war-plugin</artifactId>

<version>3.2.2</version>

</plugin>

<plugin>

<artifactId>maven-install-plugin</artifactId>

<version>2.5.2</version>

</plugin>

<plugin>

<artifactId>maven-deploy-plugin</artifactId>

<version>2.8.2</version>

</plugin>

</plugins>

</pluginManagement>

</build>

</project>

**File name: MANIFEST.MF**

Manifest-Version: 1.0

Built-By: muj

Build-Jdk: 15.0.2

Created-By: Maven Integration for Eclipse

**File name: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 <http://maven.apache.org/xsd/maven-4.0.0.xsd>">

<modelVersion>4.0.0</modelVersion>

<groupId>com.simplilearn</groupId>

<artifactId>Sporty-Shoes-Store</artifactId>

<version>1.0</version>

<packaging>war</packaging>

<name>Sporty-Shoes-Store Maven Webapp</name>

<!-- FIXME change it to the project's website -->

<url>http://www.example.com</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

</properties>

<dependencies>

<!-- <https://mvnrepository.com/artifact/org.springframework/spring-webmvc> -->

<!-- <https://mvnrepository.com/artifact/org.springframework/spring-web> -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-web</artifactId>

<version>5.2.15.RELEASE</version>

</dependency>

<!-- <https://mvnrepository.com/artifact/javax.servlet/javax.servlet-api> -->

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>javax.servlet-api</artifactId>

<version>3.1.0</version>

<scope>provided</scope>

</dependency>

<!-- <https://mvnrepository.com/artifact/org.hibernate/hibernate-core> -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>5.5.2.Final</version>

</dependency>

<!-- <https://mvnrepository.com/artifact/org.springframework/spring-tx> -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-tx</artifactId>

<version>5.3.3</version>

</dependency>

<!-- <https://mvnrepository.com/artifact/mysql/mysql-connector-java> -->

<dependency>

<groupId>mysql</groupId>

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

<version>8.0.25</version>

</dependency>

<!-- <https://mvnrepository.com/artifact/org.hibernate/hibernate-validator> -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-validator</artifactId>

<version>6.1.0.Final</version>

</dependency>

<!-- <https://mvnrepository.com/artifact/org.springframework/spring-core> -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>5.3.4</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>5.2.15.RELEASE</version>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.11</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<finalName>Sporty-Shoes-Store</finalName>

<pluginManagement><!-- lock down plugins versions to avoid using Maven

defaults (may be moved to parent pom) -->

<plugins>

<plugin>

<artifactId>maven-clean-plugin</artifactId>

<version>3.1.0</version>

</plugin>

<!-- see <http://maven.apache.org/ref/current/maven-core/default-bindings.html#Plugin_bindings_for_war_packaging> -->

<plugin>

<artifactId>maven-resources-plugin</artifactId>

<version>3.0.2</version>

</plugin>

<plugin>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.8.0</version>

</plugin>

<plugin>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.22.1</version>

</plugin>

<plugin>

<artifactId>maven-war-plugin</artifactId>

<version>3.2.2</version>

</plugin>

<plugin>

<artifactId>maven-install-plugin</artifactId>

<version>2.5.2</version>

</plugin>

<plugin>

<artifactId>maven-deploy-plugin</artifactId>

<version>2.8.2</version>

</plugin>

</plugins>

</pluginManagement>

</build>

</project>