**Envirnoment Setup:**

**1.Eclipse Oxygen IDE**

**2.Git Repository Account**

**3.H2 Database**

**4.Tomcat 8/9 Server**

**5.Java 8**

**6.JAR REQUIRED:**

**junit-4.2 , spring-4.3.6, h2-1.4.196 , hibernate-5.2.7(4.3.2),Jackson-2.8.7**

**,logback-1.2.1,hibernate-validator-5.4.1 ,common-dbcp2-2.1.1**

**and etc….**

**H2 Database Records:**

|  |  |
| --- | --- |
| **CREATE** **TABLE** category (  id IDENTITY,  name **VARCHAR**(50),  description **VARCHAR**(255),  image\_url **VARCHAR**(50),  is\_active **BOOLEAN**,  **CONSTRAINT** pk\_category\_id **PRIMARY** **KEY** (id)  ); | **CREATE** **TABLE** user\_detail (  id IDENTITY,  first\_name **VARCHAR**(50),  last\_name **VARCHAR**(50),  role **VARCHAR**(50),  enabled **BOOLEAN**,  password **VARCHAR**(60),  email **VARCHAR**(100),  contact\_number **VARCHAR**(15),  **CONSTRAINT** pk\_user\_id **PRIMARY** **KEY**(id)  ); |
| **CREATE** **TABLE** product (  id IDENTITY,  code **VARCHAR**(20),  name **VARCHAR**(50),  brand **VARCHAR**(50),  description **VARCHAR**(255),  unit\_price **DECIMAL**(10,2),  quantity **INT**,  is\_active **BOOLEAN**,  category\_id **INT**,  supplier\_id **INT**,  purchases **INT** **DEFAULT** 0,  views **INT** **DEFAULT** 0,  **CONSTRAINT** pk\_product\_id **PRIMARY** **KEY** (id),  **CONSTRAINT** fk\_product\_category\_id **FOREIGN** **KEY** (category\_id) **REFERENCES** category (id),  **CONSTRAINT** fk\_product\_supplier\_id **FOREIGN** **KEY** (supplier\_id) **REFERENCES** user\_detail(id),  ); | -- the address table to store the user billing and shipping addresses  **CREATE** **TABLE** address (  id IDENTITY,  user\_id **int**,  address\_line\_one **VARCHAR**(100),  address\_line\_two **VARCHAR**(100),  city **VARCHAR**(20),  state **VARCHAR**(20),  country **VARCHAR**(20),  postal\_code **VARCHAR**(10),  is\_billing **BOOLEAN**,  is\_shipping **BOOLEAN**,  **CONSTRAINT** fk\_address\_user\_id **FOREIGN** **KEY** (user\_id ) **REFERENCES** user\_detail (id),  **CONSTRAINT** pk\_address\_id **PRIMARY** **KEY** (id)  ); |
| -- the cart table to store the user cart top-level details  **CREATE** **TABLE** cart (  id IDENTITY,  user\_id **int**,  grand\_total **DECIMAL**(10,2),  cart\_lines **int**,  **CONSTRAINT** fk\_cart\_user\_id **FOREIGN** **KEY** (user\_id ) **REFERENCES** user\_detail (id),  **CONSTRAINT** pk\_cart\_id **PRIMARY** **KEY** (id)  ); | -- the cart line table to store the cart details  **CREATE** **TABLE** cart\_line (  id IDENTITY,  cart\_id **int**,  total **DECIMAL**(10,2),  product\_id **int**,  product\_count **int**,  buying\_price **DECIMAL**(10,2),  is\_available **boolean**,  **CONSTRAINT** fk\_cartline\_product\_id **FOREIGN** **KEY** (product\_id ) **REFERENCES** product (id),  **CONSTRAINT** pk\_cartline\_id **PRIMARY** **KEY** (id)  ); |
| -- the order detail table to store the order  **CREATE** **TABLE** order\_detail (  id IDENTITY,  user\_id **int**,  order\_total **DECIMAL**(10,2),  order\_count **int**,  shipping\_id **int**,  billing\_id **int**,  order\_date **date**,  **CONSTRAINT** fk\_order\_detail\_user\_id **FOREIGN** **KEY** (user\_id) **REFERENCES** user\_detail (id),  **CONSTRAINT** fk\_order\_detail\_shipping\_id **FOREIGN** **KEY** (shipping\_id) **REFERENCES** address (id),  **CONSTRAINT** fk\_order\_detail\_billing\_id **FOREIGN** **KEY** (billing\_id) **REFERENCES** address (id),  **CONSTRAINT** pk\_order\_detail\_id **PRIMARY** **KEY** (id)  ); | -- the order item table to store order items  **CREATE** **TABLE** order\_item (  id IDENTITY,  order\_id **int**,  total **DECIMAL**(10,2),  product\_id **int**,  product\_count **int**,  buying\_price **DECIMAL**(10,2),  **CONSTRAINT** fk\_order\_item\_product\_id **FOREIGN** **KEY** (product\_id) **REFERENCES** product (id),  **CONSTRAINT** fk\_order\_item\_order\_id **FOREIGN** **KEY** (order\_id) **REFERENCES** order\_detail (id),  **CONSTRAINT** pk\_order\_item\_id **PRIMARY** **KEY** (id)  ); |

-- adding three categories

**INSERT** **INTO** category (name, description,image\_url,is\_active) **VALUES** ('Laptop', 'This is description for Laptop category!', 'CAT\_1.png', **true**);

**INSERT** **INTO** category (name, description,image\_url,is\_active) **VALUES** ('Television', 'This is description for Television category!', 'CAT\_2.png', **true**);

**INSERT** **INTO** category (name, description,image\_url,is\_active) **VALUES** ('Mobile', 'This is description for Mobile category!', 'CAT\_3.png', **true**);

-- adding three users

**INSERT** **INTO** user\_detail

(first\_name, last\_name, role, enabled, password, email, contact\_number)

**VALUES** ('Virat', 'Kohli', 'ADMIN', **true**, '$2a$06$ORtBskA2g5Wg0HDgRE5ZsOQNDHUZSdpJqJ2.PGXv0mKyEvLnKP7SW', 'vk@gmail.com', '8888888888');

**INSERT** **INTO** user\_detail

(first\_name, last\_name, role, enabled, password, email, contact\_number)

**VALUES** ('Ravindra', 'Jadeja', 'SUPPLIER', **true**, '$2a$06$bzYMivkRjSxTK2LPD8W4te6jjJa795OwJR1Of5n95myFsu3hgUnm6', 'rj@gmail.com', '9999999999');

**INSERT** **INTO** user\_detail

(first\_name, last\_name, role, enabled, password, email, contact\_number)

**VALUES** ('Ravichandra', 'Ashwin', 'SUPPLIER', **true**, '$2a$06$i1dLNlXj2uY.UBIb9kUcAOxCigGHUZRKBtpRlmNtL5xtgD6bcVNOK', 'ra@gmail.com', '7777777777');

**INSERT** **INTO** user\_detail

(first\_name, last\_name, role, enabled, password, email, contact\_number)

**VALUES** ('Khozema', 'Nullwala', 'USER', **true**, '$2a$06$4mvvyO0h7vnUiKV57IW3oudNEaKPpH1xVSdbie1k6Ni2jfjwwminq', 'kn@gmail.com', '7777777777');

-- adding five products

**INSERT** **INTO** product (code, name, brand, description, unit\_price, quantity, is\_active, category\_id, supplier\_id, purchases, views)

**VALUES** ('PRDABC123DEFX', 'iPhone 5s', 'apple', 'This is one of the best phone available in the market right now!', 18000, 5, **true**, 3, 2, 0, 0 );

**INSERT** **INTO** product (code, name, brand, description, unit\_price, quantity, is\_active, category\_id, supplier\_id, purchases, views)

**VALUES** ('PRDDEF123DEFX', 'Samsung s7', 'samsung', 'A smart phone by samsung!', 32000, 2, **true**, 3, 3, 0, 0 );

**INSERT** **INTO** product (code, name, brand, description, unit\_price, quantity, is\_active, category\_id, supplier\_id, purchases, views)

**VALUES** ('PRDPQR123WGTX', 'Google Pixel', 'google', 'This is one of the best android smart phone available in the market right now!', 57000, 5, **true**, 3, 2, 0, 0 );

**INSERT** **INTO** product (code, name, brand, description, unit\_price, quantity, is\_active, category\_id, supplier\_id, purchases, views)

**VALUES** ('PRDMNO123PQRX', ' Macbook Pro', 'apple', 'This is one of the best laptops available in the market right now!', 54000, 3, **true**, 1, 2, 0, 0 );

**INSERT** **INTO** product (code, name, brand, description, unit\_price, quantity, is\_active, category\_id, supplier\_id, purchases, views)

**VALUES** ('PRDABCXYZDEFX', 'Dell Latitude E6510', 'dell', 'This is one of the best laptop series from dell that can be used!', 48000, 5, **true**, 1, 3, 0, 0 );

-- adding a supplier correspondece address

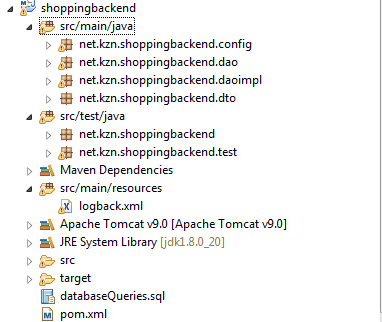
**INSERT** **INTO** address( user\_id, address\_line\_one, address\_line\_two, city, state, country, postal\_code, is\_billing, is\_shipping)

**VALUES** (4, '102 Sabarmati Society, Mahatma Gandhi Road', 'Near Salt Lake, Gandhi Nagar', 'Ahmedabad', 'Gujarat', 'India', '111111', **true**, **false** );

-- adding a cart for testing

**INSERT** **INTO** cart (user\_id, grand\_total, cart\_lines) **VALUES** (4, 0, 0);

**Shoppingbackend(StandAlone Application)**

****

|  |
| --- |
| **pom.xml** |
| <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>net.kzn</groupId>  <artifactId>shoppingbackend</artifactId>  <version>0.0.1-SNAPSHOT</version>  <packaging>jar</packaging>  <name>shoppingbackend</name>  <url>http://maven.apache.org</url>  <properties>  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  <spring.version>4.3.6.RELEASE</spring.version>  <hibernate.version>5.2.7.Final</hibernate.version>  <jackson.version>2.8.7</jackson.version>  </properties>  <dependencies>  <!-- JUNIT VERSION 4.12 -->  <dependency>  <groupId>junit</groupId>  <artifactId>junit</artifactId>  <version>4.12</version>  <scope>test</scope>  </dependency>  <!-- SPRING -->  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-core</artifactId>  <version>${spring.version}</version>  <exclusions>  <exclusion>  <groupId>commons-logging</groupId>  <artifactId>commons-logging</artifactId>  </exclusion>  </exclusions>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-context</artifactId>  <version>${spring.version}</version>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-web</artifactId>  <version>${spring.version}</version>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-orm</artifactId>  <version>${spring.version}</version>  </dependency>  <!-- H2 Database -->  <dependency>  <groupId>com.h2database</groupId>  <artifactId>h2</artifactId>  <version>1.4.196</version>  </dependency>  <!-- Hibernate Dependency -->  <dependency>  <groupId>org.hibernate</groupId>  <artifactId>hibernate-core</artifactId>  <version>${hibernate.version}</version>  </dependency>  <!-- Database Connection Pooling -->  <dependency>  <groupId>org.apache.commons</groupId>  <artifactId>commons-dbcp2</artifactId>  <version>2.1.1</version>  <exclusions>  <exclusion>  <groupId>commons-logging</groupId>  <artifactId>commons-logging</artifactId>  </exclusion>  </exclusions>  </dependency>  <!-- Jackson -->  <dependency>  <groupId>com.fasterxml.jackson.core</groupId>  <artifactId>jackson-databind</artifactId>  <version>${jackson.version}</version>  </dependency>  <dependency>  <groupId>com.fasterxml.jackson.core</groupId>  <artifactId>jackson-annotations</artifactId>  <version>${jackson.version}</version>  </dependency>  <!-- SLF4J Logging -->  <dependency>  <groupId>ch.qos.logback</groupId>  <artifactId>logback-classic</artifactId>  <version>1.2.1</version>  </dependency>  <dependency>  <groupId>org.slf4j</groupId>  <artifactId>jcl-over-slf4j</artifactId>  <version>1.7.24</version>  </dependency>  <!-- Hibernate Validator -->  <dependency>  <groupId>org.hibernate</groupId>  <artifactId>hibernate-validator</artifactId>  <version>5.4.1.Final</version>  </dependency>  <dependency>  <groupId>javax.validation</groupId>  <artifactId>validation-api</artifactId>  <version>1.1.0.Final</version>  </dependency>  </dependencies>  <build>  <!-- Updated for the latest version of JAVA -->  <plugins>  <plugin>  <artifactId>maven-compiler-plugin</artifactId>  <version>3.1</version>  <configuration>  <source>1.8</source>  <target>1.8</target>  </configuration>  </plugin>  </plugins>  </build>  </project> |

|  |
| --- |
| **In src/main/resources/logback.xml** |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <configuration>  <appender name=*"consoleAppender"* class=*"ch.qos.logback.core.ConsoleAppender"*>  <encoder>  <Pattern>%-5level %msg %d{HH:mm:ss.SSS} %logger{36} %n</Pattern>  </encoder>  </appender>  <logger name=*"net.kzn"* level=*"DEBUG"* additivity = *"false"*>  <appender-ref ref=*"consoleAppender"*/>  </logger>  <root level=*"INFO"*>  <appender-ref ref=*"consoleAppender"*/>  </root>  </configuration> |

|  |
| --- |
| HibernateConfig.java |
| package net.kzn.shoppingbackend.config;  import java.util.Properties;  import javax.sql.DataSource;  import org.apache.commons.dbcp2.BasicDataSource;  import org.hibernate.SessionFactory;  import org.springframework.context.annotation.Bean;  import org.springframework.context.annotation.ComponentScan;  import org.springframework.context.annotation.Configuration;  import org.springframework.core.annotation.Order;  import org.springframework.orm.hibernate5.HibernateTransactionManager;  import org.springframework.orm.hibernate5.LocalSessionFactoryBuilder;  import org.springframework.transaction.annotation.EnableTransactionManagement;  @Configuration  @ComponentScan(basePackages={"net.kzn.shoppingbackend.dto"})  @EnableTransactionManagement  public class HibernateConfig {  // Change the below based on the DBMS you choose  private final static String DATABASE\_URL = "jdbc:h2:tcp://localhost/~/onlineshopping";  private final static String DATABASE\_DRIVER = "org.h2.Driver";  private final static String DATABASE\_DIALECT = "org.hibernate.dialect.H2Dialect";  private final static String DATABASE\_USERNAME = "system";  private final static String DATABASE\_PASSWORD = "manager";    // dataSource bean will be available  @Bean("dataSource")  public DataSource getDataSource() {  BasicDataSource dataSource = new BasicDataSource();  // Providing the database connection information  dataSource.setDriverClassName(DATABASE\_DRIVER);  dataSource.setUrl(DATABASE\_URL);  dataSource.setUsername(DATABASE\_USERNAME);  dataSource.setPassword(DATABASE\_PASSWORD);  return dataSource;  }  // sessionFactory bean will be available  @Bean  public SessionFactory getSessionFactory(DataSource dataSource) {  LocalSessionFactoryBuilder builder = new LocalSessionFactoryBuilder(dataSource);  builder.addProperties(getHibernateProperties());  builder.scanPackages("net.kzn.shoppingbackend.dto");  return builder.buildSessionFactory();  }  // All the hibernate properties will be returned in this method  private Properties getHibernateProperties() {  Properties properties = new Properties();  properties.put("hibernate.dialect", DATABASE\_DIALECT);  properties.put("hibernate.show\_sql", "true");  properties.put("hibernate.format\_sql", "true");  //properties.put("hibernate.hbm2ddl.auto", "create");  return properties;  }  // transactionManager bean  @Bean  public HibernateTransactionManager getTransactionManager(SessionFactory sessionFactory) {  HibernateTransactionManager transactionManager = new HibernateTransactionManager(sessionFactory);  return transactionManager;  }  } |

|  |
| --- |
| CategoryDAO.java |
| package net.kzn.shoppingbackend.dao;  import java.util.List;  import net.kzn.shoppingbackend.dto.Category;  public interface CategoryDAO {  Category get(int id);  List<Category> list();  boolean add(Category category);  boolean update(Category category);  boolean delete(Category category);  } |

|  |
| --- |
| CartLineDAO.java |
| package net.kzn.shoppingbackend.dao;  import java.util.List;  import net.kzn.shoppingbackend.dto.Cart;  import net.kzn.shoppingbackend.dto.CartLine;  import net.kzn.shoppingbackend.dto.OrderDetail;  public interface CartLineDAO {  public List<CartLine> list(int cartId);  public CartLine get(int id);  public boolean add(CartLine cartLine);  public boolean update(CartLine cartLine);  public boolean remove(CartLine cartLine);  // fetch the CartLine based on cartId and productId  public CartLine getByCartAndProduct(int cartId, int productId);  // updating the cart  boolean updateCart(Cart cart);  // list of available cartLine  public List<CartLine> listAvailable(int cartId);  // adding order details  boolean addOrderDetail(OrderDetail orderDetail);  } |

|  |
| --- |
| ProductDAO.java |
| package net.kzn.shoppingbackend.dao;  import java.util.List;  import net.kzn.shoppingbackend.dto.Product;  public interface ProductDAO {  Product get(int productId);  List<Product> list();  boolean add(Product product);  boolean update(Product product);  boolean delete(Product product);  List<Product> getProductsByParam(String param, int count);  // business methods  List<Product> listActiveProducts();  List<Product> listActiveProductsByCategory(int categoryId);  List<Product> getLatestActiveProducts(int count);  } |

|  |
| --- |
| UserDAO.java |
| package net.kzn.shoppingbackend.dao;  import java.util.List;  import net.kzn.shoppingbackend.dto.Address;  import net.kzn.shoppingbackend.dto.Cart;  import net.kzn.shoppingbackend.dto.User;  public interface UserDAO {  // user related operation  User getByEmail(String email);  User get(int id);  boolean add(User user);  // adding and updating a new address  Address getAddress(int addressId);  boolean addAddress(Address address);  boolean updateAddress(Address address);  Address getBillingAddress(int userId);  List<Address> listShippingAddresses(int userId);  } |

|  |
| --- |
| CartLineDAOImpl.java |
| package net.kzn.shoppingbackend.daoimpl;  import java.util.List;  import org.hibernate.SessionFactory;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.stereotype.Repository;  import org.springframework.transaction.annotation.Transactional;  import net.kzn.shoppingbackend.dao.CartLineDAO;  import net.kzn.shoppingbackend.dto.Cart;  import net.kzn.shoppingbackend.dto.CartLine;  import net.kzn.shoppingbackend.dto.OrderDetail;  @Repository("cartLineDAO")  @Transactional  public class CartLineDAOImpl implements CartLineDAO {  @Autowired  private SessionFactory sessionFactory;  @Override  public CartLine getByCartAndProduct(int cartId, int productId) {  String query = "FROM CartLine WHERE cartId = :cartId AND product.id = :productId";  try {  return sessionFactory.getCurrentSession().createQuery(query,CartLine.class).setParameter("cartId", cartId)  .setParameter("productId", productId).getSingleResult();    }catch(Exception ex) {  return null;  }  }  @Override  public boolean add(CartLine cartLine) {  try {  sessionFactory.getCurrentSession().persist(cartLine);  return true;  }  catch(Exception ex) {  ex.printStackTrace();  return false;  }  }  @Override  public boolean update(CartLine cartLine) {  try {  sessionFactory.getCurrentSession().update(cartLine);  return true;  }  catch(Exception ex) {  ex.printStackTrace();  return false;  }  }  @Override  public boolean remove(CartLine cartLine) {  try {  sessionFactory.getCurrentSession().delete(cartLine);  return true;  }catch(Exception ex) {  return false;  }  }  @Override  public List<CartLine> list(int cartId) {  String query = "FROM CartLine WHERE cartId = :cartId";  return sessionFactory.getCurrentSession().createQuery(query, CartLine.class).setParameter("cartId", cartId).getResultList();  }  @Override  public CartLine get(int id) {  return sessionFactory.getCurrentSession().get(CartLine.class, Integer.valueOf(id));  }  @Override  public boolean updateCart(Cart cart) {  try {  sessionFactory.getCurrentSession().update(cart);  return true;  }  catch(Exception ex) {  return false;  }  }  @Override  public List<CartLine> listAvailable(int cartId) {  String query = "FROM CartLine WHERE cartId = :cartId AND available = :available";  return sessionFactory.getCurrentSession().createQuery(query, CartLine.class).setParameter("cartId", cartId)  .setParameter("available", true).getResultList();  }  @Override  public boolean addOrderDetail(OrderDetail orderDetail) {  try {  sessionFactory.getCurrentSession().persist(orderDetail);  return true;  }  catch(Exception ex) {  return false;  }  }  } |

|  |
| --- |
| CategoryDAOImpl.java |
| package net.kzn.shoppingbackend.daoimpl;  import java.util.List;  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 net.kzn.shoppingbackend.dao.CategoryDAO;  import net.kzn.shoppingbackend.dto.Category;  @Repository("categoryDAO")  @Transactional  public class CategoryDAOImpl implements CategoryDAO {  @Autowired  private SessionFactory sessionFactory;  @Override  public List<Category> list() {    String selectActiveCategory = "FROM Category WHERE active = :active";    Query query = sessionFactory.getCurrentSession().createQuery(selectActiveCategory);    query.setParameter("active", true);    return query.getResultList();  }  /\*  \* Getting single category based on id  \*/  @Override  public Category get(int id) {  return sessionFactory.getCurrentSession().get(Category.class, Integer.valueOf(id));  }  @Override  public boolean add(Category category) {  try {  // add the category to the database table  sessionFactory.getCurrentSession().persist(category);  return true;  } catch (Exception ex) {  ex.printStackTrace();  return false;  }  }  /\*  \* Updating a single category  \*/  @Override  public boolean update(Category category) {  try {  // add the category to the database table  sessionFactory.getCurrentSession().update(category);  return true;  } catch (Exception ex) {  ex.printStackTrace();  return false;  }  }  @Override  public boolean delete(Category category) {    category.setActive(false);    try {  // add the category to the database table  sessionFactory.getCurrentSession().update(category);  return true;  } catch (Exception ex) {  ex.printStackTrace();  return false;  }  }  } |

|  |
| --- |
| ProductDAOImpl.java |
| package net.kzn.shoppingbackend.daoimpl;  import java.util.List;  import org.hibernate.SessionFactory;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.stereotype.Repository;  import org.springframework.transaction.annotation.Transactional;  import net.kzn.shoppingbackend.dao.ProductDAO;  import net.kzn.shoppingbackend.dto.Product;  @Repository("productDAO")  @Transactional  public class ProductDAOImpl implements ProductDAO {  @Autowired  private SessionFactory sessionFactory;  //GET  @Override  public Product get(int productId) {  try {  return sessionFactory.getCurrentSession().get(Product.class,Integer.valueOf(productId));  }  catch(Exception ex) {  ex.printStackTrace();  }  return null;  }  //LIST  @Override  public List<Product> list() {  return sessionFactory.getCurrentSession().createQuery("FROM Product" , Product.class).getResultList();  }  // INSERT  @Override  public boolean add(Product product) {  try {  sessionFactory.getCurrentSession().persist(product);  return true;  }  catch(Exception ex) {  ex.printStackTrace();  }  return false;  }  // UPDATE  @Override  public boolean update(Product product) {  try {  sessionFactory.getCurrentSession().update(product);  return true;  }  catch(Exception ex) {  ex.printStackTrace();  }  return false;  }  / /DELETE  @Override  public boolean delete(Product product) {  try {    product.setActive(false);  // call the update method  return this.update(product);  }  catch(Exception ex) {  ex.printStackTrace();  }  return false;  }  @Override  public List<Product> listActiveProducts() {  String selectActiveProducts = "FROM Product WHERE active = :active";  return sessionFactory  .getCurrentSession()  .createQuery(selectActiveProducts, Product.class)  .setParameter("active", true)  .getResultList();  }  @Override  public List<Product> listActiveProductsByCategory(int categoryId) {  String selectActiveProductsByCategory = "FROM Product WHERE active = :active AND categoryId = :categoryId";  return sessionFactory  .getCurrentSession()  .createQuery(selectActiveProductsByCategory, Product.class)  .setParameter("active", true)  .setParameter("categoryId",categoryId)  .getResultList();  }  @Override  public List<Product> getLatestActiveProducts(int count) {  return sessionFactory  .getCurrentSession()  .createQuery("FROM Product WHERE active = :active ORDER BY id", Product.class)  .setParameter("active", true)  .setFirstResult(0)  .setMaxResults(count)  .getResultList();  }  @Override  public List<Product> getProductsByParam(String param, int count) {    String query = "FROM Product WHERE active = true ORDER BY " + param + " DESC";    return sessionFactory  .getCurrentSession()  .createQuery(query,Product.class)  .setFirstResult(0)  .setMaxResults(count)  .getResultList();  }  } |

|  |
| --- |
| UserDAOImpl.java |
| package net.kzn.shoppingbackend.daoimpl;  import java.util.List;  import org.hibernate.SessionFactory;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.stereotype.Repository;  import org.springframework.transaction.annotation.Transactional;  import net.kzn.shoppingbackend.dao.UserDAO;  import net.kzn.shoppingbackend.dto.Address;  import net.kzn.shoppingbackend.dto.User;  @Repository("userDAO")  @Transactional  public class UserDAOImpl implements UserDAO {  @Autowired  private SessionFactory sessionFactory;  @Override  public User getByEmail(String email) {  String selectQuery = "FROM User WHERE email = :email";  try {  return sessionFactory  .getCurrentSession()  .createQuery(selectQuery,User.class)  .setParameter("email",email)  .getSingleResult();  }  catch(Exception ex) {  return null;  }  }  @Override  public boolean add(User user) {  try {  sessionFactory.getCurrentSession().persist(user);  return true;  }  catch(Exception ex) {  return false;  }  }  @Override  public boolean addAddress(Address address) {  try {  // will look for this code later and why we need to change it  sessionFactory.getCurrentSession().persist(address);  return true;  }  catch(Exception ex) {  return false;  }  }  @Override  public boolean updateAddress(Address address) {  try {  sessionFactory.getCurrentSession().update(address);  return true;  }  catch(Exception ex) {  return false;  }  }  @Override  public List<Address> listShippingAddresses(int userId) {  String selectQuery = "FROM Address WHERE userId = :userId AND shipping = :isShipping ORDER BY id DESC";  return sessionFactory  .getCurrentSession()  .createQuery(selectQuery,Address.class)  .setParameter("userId", userId)  .setParameter("isShipping", true)  .getResultList();  }  @Override  public Address getBillingAddress(int userId) {  String selectQuery = "FROM Address WHERE userId = :userId AND billing = :isBilling";  try{  return sessionFactory  .getCurrentSession()  .createQuery(selectQuery,Address.class)  .setParameter("userId", userId)  .setParameter("isBilling", true)  .getSingleResult();  }  catch(Exception ex) {  return null;  }  }  @Override  public User get(int id) {  try {  return sessionFactory.getCurrentSession().get(User.class, id);  }  catch(Exception ex) {  System.out.println(ex.getMessage());  return null;  }  }  @Override  public Address getAddress(int addressId) {  try {  return sessionFactory.getCurrentSession().get(Address.class, addressId);  }  catch(Exception ex) {  System.out.println(ex.getMessage());  return null;  }  }  } |

|  |
| --- |
| Address.java |
| package net.kzn.shoppingbackend.dto;  import java.io.Serializable;  import javax.persistence.Column;  import javax.persistence.Entity;  import javax.persistence.GeneratedValue;  import javax.persistence.GenerationType;  import javax.persistence.Id;  import org.hibernate.validator.constraints.NotBlank;  @Entity  public class Address implements Serializable {  private static final long serialVersionUID = 1L;  @Id  @GeneratedValue(strategy = GenerationType.IDENTITY)  private int id;  @NotBlank(message = "Please enter address line one!")  @Column(name = "address\_line\_one")  private String addressLineOne;  @NotBlank(message = "Please enter address line two!")  @Column(name = "address\_line\_two")  private String addressLineTwo;  @NotBlank(message = "Please enter City!")  private String city;  @NotBlank(message = "Please enter State!")  private String state;  @NotBlank(message = "Please enter country!")  private String country;  @Column(name ="postal\_code")  @NotBlank(message = "Please enter Postal Code!")  private String postalCode;  @Column(name="is\_shipping")  private boolean shipping;  @Column(name="is\_billing")  private boolean billing;  public int getId() {  return id;  }  public void setId(int id) {  this.id = id;  }  public String getAddressLineOne() {  return addressLineOne;  }  public void setAddressLineOne(String addressLineOne) {  this.addressLineOne = addressLineOne;  }  public String getAddressLineTwo() {  return addressLineTwo;  }  public void setAddressLineTwo(String addressLineTwo) {  this.addressLineTwo = addressLineTwo;  }  public String getCity() {  return city;  }  public void setCity(String city) {  this.city = city;  }  public String getState() {  return state;  }  public void setState(String state) {  this.state = state;  }  public String getCountry() {  return country;  }  public void setCountry(String country) {  this.country = country;  }  public String getPostalCode() {  return postalCode;  }  public void setPostalCode(String postalCode) {  this.postalCode = postalCode;  }  public boolean isBilling() {  return billing;  }  public void setBilling(boolean billing) {  this.billing = billing;  }  @Override  public String toString() {  return "Address [id=" + id + ", addressLineOne=" + addressLineOne + ", addressLineTwo=" + addressLineTwo  + ", city=" + city + ", state=" + state + ", country=" + country + ", postalCode=" + postalCode  + ", billing=" + billing + "]";  }  @Column(name = "user\_id")  private int userId;  public boolean isShipping() {  return shipping;  }  public void setShipping(boolean shipping) {  this.shipping = shipping;  }  public int getUserId() {  return userId;  }  public void setUserId(int userId) {  this.userId = userId;  }  } |

|  |
| --- |
| Cart.java |
| package net.kzn.shoppingbackend.dto;  import java.io.Serializable;  import javax.persistence.Column;  import javax.persistence.Entity;  import javax.persistence.GeneratedValue;  import javax.persistence.GenerationType;  import javax.persistence.Id;  import javax.persistence.OneToOne;  @Entity  public class Cart implements Serializable {  private static final long serialVersionUID = 1L;  @Id  @GeneratedValue(strategy = GenerationType.IDENTITY)  private int id;  @Column(name = "grand\_total")  private double grandTotal;  @Column(name = "cart\_lines")  private int cartLines;  public int getId() {  return id;  }  public void setId(int id) {  this.id = id;  }  public double getGrandTotal() {  return grandTotal;  }  public void setGrandTotal(double grandTotal) {  this.grandTotal = grandTotal;  }  public int getCartLines() {  return cartLines;  }  public void setCartLines(int cartLines) {  this.cartLines = cartLines;  }  @Override  public String toString() {  return "Cart [id=" + id + ", grandTotal=" + grandTotal + ", cartLines=" + cartLines + "]";  }  // linking the cart with a user  @OneToOne  private User user;  public User getUser() {  return user;  }  public void setUser(User user) {  this.user = user;  }  } |

|  |
| --- |
| CartLine.java |
| package net.kzn.shoppingbackend.dto;  import java.io.Serializable;  import javax.persistence.Column;  import javax.persistence.Entity;  import javax.persistence.GeneratedValue;  import javax.persistence.GenerationType;  import javax.persistence.Id;  import javax.persistence.OneToOne;  import javax.persistence.Table;  @Entity  @Table(name = "cart\_line")  public class CartLine implements Serializable {  private static final long serialVersionUID = 1L;  @Id  @GeneratedValue(strategy = GenerationType.IDENTITY)  private int id;  @OneToOne  private Product product;  @Column(name = "cart\_id")  private int cartId;  @Column(name = "product\_count")  private int productCount;  private double total;  @Column(name = "buying\_price")  private double buyingPrice;  public double getBuyingPrice() {  return buyingPrice;  }  public void setBuyingPrice(double buyingPrice) {  this.buyingPrice = buyingPrice;  }  @Column(name = "is\_available")  private boolean available = true;    public int getId() {  return id;  }  public void setId(int id) {  this.id = id;  }  public Product getProduct() {  return product;  }  public void setProduct(Product product) {  this.product = product;  }  public int getCartId() {  return cartId;  }  public void setCartId(int cartId) {  this.cartId = cartId;  }  public int getProductCount() {  return productCount;  }  public void setProductCount(int productCount) {  this.productCount = productCount;  }  public double getTotal() {  return total;  }  public void setTotal(double total) {  this.total = total;  }  public boolean isAvailable() {  return available;  }  public void setAvailable(boolean available) {  this.available = available;  }  } |

|  |
| --- |
| Category.java |
| package net.kzn.shoppingbackend.dto;  import java.io.Serializable;  import javax.persistence.Column;  import javax.persistence.Entity;  import javax.persistence.GeneratedValue;  import javax.persistence.GenerationType;  import javax.persistence.Id;  @Entity  public class Category implements Serializable {  private static final long serialVersionUID = 1L;  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 getDescription() {  return description;  }  public void setDescription(String description) {  this.description = description;  }  public String getImageURL() {  return imageURL;  }  public void setImageURL(String imageURL) {  this.imageURL = imageURL;  }  public boolean isActive() {  return active;  }  public void setActive(boolean active) {  this.active = active;  }  @Override  public String toString() {  return "Category [id=" + id + ", name=" + name + ", description=" + description + ", imageURL=" + imageURL  + ", active=" + active + "]";  }  /\*  \* Private fields  \* \*/  @Id  @GeneratedValue(strategy = GenerationType.IDENTITY)  private int id;    private String name;    private String description;    @Column(name = "image\_url")  private String imageURL;    @Column(name = "is\_active")  private boolean active = true;  } |

|  |
| --- |
| OrderDetail.java |
| package net.kzn.shoppingbackend.dto;  import java.io.Serializable;  import java.util.ArrayList;  import java.util.Date;  import java.util.List;  import javax.persistence.CascadeType;  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.OneToMany;  import javax.persistence.Table;  @Entity  @Table(name = "order\_detail")  public class OrderDetail implements Serializable {  /\*\*  \*  \*/  private static final long serialVersionUID = 1L;  @Id  @GeneratedValue(strategy = GenerationType.IDENTITY)  private int id;  @ManyToOne  @JoinColumn(name = "user\_id")  private User user;  @Column(name = "order\_total")  private double orderTotal;  @ManyToOne  private Address shipping;  @ManyToOne  private Address billing;  @OneToMany(mappedBy="orderDetail", fetch = FetchType.EAGER, cascade = CascadeType.ALL)  private List<OrderItem> orderItems = new ArrayList<>();    @Column(name = "order\_count")  private int orderCount;    @Column(name="order\_date")  private Date orderDate;    public Date getOrderDate() {  return orderDate;  }  public void setOrderDate(Date orderDate) {  this.orderDate = orderDate;  }  public int getOrderCount() {  return orderCount;  }  public void setOrderCount(int orderCount) {  this.orderCount = orderCount;  }  public User getUser() {  return user;  }  public void setUser(User user) {  this.user = user;  }  public int getId() {  return id;  }  public void setId(int id) {  this.id = id;  }  public double getOrderTotal() {  return orderTotal;  }  public void setOrderTotal(double orderTotal) {  this.orderTotal = orderTotal;  }  public Address getShipping() {  return shipping;  }  public void setShipping(Address shipping) {  this.shipping = shipping;  }  public Address getBilling() {  return billing;  }  public void setBilling(Address billing) {  this.billing = billing;  }  public List<OrderItem> getOrderItems() {  return orderItems;  }  public void setOrderItems(List<OrderItem> orderItems) {  this.orderItems = orderItems;  }    } |

|  |
| --- |
| OrderItem.java |
| package net.kzn.shoppingbackend.dto;  import java.io.Serializable;  import javax.persistence.Column;  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.OneToOne;  import javax.persistence.Table;  @Entity  @Table(name = "order\_item")  public class OrderItem implements Serializable {  /\*\*  \*  \*/  private static final long serialVersionUID = 1L;    @Id  @GeneratedValue(strategy = GenerationType.IDENTITY)  private int id;    @OneToOne  private Product product;    @ManyToOne  @JoinColumn(name = "order\_id")  private OrderDetail orderDetail;  public OrderDetail getOrderDetail() {  return orderDetail;  }  public void setOrderDetail(OrderDetail orderDetail) {  this.orderDetail = orderDetail;  }  @Column (name = "buying\_price")  private double buyingPrice;    @Column (name = "product\_count")  private int productCount;    private double total;    public int getId() {  return id;  }  public void setId(int id) {  this.id = id;  }  public Product getProduct() {  return product;  }  public void setProduct(Product product) {  this.product = product;  }  public double getBuyingPrice() {  return buyingPrice;  }  public void setBuyingPrice(double buyingPrice) {  this.buyingPrice = buyingPrice;  }  public int getProductCount() {  return productCount;  }  public void setProductCount(int productCount) {  this.productCount = productCount;  }  public double getTotal() {  return total;  }  public void setTotal(double total) {  this.total = total;  }    } |

|  |
| --- |
| Product.java |
| package net.kzn.shoppingbackend.dto;  import java.io.Serializable;  import java.util.UUID;  import javax.persistence.Column;  import javax.persistence.Entity;  import javax.persistence.GeneratedValue;  import javax.persistence.GenerationType;  import javax.persistence.Id;  import javax.persistence.Transient;  import javax.validation.constraints.Min;  import org.hibernate.validator.constraints.NotBlank;  import org.springframework.stereotype.Component;  import org.springframework.web.multipart.MultipartFile;  import com.fasterxml.jackson.annotation.JsonIgnore;  @Component  @Entity  public class Product implements Serializable {  private static final long serialVersionUID = 1L;    // private fields  @Id  @GeneratedValue(strategy = GenerationType.IDENTITY)  private int id;  private String code;  @NotBlank(message = "Please enter the product name!")  private String name;  @NotBlank(message = "Please enter the brand name!")  private String brand;  @NotBlank(message = "Please enter the description!")  private String description;  @Column(name = "unit\_price")  @Min(value = 1, message="Please select at least one value!")  private double unitPrice;  private int quantity;  @Column(name = "is\_active")  private boolean active;  @Column(name = "category\_id")  @JsonIgnore  private int categoryId;  @Column(name = "supplier\_id")  @JsonIgnore  private int supplierId;  private int purchases;  private int views;      @Transient  private MultipartFile file;    public MultipartFile getFile() {  return file;  }  public void setFile(MultipartFile file) {  this.file = file;  }  // default constructor  public Product() {    this.code = "PRD" + UUID.randomUUID().toString().substring(26).toUpperCase();    }      // setters and getters  public int getId() {  return id;  }  public void setId(int id) {  this.id = id;  }  public String getCode() {  return code;  }  public void setCode(String code) {  this.code = code;  }  public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }  public String getBrand() {  return brand;  }  public void setBrand(String brand) {  this.brand = brand;  }  public String getDescription() {  return description;  }  public void setDescription(String description) {  this.description = description;  }  public double getUnitPrice() {  return unitPrice;  }  public void setUnitPrice(double unitPrice) {  this.unitPrice = unitPrice;  }  public int getQuantity() {  return quantity;  }  public void setQuantity(int quantity) {  this.quantity = quantity;  }  public boolean isActive() {  return active;  }  public void setActive(boolean active) {  this.active = active;  }  public int getCategoryId() {  return categoryId;  }  public void setCategoryId(int categoryId) {  this.categoryId = categoryId;  }  public int getSupplierId() {  return supplierId;  }  public void setSupplierId(int supplierId) {  this.supplierId = supplierId;  }  public int getPurchases() {  return purchases;  }  public void setPurchases(int purchases) {  this.purchases = purchases;  }  public int getViews() {  return views;  }  public void setViews(int views) {  this.views = views;  }    // toString for debugging  @Override  public String toString() {  return "Product [id=" + id + ", code=" + code + ", name=" + name + ", brand=" + brand + ", description="  + description + ", unitPrice=" + unitPrice + ", quantity=" + quantity + ", active=" + active  + ", categoryId=" + categoryId + ", supplierId=" + supplierId + ", purchases=" + purchases + ", views="  + views + "]";  }  } |

|  |
| --- |
| User.java |
| package net.kzn.shoppingbackend.dto;  import java.io.Serializable;  import javax.persistence.CascadeType;  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.OneToOne;  import javax.persistence.Table;  import javax.persistence.Transient;  import org.hibernate.validator.constraints.NotBlank;  @Entity  @Table(name = "user\_detail")  public class User implements Serializable{  private static final long serialVersionUID = 1L;  @Id  @GeneratedValue(strategy = GenerationType.IDENTITY)  private int id;  @NotBlank(message = "Please enter first name!")  @Column(name = "first\_name")  private String firstName;  @NotBlank(message = "Please enter last name!")  @Column(name = "last\_name")  private String lastName;  @NotBlank(message = "Please enter email address!")  private String email;  @NotBlank(message = "Please enter contact number!")  @Column(name = "contact\_number")  private String contactNumber;  private String role;  @NotBlank(message = "Please enter password!")  private String password;  private boolean enabled = true;  @Transient  private String confirmPassword;  public String getConfirmPassword() {  return confirmPassword;  }  public void setConfirmPassword(String confirmPassword) {  this.confirmPassword = confirmPassword;  }  public int getId() {  return id;  }  public void setId(int id) {  this.id = id;  }  public String getFirstName() {  return firstName;  }  public void setFirstName(String firstName) {  this.firstName = firstName;  }  public String getLastName() {  return lastName;  }  public void setLastName(String lastName) {  this.lastName = lastName;  }  public String getEmail() {  return email;  }  public void setEmail(String email) {  this.email = email;  }  public String getContactNumber() {  return contactNumber;  }  public void setContactNumber(String contactNumber) {  this.contactNumber = contactNumber;  }  public String getRole() {  return role;  }  public void setRole(String role) {  this.role = role;  }  public String getPassword() {  return password;  }  public void setPassword(String password) {  this.password = password;  }  public boolean isEnabled() {  return enabled;  }  public void setEnabled(boolean enabled) {  this.enabled = enabled;  }  @Override  public String toString() {  return "User [id=" + id + ", firstName=" + firstName + ", lastName=" + lastName + ", email=" + email  + ", contactNumber=" + contactNumber + ", role=" + role + ", password=" + password + ", enabled="  + enabled + "]";  }  @OneToOne(mappedBy="user", cascade = CascadeType.ALL, fetch = FetchType.EAGER)  private Cart cart;  public Cart getCart() {  return cart;  }  public void setCart(Cart cart) {  this.cart = cart;  }  } |

|  |
| --- |
| CartLineTestCase.java |
| package net.kzn.shoppingbackend.test;  import static org.junit.Assert.assertEquals;  import org.junit.BeforeClass;  import org.junit.Test;  import org.springframework.context.annotation.AnnotationConfigApplicationContext;  import net.kzn.shoppingbackend.dao.CartLineDAO;  import net.kzn.shoppingbackend.dao.ProductDAO;  import net.kzn.shoppingbackend.dao.UserDAO;  import net.kzn.shoppingbackend.dto.Cart;  import net.kzn.shoppingbackend.dto.CartLine;  import net.kzn.shoppingbackend.dto.User;  public class CartLineTestCase {  private static AnnotationConfigApplicationContext context;  private static CartLineDAO cartLineDAO;  private static ProductDAO productDAO;  private static UserDAO userDAO;  private CartLine cartLine = null;  @BeforeClass  public static void init() {  context = new AnnotationConfigApplicationContext();  context.scan("net.kzn.shoppingbackend");  context.refresh();  cartLineDAO = (CartLineDAO)context.getBean("cartLineDAO");  productDAO = (ProductDAO)context.getBean("productDAO");  userDAO = (UserDAO)context.getBean("userDAO");  }  /\*  @Test  public void testAddCartLine() {  // fetch the user and then cart of that user  User user = userDAO.getByEmail("absr@gmail.com");  Cart cart = user.getCart();  // fetch the product  Product product = productDAO.get(2);  // Create a new CartLine  cartLine = new CartLine();  cartLine.setCartId(cart.getId());  cartLine.setProduct(product);  cartLine.setProductCount(1);  double oldTotal = cartLine.getTotal();  cartLine.setTotal(product.getUnitPrice() \* cartLine.getProductCount());  cart.setCartLines(cart.getCartLines() + 1);  cart.setGrandTotal(cart.getGrandTotal() + (cartLine.getTotal() - oldTotal));  assertEquals("Failed to add the CartLine!",true, cartLineDAO.add(cartLine));  assertEquals("Failed to update the cart!",true, userDAO.updateCart(cart));  }  \*/  /\*@Test  public void testUpdateCartLine() {  // fetch the user and then cart of that user  User user = userDAO.getByEmail("absr@gmail.com");  Cart cart = user.getCart();  cartLine = cartLineDAO.getByCartAndProduct(cart.getId(), 2);  cartLine.setProductCount(cartLine.getProductCount() + 1);  double oldTotal = cartLine.getTotal();  cartLine.setTotal(cartLine.getProduct().getUnitPrice() \* cartLine.getProductCount());  cart.setGrandTotal(cart.getGrandTotal() + (cartLine.getTotal() - oldTotal));  assertEquals("Failed to update the CartLine!",true, cartLineDAO.update(cartLine));    }\*/  } |

|  |
| --- |
| CategoryTestCase.java |
| package net.kzn.shoppingbackend.test;  import static org.junit.Assert.assertEquals;  import org.junit.BeforeClass;  import org.junit.Test;  import org.springframework.context.annotation.AnnotationConfigApplicationContext;  import net.kzn.shoppingbackend.dao.CategoryDAO;  import net.kzn.shoppingbackend.dto.Category;  public class CategoryTestCase {  private static AnnotationConfigApplicationContext context;      private static CategoryDAO categoryDAO;      private Category category;      @BeforeClass  public static void init() {  context = new AnnotationConfigApplicationContext();  context.scan("net.kzn.shoppingbackend");  context.refresh();  categoryDAO = (CategoryDAO)context.getBean("categoryDAO");  }      /\* @Test  public void testAddCategory() {    category = new Category();    category.setName("Laptop");  category.setDescription("This is some description for laptop!");  category.setImageURL("CAT\_105.png");    assertEquals("Successfully added a category inside the table!",true,categoryDAO.add(category));      }  \*/    /\* @Test  public void testGetCategory() {    category = categoryDAO.get(3);      assertEquals("Successfully fetched a single category from the table!","Television",category.getName());      }  \*/    /\* @Test  public void testUpdateCategory() {    category = categoryDAO.get(3);    category.setName("TV");    assertEquals("Successfully updated a single category in the table!",true,categoryDAO.update(category));      }  \*/  /\* @Test  public void testDeleteCategory() {    category = categoryDAO.get(3);  assertEquals("Successfully deleted a single category in the table!",true,categoryDAO.delete(category));      }  \*/  /\*  @Test  public void testListCategory() {    assertEquals("Successfully fetched the list of categories from the table!",3,categoryDAO.list().size());      }    \*/  /\*@Test  public void testCRUDCategory() {  // add operation  category = new Category();  category.setName("Laptop");  category.setDescription("This is some description for laptop!");  category.setImageURL("CAT\_1.png");  assertEquals("Successfully added a category inside the table!",true,categoryDAO.add(category));  category = new Category();  category.setName("Television");  category.setDescription("This is some description for television!");  category.setImageURL("CAT\_2.png");  assertEquals("Successfully added a category inside the table!",true,categoryDAO.add(category));  // fetching and updating the category  category = categoryDAO.get(2);  category.setName("TV");  assertEquals("Successfully updated a single category in the table!",true,categoryDAO.update(category));  // delete the category  assertEquals("Successfully deleted a single category in the table!",true,categoryDAO.delete(category));  //fetching the list  assertEquals("Successfully fetched the list of categories from the table!",1,categoryDAO.list().size());    }\*/  } |

|  |
| --- |
| ProductTestCase.java |
| package net.kzn.shoppingbackend.test;  import static org.junit.Assert.assertEquals;  import org.junit.BeforeClass;  import org.junit.Test;  import org.springframework.context.annotation.AnnotationConfigApplicationContext;  import net.kzn.shoppingbackend.dao.ProductDAO;  import net.kzn.shoppingbackend.dto.Product;  public class ProductTestCase {  private static AnnotationConfigApplicationContext context;  private static ProductDAO productDAO;  private Product product;  @BeforeClass  public static void init() {  context = new AnnotationConfigApplicationContext();  context.scan("net.kzn.shoppingbackend");  context.refresh();  productDAO = (ProductDAO)context.getBean("productDAO");  }  /\* @Test  public void testCRUDProduct() {  // create operation  product = new Product();  product.setName("Oppo Selfie S53");  product.setBrand("Oppo");  product.setDescription("This is some description for oppo mobile phones!");  product.setUnitPrice(25000);  product.setActive(true);  product.setCategoryId(3);  product.setSupplierId(3);  assertEquals("Something went wrong while inserting a new product!",  true,productDAO.add(product));  // reading and updating the category  product = productDAO.get(2);  product.setName("Samsung Galaxy S7");  assertEquals("Something went wrong while updating the existing record!",  true,productDAO.update(product));  assertEquals("Something went wrong while deleting the existing record!",  true,productDAO.delete(product));  // list  assertEquals("Something went wrong while fetching the list of products!",  6,productDAO.list().size());  }  \*/  /\*@Test  public void testListActiveProducts() {  assertEquals("Something went wrong while fetching the list of products!",  5,productDAO.listActiveProducts().size());  }  @Test  public void testListActiveProductsByCategory() {  assertEquals("Something went wrong while fetching the list of products!",  3,productDAO.listActiveProductsByCategory(3).size());  assertEquals("Something went wrong while fetching the list of products!",  2,productDAO.listActiveProductsByCategory(1).size());  }  @Test  public void testGetLatestActiveProduct() {  assertEquals("Something went wrong while fetching the list of products!",  3,productDAO.getLatestActiveProducts(3).size());  }\*/  } |

|  |
| --- |
| UserTestCase.java |
| package net.kzn.shoppingbackend.test;  import org.junit.BeforeClass;  import org.springframework.context.annotation.AnnotationConfigApplicationContext;  import net.kzn.shoppingbackend.dao.UserDAO;  import net.kzn.shoppingbackend.dto.Address;  import net.kzn.shoppingbackend.dto.Cart;  import net.kzn.shoppingbackend.dto.User;  public class UserTestCase {  private static AnnotationConfigApplicationContext context;  private static UserDAO userDAO;  private User user = null;  private Cart cart = null;  private Address address = null;  @BeforeClass  public static void init() {  context = new AnnotationConfigApplicationContext();  context.scan("net.kzn.shoppingbackend");  context.refresh();  userDAO = (UserDAO) context.getBean("userDAO");  }/\*  @Test  public void testAddUser() {  user = new User() ;  user.setFirstName("Hrithik");  user.setLastName("Roshan");  user.setEmail("hr@gmail.com");  user.setContactNumber("1234512345");  user.setRole("CUSTOMER");  user.setEnabled(true);  user.setPassword("12345");  address = new Address();  address.setAddressLineOne("101/B Jadoo Society, Krissh Nagar");  address.setAddressLineTwo("Near Kaabil Store");  address.setCity("Mumbai");  address.setState("Maharashtra");  address.setCountry("India");  address.setPostalCode("400001");  address.setBilling(true);  cart = new Cart();  // linked the address with the user  address.setUser(user);  // linked the cart with the user  cart.setUser(user);  // link the user with the cart  user.setCart(cart);  // add the user  assertEquals("Failed to add the user!", true, userDAO.add(user));  // add the address  assertEquals("Failed to add the billing address!", true, userDAO.addAddress(address));  // add the shipping address  address = new Address();  address.setAddressLineOne("201/B Jadoo Society, Kishan Kanhaiya Nagar");  address.setAddressLineTwo("Near Kudrat Store");  address.setCity("Mumbai");  address.setState("Maharashtra");  address.setCountry("India");  address.setPostalCode("400001");  address.setUser(user);  assertEquals("Failed to add the shipping address!", true, userDAO.addAddress(address));  }\*/  // working for uni-directional  /\* @Test  public void testAddAddress() {  user = userDAO.get(1);  address = new Address();  address.setAddressLineOne("301/B Jadoo Society, King Uncle Nagar");  address.setAddressLineTwo("Near Store");  address.setCity("Mumbai");  address.setState("Maharashtra");  address.setCountry("India");  address.setPostalCode("400001");  address.setUser(user);  // add the address  assertEquals("Failed to add the address!", true, userDAO.addAddress(address));  }  @Test  public void testUpdateCart() {  user = userDAO.get(1);  cart = user.getCart();  cart.setGrandTotal(10000);  cart.setCartLines(1);  assertEquals("Failed to update the cart!", true, userDAO.updateCart(cart));  } \*/  } |

**OnlineShopping(Web Application)**

|  |
| --- |
| Pom.xml |
| <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/maven-v4\_0\_0.xsd"*>  <modelVersion>4.0.0</modelVersion>  <groupId>net.kzn</groupId>  <artifactId>onlineshopping</artifactId>  <packaging>war</packaging>  <version>0.0.1-SNAPSHOT</version>  <name>onlineshopping Maven Webapp</name>  <url>http://maven.apache.org</url>  <!-- Version information will be stored here -->  <properties>  <javaee.version>7.0</javaee.version>  <spring.version>4.3.6.RELEASE</spring.version>  </properties>  <dependencies>  **<dependency>**  **<groupId>net.kzn</groupId>**  **<artifactId>shoppingbackend</artifactId>**  **<version>0.0.1-SNAPSHOT</version>**  **</dependency>**  <dependency>  <groupId>org.springframework.webflow</groupId>  <artifactId>spring-webflow</artifactId>  <version>2.4.4.RELEASE</version>  </dependency>  <dependency>  <groupId>javax</groupId>  <artifactId>javaee-api</artifactId>  <version>${javaee.version}</version>  <scope>provided</scope>  </dependency>  <dependency>  <groupId>ch.qos.logback</groupId>  <artifactId>logback-classic</artifactId>  <version>1.2.1</version>  <scope>test</scope>  </dependency>  <dependency>  <groupId>org.slf4j</groupId>  <artifactId>jcl-over-slf4j</artifactId>  <version>1.7.24</version>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-webmvc</artifactId>  <version>${spring.version}</version>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-web</artifactId>  <version>${spring.version}</version>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-core</artifactId>  <version>4.3.6.RELEASE</version>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-expression</artifactId>  <version>4.3.6.RELEASE</version>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-beans</artifactId>  <version>4.3.6.RELEASE</version>  </dependency>  <dependency>  <groupId>org.springframework.security</groupId>  <artifactId>spring-security-config</artifactId>  <version>4.2.2.RELEASE</version>  </dependency>  <dependency>  <groupId>org.springframework.security</groupId>  <artifactId>spring-security-web</artifactId>  <version>4.2.2.RELEASE</version>  </dependency>  <dependency>  <groupId>org.springframework.security</groupId>  <artifactId>spring-security-taglibs</artifactId>  <version>4.2.2.RELEASE</version>  </dependency>  <dependency>  <groupId>jstl</groupId>  <artifactId>jstl</artifactId>  <version>1.2</version>  </dependency>  <dependency>  <groupId>junit</groupId>  <artifactId>junit</artifactId>  <version>3.8.1</version>  <scope>test</scope>  </dependency>  </dependencies>  <build>  <finalName>onlineshopping</finalName>  </build>  </project> |

|  |
| --- |
| CartController.java |
| package net.kzn.onlineshopping.controller;  import org.slf4j.Logger;  import org.slf4j.LoggerFactory;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.stereotype.Controller;  import org.springframework.web.bind.annotation.PathVariable;  import org.springframework.web.bind.annotation.RequestMapping;  import org.springframework.web.bind.annotation.RequestParam;  import org.springframework.web.servlet.ModelAndView;  import net.kzn.onlineshopping.service.CartService;  @Controller  @RequestMapping("/cart")  public class CartController {  private final static Logger logger = LoggerFactory.getLogger(CartController.class);  @Autowired  private CartService cartService;  @RequestMapping("/show")  public ModelAndView showCart(@RequestParam(name = "result", required = false) String result) {  ModelAndView mv = new ModelAndView("page");  mv.addObject("title", "Shopping Cart");  mv.addObject("userClickShowCart", true);  if(result!=null) {  switch(result) {  case "added":  mv.addObject("message", "Product has been successfully added inside cart!");  cartService.validateCartLine();  break;  case "unavailable":  mv.addObject("message", "Product quantity is not available!");  break;  case "updated":  mv.addObject("message", "Cart has been updated successfully!");  cartService.validateCartLine();  break;  case "modified":  mv.addObject("message", "One or more items inside cart has been modified!");  break;  case "maximum":  mv.addObject("message", "Maximum limit for the item has been reached!");  break;  case "deleted":  mv.addObject("message", "CartLine has been successfully removed!");  break;  }  }  else {  String response = cartService.validateCartLine();  if(response.equals("result=modified")) {  mv.addObject("message", "One or more items inside cart has been modified!");  }  }  mv.addObject("cartLines", cartService.getCartLines());  return mv;  }  @RequestMapping("/{cartLineId}/update")  public String udpateCartLine(@PathVariable int cartLineId, @RequestParam int count) {  String response = cartService.manageCartLine(cartLineId, count);  return "redirect:/cart/show?"+response;  }  @RequestMapping("/add/{productId}/product")  public String addCartLine(@PathVariable int productId) {  String response = cartService.addCartLine(productId);  return "redirect:/cart/show?"+response;  }  @RequestMapping("/{cartLineId}/remove")  public String removeCartLine(@PathVariable int cartLineId) {  String response = cartService.removeCartLine(cartLineId);  return "redirect:/cart/show?"+response;  }  /\* after validating it redirect to checkout if result received is success proceed to checkout else display the message to the user  about the changes in cart page \*/  @RequestMapping("/validate")  public String validateCart() {  String response = cartService.validateCartLine();  if(!response.equals("result=success")) {  return "redirect:/cart/show?"+response;  }  else {  return "redirect:/cart/checkout";  }  }  } |

|  |
| --- |
| GlobalController.java |
| package net.kzn.onlineshopping.controller;  import javax.servlet.http.HttpSession;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.security.core.Authentication;  import org.springframework.security.core.context.SecurityContextHolder;  import org.springframework.web.bind.annotation.ControllerAdvice;  import org.springframework.web.bind.annotation.ModelAttribute;  import net.kzn.onlineshopping.model.UserModel;  import net.kzn.shoppingbackend.dao.UserDAO;  import net.kzn.shoppingbackend.dto.Cart;  import net.kzn.shoppingbackend.dto.User;  @ControllerAdvice  public class GlobalController {  @Autowired  private UserDAO userDAO;  @Autowired  private HttpSession session;  private UserModel userModel = null;  private User user = null;  @ModelAttribute("userModel")  public UserModel getUserModel() {  if(session.getAttribute("userModel")==null) {  // get the authentication object  Authentication authentication = SecurityContextHolder.getContext().getAuthentication();  if(!authentication.getPrincipal().equals("anonymousUser")){  // get the user from the database  user = userDAO.getByEmail(authentication.getName());  if(user!=null) {  // create a new model  userModel = new UserModel();  // set the name and the id  userModel.setId(user.getId());  userModel.setFullName(user.getFirstName() + " " + user.getLastName());  userModel.setRole(user.getRole());  if(user.getRole().equals("USER")) {  userModel.setCart(user.getCart());  }  session.setAttribute("userModel", userModel);  return userModel;  }  }  }  return (UserModel) session.getAttribute("userModel");  }  } |

|  |
| --- |
| JsonDataController.java |
| package net.kzn.onlineshopping.controller;  import java.util.List;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.stereotype.Controller;  import org.springframework.web.bind.annotation.PathVariable;  import org.springframework.web.bind.annotation.RequestMapping;  import org.springframework.web.bind.annotation.ResponseBody;  import net.kzn.shoppingbackend.dao.ProductDAO;  import net.kzn.shoppingbackend.dto.Product;  @Controller  @RequestMapping("/json/data")  public class JsonDataController {  @Autowired  private ProductDAO productDAO;  @RequestMapping("/admin/all/products")  @ResponseBody  public List<Product> getAllProductsList() {  return productDAO.list();  }  @RequestMapping("/all/products")  @ResponseBody  public List<Product> getAllProducts() {  return productDAO.listActiveProducts();  }  @RequestMapping("/category/{id}/products")  @ResponseBody  public List<Product> getProductsByCategory(@PathVariable int id) {  return productDAO.listActiveProductsByCategory(id);  }  @RequestMapping("/mv/products")  @ResponseBody  public List<Product> getMostViewedProducts() {  return productDAO.getProductsByParam("views", 5);  }  @RequestMapping("/mp/products")  @ResponseBody  public List<Product> getMostPurchasedProducts() {  return productDAO.getProductsByParam("purchases", 5);  }  } |

|  |
| --- |
| ManagementController.java |
| package net.kzn.onlineshopping.controller;  import java.util.List;  import javax.servlet.http.HttpServletRequest;  import javax.validation.Valid;  import org.slf4j.Logger;  import org.slf4j.LoggerFactory;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.stereotype.Controller;  import org.springframework.ui.Model;  import org.springframework.validation.BindingResult;  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.ResponseBody;  import org.springframework.web.servlet.ModelAndView;  import net.kzn.onlineshopping.util.FileUtil;  import net.kzn.onlineshopping.validator.ProductValidator;  import net.kzn.shoppingbackend.dao.CategoryDAO;  import net.kzn.shoppingbackend.dao.ProductDAO;  import net.kzn.shoppingbackend.dto.Category;  import net.kzn.shoppingbackend.dto.Product;  @Controller  @RequestMapping("/manage")  public class ManagementController {  private static final Logger logger = LoggerFactory.getLogger(ManagementController.class);  @Autowired  private ProductDAO productDAO;  @Autowired  private CategoryDAO categoryDAO;  @RequestMapping("/product")  public ModelAndView manageProduct(@RequestParam(name="success",required=false)String success) {  ModelAndView mv = new ModelAndView("page");  mv.addObject("title","Product Management");  mv.addObject("userClickManageProduct",true);  Product nProduct = new Product();  // assuming that the user is ADMIN  // later we will fixed it based on user is SUPPLIER or ADMIN  nProduct.setSupplierId(1);  nProduct.setActive(true);  mv.addObject("product", nProduct);  if(success != null) {  if(success.equals("product")){  mv.addObject("message", "Product submitted successfully!");  }  else if (success.equals("category")) {  mv.addObject("message", "Category submitted successfully!");  }  }  return mv;  }  @RequestMapping("/{id}/product")  public ModelAndView manageProductEdit(@PathVariable int id) {  ModelAndView mv = new ModelAndView("page");  mv.addObject("title","Product Management");  mv.addObject("userClickManageProduct",true);  // Product nProduct = new Product();  mv.addObject("product", productDAO.get(id));  return mv;  }  @RequestMapping(value = "/product", method=RequestMethod.POST)  public String managePostProduct(@Valid @ModelAttribute("product") Product mProduct,  BindingResult results, Model model, HttpServletRequest request) {  // mandatory file upload check  if(mProduct.getId() == 0) {  new ProductValidator().validate(mProduct, results);  }  else {  // edit check only when the file has been selected  if(!mProduct.getFile().getOriginalFilename().equals("")) {  new ProductValidator().validate(mProduct, results);  }  }  if(results.hasErrors()) {  model.addAttribute("message", "Validation fails for adding the product!");  model.addAttribute("userClickManageProduct",true);  return "page";  }  if(mProduct.getId() == 0 ) {  productDAO.add(mProduct);  }  else {  productDAO.update(mProduct);  }  //upload the file  if(!mProduct.getFile().getOriginalFilename().equals("") ){  FileUtil.uploadFile(request, mProduct.getFile(), mProduct.getCode());  }  return "redirect:/manage/product?success=product";  }  @RequestMapping(value = "/product/{id}/activation", method=RequestMethod.GET)  @ResponseBody  public String managePostProductActivation(@PathVariable int id) {  Product product = productDAO.get(id);  boolean isActive = product.isActive();  product.setActive(!isActive);  productDAO.update(product);  return (isActive)? "Product Dectivated Successfully!": "Product Activated Successfully";  }  @RequestMapping(value = "/category", method=RequestMethod.POST)  public String managePostCategory(@ModelAttribute("category") Category mCategory, HttpServletRequest request) {  categoryDAO.add(mCategory);  return "redirect:" + request.getHeader("Referer") + "?success=category";  }  @ModelAttribute("categories")  public List<Category> modelCategories() {  return categoryDAO.list();  }  @ModelAttribute("category")  public Category modelCategory() {  return new Category();  }  } |

|  |
| --- |
| PageController.java |
| package net.kzn.onlineshopping.controller;  import java.io.PrintWriter;  import java.io.StringWriter;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import org.slf4j.Logger;  import org.slf4j.LoggerFactory;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.security.core.Authentication;  import org.springframework.security.core.context.SecurityContextHolder;  import org.springframework.security.web.authentication.logout.SecurityContextLogoutHandler;  import org.springframework.stereotype.Controller;  import org.springframework.web.bind.annotation.PathVariable;  import org.springframework.web.bind.annotation.RequestMapping;  import org.springframework.web.bind.annotation.RequestParam;  import org.springframework.web.servlet.ModelAndView;  import net.kzn.onlineshopping.exception.ProductNotFoundException;  import net.kzn.shoppingbackend.dao.CategoryDAO;  import net.kzn.shoppingbackend.dao.ProductDAO;  import net.kzn.shoppingbackend.dto.Category;  import net.kzn.shoppingbackend.dto.Product;  @Controller  public class PageController {  private static final Logger logger = LoggerFactory.getLogger(PageController.class);  @Autowired  private CategoryDAO categoryDAO;  @Autowired  private ProductDAO productDAO;  @RequestMapping(value = {"/", "/home", "/index"})  public ModelAndView index(@RequestParam(name="logout",required=false)String logout) {  ModelAndView mv = new ModelAndView("page");  mv.addObject("title","Home");  logger.info("Inside PageController index method - INFO");  logger.debug("Inside PageController index method - DEBUG");  //passing the list of categories  mv.addObject("categories", categoryDAO.list());  if(logout!=null) {  mv.addObject("message", "You have successfully logged out!");  }  mv.addObject("userClickHome",true);  return mv;  }  @RequestMapping(value = "/about")  public ModelAndView about() {  ModelAndView mv = new ModelAndView("page");  mv.addObject("title","About Us");  mv.addObject("userClickAbout",true);  return mv;  }  @RequestMapping(value = "/contact")  public ModelAndView contact() {  ModelAndView mv = new ModelAndView("page");  mv.addObject("title","Contact Us");  mv.addObject("userClickContact",true);  return mv;  }  /\* Methods to load all the products and based on category\*/  @RequestMapping(value = "/show/all/products")  public ModelAndView showAllProducts() {  ModelAndView mv = new ModelAndView("page");  mv.addObject("title","All Products");  //passing the list of categories  mv.addObject("categories", categoryDAO.list());  mv.addObject("userClickAllProducts",true);  return mv;  }  @RequestMapping(value = "/show/category/{id}/products")  public ModelAndView showCategoryProducts(@PathVariable("id") int id) {  ModelAndView mv = new ModelAndView("page");  // categoryDAO to fetch a single category  Category category = null;  category = categoryDAO.get(id);  mv.addObject("title",category.getName());  //passing the list of categories  mv.addObject("categories", categoryDAO.list());  // passing the single category object  mv.addObject("category", category);  mv.addObject("userClickCategoryProducts",true);  return mv;  }  /\*Viewing a single product\*/  @RequestMapping(value = "/show/{id}/product")  public ModelAndView showSingleProduct(@PathVariable int id) throws ProductNotFoundException {  ModelAndView mv = new ModelAndView("page");  Product product = productDAO.get(id);  if(product == null) throw new ProductNotFoundException();  // update the view count  product.setViews(product.getViews() + 1);  productDAO.update(product);  //---------------------------  mv.addObject("title", product.getName());  mv.addObject("product", product);  mv.addObject("userClickShowProduct", true);  return mv;  }  @RequestMapping(value="/membership")  public ModelAndView register() {  ModelAndView mv= new ModelAndView("page");  logger.info("Page Controller membership called!");  return mv;  }  @RequestMapping(value="/login")  public ModelAndView login(@RequestParam(name="error", required = false) String error,  @RequestParam(name="logout", required = false) String logout) {  ModelAndView mv= new ModelAndView("login");  mv.addObject("title", "Login");  if(error!=null) {  mv.addObject("message", "Username and Password is invalid!");  }  if(logout!=null) {  mv.addObject("logout", "You have logged out successfully!");  }  return mv;  }  @RequestMapping(value="/logout")  public String logout(HttpServletRequest request, HttpServletResponse response) {  // Invalidates HTTP Session, then unbinds any objects bound to it.  // Removes the authentication from securitycontext  Authentication auth = SecurityContextHolder.getContext().getAuthentication();  if (auth != null){  new SecurityContextLogoutHandler().logout(request, response, auth);  }  return "redirect:/login?logout";  }  @RequestMapping(value="/access-denied")  public ModelAndView accessDenied() {  ModelAndView mv = new ModelAndView("error");  mv.addObject("errorTitle", "Aha! Caught You.");  mv.addObject("errorDescription", "You are not authorized to view this page!");  mv.addObject("title", "403 Access Denied");  return mv;  }  } |

|  |
| --- |
| GlobalDefaultExceptionHandler.java |
| package net.kzn.onlineshopping.exception;  import java.io.PrintWriter;  import java.io.StringWriter;  import org.springframework.web.bind.annotation.ControllerAdvice;  import org.springframework.web.bind.annotation.ExceptionHandler;  import org.springframework.web.servlet.ModelAndView;  import org.springframework.web.servlet.NoHandlerFoundException;  @ControllerAdvice  public class GlobalDefaultExceptionHandler {  @ExceptionHandler(NoHandlerFoundException.class)  public ModelAndView handlerNoHandlerFoundException() {  ModelAndView mv = new ModelAndView("error");  mv.addObject("errorTitle", "The page is not constructed!");  mv.addObject("errorDescription", "The page you are looking for is not available now!");  mv.addObject("title", "404 Error Page");  return mv;  }  @ExceptionHandler(ProductNotFoundException.class)  public ModelAndView handlerProductNotFoundException() {  ModelAndView mv = new ModelAndView("error");  mv.addObject("errorTitle", "Product not available!");  mv.addObject("errorDescription", "The product you are looking for is not available right now!");  mv.addObject("title", "Product Unavailable");  return mv;  }  @ExceptionHandler(Exception.class)  public ModelAndView handlerException(Exception ex) {  ModelAndView mv = new ModelAndView("error");  mv.addObject("errorTitle", "Contact Your Administrator!!");  /\* only for debugging your application\*/  StringWriter sw = new StringWriter();  PrintWriter pw = new PrintWriter(sw);  ex.printStackTrace(pw);  mv.addObject("errorDescription", sw.toString());  mv.addObject("title", "Error");  return mv;  }  } |

|  |
| --- |
| ProductNotFoundException.java |
| **package** net.kzn.onlineshopping.exception;  **import** java.io.Serializable;  **public** **class** ProductNotFoundException **extends** Exception **implements** Serializable {  **private** **static** **final** **long** ***serialVersionUID*** = 1L;  **private** String message;  **public** ProductNotFoundException() {  **this**("Product is not available!");  }  **public** ProductNotFoundException(String message) {  **this**.message = System.*currentTimeMillis*() + ": " + message;  }  **public** String getMessage() {  **return** message;  }  } |

|  |
| --- |
| CheckoutHandler.java |
| package net.kzn.onlineshopping.handler;  import java.util.ArrayList;  import java.util.Date;  import java.util.List;  import javax.servlet.http.HttpSession;  import org.slf4j.Logger;  import org.slf4j.LoggerFactory;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.stereotype.Component;  import net.kzn.onlineshopping.model.CheckoutModel;  import net.kzn.onlineshopping.model.UserModel;  import net.kzn.shoppingbackend.dao.CartLineDAO;  import net.kzn.shoppingbackend.dao.ProductDAO;  import net.kzn.shoppingbackend.dao.UserDAO;  import net.kzn.shoppingbackend.dto.Address;  import net.kzn.shoppingbackend.dto.Cart;  import net.kzn.shoppingbackend.dto.CartLine;  import net.kzn.shoppingbackend.dto.OrderDetail;  import net.kzn.shoppingbackend.dto.OrderItem;  import net.kzn.shoppingbackend.dto.Product;  import net.kzn.shoppingbackend.dto.User;  @Component  public class CheckoutHandler {  private static final Logger logger = LoggerFactory.getLogger(CheckoutHandler.class);  @Autowired  private UserDAO userDAO;  @Autowired  private ProductDAO productDAO;  @Autowired  private CartLineDAO cartLineDAO;  @Autowired  private HttpSession session;  public CheckoutModel init(String name) throws Exception{  User user = userDAO.getByEmail(name);  CheckoutModel checkoutModel = null;  if(user!=null) {  checkoutModel = new CheckoutModel();  checkoutModel.setUser(user);  checkoutModel.setCart(user.getCart());  double checkoutTotal = 0.0;  List<CartLine> cartLines = cartLineDAO.listAvailable(user.getCart().getId());  if(cartLines.size() == 0 ) {  throw new Exception("There are no products available for checkout now!");  }  for(CartLine cartLine: cartLines) {  checkoutTotal += cartLine.getTotal();  }  checkoutModel.setCartLines(cartLines);  checkoutModel.setCheckoutTotal(checkoutTotal);  }  return checkoutModel;  }  public List<Address> getShippingAddresses(CheckoutModel model) {  List<Address> addresses = userDAO.listShippingAddresses(model.getUser().getId());  if(addresses.size() == 0) {  addresses = new ArrayList<>();  }  addresses.add(addresses.size(), userDAO.getBillingAddress(model.getUser().getId()));  return addresses;  }  public String saveAddressSelection(CheckoutModel checkoutModel, int shippingId) {  String transitionValue = "success";  //logger.info(String.valueOf(shippingId));  Address shipping = userDAO.getAddress(shippingId);  checkoutModel.setShipping(shipping);  return transitionValue;  }  public String saveAddress(CheckoutModel checkoutModel, Address shipping) {  String transitionValue = "success";  // set the user id  // set shipping as true  shipping.setUserId(checkoutModel.getUser().getId());  shipping.setShipping(true);  userDAO.addAddress(shipping);  // set the shipping id to flowScope object  checkoutModel.setShipping(shipping);  return transitionValue;  }  public String saveOrder(CheckoutModel checkoutModel) {  String transitionValue = "success";  // create a new order object  OrderDetail orderDetail = new OrderDetail();  // attach the user  orderDetail.setUser(checkoutModel.getUser());  // attach the shipping address  orderDetail.setShipping(checkoutModel.getShipping());  // fetch the billing address  Address billing = userDAO.getBillingAddress(checkoutModel.getUser().getId());  orderDetail.setBilling(billing);  List<CartLine> cartLines = checkoutModel.getCartLines();  OrderItem orderItem = null;  double orderTotal = 0.0;  int orderCount = 0;  Product product = null;  for(CartLine cartLine : cartLines) {  orderItem = new OrderItem();  orderItem.setBuyingPrice(cartLine.getBuyingPrice());  orderItem.setProduct(cartLine.getProduct());  orderItem.setProductCount(cartLine.getProductCount());  orderItem.setTotal(cartLine.getTotal());  orderItem.setOrderDetail(orderDetail);  orderDetail.getOrderItems().add(orderItem);  orderTotal += orderItem.getTotal();  orderCount++;  // update the product  // reduce the quantity of product  product = cartLine.getProduct();  product.setQuantity(product.getQuantity() - cartLine.getProductCount());  product.setPurchases(product.getPurchases() + cartLine.getProductCount());  productDAO.update(product);  // delete the cartLine  cartLineDAO.remove(cartLine);  }  orderDetail.setOrderTotal(orderTotal);  orderDetail.setOrderCount(orderCount);  orderDetail.setOrderDate(new Date());  // save the order  cartLineDAO.addOrderDetail(orderDetail);  // set it to the orderDetails of checkoutModel  checkoutModel.setOrderDetail(orderDetail);  // update the cart  Cart cart = checkoutModel.getCart();  cart.setGrandTotal(cart.getGrandTotal() - orderTotal);  cart.setCartLines(cart.getCartLines() - orderCount);  cartLineDAO.updateCart(cart);  // update the cart if its in the session  UserModel userModel = (UserModel) session.getAttribute("userModel");  if(userModel!=null) {  userModel.setCart(cart);  }  return transitionValue;  }  public OrderDetail getOrderDetail(CheckoutModel checkoutModel) {  return checkoutModel.getOrderDetail();  }  } |

|  |
| --- |
| RegisterHandler.java |
| package net.kzn.onlineshopping.handler;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.binding.message.MessageBuilder;  import org.springframework.binding.message.MessageContext;  import org.springframework.security.crypto.password.PasswordEncoder;  import org.springframework.stereotype.Component;  import net.kzn.onlineshopping.model.RegisterModel;  import net.kzn.shoppingbackend.dao.UserDAO;  import net.kzn.shoppingbackend.dto.Address;  import net.kzn.shoppingbackend.dto.Cart;  import net.kzn.shoppingbackend.dto.User;  @Component  public class RegisterHandler  @Autowired  private PasswordEncoder passwordEncoder;  @Autowired  private UserDAO userDAO;  public RegisterModel init() {  return new RegisterModel();  }  public void addUser(RegisterModel registerModel, User user) {  registerModel.setUser(user);  }  public void addBilling(RegisterModel registerModel, Address billing) {  registerModel.setBilling(billing);  }  public String validateUser(User user, MessageContext error) {  String transitionValue = "success";  if(!user.getPassword().equals(user.getConfirmPassword())) {  error.addMessage(new MessageBuilder().error().source(  "confirmPassword").defaultText("Password does not match confirm password!").build());  transitionValue = "failure";  }  if(userDAO.getByEmail(user.getEmail())!=null) {  error.addMessage(new MessageBuilder().error().source(  "email").defaultText("Email address is already taken!").build());  transitionValue = "failure";  }  return transitionValue;  }  public String saveAll(RegisterModel registerModel) {  String transitionValue = "success";  User user = registerModel.getUser();  if(user.getRole().equals("USER")) {  // create a new cart  Cart cart = new Cart();  cart.setUser(user);  user.setCart(cart);  }  // encode the password  user.setPassword(passwordEncoder.encode(user.getPassword()));  // save the user  userDAO.add(user);  // save the billing address  Address billing = registerModel.getBilling();  billing.setUserId(user.getId());  billing.setBilling(true);  userDAO.addAddress(billing);  return transitionValue ;  }  } |

|  |
| --- |
| CheckOutModel.java |
| package net.kzn.onlineshopping.model;  import java.io.Serializable;  import java.util.List;  import net.kzn.shoppingbackend.dto.Address;  import net.kzn.shoppingbackend.dto.Cart;  import net.kzn.shoppingbackend.dto.CartLine;  import net.kzn.shoppingbackend.dto.OrderDetail;  import net.kzn.shoppingbackend.dto.User;  public class CheckoutModel implements Serializable {  private static final long serialVersionUID = 1L;  private User user;  private Address shipping;  private Cart cart;  private List<CartLine> cartLines;  private OrderDetail orderDetail;  private double checkoutTotal;  public OrderDetail getOrderDetail() {  return orderDetail;  }  public void setOrderDetail(OrderDetail orderDetail) {  this.orderDetail = orderDetail;  }  public Cart getCart() {  return cart;  }  public void setCart(Cart cart) {  this.cart = cart;  }  public double getCheckoutTotal() {  return checkoutTotal;  }  public void setCheckoutTotal(double checkoutTotal) {  this.checkoutTotal = checkoutTotal;  }  public List<CartLine> getCartLines() {  return cartLines;  }  public void setCartLines(List<CartLine> cartLines) {  this.cartLines = cartLines;  }  public User getUser() {  return user;  }  public void setUser(User user) {  this.user = user;  }  public Address getShipping() {  return shipping;  }  public void setShipping(Address shipping) {  this.shipping = shipping;  }  } |

|  |
| --- |
| RegisterModel.java |
| package net.kzn.onlineshopping.model;  import java.io.Serializable;  import net.kzn.shoppingbackend.dto.Address;  import net.kzn.shoppingbackend.dto.User;  public class RegisterModel implements Serializable {  private static final long serialVersionUID = 1L;  private User user;  private Address billing;  public User getUser() {  return user;  }  public void setUser(User user) {  this.user = user;  }  public Address getBilling() {  return billing;  }  public void setBilling(Address billing) {  this.billing = billing;  }  } |

|  |
| --- |
| UserModel.java |
| package net.kzn.onlineshopping.model;  import java.io.Serializable;  import net.kzn.shoppingbackend.dto.Cart;  public class UserModel implements Serializable {  private static final long serialVersionUID = 1L;  private int id;  private String fullName;  private String role;  public String getRole() {  return role;  }  public void setRole(String role) {  this.role = role;  }  private Cart cart;  public int getId() {  return id;  }  public void setId(int id) {  this.id = id;  }  public String getFullName() {  return fullName;  }  public void setFullName(String fullName) {  this.fullName = fullName;  }  public Cart getCart() {  return cart;  }  public void setCart(Cart cart) {  this.cart = cart;  }  } |

|  |
| --- |
| CartService.java |
| package net.kzn.onlineshopping.service;  import java.util.List;  import javax.servlet.http.HttpSession;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.stereotype.Service;  import net.kzn.onlineshopping.model.UserModel;  import net.kzn.shoppingbackend.dao.CartLineDAO;  import net.kzn.shoppingbackend.dao.ProductDAO;  import net.kzn.shoppingbackend.dao.UserDAO;  import net.kzn.shoppingbackend.dto.Cart;  import net.kzn.shoppingbackend.dto.CartLine;  import net.kzn.shoppingbackend.dto.Product;  @Service("cartService")  public class CartService {  @Autowired  private CartLineDAO cartLineDAO;  @Autowired  private ProductDAO productDAO;  @Autowired  private HttpSession session;  public List<CartLine> getCartLines() {  return cartLineDAO.list(getCart().getId());  }  /\* to update the cart count \*/  public String manageCartLine(int cartLineId, int count) {  CartLine cartLine = cartLineDAO.get(cartLineId);  double oldTotal = cartLine.getTotal();  Product product = cartLine.getProduct();  // check if that much quantity is available or not  if(product.getQuantity() < count) {  return "result=unavailable";  }  // update the cart line  cartLine.setProductCount(count);  cartLine.setBuyingPrice(product.getUnitPrice());  cartLine.setTotal(product.getUnitPrice() \* count);  cartLineDAO.update(cartLine);  // update the cart  Cart cart = this.getCart();  cart.setGrandTotal(cart.getGrandTotal() - oldTotal + cartLine.getTotal());  cartLineDAO.updateCart(cart);  return "result=updated";  }  public String addCartLine(int productId) {  Cart cart = this.getCart();  String response = null;  CartLine cartLine = cartLineDAO.getByCartAndProduct(cart.getId(), productId);  if(cartLine==null) {  // add a new cartLine if a new product is getting added  cartLine = new CartLine();  Product product = productDAO.get(productId);  // transfer the product details to cartLine  cartLine.setCartId(cart.getId());  cartLine.setProduct(product);  cartLine.setProductCount(1);  cartLine.setBuyingPrice(product.getUnitPrice());  cartLine.setTotal(product.getUnitPrice());  // insert a new cartLine  cartLineDAO.add(cartLine);  // update the cart  cart.setGrandTotal(cart.getGrandTotal() + cartLine.getTotal());  cart.setCartLines(cart.getCartLines() + 1);  cartLineDAO.updateCart(cart);  response = "result=added";  }  else {  // check if the cartLine has been already reached to maximum count  if(cartLine.getProductCount() < 3) {  // call the manageCartLine method to increase the count  response = this.manageCartLine(cartLine.getId(), cartLine.getProductCount() + 1);  }  else {  response = "result=maximum";  }  }  return response;  }  private Cart getCart() {  return ((UserModel)session.getAttribute("userModel")).getCart();  }  public String removeCartLine(int cartLineId) {  CartLine cartLine = cartLineDAO.get(cartLineId);  // deduct the cart  // update the cart  Cart cart = this.getCart();  cart.setGrandTotal(cart.getGrandTotal() - cartLine.getTotal());  cart.setCartLines(cart.getCartLines() - 1);  cartLineDAO.updateCart(cart);  // remove the cartLine  cartLineDAO.remove(cartLine);  return "result=deleted";  }  public String validateCartLine() {  Cart cart = this.getCart();  List<CartLine> cartLines = cartLineDAO.list(cart.getId());  double grandTotal = 0.0;  int lineCount = 0;  String response = "result=success";  boolean changed = false;  Product product = null;  for(CartLine cartLine : cartLines) {  product = cartLine.getProduct();  changed = false;  // check if the product is active or not  // if it is not active make the availability of cartLine as false  if((!product.isActive() && product.getQuantity() == 0) && cartLine.isAvailable()) {  cartLine.setAvailable(false);  changed = true;  }  // check if the cartLine is not available  // check whether the product is active and has at least one quantity available  if((product.isActive() && product.getQuantity() > 0) && !(cartLine.isAvailable())) {  cartLine.setAvailable(true);  changed = true;  }  // check if the buying price of product has been changed  if(cartLine.getBuyingPrice() != product.getUnitPrice()) {  // set the buying price to the new price  cartLine.setBuyingPrice(product.getUnitPrice());  // calculate and set the new total  cartLine.setTotal(cartLine.getProductCount() \* product.getUnitPrice());  changed = true;  }  // check if that much quantity of product is available or not  if(cartLine.getProductCount() > product.getQuantity()) {  cartLine.setProductCount(product.getQuantity());  cartLine.setTotal(cartLine.getProductCount() \* product.getUnitPrice());  changed = true;  }  // changes has happened  if(changed) {  //update the cartLine  cartLineDAO.update(cartLine);  // set the result as modified  response = "result=modified";  }  grandTotal += cartLine.getTotal();  lineCount++;  }  cart.setCartLines(lineCount++);  cart.setGrandTotal(grandTotal);  cartLineDAO.updateCart(cart);  return response;  }  } |

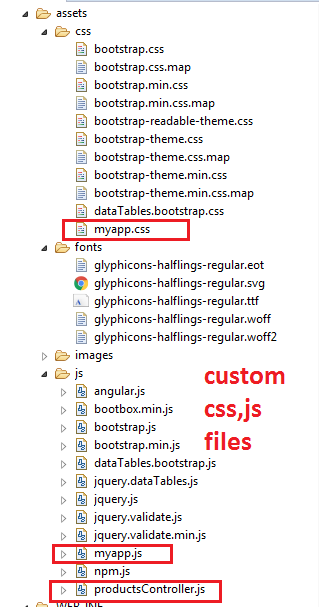
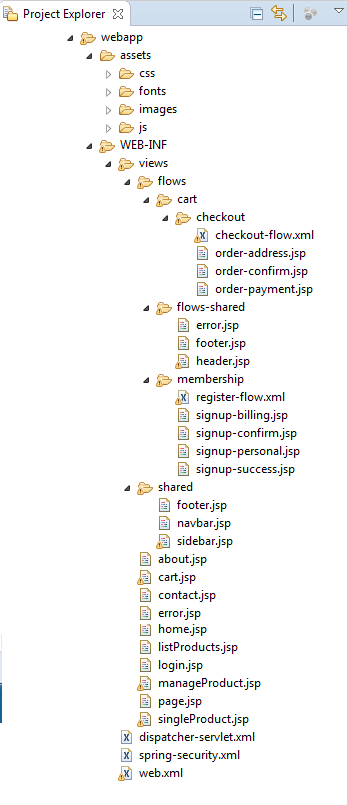
|  |
| --- |
| FileUtil.java |
| package net.kzn.onlineshopping.util;  import java.io.File;  import java.io.FileOutputStream;  import java.io.IOException;  import java.io.InputStream;  import java.io.OutputStream;  import java.net.URL;  import javax.servlet.http.HttpServletRequest;  import org.slf4j.Logger;  import org.slf4j.LoggerFactory;  import org.springframework.web.multipart.MultipartFile;  public class FileUtil {  private static final String ABS\_PATH = "D:/WORKSPACE/LiveProject\_MASTER/onlineshopping/src/main/webapp/assets/images/";  private static String REAL\_PATH = null;  private static final Logger logger = LoggerFactory.getLogger(FileUtil.class);  public static boolean uploadFile(HttpServletRequest request, MultipartFile file, String code)  {  // get the real server path  REAL\_PATH = request.getSession().getServletContext().getRealPath("/assets/images/");  logger.info(REAL\_PATH);  // create the directories if it does not exist  if(!new File(REAL\_PATH).exists()) {  new File(REAL\_PATH).mkdirs();  }  if(!new File(ABS\_PATH).exists()) {  new File(ABS\_PATH).mkdirs();  }  try {  //transfer the file to both the location  file.transferTo(new File(REAL\_PATH + code + ".jpg"));  file.transferTo(new File(ABS\_PATH + code + ".jpg"));  }  catch(IOException ex) {  ex.printStackTrace();  }  return true;  }  public static void uploadNoImage(HttpServletRequest request, String code) {  // get the real server path  REAL\_PATH = request.getSession().getServletContext().getRealPath("/assets/images/");  String imageURL = "http://placehold.it/640X480?text=No Image";  String destinationServerFile = REAL\_PATH + code + ".jpg";  String destinationProjectFile = REAL\_PATH + code + ".jpg";  try {  URL url = new URL(imageURL);  try (  InputStream is = url.openStream();  OutputStream osREAL\_PATH = new FileOutputStream(destinationServerFile);  OutputStream osABS\_PATH = new FileOutputStream(destinationProjectFile);  ){  byte[] b = new byte[2048];  int length;  while((length = is.read(b))!= -1) {  osREAL\_PATH.write(b, 0, length);  osABS\_PATH.write(b, 0, length);  }  }  }  catch(IOException ex) {  ex.printStackTrace();  }  }  } |

|  |
| --- |
| ProductValidator.java |
| package net.kzn.onlineshopping.validator;  import org.springframework.validation.Errors;  import org.springframework.validation.Validator;  import net.kzn.shoppingbackend.dto.Product;  public class ProductValidator implements Validator {  @Override  public boolean supports(Class<?> clazz) {  return Product.class.equals(clazz);  }  @Override  public void validate(Object target, Errors errors) {  Product product = (Product) target;  if(product.getFile() == null || product.getFile().getOriginalFilename().equals("")) {  errors.rejectValue("file", null, "Please select a file to upload!");  return;  }  if(! (product.getFile().getContentType().equals("image/jpeg") ||  product.getFile().getContentType().equals("image/png")) ||  product.getFile().getContentType().equals("image/gif")  )  {  errors.rejectValue("file", null, "Please select an image file to upload!");  return;  }  }  } |

|  |
| --- |
| In src/main/resources/logback.xml |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <configuration>  <appender name=*"consoleAppender"* class=*"ch.qos.logback.core.ConsoleAppender"*>  <encoder>  <Pattern>%-5level %msg %d{HH:mm:ss.SSS} %logger{36} %n</Pattern>  </encoder>  </appender>  <logger name=*"net.kzn"* level=*"DEBUG"* additivity = *"false"*>  <appender-ref ref=*"consoleAppender"*/>  </logger>  <root level=*"INFO"*>  <appender-ref ref=*"consoleAppender"*/>  </root>  </configuration> |

UI Design:

webapp directory structure. And inside asserts (sub folder)



|  |
| --- |
| Inside webapp/asserts/css/myapp.css |
| /\*!  \* Start Bootstrap - Shop Homepage (http://startbootstrap.com/)  \* Copyright 2013-2016 Start Bootstrap  \* Licensed under MIT (https://github.com/BlackrockDigital/startbootstrap/blob/gh-pages/LICENSE)  \*/  **html** {  height:*100%*;  }  **body** {  padding-top: *70px*; /\* Required padding for .navbar-fixed-top. Remove if using .navbar-static-top. Change if height of navigation changes. \*/  height:*100%*;  }  *.slide-image* {  width: *100%*;  }  *.carousel-holder* {  margin-bottom: *30px*;  }  *.carousel-control***,**  *.item* {  border-radius: *4px*;  }  *.caption* {  height: *130px*;  overflow: *hidden*;  }  *.caption* **h4** {  white-space: *nowrap*;  }  *.thumbnail* **img** {  width: *100%*;  }  *.ratings* {  padding-right: *10px*;  padding-left: *10px*;  color: *#d17581*;  }  *.thumbnail* {  padding: *0*;  }  *.thumbnail* *.caption-full* {  padding: *9px*;  color: *#333*;  }  **footer** {  margin: *0*;  }  /\*custom styling\*/  *.wrapper* {  min-height:*100%*;  position: *relative*;  }  *.content* {  padding-bottom: *60px*; /\*height of the footer\*/  }  *.footer* {  width:*100%*;  position:*absolute*;  bottom:*20px*;  height: *60px*;  }  *.dataTableImg* {  width:*100px*;  height: *100px*;    }  *.has-success* {  color: *#4582ec*;  }  /\*Toggle Switch code\*/  /\* The switch - the box around the slider \*/  *.switch* {  position: *relative*;  display: *inline-block*;  width: *60px*;  height: *34px*;  }  /\* Hide default HTML checkbox \*/  *.switch* **input** {display:*none*;}  /\* The slider \*/  *.slider* {  position: *absolute*;  cursor: *pointer*;  top: *0*;  left: *0*;  right: *0*;  bottom: *0*;  background-color: *#ccc*;  -webkit-transition: *.4s*;  transition: *.4s*;  }  *.slider:before* {  position: *absolute*;  content: *""*;  height: *26px*;  width: *26px*;  left: *4px*;  bottom: *4px*;  background-color: *white*;  -webkit-transition: *.4s*;  transition: *.4s*;  }  **input***:checked* + *.slider* {  background-color: *#2196F3*;  }  **input***:focus* + *.slider* {  box-shadow: *0* *0* *1px* *#2196F3*;  }  **input***:checked* + *.slider:before* {  -webkit-transform: *translateX(26px)*;  -ms-transform: *translateX(26px)*;  transform: *translateX(26px)*;  }  /\* Rounded sliders \*/  *.slider.round* {  border-radius: *34px*;  }  *.slider.round:before* {  border-radius: *50%*;  }  /\* overridden css \*/  *.help-block* {  color: *#ff0000*;  }  /\*Shopping Cart\*/  *.table*>**tbody**>**tr**>**td,** *.table*>**tfoot**>**tr**>**td**{  vertical-align: *middle*;  }  @media screen and (max-width: 600px) {  **table***#cart* **tbody** **td** *.form-control*{  width:*20%*;  display: *inline* *!important*;  }  *.actions* *.btn*{  width:*36%*;  margin:*1.5em* *0*;  }  *.actions* *.btn-info*{  float:*left*;  }  *.actions* *.btn-danger*{  float:*right*;  }  **table***#cart* **thead** { display: *none*; }  **table***#cart* **tbody** **td** { display: *block*; padding: *.6rem*; min-width:*320px*;}  **table***#cart* **tbody** **tr** **td***:first-child* { background: *#333*; color: *#fff*; }  **table***#cart* **tbody** **td***:before* {  content: *attr(data-th)*; font-weight: *bold*;  display: *inline-block*; width: *8rem*;  }  **table***#cart* **tfoot** **td**{display:*block*; }  **table***#cart* **tfoot** **td** *.btn*{display:*block*;}  }  /\* cart checkout \*/  *.panel-title* {display: *inline*;font-weight: *bold*;}  *.checkbox.pull-right* { margin: *0*; }  *.pl-ziro* { padding-left: *0px*; }  /\* order receipt \*/  *.invoice-title* **h2,** *.invoice-title* **h3** {  display: *inline-block*;  }  *.table* > **tbody** > **tr** > *.no-line* {  border-top: *none*;  }  *.table* > **thead** > **tr** > *.no-line* {  border-bottom: *none*;  }  *.table* > **tbody** > **tr** > *.thick-line* {  border-top: *2px* *solid*;  }  /\* landing css \*/  *.landingImg* {  width: *auto*;  height : *180px*;  }  *.se-pre-con*{  position:*fixed*;  left:*0*;  top:*0*;  width:*98%*;  height:*100%*;  z-index:*9999*;  background:*url(../images/pre-loader.gif)* *50%* *no-repeat* *#fff*  } |

|  |
| --- |
| Inside webapp/asserts/js/myapp.js |
| $(**function**() {  // for adding a loader  $(window).load(**function**(){  setTimeout(**function**() {  $(".se-pre-con").fadeOut("slow");  }, 500);  });  // for handling CSRF token  **var** token = $('meta[name="\_csrf"]').attr('content');  **var** header = $('meta[name="\_csrf\_header"]').attr('content');    **if**((token!=**undefined** && header !=**undefined**) && (token.length > 0 && header.length > 0)) {  // set the token header for the ajax request  $(document).ajaxSend(**function**(e, xhr, options) {  xhr.setRequestHeader(header,token);  });  }  // solving the active menu problem  **switch** (menu) {  **case** 'About Us':  $('#about').addClass('active');  **break**;  **case** 'Contact Us':  $('#contact').addClass('active');  **break**;  **case** 'All Products':  $('#listProducts').addClass('active');  **break**;  **case** 'Product Management':  $('#manageProduct').addClass('active');  **break**;  **case** 'Shopping Cart':  $('#userModel').addClass('active');  **break**;  **default**:  **if** (menu == "Home")  **break**;  $('#listProducts').addClass('active');  $('#a\_' + menu).addClass('active');  **break**;  }  // code for jquery dataTable  **var** $table = $('#productListTable');  // execute the below code only where we have this table  **if** ($table.length) {  // console.log('Inside the table!');  **var** jsonUrl = '';  **if** (window.categoryId == '') {  jsonUrl = window.contextRoot + '/json/data/all/products';  } **else** {  jsonUrl = window.contextRoot + '/json/data/category/'  + window.categoryId + '/products';  }  $table  .DataTable({  lengthMenu : [ [ 3, 5, 10, -1 ],  [ '3 Records', '5 Records', '10 Records', 'ALL' ] ],  pageLength : 5,  ajax : {  url : jsonUrl,  dataSrc : ''  },  columns : [  {  data : 'code',  bSortable : **false**,  mRender : **function**(data, type, row) {  **return** '<img src="' + window.contextRoot + '/resources/images/' + data+ '.jpg" class="dataTableImg"/>';  }  },  {  data : 'name'  },  {  data : 'brand'  },  {  data : 'unitPrice',  mRender : **function**(data, type, row) {  **return** '&#8377; ' + data  }  },  {  data : 'quantity',  mRender : **function**(data, type, row) {  **if** (data < 1) {  **return** '<span style="color:red">Out of Stock!</span>';  }  **return** data;  }  },  {  data : 'id',  bSortable : **false**,  mRender : **function**(data, type, row) {  **var** str = '';  str += '<a href="'+window.contextRoot+ '/show/'+ data + '/product" class="btn btn-primary"><span class="glyphicon glyphicon-eye-open"></span></a> &#160;';    **if**(userRole !== 'ADMIN') {  **if** (row.quantity < 1) {  str += '<a href="javascript:void(0)" class="btn btn-success disabled"><span class="glyphicon glyphicon-shopping-cart"></span></a>';  } **else** {  str += '<a href="'+window.contextRoot+ '/cart/add/'+ data+ '/product" class="btn btn-success"><span class="glyphicon glyphicon-shopping-cart"></span></a>';  }  }  **else** {  str += '<a href="'+ window.contextRoot+ '/manage/'+ data+ '/product" class="btn btn-warning"><span class="glyphicon glyphicon-pencil"></span></a>';  }  **return** str;  }  } ]  });  }  // list of all products for admin  **var** $productsTable = $('#productsTable');  **if**($productsTable.length) {  **var** jsonUrl = window.contextRoot + '/json/data/admin/all/products';  console.log(jsonUrl);  $productsTable.DataTable({  lengthMenu : [ [ 10, 30, 50, -1 ], [ '10 Records', '30 Records', '50 Records', 'ALL' ] ],  pageLength : 30,  ajax : {  url : jsonUrl,  dataSrc : ''  },  columns : [  {data: 'id'},  {data: 'code',  bSortable: **false**,  mRender: **function**(data,type,row) {  **return** '<img src="' + window.contextRoot  + '/resources/images/' + data  + '.jpg" class="dataTableImg"/>';  }  },  {  data : 'name'  },  {  data : 'brand'  },  {  data : 'quantity',  mRender : **function**(data, type, row) {  **if** (data < 1) {  **return** '<span style="color:red">Out of Stock!</span>';  }  **return** data;  }  },  {  data : 'unitPrice',  mRender : **function**(data, type, row) {  **return** '&#8377; ' + data  }  },  {  data : 'active',  bSortable : **false**,  mRender : **function**(data, type, row) {  **var** str = '';  **if**(data) {  str += '<label class="switch"> <input type="checkbox" value="'+row.id+'" checked="checked"> <div class="slider round"> </div></label>';  }**else** {  str += '<label class="switch"> <input type="checkbox" value="'+row.id+'"> <div class="slider round"> </div></label>';  }  **return** str;  }  },  {  data : 'id',  bSortable : **false**,  mRender : **function**(data, type, row) {  **var** str = '';  str += '<a href="'  + window.contextRoot  + '/manage/'  + data  + '/product" class="btn btn-primary"><span class="glyphicon glyphicon-pencil"></span></a> &#160;';  **return** str;  }  }  ],  initComplete: **function** () {  **var** api = **this**.api();  api.$('.switch input[type="checkbox"]').on('change' , **function**() {  **var** dText = (**this**.checked)? 'You want to activate the Product?': 'You want to de-activate the Product?';  **var** checked = **this**.checked;  **var** checkbox = $(**this**);  **debugger**;  bootbox.confirm({  size: 'medium',  title: 'Product Activation/Deactivation',  message: dText,  callback: **function** (confirmed) {  **if** (confirmed) {  $.ajax({  type: 'GET',  url: window.contextRoot + '/manage/product/'+checkbox.prop('value')+'/activation', timeout : 100000,  success : **function**(data) {  bootbox.alert(data);  },  error : **function**(e) {  bootbox.alert('ERROR: '+ e);  //display(e);  }  });  }  **else** {  checkbox.prop('checked', !checked);  }  }  });  });  }  });  }  // jQuery Validation Code  //methods required for validation  **function** errorPlacement(error, element) {  // Add the 'help-block' class to the error element  error.addClass("help-block");  // add the error label after the input element  error.insertAfter(element);  // add the has-feedback class to the  // parent div.validate in order to add icons to inputs  element.parents(".validate").addClass("has-feedback");  }  // validating the product form element  // fetch the form element  $categoryForm = $('#categoryForm');  **if**($categoryForm.length) {  $categoryForm.validate({  rules: {  name: {  required: **true**,  minlength: 3  },  description: {  required: **true**,  minlength: 3  }  },  messages: {  name: {  required: 'Please enter product name!',  minlength: 'Please enter atleast five characters'  },  description: {  required: 'Please enter product name!',  minlength: 'Please enter atleast five characters'  }  },  errorElement : "em",  errorPlacement : **function**(error, element) {  errorPlacement(error, element);  }  }  );  }  /\*validating the loginform\*/  // validating the product form element  // fetch the form element  $loginForm = $('#loginForm');  **if**($loginForm.length) {    $loginForm.validate({  rules: {  username: {  required: **true**,  email: **true**    },  password: {  required: **true**  }  },  messages: {  username: {  required: 'Please enter your email!',  email: 'Please enter a valid email address!'  },  password: {  required: 'Please enter your password!'  }  },  errorElement : "em",  errorPlacement : **function**(error, element) {  // Add the 'help-block' class to the error element  error.addClass("help-block");  // add the error label after the input element  error.insertAfter(element);  }  }  );  }  /\* for fading out the alert message after 3 seconds \*/  $alert = $('.alert');  **if**($alert.length) {  setTimeout(**function**() {  $alert.fadeOut('slow');  }, 3000  );  }  /\* handle refresh cart\*/  $('button[name="refreshCart"]').click(**function**(){  **var** cartLineId = $(**this**).attr('value');  **var** countField = $('#count\_' + cartLineId);  **var** originalCount = countField.attr('value');  // do the checking only the count has changed  **if**(countField.val() !== originalCount) {  // check if the quantity is within the specified range  **if**(countField.val() < 1 || countField.val() > 3) {  // set the field back to the original field  countField.val(originalCount);  bootbox.alert({  size: 'medium',  title: 'Error',  message: 'Product Count should be minimum 1 and maximum 3!'  });  }  **else** {  // use the window.location.href property to send the request to the server  **var** updateUrl = window.contextRoot + '/cart/' + cartLineId + '/update?count=' + countField.val();  window.location.href = updateUrl;  }  }  });  }); |

|  |
| --- |
| Inside webapp/asserts/js/productsController.js |
| **var** app = angular.module('ShoppingApp', []);  app.controller('ProductController', **function**($http) {  **var** me = **this**;  me.mvProducts = [];  me.mpProducts = [];  me.fetchProducts = **function**() {  $http.get('/onlineshopping/json/data/mv/products')  .then(**function**(response) {  me.mvProducts = response.data;  });  $http.get('/onlineshopping/json/data/mp/products')  .then(**function**(response) {  me.mpProducts = response.data;  });  }  }); |

**CheckOut webflow**

|  |
| --- |
| Inside webapp/WEB-INF/views/flows/cart/checkout/checkout-flow.xml |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <flow xmlns=*"http://www.springframework.org/schema/webflow"*  xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*  xsi:schemaLocation=*"http://www.springframework.org/schema/webflow*  *http://www.springframework.org/schema/webflow/spring-webflow-2.0.xsd"*>  <var name=*"shipping"* class=*"net.kzn.shoppingbackend.dto.Address"*/>  <on-start>  <evaluate expression=*"checkoutHandler.init(currentUser.name)"* result=*"flowScope.checkoutModel"*/>  </on-start>  <!-- view the address selection page-->  <view-state id=*"shipping"* view=*"flows/cart/checkout/order-address"* model=*"shipping"*>  <on-entry>  <evaluate expression=*"checkoutHandler.getShippingAddresses(flowScope.checkoutModel)"* result=*"viewScope.addresses"*/>  </on-entry>  <transition on=*"addressSelection"* to=*"saveAddressSelection"* validate=*"false"*/>  <transition on=*"saveAddress"* to=*"saveAddress"* />  </view-state>  <!-- action state to save the address -->  <action-state id=*"saveAddressSelection"*>  <evaluate expression=*"checkoutHandler.saveAddressSelection(flowScope.checkoutModel, requestParameters.shippingId)"* />  <transition on=*"success"* to=*"payment"* />  </action-state>  <!-- action state to save the address -->  <action-state id=*"saveAddress"*>  <evaluate expression=*"checkoutHandler.saveAddress(flowScope.checkoutModel, shipping)"* />  <transition on=*"success"* to=*"payment"* />  </action-state>  <!-- view the payment mode page-->  <view-state id=*"payment"* view=*"flows/cart/checkout/order-payment"* model=*"flowScope.checkoutModel"*>  <transition on=*"pay"* to=*"saveOrder"* />  </view-state>  <!-- action state to save the address -->  <action-state id=*"saveOrder"*>  <evaluate expression=*"checkoutHandler.saveOrder(flowScope.checkoutModel)"* />  <transition on=*"success"* to=*"receipt"*/>  </action-state>  <!-- end state -->  <end-state id=*"receipt"* view=*"flows/cart/checkout/order-confirm"*>  <on-entry>  <evaluate expression=*"checkoutHandler.getOrderDetail(flowScope.checkoutModel)"* result=*"requestScope.orderDetail"*/>  </on-entry>  </end-state>  <end-state id=*"home"* view=*"externalRedirect:contextRelative:/home"* />  <end-state id=*"error"* view=*"flows/flows-shared/error"* />  <!-- global transition -->  <global-transitions>  <transition on=*"home"* to=*"home"* validate=*"false"* />  <transition on-exception=*"java.lang.Exception"* to=*"error"* />  </global-transitions>  </flow> |

|  |
| --- |
| Inside webapp/WEB-INF/views/flows/cart/checkout/order-address.jsp |
| <%@include file=*"../../flows-shared/header.jsp"* %>  <%@taglib prefix=*"sf"* uri=*"http://www.springframework.org/tags/form"* %>  <div class=*"container"*>  <div class=*"row"*>  <div class=*"col-md-4"*>  <h4>Select Shipping Address</h4>  <hr/>  <div class=*"row"*>  <c:forEach items=*"*${addresses}*"* var=*"address"*>  <div class=*"cols-xs-12"*>  <h3>${address.addressLineOne}</h3>  <h3>${address.addressLineTwo}</h3>  <h4>${address.city} - ${address.postalCode}</h4>  <h4>${address.state} - ${address.country}</h4>  <hr/>  <div class=*"text-center"*>  <a href=*"*${flowExecutionUrl}*&\_eventId\_addressSelection&shippingId=*${address.id}*"* class=*"btn btn-primary"*>Select</a>  </div>  </div>  </c:forEach>  </div>  </div>  <div class=*"col-md-8"*>  <div class=*"panel panel-primary"*>  <div class=*"panel-heading"*>  <h4>Sign Up - Address</h4>  </div>  <div class=*"panel-body"*>  <sf:form  method=*"POST"*  modelAttribute=*"shipping"*  class=*"form-horizontal"*  id=*"billingForm"*  >  <div class=*"form-group"*>  <label class=*"control-label col-md-4"* for=*"addressLineOne"*>Address Line One</label>  <div class=*"col-md-8"*>  <sf:input type=*"text"* path=*"addressLineOne"* class=*"form-control"*  placeholder=*"Enter Address Line One"* />  <sf:errors path=*"addressLineOne"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"* for=*"addressLineTwo"*>Address Line Two</label>  <div class=*"col-md-8"*>  <sf:input type=*"text"* path=*"addressLineTwo"* class=*"form-control"* placeholder=*"Enter Address Line Two"* />  <sf:errors path=*"addressLineTwo"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"* for=*"city"*>City</label>  <div class=*"col-md-8"*>  <sf:input type=*"text"* path=*"city"* class=*"form-control"* placeholder=*"Enter City Name"* />  <sf:errors path=*"city"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"* for=*"postalCode"*>Postal Code</label>  <div class=*"col-md-8"*>  <sf:input type=*"text"* path=*"postalCode"* class=*"form-control"* placeholder=*"XXXXXX"* />  <sf:errors path=*"postalCode"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"* for=*"state"*>State</label>  <div class=*"col-md-8"*>  <sf:input type=*"text"* path=*"state"* class=*"form-control"* placeholder=*"Enter State Name"* />  <sf:errors path=*"state"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"* for=*"country"*>Country</label>  <div class=*"col-md-8"*>  <sf:input type=*"text"* path=*"country"* class=*"form-control"* placeholder=*"Enter Country Name"* />  <sf:errors path=*"country"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <div class=*"col-md-offset-4 col-md-8"*>  <button type=*"submit"* name=*"\_eventId\_saveAddress"* class=*"btn btn-primary"*>  <span class=*"glyphicon glyphicon-plus"*></span> Add Address  </button>  </div>  </div>  </sf:form>  </div>  </div>  </div>  </div>  </div>  <%@include file=*"../../flows-shared/footer.jsp"* %> |

|  |
| --- |
| Inside webapp/WEB-INF/views/flows/cart/checkout/order-confirm.jsp |
| <%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*  pageEncoding=*"ISO-8859-1"*%>  <%@taglib prefix=*"c"* uri=*"http://java.sun.com/jsp/jstl/core"*%>  <%@taglib prefix=*"spring"* uri=*"http://www.springframework.org/tags"*%>  <spring:url var=*"css"* value=*"/resources/css"* />  <spring:url var=*"js"* value=*"/resources/js"* />  <spring:url var=*"images"* value=*"/resources/images"* />  <c:set var=*"contextRoot"* value=*"*${pageContext.request.contextPath}*"* />  <!DOCTYPE html>  <html lang=*"en"*>  <head>  <meta charset=*"utf-8"*>  <meta http-equiv=*"X-UA-Compatible"* content=*"IE=edge"*>  <meta name=*"viewport"* content=*"width=device-width, initial-scale=1"*>  <meta name=*"description"* content=*""*>  <meta name=*"author"* content=*""*>  <title>Online Shopping - Membership</title>  <script>window.contextRoot = '${contextRoot}'</script>  <!-- Bootstrap Core CSS -->  <link href=*"*${css}*/bootstrap.min.css"* rel=*"stylesheet"*>  <!-- Bootstrap Readable Theme -->  <link href=*"*${css}*/bootstrap-readable-theme.css"* rel=*"stylesheet"*>  <!-- Bootstrap DataTables -->  <link href=*"*${css}*/dataTables.bootstrap.css"* rel=*"stylesheet"*>  <!-- Custom CSS -->  <link href=*"*${css}*/myapp.css"* rel=*"stylesheet"*>  <!-- HTML5 Shim and Respond.js IE8 support of HTML5 elements and media queries -->  <!-- WARNING: Respond.js doesn't work if you view the page via file:// -->  <!--[if lt IE 9]>  <script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>  <script src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js"></script>  <![endif]-->  </head>  <body>  <div class=*"wrapper"*>  <nav class=*"navbar navbar-inverse navbar-fixed-top"* role=*"navigation"*>  <div class=*"container"*>  <!-- Brand and toggle get grouped for better mobile display -->  <div class=*"navbar-header"*>  <a class=*"navbar-brand"* href=*"*${contextRoot}*/home"*>Online Shopping</a>  </div>  </div>  </nav>  <!-- Page Content -->  <div class=*"content"*>  <div class=*"container"*>  <div class=*"alert alert-success"*>  <h3 class=*"text-center"*>Your Order is Confirmed!!</h3>  </div>  <div class=*"row"*>  <div class=*"col-xs-12"*>  <div class=*"invoice-title"*>  <h2>Invoice</h2><h3 class=*"pull-right"*>Order # ${orderDetail.id}</h3>  </div>  <hr>  <div class=*"row"*>  <div class=*"col-xs-6"*>  <address>  <strong>Billed To:</strong><br>  ${orderDetail.user.firstName} ${orderDetail.user.lastName}<br>  ${orderDetail.billing.addressLineOne}<br>  ${orderDetail.billing.addressLineTwo}<br>  ${orderDetail.billing.city} - ${orderDetail.billing.postalCode}<br>  ${orderDetail.billing.state} - ${orderDetail.billing.country}  </address>  </div>  <div class=*"col-xs-6 text-right"*>  <address>  <strong>Shipped To:</strong><br>  ${orderDetail.user.firstName} ${orderDetail.user.lastName}<br>  ${orderDetail.shipping.addressLineOne}<br>  ${orderDetail.shipping.addressLineTwo}<br>  ${orderDetail.shipping.city} - ${orderDetail.shipping.postalCode}<br>  ${orderDetail.shipping.state} - ${orderDetail.shipping.country}  </address>  </div>  </div>  <div class=*"row"*>  <div class=*"col-xs-6"*>  <address>  <strong>Payment Method:</strong><br>  Card Payment <br>  ${orderDetail.user.email}  </address>  </div>  <div class=*"col-xs-6 text-right"*>  <address>  <strong>Order Date:</strong><br>  ${orderDetail.orderDate}<br><br>  </address>  </div>  </div>  </div>  </div>    <div class=*"row"*>  <div class=*"col-md-12"*>  <div class=*"panel panel-default"*>  <div class=*"panel-heading"*>  <h3 class=*"panel-title"*><strong>Order summary</strong></h3>  </div>  <div class=*"panel-body"*>  <div class=*"table-responsive"*>  <table class=*"table table-condensed"*>  <thead><tr>  <td><strong>Item</strong></td>  <td class=*"text-center"*><strong>Price</strong></td>  <td class=*"text-center"*><strong>Quantity</strong></td>  <td class=*"text-right"*><strong>Totals</strong></td>  </tr>  </thead>  <tbody>  <!-- foreach ($order->lineItems as $line) or some such thing here -->  <c:forEach items=*"*${orderDetail.orderItems}*"* var=*"orderItem"*><tr>  <td>${orderItem.product.name}</td>  <td class=*"text-center"*>&#8377; ${orderItem.buyingPrice}</td>  <td class=*"text-center"*>${orderItem.productCount}</td>  <td class=*"text-right"*>&#8377; ${orderItem.total}</td>  </tr>  </c:forEach>  </tbody>  </table>  </div>  </div>  </div>  </div>  </div>  <div class=*"text-center"*>  <a href=*"*${contextRoot}*/show/all/products"* class=*"btn btn-lg btn-warning"*>Continue Shopping</a>  </div>  </div>  <%@include file=*"../../flows-shared/footer.jsp"* %> |

|  |
| --- |
| Inside webapp/WEB-INF/views/flows/cart/checkout/order-payment.jsp |
| <%@include file=*"../../flows-shared/header.jsp"* %>  <div class=*"container"*>  <div class=*"row"*>  <!-- To display all the goods -->  <div class=*"col-md-6"*>  <div class=*"row"*>  <c:forEach items=*"*${checkoutModel.cartLines}*"* var=*"cartLine"*>  <div class=*"col-xs-12"*>  <div>  <h3>${cartLine.product.name}</h3>  <hr/>  <h4>Quantity -${cartLine.productCount}</h4>  <h5>Buying Price - &#8377; ${cartLine.buyingPrice}/-</h5>  </div>  <hr/>  <div class=*"text-right"*>  <h3>Grand Total - &#8377; ${cartLine.total}/-</h3>  </div>  </div>  </c:forEach>  </div>  </div>  <div class=*"col-md-6"*>  <div class=*"panel panel-default"*>  <div class=*"panel-heading"*>  <h3 class=*"panel-title"*>Payment Details</h3>  </div>  <div class=*"panel-body"*>  <form role=*"form"*>  <div class=*"form-group"*>  <label for=*"cardNumber"*>CARD NUMBER</label>  <div class=*"input-group"*>  <input type=*"text"* class=*"form-control"* id=*"cardNumber"* placeholder=*"Valid Card Number"*  required autofocus />  <span class=*"input-group-addon"*><span class=*"glyphicon glyphicon-lock"*></span></span>  </div>  </div>  <div class=*"row"*>  <div class=*"col-xs-7 col-md-7"*>  <div class=*"form-group"*>  <label for=*"expityMonth"*>EXPIRY DATE</label>  <br/>  <div class=*"col-xs-6 col-lg-6 pl-ziro"*>  <input type=*"text"* class=*"form-control"* id=*"expityMonth"* placeholder=*"MM"* required />  </div>  <div class=*"col-xs-6 col-lg-6 pl-ziro"*>  <input type=*"text"* class=*"form-control"* id=*"expityYear"* placeholder=*"YY"* required /></div>  </div>  </div>  <div class=*"col-xs-5 col-md-5 pull-right"*>  <div class=*"form-group"*>  <label for=*"cvCode"*>  CV CODE</label>  <input type=*"password"* class=*"form-control"* id=*"cvCode"* placeholder=*"CV"* required />  </div>  </div>  </div>  </form>  </div>  </div>  <ul class=*"nav nav-pills nav-stacked"*>  <li class=*"active"*><a href=*"#"*><span class=*"badge pull-right"*> &#8377; ${checkoutModel.checkoutTotal}/-</span> Final Payment</a></li>  </ul>  <br/>  <a href=*"*${flowExecutionUrl}*&\_eventId\_pay"* class=*"btn btn-success btn-lg btn-block"* role=*"button"*>Pay</a>  </div>  </div>  </div>  <%@include file=*"../../flows-shared/footer.jsp"* %> |

|  |
| --- |
| Inside webapp/WEB-INF/views/flows/flows-shared/error.jsp |
| <%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*  pageEncoding=*"ISO-8859-1"*%>  <%@taglib prefix=*"c"* uri=*"http://java.sun.com/jsp/jstl/core"*%>  <%@taglib prefix=*"spring"* uri=*"http://www.springframework.org/tags"*%>  <spring:url var=*"css"* value=*"/resources/css"* />  <c:set var=*"contextRoot"* value=*"*${pageContext.request.contextPath}*"* />  <!DOCTYPE html>  <html lang=*"en"*>  <head>  <meta charset=*"utf-8"*>  <meta http-equiv=*"X-UA-Compatible"* content=*"IE=edge"*>  <meta name=*"viewport"* content=*"width=device-width, initial-scale=1"*>  <meta name=*"description"* content=*""*>  <meta name=*"author"* content=*""*>  <!-- Bootstrap Core CSS -->  <link href=*"*${css}*/bootstrap.min.css"* rel=*"stylesheet"*>  <!-- Bootstrap Readable Theme -->  <link href=*"*${css}*/bootstrap-readable-theme.css"* rel=*"stylesheet"*>  <!-- Custom CSS -->  <link href=*"*${css}*/myapp.css"* rel=*"stylesheet"*>  <title>Online Shopping - FlowException</title>  </head>  <body>  <div class=*"wrapper"*>  <nav class=*"navbar navbar-inverse navbar-fixed-top"* role=*"navigation"*>  <div class=*"container"*>  <!-- Brand and toggle get grouped for better mobile display -->  <div class=*"navbar-header"*>  <a class=*"navbar-brand"* href=*"*${contextRoot}*/home"*>Home</a>  </div>  </div>  </nav>  <div class=*"content"*>  <div class=*"container"*>  <div class=*"row"*>  <div class=*"col-xs-12"*>  <div class=*"jumbotron"*>  <h1>Please contact your administrator!</h1>  <hr/>  <blockquote style="word-wrap:*break-word*">  ${flowExecutionException}  </blockquote>  <blockquote style="word-wrap:*break-word*">  ${rootCauseException}  </blockquote>  </div>  </div>  </div>  </div>  </div>  <%@include file=*"../../shared/footer.jsp"* %>  </div>  </body>  </html> |

|  |
| --- |
| Inside webapp/WEB-INF/views/flows/flows-shared/footer.jsp |
| </div>  <!-- Footer comes here -->  <%@include file=*"../../shared/footer.jsp"*%>  <!-- jQuery -->  <script src=*"*${js}*/jquery.js"*></script>  <script src=*"*${js}*/jquery.validate.js"*></script>  <!-- Bootstrap Core JavaScript -->  <script src=*"*${js}*/bootstrap.min.js"*></script>  <!-- DataTable Plugin -->  <script src=*"*${js}*/jquery.dataTables.js"*></script>  <!-- DataTable Bootstrap Script -->  <script src=*"*${js}*/dataTables.bootstrap.js"*></script>  <!-- DataTable Bootstrap Script -->  <script src=*"*${js}*/bootbox.min.js"*></script>  <!-- Self coded javascript -->  <script src=*"*${js}*/myapp.js"*></script>  </div>  </body>  </html> |

|  |
| --- |
| Inside webapp/WEB-INF/views/flows/flows-shared/header.jsp |
| <%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*  pageEncoding=*"ISO-8859-1"*%>  <%@taglib prefix=*"c"* uri=*"http://java.sun.com/jsp/jstl/core"*%>  <%@taglib prefix=*"spring"* uri=*"http://www.springframework.org/tags"*%>  <spring:url var=*"css"* value=*"/resources/css"* />  <spring:url var=*"js"* value=*"/resources/js"* />  <spring:url var=*"images"* value=*"/resources/images"* />  <c:set var=*"contextRoot"* value=*"*${pageContext.request.contextPath}*"* />  <!DOCTYPE html>  <html lang=*"en"*>  <head>  <meta charset=*"utf-8"*>  <meta http-equiv=*"X-UA-Compatible"* content=*"IE=edge"*>  <meta name=*"viewport"* content=*"width=device-width, initial-scale=1"*>  <meta name=*"description"* content=*""*>  <meta name=*"author"* content=*""*>  <title>Online Shopping - Membership</title>  <script>  window.contextRoot = '${contextRoot}'  </script>  <!-- Bootstrap Core CSS -->  <link href=*"*${css}*/bootstrap.min.css"* rel=*"stylesheet"*>  <!-- Bootstrap Readable Theme -->  <link href=*"*${css}*/bootstrap-readable-theme.css"* rel=*"stylesheet"*>  <!-- Bootstrap DataTables -->  <link href=*"*${css}*/dataTables.bootstrap.css"* rel=*"stylesheet"*>  <!-- Custom CSS -->  <link href=*"*${css}*/myapp.css"* rel=*"stylesheet"*>  <!-- HTML5 Shim and Respond.js IE8 support of HTML5 elements and media queries -->  <!-- WARNING: Respond.js doesn't work if you view the page via file:// -->  <!--[if lt IE 9]>  <script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>  <script src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js"></script>  <![endif]-->  </head>  <body>  <div class=*"se-pre-con"*></div>  <div class=*"wrapper"*>  <nav class=*"navbar navbar-inverse navbar-fixed-top"* role=*"navigation"*>  <div class=*"container"*>  <!-- Brand and toggle get grouped for better mobile display -->  <div class=*"navbar-header"*>  <a class=*"navbar-brand"* href=*"*${flowExecutionUrl}*&\_eventId=home"*>Online Shopping</a>  </div>  </div>  </nav>  <!-- Page Content -->  <div class=*"content"*> |

|  |
| --- |
| Inside webapp/WEB-INF/views/flows/membership/register-flow.xml |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <flow xmlns=*"http://www.springframework.org/schema/webflow"*  xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*  xsi:schemaLocation=*"http://www.springframework.org/schema/webflow*  *http://www.springframework.org/schema/webflow/spring-webflow-2.0.xsd"*>  <var name=*"user"* class=*"net.kzn.shoppingbackend.dto.User"*/>  <var name=*"billing"* class=*"net.kzn.shoppingbackend.dto.Address"*/>  <on-start>  <evaluate expression=*"registerHandler.init()"* result=*"flowScope.registerModel"*/>  </on-start>  <!-- view the signup-personal page-->  <view-state id=*"personal"* view=*"flows/membership/signup-personal"* model=*"user"*>  <transition on=*"billing"* to=*"validate"* />  <on-exit>  <evaluate expression=*"registerHandler.addUser(flowScope.registerModel, user)"* />  </on-exit>  </view-state>  <!-- validate signup-personal page-->  <action-state id=*"validate"*>  <evaluate expression=*"registerHandler.validateUser(user,messageContext)"* />  <transition on=*"success"* to=*"billing"*/>  <transition on=*"failure"* to=*"personal"*/>  </action-state>  <!-- view the signup-billing page-->  <view-state id=*"billing"* view=*"flows/membership/signup-billing"* model=*"billing"*>  <transition on=*"confirm"* to=*"confirm"* />  <transition on=*"personal"* to=*"personal"* />  <on-exit>  <evaluate expression=*"registerHandler.addBilling(flowScope.registerModel, billing)"* />  </on-exit>  </view-state>  <!-- view the signup-confirm page -->  <view-state id=*"confirm"* view=*"flows/membership/signup-confirm"* model=*"flowScope.registerModel"*>  <transition on=*"submit"* to=*"submit"* />  <transition on=*"billing"* to=*"billing"* />  <transition on=*"personal"* to=*"personal"* />  </view-state>  <!-- handle the submit action -->  <action-state id=*"submit"*>  <evaluate expression=*"registerHandler.saveAll(flowScope.registerModel)"* />  <transition on=*"success"* to=*"success"*/>  </action-state>  <!-- end state -->  <end-state id=*"success"* view=*"flows/membership/signup-success"* />  <end-state id=*"home"* view=*"externalRedirect:contextRelative:/"* />  <!-- global transition -->  <global-transitions>  <transition on=*"home"* to=*"home"* validate=*"false"* />  </global-transitions>  </flow> |

|  |
| --- |
| Inside webapp/WEB-INF/views/flows/membership/Signup-billing.jsp |
| <%@include file=*"../flows-shared/header.jsp"* %>  <%@taglib prefix=*"sf"* uri=*"http://www.springframework.org/tags/form"* %>  <div class=*"container"*>  <div class=*"row"*>  <div class=*"col-md-6 col-md-offset-3"*>  <div class=*"panel panel-primary"*>  <div class=*"panel-heading"*>  <h4>Sign Up - Address</h4>  </div>  <div class=*"panel-body"*>  <sf:form  method=*"POST"*  modelAttribute=*"billing"*  class=*"form-horizontal"*  id=*"billingForm"*>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"* for=*"addressLineOne"*>Address Line One</label>  <div class=*"col-md-8"*>  <sf:input type=*"text"* path=*"addressLineOne"* class=*"form-control"* placeholder=*"Enter Address Line One"* />  <sf:errors path=*"addressLineOne"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"* for=*"addressLineTwo"*>Address Line Two</label>  <div class=*"col-md-8"*>  <sf:input type=*"text"* path=*"addressLineTwo"* class=*"form-control"* placeholder=*"Enter Address Line Two"* />  <sf:errors path=*"addressLineTwo"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"* for=*"city"*>City</label>  <div class=*"col-md-8"*>  <sf:input type=*"text"* path=*"city"* class=*"form-control"* placeholder=*"Enter City Name"* />  <sf:errors path=*"city"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"* for=*"postalCode"*>Postal Code</label>  <div class=*"col-md-8"*>  <sf:input type=*"text"* path=*"postalCode"* class=*"form-control"* placeholder=*"XXXXXX"* />  <sf:errors path=*"postalCode"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"* for=*"state"*>State</label>  <div class=*"col-md-8"*>  <sf:input type=*"text"* path=*"state"* class=*"form-control"* placeholder=*"Enter State Name"* />  <sf:errors path=*"state"* cssClass=*"help-block"* element=*"em"*/>  <div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"* for=*"country"*>Country</label>  <div class=*"col-md-8"*>  <sf:input type=*"text"* path=*"country"* class=*"form-control"* placeholder=*"Enter Country Name"* />  <sf:errors path=*"country"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <div class=*"col-md-offset-4 col-md-8"*>  <button type=*"submit"* name=*"\_eventId\_personal"* class=*"btn btn-primary"*>  <span class=*"glyphicon glyphicon-chevron-left"*></span> Back – Personal </button>  <button type=*"submit"* name=*"\_eventId\_confirm"* class=*"btn btn-primary"*>Next - Confirm<span class=*"glyphicon glyphicon-chevron-right"*></span></button>  </div>  </div>  </sf:form>  </div>  </div>  </div>  </div>  </div>  <%@include file=*"../flows-shared/footer.jsp"* %> |

|  |
| --- |
| Inside webapp/WEB-INF/views/flows/membership/Signup-confirm.jsp |
| <%@include file=*"../flows-shared/header.jsp"* %>  <div class=*"container"*>  <div class=*"row"*>  <div class=*"col-sm-6"*>  <div class=*"panel panel-primary"*>  <div class=*"panel-heading"*>  <h4>Personal Details</h4>  </div>  <div class=*"panel-body"*>  <div class=*"text-center"*>  <h3>Name : <strong>${registerModel.user.firstName} ${registerModel.user.lastName}</strong></h3>  <h4>Email : <strong>${registerModel.user.email}</strong></h4>  <h4>Contact : <strong>${registerModel.user.contactNumber}</strong></h4>  <h4>Role : <strong>${registerModel.user.role}</strong></h4>  <p>  <a href=*"*${flowExecutionUrl}*&\_eventId\_personal"* class=*"btn btn-primary"*>Edit</a>  </p>  </div>  </div>  </div>  </div>  <div class=*"col-sm-6"*>  <div class=*"panel panel-primary"*>  <div class=*"panel-heading"*>  <h4>Billing Address</h4>  </div>  <div class=*"panel-body"*>  <div class=*"text-center"*>  <p>${registerModel.billing.addressLineOne}, </p>  <p>${registerModel.billing.addressLineTwo}, </p>  <p>${registerModel.billing.city} - ${registerModel.billing.postalCode}, </p>  <p>${registerModel.billing.state}</p>  <p>${registerModel.billing.country}</p>  <p>  <a href=*"*${flowExecutionUrl}*&\_eventId\_billing"* class=*"btn btn-primary"*>Edit</a>  </p>  </div>  </div>  </div>  </div>  </div>  <div class=*"row"*>  <div class=*"col-sm-4 col-sm-offset-4"*>  <div class=*"text-center"*>  <a href=*"*${flowExecutionUrl}*&\_eventId\_submit"* class=*"btn btn-lg btn-primary"*>Confirm</a>  </div>  </div>  </div>  </div>  <%@include file=*"../flows-shared/footer.jsp"* %> |

|  |
| --- |
| Inside webapp/WEB-INF/views/flows/membership/Signup-personal.jsp |
| <%@include file=*"../flows-shared/header.jsp"* %>  <%@taglib prefix=*"sf"* uri=*"http://www.springframework.org/tags/form"* %>  <div class=*"container"*>  <div class=*"row"*>  <div class=*"col-md-6 col-md-offset-3"*>  <div class=*"panel panel-primary"*>  <div class=*"panel-heading"*>  <h4>Sign Up - Personal</h4>  </div>  <div class=*"panel-body”>*  <sf:form  method=*"POST"*  modelAttribute=*"user"*  class=*"form-horizontal"*  id=*"registerForm"*  >      <div class=*"form-group"*>  <label class=*"control-label col-md-4"*>First Name</label>  <div class=*"col-md-8"*>  <sf:input type=*"text"* path=*"firstName"* class=*"form-control"* placeholder=*"First Name"* />  <sf:errors path=*"firstName"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"*>Last Name</label>  <div class=*"col-md-8"*>  <sf:input type=*"text"* path=*"lastName"* class=*"form-control"* placeholder=*"Last Name"* />  <sf:errors path=*"lastName"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"*>Email</label>  <div class=*"col-md-8"*>  <sf:input type=*"text"* path=*"email"* class=*"form-control"* placeholder=*"abc@zyx.com"* />  <sf:errors path=*"email"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"*>Contact Number</label>  <div class=*"col-md-8"*>  <sf:input type=*"text"* path=*"contactNumber"* class=*"form-control"* placeholder=*"XXXXXXXXXX"* maxlength=*"10"* />  <sf:errors path=*"contactNumber"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"*>Password</label>  <div class=*"col-md-8"*>  <sf:input type=*"password"* path=*"password"* class=*"form-control"* placeholder=*"Password"* />  <sf:errors path=*"password"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"*>Confirm Password</label>  <div class=*"col-md-8"*>  <sf:input type=*"password"* path=*"confirmPassword"*class=*"form-control"*placeholder=*"Re-type password"*/>  <sf:errors path=*"confirmPassword"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"*>Select Role</label>  <div class=*"col-md-8"*>  <label class=*"radio-inline"*>  <sf:radiobutton path=*"role"* value=*"USER"* checked=*"checked"*/> User  </label>  <label class=*"radio-inline"*>  <sf:radiobutton path=*"role"* value=*"SUPPLIER"*/> Supplier  </label>  </div>  </div>  <div class=*"form-group"*>  <div class=*"col-md-offset-4 col-md-8"*>  <button type=*"submit"* name=*"\_eventId\_billing"* class=*"btn btn-primary"*>  Next - Billing <span class=*"glyphicon glyphicon-chevron-right"*></span>  </button>  </div>  </div>  </sf:form>  </div>  </div>  </div>  </div>  </div>  <%@include file=*"../flows-shared/footer.jsp"* %> |

|  |
| --- |
| Inside webapp/WEB-INF/views/flows/membership/Signup-success.jsp |
| <%@include file=*"../flows-shared/header.jsp"* %>  <div class=*"container"*>  <div class=*"row"*>  <div class=*"col-sm-offset-4 col-sm-4"*>  <div class=*"text-center"*>  <h1>Welcome!</h1>  <h3>onlineshopping.com</h3>  <h6>You can use your email address as username to login!</h6>  <div>  <a href=*"*${contextRoot}*/login"* class=*"btn btn-lg btn-success"*>Login Here</a>  </div>  </div>  </div>  </div>  </div>  <%@include file=*"../flows-shared/footer.jsp"* %> |

|  |
| --- |
| I nside webapp/WEB-INF/views/shared/footer.jsp |
| <div class=*"container footer"*>  <hr>  <!-- Footer -->  <footer>  <div class=*"row"*>  <div class=*"col-lg-12"*>  <p>Copyright &copy; Online Shopping 2017 - Designed and Developed By SUSMAN MAHARANA</p>  </div>  </div>  </footer>  </div> |

|  |
| --- |
| I nside webapp/WEB-INF/views/shared/navbar.jsp |
| <%@taglib prefix=*"security"* uri=*"http://www.springframework.org/security/tags"*%>  <script>  window.userRole = '${userModel.role}';  </script>  <nav class=*"navbar navbar-inverse navbar-fixed-top"* role=*"navigation"*>  <div class=*"container"*>  <!-- Brand and toggle get grouped for better mobile display -->  <div class=*"navbar-header"*>  <button type=*"button"* class=*"navbar-toggle"* data-toggle=*"collapse"* data-target=*"#bs-example-navbar-collapse-1"*>  <span class=*"sr-only"*>Toggle navigation</span>  <span class=*"icon-bar"*></span>  <span class=*"icon-bar"*></span>  <span class=*"icon-bar"*></span>  </button>  <a class=*"navbar-brand"* href=*"*${contextRoot}*/home"*>Online Shopping</a>  </div>  <!-- Collect the nav links, forms, and other content for toggling -->  <div class=*"collapse navbar-collapse"* id=*"bs-example-navbar-collapse-1"*>  <ul class=*"nav navbar-nav"*>  <li id=*"about"*>  <a href=*"*${contextRoot}*/about"*>About</a>  </li>  <li id=*"contact"*>  <a href=*"*${contextRoot}*/contact"*>Contact</a>  </li>  <li id=*"listProducts"*>  <a href=*"*${contextRoot}*/show/all/products"*>View Products</a>  </li>  <security:authorize access=*"hasAuthority('ADMIN')"*>  <li id=*"manageProduct"*>  <a href=*"*${contextRoot}*/manage/product"*>Manage Product</a>  </li>  </security:authorize>  </ul>  <ul class=*"nav navbar-nav navbar-right"*>  <security:authorize access=*"isAnonymous()"*>  <li id=*"signup"*>  <a href=*"*${contextRoot}*/membership"*>Sign Up</a>  </li>  <li id=*"login"*>  <a href=*"*${contextRoot}*/login"*>Login</a>  </li>  </security:authorize>  <security:authorize access=*"isAuthenticated()"*>  <li class=*"dropdown"* id=*"userModel"*>  <a class=*"btn btn-default dropdown-toggle"* href=*"javascript:void(0)"* id=*"dropdownMenu1"* data-toggle=*"dropdown"* aria-haspopup=*"true"* aria-expanded=*"true"*>  ${userModel.fullName}  <span class=*"caret"*></span>  </a>  <ul class=*"dropdown-menu"* aria-labelledby=*"dropdownMenu1"*>  <security:authorize access=*"hasAuthority('USER')"*>  <li id=*"cart"*>  <a href=*"*${contextRoot}*/cart/show"*>  <span class=*"glyphicon glyphicon-shopping-cart"*></span>&#160;<span class=*"badge"*>${userModel.cart.cartLines}</span> - &#8377; ${userModel.cart.grandTotal}  </a>  </li>  <li role=*"separator"* class=*"divider"*></li>  </security:authorize>  <li id=*"logout"*>  <a href=*"*${contextRoot}*/logout"*>Logout</a>  </li>  </ul>  </li>  </security:authorize>  </ul>  </div>  <!-- /.navbar-collapse -->  </div>  <!-- /.container -->  </nav> |

|  |
| --- |
| I nside webapp/WEB-INF/views/shared/sidebar.jsp |
| <p class=*"lead"*>Shop Name</p>  <div class=*"list-group"*>  <c:forEach items="${categories}" var=*"category"*>  <a href=*"*${contextRoot}*/show/category/*${category.id}*/products"* class=*"list-group-item"* id=*"a\_*${category.name}*"*>${category.name}</a>  </c:forEach>  </div> |

|  |
| --- |
| Inside webapp/WEB-INF/views/about.jsp |
| <div class=*"container"*>  <div class=*"row"*>  <h1>About Us</h1>  <hr/>  <p>This is an about us page</p>  </div>  </div> |

|  |
| --- |
| Inside webapp/WEB-INF/views/cart.jsp |
| <c:set var=*"availableCount"* value="${userModel.cart.cartLines}" />  <div class=*"container"*>  <c:if test="${**not empty** message}">  <div class=*"alert alert-info"*>  <h3 class=*"text-center"*>${message}</h3>  </div>  </c:if>  <c:choose>  <c:when test="${**not empty** cartLines}">  <table id=*"cart"* class=*"table table-hover table-condensed"*>  <thead>  <tr>  <th style="width:*50%*">Product</th>  <th style="width:*10%*">Price</th>  <th style="width:*8%*">Quantity</th>  <th style="width:*22%*" class=*"text-center"*>Subtotal</th>  <th style="width:*10%*"></th>  </tr>  </thead>  <tbody>  <c:forEach items="${cartLines}" var=*"cartLine"*>  <c:if test="${cartLine.available == **false**}">  <c:set var=*"availableCount"* value="${availableCount - 1}"/>  </c:if>  <tr>  <td data-th=*"Product"*>  <div class=*"row"*>  <div class=*"col-sm-2 hidden-xs"*><img src=*"*${images}*/*${cartLine.product.code}*.jpg"* alt=*"*${cartLine.product.name}*"* class=*"img-responsive dataTableImg"*/></div>  <div class=*"col-sm-10"*>  <h4 class=*"nomargin"*>${cartLine.product.name}  <c:if test="${cartLine.available == **false**}">  <strong style="color:*red*">(Not Available)</strong>  </c:if>  </h4>  <p>Brand : ${cartLine.product.brand}</p>  <p>Description : ${cartLine.product.description}  </div>  </div>  </td>  <td data-th=*"Price"*> &#8377; ${cartLine.buyingPrice} /-</td>  <td data-th=*"Quantity"*>  <input type=*"number"* id=*"count\_*${cartLine.id}*"* class=*"form-control text-center"* value=*"*${cartLine.productCount}*"* min=*"1"* max=*"3"*>  </td>  <td data-th=*"Subtotal"* class=*"text-center"*>&#8377; ${cartLine.total} /-</td>  <td class=*"actions"* data-th=*""*>  <c:if test="${cartLine.available == **true**}">  <button type=*"button"* name=*"refreshCart"* class=*"btn btn-info btn-sm"* value=*"*${cartLine.id}*"*><span class=*"glyphicon glyphicon-refresh"*></span></button>  </c:if>  <a href=*"*${contextRoot}*/cart/*${cartLine.id}*/remove"* class=*"btn btn-danger btn-sm"*><span class=*"glyphicon glyphicon-trash"*></span></a>  </td>  </tr>  </c:forEach>  </tbody>  <tfoot>  <tr class=*"visible-xs"*>  <td class=*"text-center"*><strong>Total &#8377; ${userModel.cart.grandTotal}</strong></td>  </tr>  <tr><td><a href=*"*${contextRoot}*/show/all/products"* class=*"btn btn-warning"*><span class=*"glyphicon glyphicon-chevron-left"*></span> Continue Shopping</a></td>  <td colspan=*"2"* class=*"hidden-xs"*></td>  <td class=*"hidden-xs text-center"*><strong>Total &#8377; ${userModel.cart.grandTotal}/-</strong></td>    <c:choose>  <c:when test="${availableCount != 0}">  <td><a href=*"*${contextRoot}*/cart/validate"* class=*"btn btn-success btn-block"*>Checkout <span class=*"glyphicon glyphicon-chevron-right"*></span></a></td>  </c:when>  <c:otherwise>  <td><a href=*"javascript:void(0)"* class=*"btn btn-success btn-block disabled"*><strike>Checkout <span class=*"glyphicon glyphicon-chevron-right"*></span></strike></a></td>  </c:otherwise>  </c:choose>  </tr>  </tfoot>  </table>  </c:when>  <c:otherwise>  <div class=*"jumbotron"*>  <h3 class=*"text-center"*>Your Cart is Empty!</h3>  </div>  </c:otherwise>  </c:choose>  </div> |

|  |
| --- |
| Inside webapp/WEB-INF/views/contact.jsp |
| <div class=*"container"*>  <div class=*"row"*>  <h1>Contact Us</h1>  <hr/>  <p>This is an contact us page</p>  </div>  </div> |

|  |
| --- |
| Inside webapp/WEB-INF/views/error.jsp |
| <%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"* pageEncoding=*"ISO-8859-1"*%>  <%@taglib prefix=*"c"* uri=*"http://java.sun.com/jsp/jstl/core"*%>  <%@taglib prefix=*"spring"* uri=*"http://www.springframework.org/tags"*%>  <spring:url var=*"css"* value=*"/resources/css"* />  <c:set var=*"contextRoot"* value=*"*${pageContext.request.contextPath}*"* />  <!DOCTYPE html>  <html lang=*"en"*>  <head>  <meta charset=*"utf-8"*>  <meta http-equiv=*"X-UA-Compatible"* content=*"IE=edge"*>  <meta name=*"viewport"* content=*"width=device-width, initial-scale=1"*>  <meta name=*"description"* content=*""*>  <meta name=*"author"* content=*""*>  <!-- Bootstrap Core CSS -->  <link href=*"*${css}*/bootstrap.min.css"* rel=*"stylesheet"*>  <!-- Bootstrap Readable Theme -->  <link href=*"*${css}*/bootstrap-readable-theme.css"* rel=*"stylesheet"*>  <!-- Custom CSS -->  <link href=*"*${css}*/myapp.css"* rel=*"stylesheet"*>  <title>Online Shopping - ${title}</title>  </head>  <body>  <div class=*"wrapper"*>  <nav class=*"navbar navbar-inverse navbar-fixed-top"* role=*"navigation"*>  <div class=*"container"*>  <!-- Brand and toggle get grouped for better mobile display -->  <div class=*"navbar-header"*>  <a class=*"navbar-brand"* href=*"*${contextRoot}*/home"*>Home</a>  </div>  </div>  </nav>  <div class=*"content"*>  <div class=*"container"*>  <div class=*"row"*>  <div class=*"col-xs-12"*>  <div class=*"jumbotron"*>  <h1>${errorTitle}</h1>  <hr/>  <blockquote style="word-wrap:*break-word*">  ${errorDescription}  </blockquote>  </div>  </div>  </div>  </div>  </div>  <%@include file=*"./shared/footer.jsp"* %>  </div>  </body>  </html> |

|  |
| --- |
| Inside webapp/WEB-INF/views/home.jsp |
| <!-- DataTable Bootstrap Script -->  <script src=*"*${js}*/angular.js"*></script>  <!-- DataTable Bootstrap Script -->  <script src=*"*${js}*/productsController.js"*></script>  <div class=*"container"* ng-app=*"ShoppingApp"* ng-controller=*"ProductController as pCtrl"* >  <div class=*"row"* ng-init=*"pCtrl.fetchProducts()"*>  <div class=*"col-md-3"*>  <%@include file=*"./shared/sidebar.jsp"*%>  </div>  <div class=*"col-md-9"*>  <div class=*"row carousel-holder"*>  <div class=*"col-md-12"*>  <div id=*"carousel-example-generic"* class=*"carousel slide"* data-ride=*"carousel"*>  <ol class=*"carousel-indicators"*>  <li data-target=*"#carousel-example-generic"* data-slide-to=*"0"* class=*"active"*></li>  <li data-target=*"#carousel-example-generic"* data-slide-to=*"1"*></li>  <li data-target=*"#carousel-example-generic"* data-slide-to=*"2"*></li>  <li data-target=*"#carousel-example-generic"* data-slide-to=*"3"*></li>  </ol>  <div class=*"carousel-inner"*>  <div class=*"item active"*>  <img class=*"slide-image"* src=*"*${images}*/banner1.jpg"* alt=*""*>  </div>  <div class=*"item"*>  <img class=*"slide-image"* src=*"*${images}*/banner2.jpg"* alt=*""*>  </div>  <div class=*"item"*>  <img class=*"slide-image"* src=*"*${images}*/banner3.jpg"* alt=*""*>  </div>  <div class=*"item"*>  <img class=*"slide-image"* src=*"*${images}*/banner4.jpg"* alt=*""*>  </div>  </div>  <a class=*"left carousel-control"* href=*"#carousel-example-generic"* data-slide=*"prev"*>  <span class=*"glyphicon glyphicon-chevron-left"*></span>  </a> <a class=*"right carousel-control"* href=*"#carousel-example-generic"* data-slide=*"next"*>  <span class=*"glyphicon glyphicon-chevron-right"*></span>  </a>  </div>  </div>  </div>  <div class=*"row"*>  <div class=*"col-xs-12"*>  <h3>Our Most Viewed Products</h3>  <hr/>  </div>  </div>  <div class=*"row is-table-row"*>  <div class=*"col-sm-4"* ng-repeat=*"product in pCtrl.mvProducts"*>  <div class=*"thumbnail"*>  <img ng-src=*"*${images}*/{{product.code}}.jpg"* alt=*"{{product.name}}"* class=*"landingImg"*>  <h5>{{product.name}}</h5>  <hr/>  <div class=*"caption"*>  <h4 class=*"pull-right"*>&#8377; {{product.unitPrice}}</h4>  <p>{{product.description}}</p>  <a ng-href=*"*${contextRoot}*/show/{{product.id}}/product"* class=*"btn btn-primary pull-right"*>View</a>  </div>  </div>  </div>  <div class=*"col-sm-4 col-lg-4 col-md-4"*>  <h4>Checkout more products!</h4>  <hr/>  <a class=*"btn btn-primary"* href=*"*${contextRoot}*/show/all/products"*>More Products</a>  </div>  </div>  <div class=*"row"*>  <div class=*"col-xs-12"*>  <h3>Our Most Purchased Products</h3>  <hr/>  </div>  </div>  <div class=*"row is-table-row"*>  <div class=*"col-sm-4"* ng-repeat=*"product in pCtrl.mpProducts"*>  <div class=*"thumbnail"*>  <img ng-src=*"*${images}*/{{product.code}}.jpg"* alt=*"{{product.name}}"* class=*"landingImg"*>  <h5>{{product.name}}</h5>  <hr/>  <div class=*"caption"*>  <h4 class=*"pull-right"*>&#8377; {{product.unitPrice}}</h4>  <p>{{product.description}}</p>  <a ng-href=*"*${contextRoot}*/show/{{product.id}}/product"* class=*"btn btn-primary pull-right"*>View</a>  </div>  </div>  </div>  <div class=*"col-sm-4 col-lg-4 col-md-4"*>  <h4>Checkout more products!</h4>  <hr/>  <a class=*"btn btn-primary"* href=*"*${contextRoot}*/show/all/products"*>More Products</a>  </div>  </div>  </div>  </div>  </div>  <!-- /.container --> |

|  |
| --- |
| Inside webapp/WEB-INF/views/listProduct.jsp |
| <div class=*"container"*>  <div class=*"row"*>  <!-- Would be to display sidebar -->  <div class=*"col-md-3"*>  <%@include file=*"./shared/sidebar.jsp"*%>  </div>  <!-- to display the actual products -->  <div class=*"col-md-9"*>  <!-- Added breadcrumb component -->  <div class=*"row"*>  <div class=*"col-lg-12"*>  <c:if test="${userClickAllProducts == **true**}">  <script>  window.categoryId = '';  </script>  <ol class=*"breadcrumb"*>  <li><a href=*"*${contextRoot}*/home"*>Home</a></li>  <li class=*"active"*>All Products</li>  </ol>  </c:if>  <c:if test="${userClickCategoryProducts == **true**}">  <script>  window.categoryId = '${category.id}';  </script>  <ol class=*"breadcrumb"*>  <li><a href=*"*${contextRoot}*/home"*>Home</a></li>  <li class=*"active"*>Category</li>  <li class=*"active"*>${category.name}</li>  </ol>  </c:if>  </div>  </div>  <div class=*"row"*>  <div class=*"col-xs-12"*>  <table id=*"productListTable"* class=*"table table-striped table-borderd"*>  <thead>  <tr>  <th></th>  <th>Name</th>  <th>Brand</th>  <th>Price</th>  <th>Qty. Available</th>  <th></th>  </tr>  </thead>  <tfoot>  <tr>  <th></th>  <th>Name</th>  <th>Brand</th>  <th>Price</th>  <th>Qty. Available</th>  <th></th>  </tr>  </tfoot>  </table>  </div>  </div>  </div>  </div>  </div> |

|  |
| --- |
| Inside webapp/WEB-INF/views/login.jsp |
| <%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*  pageEncoding=*"ISO-8859-1"*%>  <%@taglib prefix=*"spring"* uri=*"http://www.springframework.org/tags"*%>  <spring:url var=*"css"* value=*"/resources/css"* />  <spring:url var=*"js"* value=*"/resources/js"* />  <spring:url var=*"images"* value=*"/resources/images"* />  <%@taglib prefix=*"c"* uri=*"http://java.sun.com/jsp/jstl/core"*%>  <c:set var=*"contextRoot"* value=*"*${pageContext.request.contextPath}*"* />  <!DOCTYPE html>  <html lang=*"en"*>  <head>  <meta charset=*"utf-8"*>  <meta http-equiv=*"X-UA-Compatible"* content=*"IE=edge"*>  <meta name=*"viewport"* content=*"width=device-width, initial-scale=1"*>  <meta name=*"description"* content=*""*>  <meta name=*"author"* content=*""*>  <title>Online Shopping - ${title}</title>  <!-- Bootstrap Core CSS -->  <link href=*"*${css}*/bootstrap.min.css"* rel=*"stylesheet"*>  <!-- Bootstrap Readable Theme -->  <link href=*"*${css}*/bootstrap-readable-theme.css"* rel=*"stylesheet"*>  <!-- Custom CSS -->  <link href=*"*${css}*/myapp.css"* rel=*"stylesheet"*>  <!-- HTML5 Shim and Respond.js IE8 support of HTML5 elements and media queries -->  <!-- WARNING: Respond.js doesn't work if you view the page via file:// -->  <!--[if lt IE 9]>  <script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>  <script src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js"></script>  <![endif]-->  <script>  window.menu = '${title}';  window.contextRoot = '${contextRoot}'  </script>  </head>  <body>  <div class=*"wrapper"*>  <!-- Navigation -->  <nav class=*"navbar navbar-inverse navbar-fixed-top"* role=*"navigation"*>  <div class=*"container"*>  <!-- Brand and toggle get grouped for better mobile display -->  <div class=*"navbar-header"*>  <a class=*"navbar-brand"* href=*"*${contextRoot}*/home"*>Online Shopping</a>  </div>  </div>  </nav>  <!-- Page Content -->  <div class=*"content"*>  <div class=*"container"*>  <c:if test=*"*${**not empty** message}*"*>  <div class=*"row"*>  <div class=*"col-xs-12 col-md-offset-2 col-md-8"*>  <div class=*"alert alert-danger fade in"*>${message}</div>  </div>  </div>  </c:if>  <c:if test=*"*${**not empty** logout}*"*>  <div class=*"row"*>  <div class=*"col-xs-12 col-md-offset-2 col-md-8"*>  <div class=*"alert alert-success"*>${logout}</div>  </div>  </div>  </c:if>  <div class=*"row"*>  <div class=*"col-md-offset-3 col-md-6"*>  <div class=*"panel panel-primary"*>  <div class=*"panel-heading"*>  <h4>Login</h4>  </div>  <div class=*"panel-body"*>  <form action=*"*${contextRoot}*/login"* method=*"POST"* class=*"form-horizontal"*  id=*"loginForm"*>  <div class=*"form-group"*>  <label for=*"username"* class=*"col-md-4 control-label"*>Email: </label>  <div class=*"col-md-8"*>  <input type=*"text"* name=*"username"* id=*"username"* class=*"form-control"*/>  </div>  </div>  <div class=*"form-group"*>  <label for=*"password"* class=*"col-md-4 control-label"*>Password: </label>  <div class=*"col-md-8"*>  <input type=*"password"* name=*"password"* id=*"password"* class=*"form-control"*/>  </div>  </div>  <div class=*"form-group"*>  <div class=*"col-md-offset-4 col-md-8"*>  <input type=*"hidden"* name=*"*${\_csrf.parameterName}*"* value=*"*${\_csrf.token}*"*/>  <input type=*"submit"* value=*"Login"* class=*"btn btn-primary"*/>  </div>  </div>  </form>  </div>  <div class=*"panel-footer"*>  <div class=*"text-right"*>  New User - <a href=*"*${contextRoot}*/register"*>Register Here</a>  </div>  </div>  </div>  </div>  </div>  </div>  </div>  <!-- Footer comes here -->  <%@include file=*"./shared/footer.jsp"*%>  <!-- jQuery -->  <script src=*"*${js}*/jquery.js"*></script>  <script src=*"*${js}*/jquery.validate.js"*></script>  <!-- Bootstrap Core JavaScript -->  <script src=*"*${js}*/bootstrap.min.js"*></script>  <!-- Self coded javascript -->  <script src=*"*${js}*/myapp.js"*></script>  </div>  </body>  </html> |

|  |
| --- |
| Inside webapp/WEB-INF/views/manageProduct.jsp |
| <%@taglib prefix=*"sf"* uri=*"http://www.springframework.org/tags/form"* %>  <div class=*"container"*>  <c:if test="${**not empty** message}">  <div class=*"row"*>  <div class=*"col-xs-12 col-md-offset-2 col-md-8"*>  <div class=*"alert alert-info fade in"*>${message}</div>  </div>  </div>  </c:if>  <div class=*"row"*>  <div class=*"col-md-offset-2 col-md-8"*>  <div class=*"panel panel-primary"*>  <div class=*"panel-heading"*>  <h4>Product Management</h4>  </div>  <div class=*"panel-body"*>  <sf:form class=*"form-horizontal"* modelAttribute=*"product"* action=*"*${contextRoot}*/manage/product"* method=*"POST"* enctype=*"multipart/form-data"*>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"*>Name</label>  <div class=*"col-md-8"*>  <sf:input type=*"text"* path=*"name"* class=*"form-control"* placeholder=*"Product Name"* />  <sf:errors path=*"name"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"*>Brand</label>  <div class=*"col-md-8"*>  <sf:input type=*"text"* path=*"brand"* class=*"form-control"* placeholder=*"Brand Name"* />  <sf:errors path=*"brand"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"*>Description</label>  <div class=*"col-md-8"*>  <sf:textarea path=*"description"* class=*"form-control"* placeholder=*"Enter your description here!"* />  <sf:errors path=*"description"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"*>Unit Price</label>  <div class=*"col-md-8"*>  <sf:input type=*"number"* path=*"unitPrice"* class=*"form-control"* placeholder=*"Enter Unit Price"* />  <sf:errors path=*"unitPrice"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"*>Quantity</label>  <div class=*"col-md-8"*>  <sf:input type=*"number"* path=*"quantity"* class=*"form-control"* placeholder=*"Enter Quantity"* />  <sf:errors path=*"quantity"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"*>Upload a file</label>  <div class=*"col-md-8"*>  <sf:input type=*"file"* path=*"file"* class=*"form-control"*/>  <sf:errors path=*"file"* cssClass=*"help-block"* element=*"em"*/>  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"*>Category</label>  <div class=*"col-md-8"*>  <sf:select path=*"categoryId"* items=*"*${categories}*"* itemLabel=*"name"* itemValue=*"id"* class=*"form-control"*/>  <div class=*"text-right"*>  <br/>  <sf:hidden path=*"id"*/>  <sf:hidden path=*"code"*/>  <sf:hidden path=*"supplierId"*/>  <sf:hidden path=*"active"*/>  <button type=*"button"* class=*"btn btn-warning btn-xs"* data-toggle=*"modal"* data-target=*"#myCategoryModal"*>Add New Category</button>  </div>  </div>  </div>  <div class=*"form-group"*>  <div class=*"col-md-offset-4 col-md-4"*>  <input type=*"submit"* name=*"submit"* value=*"Save"* class=*"btn btn-primary"*/>  </div>  </div>  </sf:form>  </div>  </div>  </div>  </div>  <!-- Modal -->  <div class=*"modal fade"* id=*"myCategoryModal"* tabindex=*"-1"* role=*"dialog"* aria-labelledby=*"myModalLabel"*>  <div class=*"modal-dialog"* role=*"document"*>  <div class=*"modal-content"*>  <div class=*"modal-header"*>  <button type=*"button"* class=*"close"* data-dismiss=*"modal"* aria-label=*"Close"*><span aria-hidden=*"true"*>&times;</span></button>  <h4 class=*"modal-title"* id=*"myModalLabel"*>New Category</h4>  </div>  <div class=*"modal-body"*>  <sf:form id=*"categoryForm"* class=*"form-horizontal"* modelAttribute=*"category"* action=*"*${contextRoot}*/manage/category"* method=*"POST"*>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"*>Name</label>  <div class=*"col-md-8 validate"*>  <sf:input type=*"text"* path=*"name"* class=*"form-control"*  placeholder=*"Category Name"* />  </div>  </div>  <div class=*"form-group"*>  <label class=*"control-label col-md-4"*>Description</label>  <div class=*"col-md-8 validate"*>  <sf:textarea path=*"description"* class=*"form-control"*  placeholder=*"Enter category description here!"* />  </div>  </div>  <div class=*"form-group"*>  <div class=*"col-md-offset-4 col-md-4"*>  <input type=*"submit"* name=*"submit"* value=*"Save"* class=*"btn btn-primary"*/>  </div>  </div>  </sf:form>  </div>  </div>  </div>  </div>  <hr/>  <h1>Available Products</h1>  <hr/>  <div class=*"row"*>  <div class=*'col-xs-12'*>  <table id=*"productsTable"* class=*"table table-condensed table-bordered"*>  <thead>  <tr>  <th>Id</th>  <th>&#160;</th>  <th>Name</th>  <th>Brand</th>  <th>Qty. Avail</th>  <th>Unit Price</th>  <th>Activate</th>  <th>Edit</th>  </tr>  </thead>  <tfoot>  <tr>  <th>Id</th>  <th>&#160;</th>  <th>Name</th>  <th>Brand</th>  <th>Qty. Avail</th>  <th>Unit Price</th>  <th>Activate</th>  <th>Edit</th>  </tr>  </tfoot>  </table>  </div>  </div>  </div> |

|  |
| --- |
| Inside webapp/WEB-INF/views/page.jsp |
| <%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"* pageEncoding=*"ISO-8859-1"*%>  <%@taglib prefix=*"c"* uri=*"http://java.sun.com/jsp/jstl/core"*%>  <%@taglib prefix=*"spring"* uri=*"http://www.springframework.org/tags"*%>  <spring:url var=*"css"* value=*"/resources/css"* />  <spring:url var=*"js"* value=*"/resources/js"* />  <spring:url var=*"images"* value=*"/resources/images"* />  <c:set var=*"contextRoot"* value=*"*${pageContext.request.contextPath}*"* />  <!DOCTYPE html>  <html lang=*"en"*>  <head>  <meta charset=*"utf-8"*>  <meta http-equiv=*"X-UA-Compatible"* content=*"IE=edge"*>  <meta name=*"viewport"* content=*"width=device-width, initial-scale=1"*>  <meta name=*"description"* content=*"Online Shopping Website Using Spring MVC and Hibernate"*>  <meta name=*"author"* content=*"Khozema Nullwala"*>  <meta name=*"\_csrf"* content=*"*${\_csrf.token}*"*>  <meta name=*"\_csrf\_header"* content=*"*${\_csrf.headerName}*"*>  <title>Online Shopping - ${title}</title>  <script>  window.menu = '${title}';  window.contextRoot = '${contextRoot}'  </script>  <!-- Bootstrap Core CSS -->  <link href=*"*${css}*/bootstrap.min.css"* rel=*"stylesheet"*>  <!-- Bootstrap Readable Theme -->  <link href=*"*${css}*/bootstrap-readable-theme.css"* rel=*"stylesheet"*>  <!-- Bootstrap DataTables -->  <link href=*"*${css}*/dataTables.bootstrap.css"* rel=*"stylesheet"*>  <!-- Custom CSS -->  <link href=*"*${css}*/myapp.css"* rel=*"stylesheet"*>  <!-- HTML5 Shim and Respond.js IE8 support of HTML5 elements and media queries -->  <!-- WARNING: Respond.js doesn't work if you view the page via file:// -->  <!--[if lt IE 9]>  <script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>  <script src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js"></script>  <![endif]-->  </head>  <body>  <div class=*"se-pre-con"*></div>  <div class=*"wrapper"*>  <!-- Navigation -->  <%@include file=*"./shared/navbar.jsp"*%>  <!-- Page Content -->  <div class=*"content"*>  <!-- Loading the home content -->  <c:if test=*"*${userClickHome == **true** }*"*>  <%@include file=*"home.jsp"*%>  </c:if>  <!-- Load only when user clicks about -->  <c:if test=*"*${userClickAbout == **true** }*"*>  <%@include file=*"about.jsp"*%>  </c:if>  <!-- Load only when user clicks contact -->  <c:if test=*"*${userClickContact == **true** }*"*>  <%@include file=*"contact.jsp"*%>  </c:if>  <!-- Load only when user clicks contact -->  <c:if test=*"*${userClickAllProducts == **true or** userClickCategoryProducts == **true** }*"*>  <%@include file=*"listProducts.jsp"*%>  </c:if>  <!-- Load only when user clicks show product -->  <c:if test=*"*${userClickShowProduct == **true**}*"*>  <%@include file=*"singleProduct.jsp"*%>  </c:if>  <!-- Load only when user clicks manage product -->  <c:if test=*"*${userClickManageProduct == **true**}*"*>  <%@include file=*"manageProduct.jsp"*%>  </c:if>  <!-- Load only when user clicks manage product -->  <c:if test=*"*${userClickShowCart == **true**}*"*>  <%@include file=*"cart.jsp"*%>  </c:if>  </div>  <!-- Footer comes here -->  <%@include file=*"./shared/footer.jsp"*%>  <!-- jQuery -->  <script src=*"*${js}*/jquery.js"*></script>  <script src=*"*${js}*/jquery.validate.js"*></script>  <!-- Bootstrap Core JavaScript -->  <script src=*"*${js}*/bootstrap.min.js"*></script>  <!-- DataTable Plugin -->  <script src=*"*${js}*/jquery.dataTables.js"*></script>  <!-- DataTable Bootstrap Script -->  <script src=*"*${js}*/dataTables.bootstrap.js"*></script>  <!-- DataTable Bootstrap Script -->  <script src=*"*${js}*/bootbox.min.js"*></script>  <!-- Self coded javascript -->  <script src=*"*${js}*/myapp.js"*></script>  </div>  </body>  </html> |

|  |
| --- |
| Inside webapp/WEB-INF/views/singleProduct.jsp |
| <div class=*"container"*>  <!-- Breadcrumb -->  <div class=*"row"*>  <div class=*"col-xs-12"*>  <ol class=*"breadcrumb"*>  <li><a href=*"*${contextRoot}*/home"*>Home</a></li>  <li><a href=*"*${contextRoot}*/show/all/products"*>Products</a></li>  <li class=*"active"*>${product.name}</li>  </ol>  </div>  </div>  <div class=*"row"*>  <!-- Display the product image -->  <div class=*"col-xs-12 col-sm-4"*>  <div class=*"thumbnail"*>  <img src=*"*${images}*/*${product.code}*.jpg"* class=*"img img-responsive"*/>  </div>  </div>  <!-- Display the product description -->  <div class=*"col-xs-12 col-sm-8"*>  <h3>${product.name}</h3>  <hr/>  <p>${product.description}</p>  <hr/>  <h4>Price: <strong> &#8377; ${product.unitPrice} /-</strong></h4>  <hr/>  <c:choose>  <c:when test="${product.quantity < 1}">    <h6>Qty. Available: <span style="color:*red*">Out of Stock!</span></h6>  </c:when>  <c:otherwise>  <h6>Qty. Available: ${product.quantity}</h6>  </c:otherwise>  </c:choose>  <security:authorize access=*"isAnonymous() or hasAuthority('USER')"*>  <c:choose>  <c:when test="${product.quantity < 1}">  <a href=*"javascript:void(0)"* class=*"btn btn-success disabled"*><strike>  <span class=*"glyphicon glyphicon-shopping-cart"*></span> Add to Cart</strike></a>  </c:when>  <c:otherwise>  <a href=*"*${contextRoot}*/cart/add/*${product.id}*/product"* class=*"btn btn-success"*>  <span class=*"glyphicon glyphicon-shopping-cart"*></span> Add to Cart</a>  </c:otherwise>  </c:choose>  </security:authorize>  <security:authorize access=*"hasAuthority('ADMIN')"*>  <a href=*"*${contextRoot}*/manage/*${product.id}*/product"* class=*"btn btn-success"*>  <span class=*"glyphicon glyphicon-pencil"*></span> Edit</a>  </security:authorize>  <a href=*"*${contextRoot}*/show/all/products"* class=*"btn btn-warning"*>Continue Shopping</a>  </div>  </div>  </div> |

|  |
| --- |
| Inside webapp/WEB-INF/dispatcher-servlet.xml |
| <beans xmlns=*"http://www.springframework.org/schema/beans"*  xmlns:context=*"http://www.springframework.org/schema/context"*  xmlns:mvc=*"http://www.springframework.org/schema/mvc"*  xmlns:wf=*"http://www.springframework.org/schema/webflow-config"*  xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*  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/webflow-config*  *http://www.springframework.org/schema/webflow-config/spring-webflow-config.xsd*  *"*>  <context:component-scan base-package=*"net.kzn.onlineshopping"*/>  <bean id=*"viewResolver"*  class=*"org.springframework.web.servlet.view.InternalResourceViewResolver"*>  <property name=*"prefix"* value=*"/WEB-INF/views/"* />  <property name=*"suffix"* value=*".jsp"* />  </bean>  <bean id=*"multipartResolver"*  class=*"org.springframework.web.multipart.support.StandardServletMultipartResolver"*/>  <!-- Loading static resources -->  <mvc:annotation-driven />  <mvc:resources location=*"/assets/"* mapping=*"/resources/\*\*"* />  <!-- Webflow configuration -->  <wf:flow-executor id=*"flowExecutor"* />  <wf:flow-registry id=*"flowRegistry"* base-path=*"/WEB-INF/views/flows"* flow-builder-services=*"flowBuilderServices"*>  <wf:flow-location-pattern value=*"/\*\*/\*-flow.xml"*/>  </wf:flow-registry>  <wf:flow-builder-services id=*"flowBuilderServices"*  view-factory-creator=*"viewFactoryCreator"*  validator=*"validator"*  />  <bean class=*"org.springframework.webflow.mvc.servlet.FlowHandlerAdapter"*>  <property name=*"flowExecutor"* ref=*"flowExecutor"* />  </bean>  <bean class=*"org.springframework.webflow.mvc.servlet.FlowHandlerMapping"*>  <property name=*"flowRegistry"* ref=*"flowRegistry"* />  <property name=*"order"* value=*"-1"* />  </bean>  <bean id=*"viewFactoryCreator"* class=*"org.springframework.webflow.mvc.builder.MvcViewFactoryCreator"*>  <property name=*"viewResolvers"* ref=*"viewResolver"* />  </bean>  <bean id=*"validator"* class=*"org.springframework.validation.beanvalidation.LocalValidatorFactoryBean"*/>  </beans> |

|  |
| --- |
| Inside webapp/WEB-INF/spring-security.xml |
| <beans:beans xmlns=*"http://www.springframework.org/schema/security"*  xmlns:beans=*"http://www.springframework.org/schema/beans"*  xmlns:context=*"http://www.springframework.org/schema/context"*  xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*  xsi:schemaLocation=*"http://www.springframework.org/schema/beans*  *http://www.springframework.org/schema/beans/spring-beans.xsd*  *http://www.springframework.org/schema/security*  *http://www.springframework.org/schema/security/spring-security.xsd*  *http://www.springframework.org/schema/context*  *http://www.springframework.org/schema/context/spring-context.xsd"*>  <context:component-scan base-package=*"net.kzn.shoppingbackend"*/>  <beans:bean id=*"passwordEncoder"* class=*"org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder"*/>  <http pattern=*"/resources/\*\*"* security=*"none"*/>  <http>  <!-- only admin access -->  <intercept-url pattern=*"/manage/\*\*"* access=*"hasAuthority('ADMIN')"* />  <!-- only user access (who is registered) -->  <intercept-url pattern=*"/cart/\*\*"* access=*"hasAuthority('USER')"* />  <!-- rest of the world -->  <intercept-url pattern=*"/\*\*"* access=*"permitAll"* />  <form-login login-page=*"/login"* />  <access-denied-handler error-page=*"/access-denied"*/>  </http>  <authentication-manager>  <authentication-provider>  <password-encoder ref=*"passwordEncoder"* />  <jdbc-user-service data-source-ref=*"dataSource"*  authorities-by-username-query=*"select email, role from user\_detail where email = ?"*  users-by-username-query=*"select email, password, enabled from user\_detail where email = ?"*/>  </authentication-provider>  </authentication-manager>  </beans:beans> |

|  |
| --- |
| Inside webapp/WEB-INF/web.xml |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <web-app>  <!-- Configuring front-controller -->  <servlet>  <servlet-name>dispatcher</servlet-name>  <servlet-class>org.springframework.web.servlet.DispatcherServlet  </servlet-class>  <!-- FOr throwing exception -->  <init-param>  <param-name>throwExceptionIfNoHandlerFound</param-name>  <param-value>true</param-value>  </init-param>  <multipart-config>  <max-file-size>20848820</max-file-size>  <max-request-size>418018841</max-request-size>  <file-size-threshold>1048576</file-size-threshold>  </multipart-config>  </servlet>  <servlet-mapping>  <servlet-name>dispatcher</servlet-name>  <url-pattern>/</url-pattern>  </servlet-mapping>  <!-- Configuration for spring security -->  <listener>  <listener-class>org.springframework.web.context.ContextLoaderListener  </listener-class>  </listener>  <!-- Load spring security config file -->  <context-param>  <param-name>contextConfigLocation</param-name>  <param-value>  /WEB-INF/spring-security.xml  </param-value>  </context-param>  <!-- Spring Security filter -->  <filter>  <filter-name>springSecurityFilterChain</filter-name>  <filter-class>org.springframework.web.filter.DelegatingFilterProxy  </filter-class>  </filter>  <filter-mapping>  <filter-name>springSecurityFilterChain</filter-name>  <url-pattern>/\*</url-pattern>  </filter-mapping>  </web-app> |