**Source code of SportyShoes.com online store ---By Dr. Jyothi N.M**

**Git link**

[**https://github.com/jyothiarunkr/simplilearn\_Phase3\_project.git**](https://github.com/jyothiarunkr/simplilearn_Phase3_project.git)

**Source code with output screen**

**Step1 create database**

**MYSQL**

?

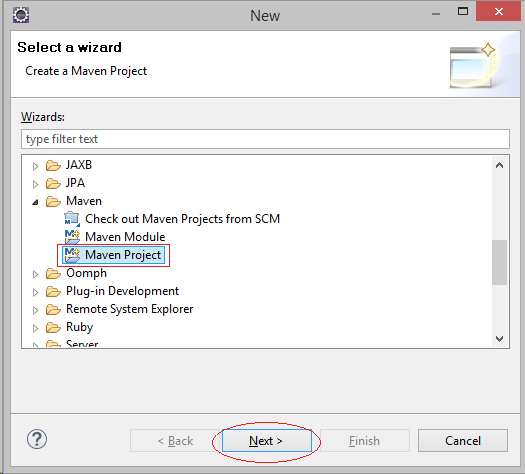
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74 | create table Accounts (          User\_Name varchar(20) not null,          Active bit not null,          Password varchar(20) not null,          User\_Role varchar(20) not null,          primary key (User\_Name)      );        create table Order\_Details (          ID varchar(50) not null,          Amount double precision not null,          Price double precision not null,          Quanity integer not null,          ORDER\_ID varchar(50) not null,          PRODUCT\_ID varchar(20) not null,          primary key (ID)      );        create table Orders (          ID varchar(50) not null,          Amount double precision not null,          Customer\_Address varchar(255) not null,          Customer\_Email varchar(128) not null,          Customer\_Name varchar(255) not null,          Customer\_Phone varchar(128) not null,          Order\_Date datetime not null,          Order\_Num integer not null,          primary key (ID)      );        create table Products (          Code varchar(20) not null,          Create\_Date datetime not null,          Image longblob,          Name varchar(255) not null,          Price double precision not null,          primary key (Code)      );        alter table Orders          add constraint UK\_sxhpvsj665kmi4f7jdu9d2791  unique (Order\_Num);        alter table Order\_Details          add constraint ORDER\_DETAIL\_ORD\_FK          foreign key (ORDER\_ID)          references Orders (ID);        alter table Order\_Details          add constraint ORDER\_DETAIL\_PROD\_FK          foreign key (PRODUCT\_ID)          references Products (Code);    ---------------------------------------  insert into Accounts (USER\_NAME, ACTIVE, PASSWORD, USER\_ROLE)  values ('employee1', 1, '123', 'EMPLOYEE');    insert into Accounts (USER\_NAME, ACTIVE, PASSWORD, USER\_ROLE)  values ('manager1', 1, '123', 'MANAGER');    ----------------  insert into products (CODE, NAME, PRICE, CREATE\_DATE)  values ('S001', 'Core Java', 100, current\_timestamp() );    insert into products (CODE, NAME, PRICE, CREATE\_DATE)  values ('S002', 'Spring for Beginners', 50, current\_timestamp() );    insert into products (CODE, NAME, PRICE, CREATE\_DATE)  values ('S003', 'Swift for Beginners', 120, current\_timestamp() );    insert into products (CODE, NAME, PRICE, CREATE\_DATE)  values ('S004', 'Oracle XML Parser', 120, current\_timestamp() );    insert into products (CODE, NAME, PRICE, CREATE\_DATE)  values ('S005', 'CSharp Tutorial for Beginers', 110, current\_timestamp() ); |

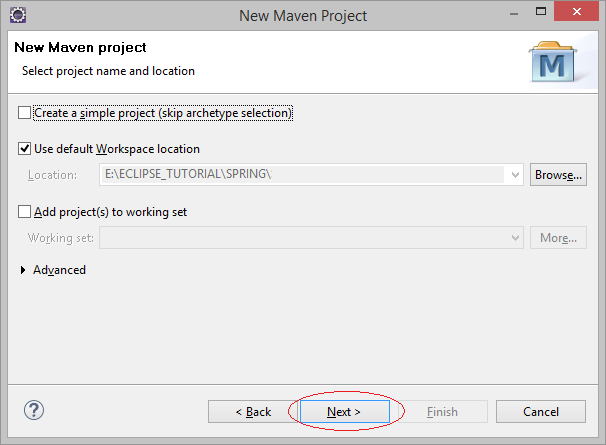
**Step 2**

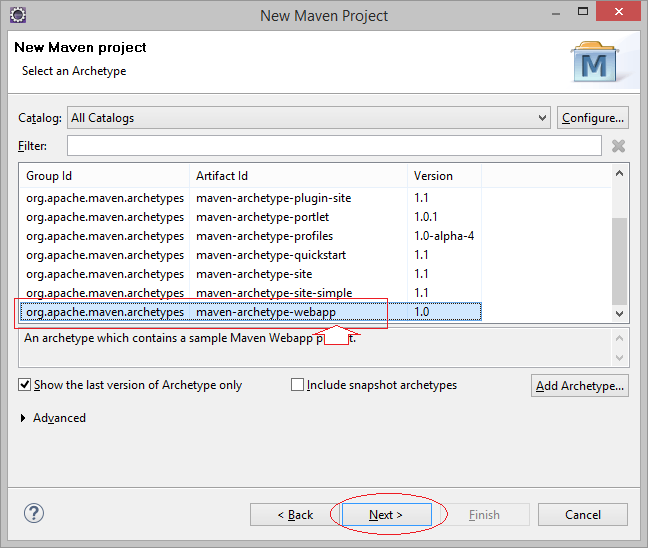
**CREATE PROJECT**

In Eclipse select:

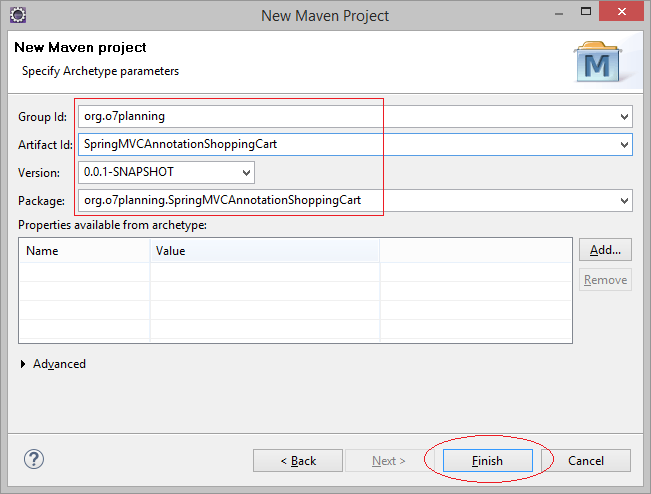
* **File/New/Others..**



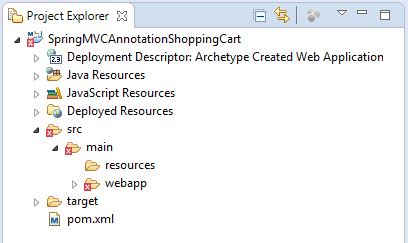




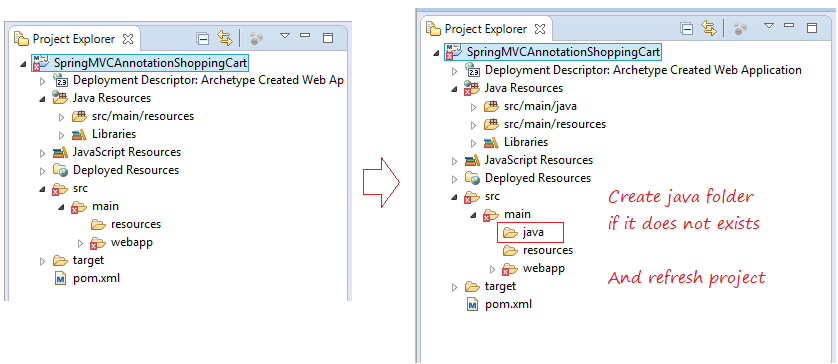
* **Group Id:** org.o7planning
* **Artifact Id:** SpringMVCAnnotationShoppingCart



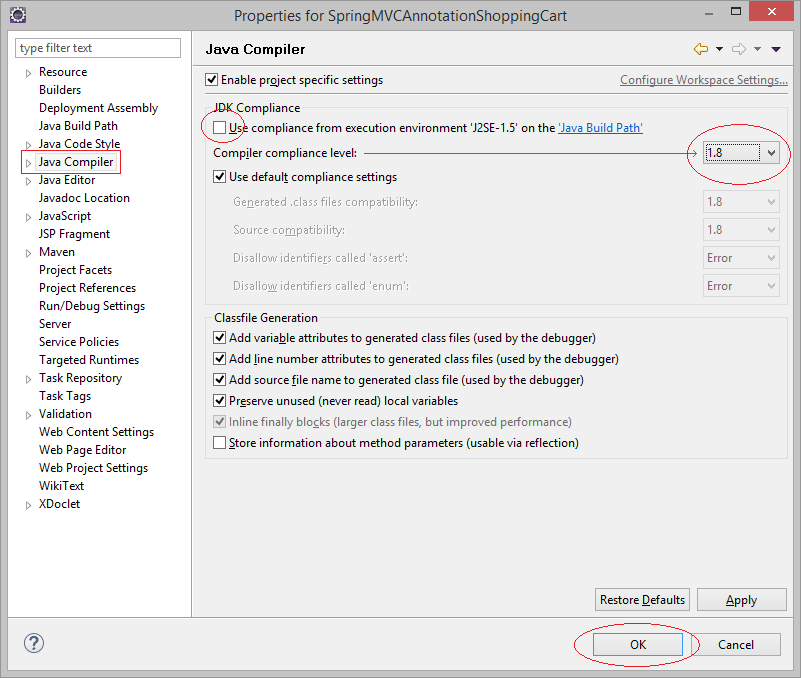
Project is created.



Fix Project:



Right click Project, select Properties.  
  
Select Java Compiler 7 or 8:



**5- Configure Web.xml & Maven**

You need to edit **web.xml** to use **Web App >= 3.x**

**web.xml**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | <?xml version="1.0" encoding="UTF-8"?>  <web-app xmlns:xsi="<http://www.w3.org/2001/XMLSchema-instance>"   xmlns="<http://java.sun.com/xml/ns/javaee>"   xsi:schemaLocation="<http://java.sun.com/xml/ns/javaee>  <http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd>"   id="WebApp\_ID" version="3.0">     <display-name>Spring MVC ShoppingCart</display-name>      </web-app> |

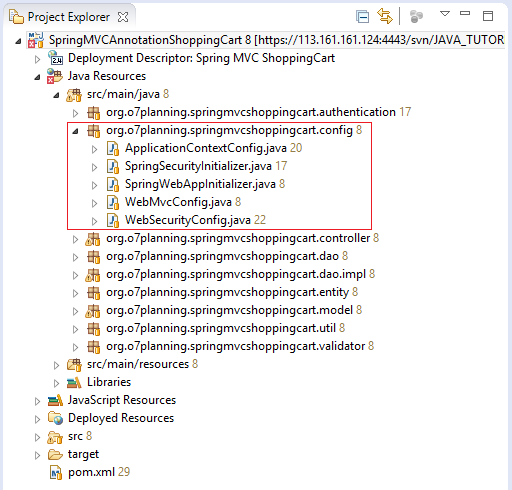
Configure Maven:

**pom.xml**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149  150  151  152  153  154  155  156  157  158  159  160  161  162  163  164  165  166  167  168  169  170  171  172  173  174  175  176  177  178  179  180  181  182  183  184  185  186  187  188  189  190  191  192  193  194  195  196  197  198  199  200  201  202  203  204  205  206  207  208  209  210  211  212  213  214  215  216  217  218 | <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>org.o7planning</groupId>      <artifactId>SpringMVCAnnotationShoppingCart</artifactId>      <packaging>war</packaging>      <version>1.0.0</version>      <name>SpringMVCAnnotationShoppingCart Maven Webapp</name>      <url>[http://maven.apache.org](http://maven.apache.org/)</url>          <properties>          <java-version>1.7</java-version>      </properties>        <repositories>          <!-- Repository for ORACLE JDBC Driver -->          <repository>              <id>codelds</id>              <url><https://code.lds.org/nexus/content/groups/main-repo></url>          </repository>      </repositories>        <dependencies>            <dependency>              <groupId>junit</groupId>              <artifactId>junit</artifactId>              <version>3.8.1</version>              <scope>test</scope>          </dependency>            <!-- Servlet API -->          <!-- <http://mvnrepository.com/artifact/javax.servlet/javax.servlet-api> -->          <dependency>              <groupId>javax.servlet</groupId>              <artifactId>javax.servlet-api</artifactId>              <version>3.1.0</version>              <scope>provided</scope>          </dependency>            <!-- Jstl for jsp page -->          <!-- <http://mvnrepository.com/artifact/javax.servlet/jstl> -->          <dependency>              <groupId>javax.servlet</groupId>              <artifactId>jstl</artifactId>              <version>1.2</version>          </dependency>              <!-- JSP API -->          <!-- <http://mvnrepository.com/artifact/javax.servlet.jsp/jsp-api> -->          <dependency>              <groupId>javax.servlet.jsp</groupId>              <artifactId>jsp-api</artifactId>              <version>2.2</version>              <scope>provided</scope>          </dependency>           <!-- Apache Commons FileUpload -->         <!-- <http://mvnrepository.com/artifact/commons-fileupload/commons-fileupload> -->         <dependency>             <groupId>commons-fileupload</groupId>             <artifactId>commons-fileupload</artifactId>             <version>1.3.1</version>         </dependency>            <!-- Spring dependencies -->          <!-- <http://mvnrepository.com/artifact/org.springframework/spring-core> -->          <dependency>              <groupId>org.springframework</groupId>              <artifactId>spring-core</artifactId>              <version>4.2.5.RELEASE</version>          </dependency>            <!-- <http://mvnrepository.com/artifact/org.springframework/spring-web> -->          <dependency>              <groupId>org.springframework</groupId>              <artifactId>spring-web</artifactId>              <version>4.2.5.RELEASE</version>          </dependency>            <!-- <http://mvnrepository.com/artifact/org.springframework/spring-webmvc> -->          <dependency>              <groupId>org.springframework</groupId>              <artifactId>spring-webmvc</artifactId>              <version>4.2.5.RELEASE</version>          </dependency>            <!-- <http://mvnrepository.com/artifact/org.springframework/spring-orm> -->          <dependency>              <groupId>org.springframework</groupId>              <artifactId>spring-orm</artifactId>              <version>4.2.5.RELEASE</version>          </dependency>            <!-- Spring Security Artifacts - START -->          <!-- <http://mvnrepository.com/artifact/org.springframework.security/spring-security-web> -->          <dependency>              <groupId>org.springframework.security</groupId>              <artifactId>spring-security-web</artifactId>              <version>4.0.4.RELEASE</version>          </dependency>            <!-- <http://mvnrepository.com/artifact/org.springframework.security/spring-security-config> -->          <dependency>              <groupId>org.springframework.security</groupId>              <artifactId>spring-security-config</artifactId>              <version>4.0.4.RELEASE</version>          </dependency>            <!-- <http://mvnrepository.com/artifact/org.springframework.security/spring-security-taglibs> -->          <dependency>              <groupId>org.springframework.security</groupId>              <artifactId>spring-security-taglibs</artifactId>              <version>4.0.4.RELEASE</version>          </dependency>            <!-- Spring Security Artifacts - END -->            <!-- Hibernate -->          <!-- <http://mvnrepository.com/artifact/org.hibernate/hibernate-core> -->          <dependency>              <groupId>org.hibernate</groupId>              <artifactId>hibernate-core</artifactId>              <version>5.2.10.Final</version>          </dependency>            <!-- <http://mvnrepository.com/artifact/org.hibernate/hibernate-entitymanager> -->          <dependency>              <groupId>org.hibernate</groupId>              <artifactId>hibernate-entitymanager</artifactId>              <version>5.2.10.Final</version>          </dependency>              <!-- <http://mvnrepository.com/artifact/org.hibernate/hibernate-c3p0> -->          <dependency>              <groupId>org.hibernate</groupId>              <artifactId>hibernate-c3p0</artifactId>              <version>5.2.10.Final</version>          </dependency>              <!-- MySQL JDBC driver -->          <!-- <http://mvnrepository.com/artifact/mysql/mysql-connector-java> -->          <dependency>              <groupId>mysql</groupId>              <artifactId>mysql-connector-java</artifactId>              <version>5.1.34</version>          </dependency>            <!-- Oracle JDBC driver -->          <dependency>              <groupId>com.oracle</groupId>              <artifactId>ojdbc6</artifactId>              <version>11.2.0.3</version>          </dependency>            <!-- SQLServer JDBC driver (JTDS) -->          <!-- <http://mvnrepository.com/artifact/net.sourceforge.jtds/jtds> -->          <dependency>              <groupId>net.sourceforge.jtds</groupId>              <artifactId>jtds</artifactId>              <version>1.3.1</version>          </dependency>            <!-- SQLServer JDBC driver (MSSQL-JDBC) -->          <!-- <https://mvnrepository.com/artifact/com.microsoft.sqlserver/mssql-jdbc> -->          <dependency>              <groupId>com.microsoft.sqlserver</groupId>              <artifactId>mssql-jdbc</artifactId>              <version>6.1.0.jre7</version>          </dependency>            <!-- PostgreSQL driver -->          <!-- <https://mvnrepository.com/artifact/postgresql/postgresql> -->          <dependency>              <groupId>postgresql</groupId>              <artifactId>postgresql</artifactId>              <version>9.1-901-1.jdbc4</version>          </dependency>                <!-- Email validator,... -->          <!-- <http://mvnrepository.com/artifact/commons-validator/commons-validator> -->          <dependency>              <groupId>commons-validator</groupId>              <artifactId>commons-validator</artifactId>              <version>1.5.0</version>          </dependency>        </dependencies>          <build>          <finalName>SpringMVCAnnotationShoppingCart</finalName>          <plugins>                <!-- Config: Maven Tomcat Plugin -->              <!-- <http://mvnrepository.com/artifact/org.apache.tomcat.maven/tomcat7-maven-plugin> -->              <plugin>                  <groupId>org.apache.tomcat.maven</groupId>                  <artifactId>tomcat7-maven-plugin</artifactId>                  <version>2.2</version>                  <!-- Config: contextPath and Port (Default - /SpringMVCAnnotationShoppingCart                      : 8080) -->                  <!-- <configuration> <path>/</path> <port>8899</port> </configuration> -->              </plugin>          </plugins>      </build>      </project> |

**6- Configure Spring MVC**



**SpringWebAppInitializer.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40 | package org.o7planning.springmvcshoppingcart.config;    import javax.servlet.FilterRegistration;  import javax.servlet.ServletContext;  import javax.servlet.ServletException;  import javax.servlet.ServletRegistration;    import org.springframework.web.WebApplicationInitializer;  import org.springframework.web.context.ContextLoaderListener;  import org.springframework.web.context.support.AnnotationConfigWebApplicationContext;  import org.springframework.web.filter.CharacterEncodingFilter;  import org.springframework.web.servlet.DispatcherServlet;    public class SpringWebAppInitializer implements WebApplicationInitializer {        @Override      public void onStartup(ServletContext servletContext) throws ServletException {          AnnotationConfigWebApplicationContext appContext = new AnnotationConfigWebApplicationContext();          appContext.register(ApplicationContextConfig.class);            ServletRegistration.Dynamic dispatcher = servletContext.addServlet("SpringDispatcher",                  new DispatcherServlet(appContext));          dispatcher.setLoadOnStartup(1);          dispatcher.addMapping("/");              ContextLoaderListener contextLoaderListener = new ContextLoaderListener(appContext);            servletContext.addListener(contextLoaderListener);              // Filter.          FilterRegistration.Dynamic fr = servletContext.addFilter("encodingFilter", CharacterEncodingFilter.class);            fr.setInitParameter("encoding", "UTF-8");          fr.setInitParameter("forceEncoding", "true");          fr.addMappingForUrlPatterns(null, true, "/\*");      }    } |

**ApplicationContextConfig** class used to configure Spring MVC Context, including:

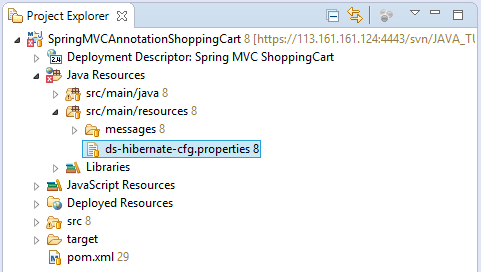
* View Resolver
* Datasouce
* Hiberante (Hibernate Transaction Manager, Hibernate Session,..)
* DAO
* Bean
* ....

**ApplicationContextConfig.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134 | package org.o7planning.springmvcshoppingcart.config;    import java.util.Properties;    import javax.sql.DataSource;    import org.hibernate.SessionFactory;  import org.o7planning.springmvcshoppingcart.dao.AccountDAO;  import org.o7planning.springmvcshoppingcart.dao.OrderDAO;  import org.o7planning.springmvcshoppingcart.dao.ProductDAO;  import org.o7planning.springmvcshoppingcart.dao.impl.AccountDAOImpl;  import org.o7planning.springmvcshoppingcart.dao.impl.OrderDAOImpl;  import org.o7planning.springmvcshoppingcart.dao.impl.ProductDAOImpl;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.context.annotation.Bean;  import org.springframework.context.annotation.ComponentScan;  import org.springframework.context.annotation.Configuration;  import org.springframework.context.annotation.PropertySource;  import org.springframework.context.support.ResourceBundleMessageSource;  import org.springframework.core.env.Environment;  import org.springframework.jdbc.datasource.DriverManagerDataSource;  import org.springframework.orm.hibernate5.HibernateTransactionManager;  import org.springframework.orm.hibernate5.LocalSessionFactoryBean;  import org.springframework.transaction.annotation.EnableTransactionManagement;  import org.springframework.web.multipart.commons.CommonsMultipartResolver;  import org.springframework.web.servlet.view.InternalResourceViewResolver;    @Configuration  @ComponentScan("org.o7planning.springmvcshoppingcart.\*")  @EnableTransactionManagement  // Load to Environment.  @PropertySource("classpath:ds-hibernate-cfg.properties")  public class ApplicationContextConfig {        // The Environment class serves as the property holder      // and stores all the properties loaded by the @PropertySource      @Autowired      private Environment env;        @Bean      public ResourceBundleMessageSource messageSource() {          ResourceBundleMessageSource rb = new ResourceBundleMessageSource();          // Load property in message/validator.properties          rb.setBasenames(new String[] { "messages/validator" });          return rb;      }        @Bean(name = "viewResolver")      public InternalResourceViewResolver getViewResolver() {          InternalResourceViewResolver viewResolver = new InternalResourceViewResolver();          viewResolver.setPrefix("/WEB-INF/pages/");          viewResolver.setSuffix(".jsp");          return viewResolver;      }        // Config for Upload.      @Bean(name = "multipartResolver")      public CommonsMultipartResolver multipartResolver() {          CommonsMultipartResolver commonsMultipartResolver = new CommonsMultipartResolver();            // Set Max Size...          // commonsMultipartResolver.setMaxUploadSize(...);            return commonsMultipartResolver;      }        @Bean(name = "dataSource")      public DataSource getDataSource() {          DriverManagerDataSource dataSource = new DriverManagerDataSource();            // See: ds-hibernate-cfg.properties          dataSource.setDriverClassName(env.getProperty("ds.database-driver"));          dataSource.setUrl(env.getProperty("ds.url"));          dataSource.setUsername(env.getProperty("ds.username"));          dataSource.setPassword(env.getProperty("ds.password"));            System.out.println("## getDataSource: " + dataSource);            return dataSource;      }        @Autowired      @Bean(name = "sessionFactory")      public SessionFactory getSessionFactory(DataSource dataSource) throws Exception {          Properties properties = new Properties();            // See: ds-hibernate-cfg.properties          properties.put("hibernate.dialect", env.getProperty("hibernate.dialect"));          properties.put("hibernate.show\_sql", env.getProperty("hibernate.show\_sql"));          properties.put("current\_session\_context\_class", env.getProperty("current\_session\_context\_class"));              LocalSessionFactoryBean factoryBean = new LocalSessionFactoryBean();            // Package contain entity classes          factoryBean.setPackagesToScan(new String[] { "org.o7planning.springmvcshoppingcart.entity" });          factoryBean.setDataSource(dataSource);          factoryBean.setHibernateProperties(properties);          factoryBean.afterPropertiesSet();          //          SessionFactory sf = factoryBean.getObject();          System.out.println("## getSessionFactory: " + sf);          return sf;      }        @Autowired      @Bean(name = "transactionManager")      public HibernateTransactionManager getTransactionManager(SessionFactory sessionFactory) {          HibernateTransactionManager transactionManager = new HibernateTransactionManager(sessionFactory);            return transactionManager;      }        @Bean(name = "accountDAO")      public AccountDAO getApplicantDAO() {          return new AccountDAOImpl();      }        @Bean(name = "productDAO")      public ProductDAO getProductDAO() {          return new ProductDAOImpl();      }        @Bean(name = "orderDAO")      public OrderDAO getOrderDAO() {          return new OrderDAOImpl();      }        @Bean(name = "accountDAO")      public AccountDAO getAccountDAO()  {          return new AccountDAOImpl();      }    } |

**ApplicationContextConfig** will read the Datasource configuration information and **Hibernate Properties** in ***ds-hibernate-cfg.properties*** file.



Configurations for  **MySQL**

# DataSource

ds.database-driver=com.mysql.jdbc.Driver

ds.url=jdbc:mysql://localhost:3306/mydatabase

ds.username=root

ds.password=12345

# Hibernate Config

hibernate.dialect=org.hibernate.dialect.MySQLDialect

hibernate.show\_sql=true

current\_session\_context\_class=thread

hibernate.hbm2ddl.auto=update

Config for static resource (html, image, css,..)

**WebMvcConfig.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47 | package org.o7planning.springmvcshoppingcart.config;    import java.nio.charset.Charset;  import java.util.Arrays;  import java.util.List;    import org.springframework.context.annotation.Configuration;  import org.springframework.http.MediaType;  import org.springframework.http.converter.HttpMessageConverter;  import org.springframework.http.converter.StringHttpMessageConverter;  import org.springframework.web.servlet.config.annotation.DefaultServletHandlerConfigurer;  import org.springframework.web.servlet.config.annotation.EnableWebMvc;  import org.springframework.web.servlet.config.annotation.ResourceHandlerRegistry;  import org.springframework.web.servlet.config.annotation.WebMvcConfigurerAdapter;    @Configuration  @EnableWebMvc  public class WebMvcConfig extends WebMvcConfigurerAdapter {        private static final Charset UTF8 = Charset.forName("UTF-8");        // Config UTF-8 Encoding.      @Override      public void configureMessageConverters(List<HttpMessageConverter<?>> converters) {          StringHttpMessageConverter stringConverter = new StringHttpMessageConverter();          stringConverter.setSupportedMediaTypes(Arrays.asList(new MediaType("text", "plain", UTF8)));          converters.add(stringConverter);            // Add other converters ...      }        // Static Resource Config      // equivalents for <mvc:resources/> tags      @Override      public void addResourceHandlers(ResourceHandlerRegistry registry) {          registry.addResourceHandler("/css/\*\*").addResourceLocations("/css/").setCachePeriod(31556926);          registry.addResourceHandler("/img/\*\*").addResourceLocations("/img/").setCachePeriod(31556926);          registry.addResourceHandler("/js/\*\*").addResourceLocations("/js/").setCachePeriod(31556926);      }        // equivalent for <mvc:default-servlet-handler/> tag      @Override      public void configureDefaultServletHandling(DefaultServletHandlerConfigurer configurer) {          configurer.enable();      }    } |

Configure for security:

**SpringSecurityInitializer.java**

?

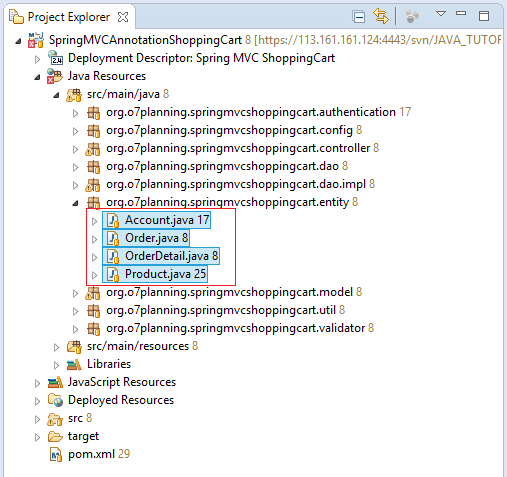
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | package org.o7planning.springmvcshoppingcart.config;    import org.springframework.security.web.context.AbstractSecurityWebApplicationInitializer;      public class SpringSecurityInitializer extends AbstractSecurityWebApplicationInitializer {        // Do nothing    } |

**WebSecurityConfig.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59 | package org.o7planning.springmvcshoppingcart.config;    import org.o7planning.springmvcshoppingcart.authentication.MyDBAuthenticationService;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.context.annotation.Configuration;  import org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;  import org.springframework.security.config.annotation.web.builders.HttpSecurity;  import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;  import org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;    @Configuration  // @EnableWebSecurity = @EnableWebMVCSecurity + Extra features  @EnableWebSecurity  public class WebSecurityConfig extends WebSecurityConfigurerAdapter {       @Autowired     MyDBAuthenticationService myDBAauthenticationService;       @Autowired     public void configureGlobal(AuthenticationManagerBuilder auth) throws Exception {           // For User in database.         auth.userDetailsService(myDBAauthenticationService);       }       @Override     protected void configure(HttpSecurity http) throws Exception {           http.csrf().disable();           // The pages requires login as EMPLOYEE or MANAGER.         // If no login, it will redirect to /login page.         http.authorizeRequests().antMatchers("/orderList","/order", "/accountInfo")//                 .access("hasAnyRole('ROLE\_EMPLOYEE', 'ROLE\_MANAGER')");           // For MANAGER only.         http.authorizeRequests().antMatchers("/product").access("hasRole('ROLE\_MANAGER')");           // When the user has logged in as XX.         // But access a page that requires role YY,         // AccessDeniedException will throw.         http.authorizeRequests().and().exceptionHandling().accessDeniedPage("/403");           // Config for Login Form         http.authorizeRequests().and().formLogin()//                 // Submit URL of login page.                 .loginProcessingUrl("/j\_spring\_security\_check") // Submit URL                 .loginPage("/login")//                 .defaultSuccessUrl("/accountInfo")//                 .failureUrl("/login?error=true")//                 .usernameParameter("userName")//                 .passwordParameter("password")                 // Config for Logout Page                 // (Go to home page).                 .and().logout().logoutUrl("/logout").logoutSuccessUrl("/");       }  } |

## 7- Entity Classes



**Account.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67 | package org.o7planning.springmvcshoppingcart.entity;    import java.io.Serializable;    import javax.persistence.Column;  import javax.persistence.Entity;  import javax.persistence.Id;  import javax.persistence.Table;    @Entity  @Table(name = "Accounts")  public class Account implements Serializable {        private static final long serialVersionUID = -2054386655979281969L;          public static final String ROLE\_MANAGER = "MANAGER";      public static final String ROLE\_EMPLOYEE = "EMPLOYEE";        private String userName;      private String password;      private boolean active;      private String userRole;        @Id      @Column(name = "User\_Name", length = 20, nullable = false)      public String getUserName() {          return userName;      }        public void setUserName(String userName) {          this.userName = userName;      }        @Column(name = "Password", length = 20, nullable = false)      public String getPassword() {          return password;      }        public void setPassword(String password) {          this.password = password;      }        @Column(name = "Active", length = 1, nullable = false)      public boolean isActive() {          return active;      }        public void setActive(boolean active) {          this.active = active;      }        @Column(name = "User\_Role", length = 20, nullable = false)      public String getUserRole() {          return userRole;      }        public void setUserRole(String userRole) {          this.userRole = userRole;      }        @Override      public String toString()  {          return "["+ this.userName+","+ this.password+","+ this.userRole+"]";      }    } |

**Order.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102 | package org.o7planning.springmvcshoppingcart.entity;    import java.io.Serializable;  import java.util.Date;    import javax.persistence.Column;  import javax.persistence.Entity;  import javax.persistence.Id;  import javax.persistence.Table;  import javax.persistence.UniqueConstraint;    @Entity  @Table(name = "Orders", //  uniqueConstraints = { @UniqueConstraint(columnNames = "Order\_Num") })  public class Order implements Serializable {        private static final long serialVersionUID = -2576670215015463100L;        private String id;      private Date orderDate;      private int orderNum;      private double amount;        private String customerName;      private String customerAddress;      private String customerEmail;      private String customerPhone;        @Id      @Column(name = "ID", length = 50)      public String getId() {          return id;      }        public void setId(String id) {          this.id = id;      }        @Column(name = "Order\_Date", nullable = false)      public Date getOrderDate() {          return orderDate;      }        public void setOrderDate(Date orderDate) {          this.orderDate = orderDate;      }        @Column(name = "Order\_Num", nullable = false)      public int getOrderNum() {          return orderNum;      }        public void setOrderNum(int orderNum) {          this.orderNum = orderNum;      }        @Column(name = "Amount", nullable = false)      public double getAmount() {          return amount;      }        public void setAmount(double amount) {          this.amount = amount;      }        @Column(name = "Customer\_Name", length = 255, nullable = false)      public String getCustomerName() {          return customerName;      }        public void setCustomerName(String customerName) {          this.customerName = customerName;      }        @Column(name = "Customer\_Address", length = 255, nullable = false)      public String getCustomerAddress() {          return customerAddress;      }        public void setCustomerAddress(String customerAddress) {          this.customerAddress = customerAddress;      }        @Column(name = "Customer\_Email", length = 128, nullable = false)      public String getCustomerEmail() {          return customerEmail;      }        public void setCustomerEmail(String customerEmail) {          this.customerEmail = customerEmail;      }        @Column(name = "Customer\_Phone", length = 128, nullable = false)      public String getCustomerPhone() {          return customerPhone;      }        public void setCustomerPhone(String customerPhone) {          this.customerPhone = customerPhone;      }    } |

**OrderDetail.java**

?

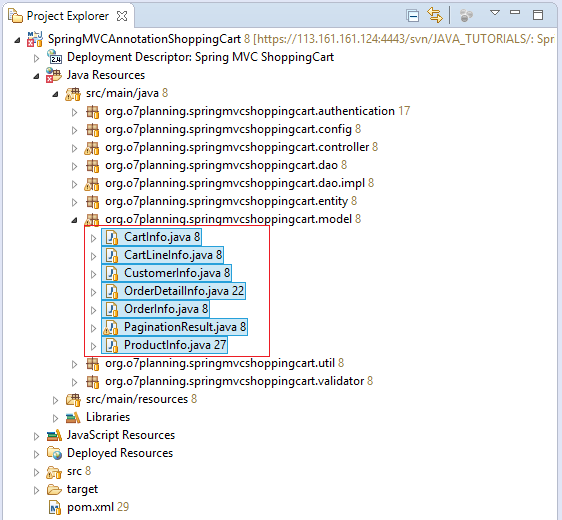
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87 | package org.o7planning.springmvcshoppingcart.entity;    import java.io.Serializable;    import javax.persistence.Column;  import javax.persistence.Entity;  import javax.persistence.FetchType;  import javax.persistence.ForeignKey;  import javax.persistence.Id;  import javax.persistence.JoinColumn;  import javax.persistence.ManyToOne;  import javax.persistence.Table;    @Entity  @Table(name = "Order\_Details")  public class OrderDetail implements Serializable {        private static final long serialVersionUID = 7550745928843183535L;        private String id;      private Order order;        private Product product;      private int quanity;      private double price;      private double amount;        @Id      @Column(name = "ID", length = 50, nullable = false)      public String getId() {          return id;      }        public void setId(String id) {          this.id = id;      }        @ManyToOne(fetch = FetchType.LAZY)      @JoinColumn(name = "ORDER\_ID", nullable = false, //      foreignKey = @ForeignKey(name = "ORDER\_DETAIL\_ORD\_FK") )      public Order getOrder() {          return order;      }        public void setOrder(Order order) {          this.order = order;      }        @ManyToOne(fetch = FetchType.LAZY)      @JoinColumn(name = "PRODUCT\_ID", nullable = false, //      foreignKey = @ForeignKey(name = "ORDER\_DETAIL\_PROD\_FK") )      public Product getProduct() {          return product;      }        public void setProduct(Product product) {          this.product = product;      }        @Column(name = "Quanity", nullable = false)      public int getQuanity() {          return quanity;      }        public void setQuanity(int quanity) {          this.quanity = quanity;      }        @Column(name = "Price", nullable = false)      public double getPrice() {          return price;      }        public void setPrice(double price) {          this.price = price;      }        @Column(name = "Amount", nullable = false)      public double getAmount() {          return amount;      }        public void setAmount(double amount) {          this.amount = amount;      }    } |

**Product.java**

?

|  |
| --- |
| package org.o7planning.springmvcshoppingcart.entity;    import java.io.Serializable;  import java.util.Date;    import javax.persistence.Column;  import javax.persistence.Entity;  import javax.persistence.Id;  import javax.persistence.Lob;  import javax.persistence.Table;  import javax.persistence.Temporal;  import javax.persistence.TemporalType;    @Entity  @Table(name = "Products")  public class Product implements Serializable {        private static final long serialVersionUID = -1000119078147252957L;        private String code;      private String name;      private double price;      private byte[] image;        // For sort.      private Date createDate;        public Product() {      }        @Id      @Column(name = "Code", length = 20, nullable = false)      public String getCode() {          return code;      }        public void setCode(String code) {          this.code = code;      }        @Column(name = "Name", length = 255, nullable = false)      public String getName() {          return name;      }        public void setName(String name) {          this.name = name;      }        @Column(name = "Price", nullable = false)      public double getPrice() {          return price;      }        public void setPrice(double price) {          this.price = price;      }        @Temporal(TemporalType.TIMESTAMP)      @Column(name = "Create\_Date", nullable = false)      public Date getCreateDate() {          return createDate;      }        public void setCreateDate(Date createDate) {          this.createDate = createDate;      }        @Lob      @Column(name = "Image", length = Integer.MAX\_VALUE, nullable = true)      public byte[] getImage() {          return image;      }        public void setImage(byte[] image) {          this.image = image;      }    } |

## 8- Model Classes



**CartInfo.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121 | package org.o7planning.springmvcshoppingcart.model;    import java.util.ArrayList;  import java.util.List;    public class CartInfo {        private int orderNum;        private CustomerInfo customerInfo;        private final List<CartLineInfo> cartLines = new ArrayList<CartLineInfo>();        public CartInfo() {        }        public int getOrderNum() {          return orderNum;      }        public void setOrderNum(int orderNum) {          this.orderNum = orderNum;      }        public CustomerInfo getCustomerInfo() {          return customerInfo;      }        public void setCustomerInfo(CustomerInfo customerInfo) {          this.customerInfo = customerInfo;      }        public List<CartLineInfo> getCartLines() {          return this.cartLines;      }        private CartLineInfo findLineByCode(String code) {          for (CartLineInfo line : this.cartLines) {              if (line.getProductInfo().getCode().equals(code)) {                  return line;              }          }          return null;      }        public void addProduct(ProductInfo productInfo, int quantity) {          CartLineInfo line = this.findLineByCode(productInfo.getCode());            if (line == null) {              line = new CartLineInfo();              line.setQuantity(0);              line.setProductInfo(productInfo);              this.cartLines.add(line);          }          int newQuantity = line.getQuantity() + quantity;          if (newQuantity <= 0) {              this.cartLines.remove(line);          } else {              line.setQuantity(newQuantity);          }      }        public void validate() {        }        public void updateProduct(String code, int quantity) {          CartLineInfo line = this.findLineByCode(code);            if (line != null) {              if (quantity <= 0) {                  this.cartLines.remove(line);              } else {                  line.setQuantity(quantity);              }          }      }        public void removeProduct(ProductInfo productInfo) {          CartLineInfo line = this.findLineByCode(productInfo.getCode());          if (line != null) {              this.cartLines.remove(line);          }      }        public boolean isEmpty() {          return this.cartLines.isEmpty();      }        public boolean isValidCustomer() {          return this.customerInfo != null && this.customerInfo.isValid();      }        public int getQuantityTotal() {          int quantity = 0;          for (CartLineInfo line : this.cartLines) {              quantity += line.getQuantity();          }          return quantity;      }        public double getAmountTotal() {          double total = 0;          for (CartLineInfo line : this.cartLines) {              total += line.getAmount();          }          return total;      }        public void updateQuantity(CartInfo cartForm) {          if (cartForm != null) {              List<CartLineInfo> lines = cartForm.getCartLines();              for (CartLineInfo line : lines) {                  this.updateProduct(line.getProductInfo().getCode(), line.getQuantity());              }          }        }    } |

**CartLineInfo.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32 | package org.o7planning.springmvcshoppingcart.model;    public class CartLineInfo {        private ProductInfo productInfo;      private int quantity;        public CartLineInfo() {          this.quantity = 0;      }        public ProductInfo getProductInfo() {          return productInfo;      }        public void setProductInfo(ProductInfo productInfo) {          this.productInfo = productInfo;      }        public int getQuantity() {          return quantity;      }        public void setQuantity(int quantity) {          this.quantity = quantity;      }        public double getAmount() {          return this.productInfo.getPrice() \* this.quantity;      }    } |

**CustomerInfo.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53 | package org.o7planning.springmvcshoppingcart.model;    public class CustomerInfo {        private String name;      private String address;      private String email;      private String phone;        private boolean valid;          public String getName() {          return name;      }        public void setName(String name) {          this.name = name;      }        public String getEmail() {          return email;      }        public void setEmail(String email) {          this.email = email;      }        public String getAddress() {          return address;      }        public void setAddress(String address) {          this.address = address;      }        public String getPhone() {          return phone;      }        public void setPhone(String phone) {          this.phone = phone;      }        public boolean isValid() {          return valid;      }        public void setValid(boolean valid) {          this.valid = valid;      }    } |

**OrderDetailInfo.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78 | package org.o7planning.springmvcshoppingcart.model;    public class OrderDetailInfo {      private String id;        private String productCode;      private String productName;        private int quanity;      private double price;      private double amount;        public OrderDetailInfo() {        }        // Using for Hibernate Query.      // Sử dụng cho Hibernate Query.      public OrderDetailInfo(String id, String productCode, //              String productName, int quanity, double price, double amount) {          this.id = id;          this.productCode = productCode;          this.productName = productName;          this.quanity = quanity;          this.price = price;          this.amount = amount;      }        public String getId() {          return id;      }        public void setId(String id) {          this.id = id;      }        public String getProductCode() {          return productCode;      }        public void setProductCode(String productCode) {          this.productCode = productCode;      }        public String getProductName() {          return productName;      }        public void setProductName(String productName) {          this.productName = productName;      }        public int getQuanity() {          return quanity;      }        public void setQuanity(int quanity) {          this.quanity = quanity;      }        public double getPrice() {          return price;      }        public void setPrice(double price) {          this.price = price;      }        public double getAmount() {          return amount;      }        public void setAmount(double amount) {          this.amount = amount;      }  } |

**OrderInfo.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112 | package org.o7planning.springmvcshoppingcart.model;    import java.util.Date;  import java.util.List;    public class OrderInfo {        private String id;      private Date orderDate;      private int orderNum;      private double amount;        private String customerName;      private String customerAddress;      private String customerEmail;      private String customerPhone;        private List<OrderDetailInfo> details;        public OrderInfo() {        }        // Using for Hibernate Query.      // Sử dụng cho Hibernate Query.      public OrderInfo(String id, Date orderDate, int orderNum, //              double amount, String customerName, String customerAddress, //              String customerEmail, String customerPhone) {          this.id = id;          this.orderDate = orderDate;          this.orderNum = orderNum;          this.amount = amount;            this.customerName = customerName;          this.customerAddress = customerAddress;          this.customerEmail = customerEmail;          this.customerPhone = customerPhone;      }        public String getId() {          return id;      }        public void setId(String id) {          this.id = id;      }        public Date getOrderDate() {          return orderDate;      }        public void setOrderDate(Date orderDate) {          this.orderDate = orderDate;      }        public int getOrderNum() {          return orderNum;      }        public void setOrderNum(int orderNum) {          this.orderNum = orderNum;      }        public double getAmount() {          return amount;      }        public void setAmount(double amount) {          this.amount = amount;      }        public String getCustomerName() {          return customerName;      }        public void setCustomerName(String customerName) {          this.customerName = customerName;      }        public String getCustomerAddress() {          return customerAddress;      }        public void setCustomerAddress(String customerAddress) {          this.customerAddress = customerAddress;      }        public String getCustomerEmail() {          return customerEmail;      }        public void setCustomerEmail(String customerEmail) {          this.customerEmail = customerEmail;      }        public String getCustomerPhone() {          return customerPhone;      }        public void setCustomerPhone(String customerPhone) {          this.customerPhone = customerPhone;      }        public List<OrderDetailInfo> getDetails() {          return details;      }        public void setDetails(List<OrderDetailInfo> details) {          this.details = details;      }    } |

**PaginationResult.java**

?

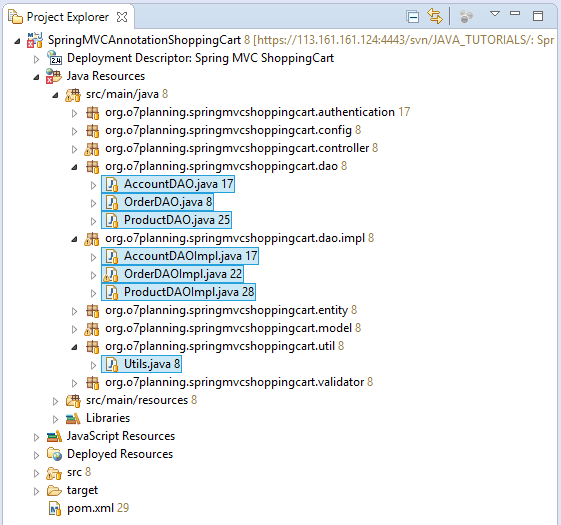
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121 | package org.o7planning.springmvcshoppingcart.model;    import java.util.ArrayList;  import java.util.List;    import org.hibernate.Query;  import org.hibernate.ScrollMode;  import org.hibernate.ScrollableResults;    public class PaginationResult<E> {       private int totalRecords;     private int currentPage;     private List<E> list;     private int maxResult;     private int totalPages;       private int maxNavigationPage;       private List<Integer> navigationPages;       // @page: 1, 2, ..     public PaginationResult(Query query, int page, int maxResult, int maxNavigationPage) {         final int pageIndex = page - 1 < 0 ? 0 : page - 1;           int fromRecordIndex = pageIndex \* maxResult;         int maxRecordIndex = fromRecordIndex + maxResult;           ScrollableResults resultScroll = query.scroll(ScrollMode.SCROLL\_INSENSITIVE);           List results = new ArrayList();           boolean hasResult = resultScroll.first();           if (hasResult) {             // Scroll to position:             hasResult = resultScroll.scroll(fromRecordIndex);               if (hasResult) {                 do {                     E record = (E) resultScroll.get(0);                     results.add(record);                 } while (resultScroll.next()//                         && resultScroll.getRowNumber() >= fromRecordIndex                         && resultScroll.getRowNumber() < maxRecordIndex);               }             // Go to last record.             resultScroll.last();         }         // Total records.         this.totalRecords = resultScroll.getRowNumber() + 1;         this.currentPage = pageIndex + 1;         this.list = results;         this.maxResult = maxResult;           this.totalPages = (this.totalRecords / this.maxResult) + 1;         this.maxNavigationPage = maxNavigationPage;           if (maxNavigationPage < totalPages) {             this.maxNavigationPage = maxNavigationPage;         }           this.calcNavigationPages();     }       private void calcNavigationPages() {           this.navigationPages = new ArrayList<Integer>();           int current = this.currentPage > this.totalPages ? this.totalPages : this.currentPage;           int begin = current - this.maxNavigationPage / 2;         int end = current + this.maxNavigationPage / 2;           // First page         navigationPages.add(1);         if (begin > 2) {             // For '...'             navigationPages.add(-1);         }           for (int i = begin; i < end; i++) {             if (i > 1 && i < this.totalPages) {                 navigationPages.add(i);             }         }           if (end < this.totalPages - 2) {             // For '...'             navigationPages.add(-1);         }         // Last page.         navigationPages.add(this.totalPages);     }       public int getTotalPages() {         return totalPages;     }       public int getTotalRecords() {         return totalRecords;     }       public int getCurrentPage() {         return currentPage;     }       public List<E> getList() {         return list;     }       public int getMaxResult() {         return maxResult;     }       public List<Integer> getNavigationPages() {         return navigationPages;     }    } |

**ProductInfo.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73 | package org.o7planning.springmvcshoppingcart.model;    import org.o7planning.springmvcshoppingcart.entity.Product;  import org.springframework.web.multipart.commons.CommonsMultipartFile;    public class ProductInfo {      private String code;      private String name;      private double price;        private boolean newProduct=false;        // Upload file.      private CommonsMultipartFile fileData;        public ProductInfo() {      }        public ProductInfo(Product product) {          this.code = product.getCode();          this.name = product.getName();          this.price = product.getPrice();      }        // Không thay đổi Constructor này,      // nó được sử dụng trong Hibernate query.      public ProductInfo(String code, String name, double price) {          this.code = code;          this.name = name;          this.price = price;      }        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 double getPrice() {          return price;      }        public void setPrice(double price) {          this.price = price;      }        public CommonsMultipartFile getFileData() {          return fileData;      }        public void setFileData(CommonsMultipartFile fileData) {          this.fileData = fileData;      }        public boolean isNewProduct() {          return newProduct;      }        public void setNewProduct(boolean newProduct) {          this.newProduct = newProduct;      }    } |

## 9- Utils & DAO Classes



**AccountDAO.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | package org.o7planning.springmvcshoppingcart.dao;    import org.o7planning.springmvcshoppingcart.entity.Account;    public interface AccountDAO {          public Account findAccount(String userName );    } |

**OrderDAO.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | package org.o7planning.springmvcshoppingcart.dao;    import java.util.List;    import org.o7planning.springmvcshoppingcart.model.CartInfo;  import org.o7planning.springmvcshoppingcart.model.OrderDetailInfo;  import org.o7planning.springmvcshoppingcart.model.OrderInfo;  import org.o7planning.springmvcshoppingcart.model.PaginationResult;    public interface OrderDAO {        public void saveOrder(CartInfo cartInfo);        public PaginationResult<OrderInfo> listOrderInfo(int page,              int maxResult, int maxNavigationPage);        public OrderInfo getOrderInfo(String orderId);        public List<OrderDetailInfo> listOrderDetailInfos(String orderId);    } |

**ProductDAO.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24 | package org.o7planning.springmvcshoppingcart.dao;    import org.o7planning.springmvcshoppingcart.entity.Product;  import org.o7planning.springmvcshoppingcart.model.PaginationResult;  import org.o7planning.springmvcshoppingcart.model.ProductInfo;    public interface ProductDAO {            public Product findProduct(String code);        public ProductInfo findProductInfo(String code) ;          public PaginationResult<ProductInfo> queryProducts(int page,                         int maxResult, int maxNavigationPage  );        public PaginationResult<ProductInfo> queryProducts(int page, int maxResult,                         int maxNavigationPage, String likeName);        public void save(ProductInfo productInfo);    } |

**AccountDAOImpl.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27 | package org.o7planning.springmvcshoppingcart.dao.impl;    import org.hibernate.Criteria;  import org.hibernate.Session;  import org.hibernate.SessionFactory;  import org.hibernate.criterion.Restrictions;  import org.o7planning.springmvcshoppingcart.dao.AccountDAO;  import org.o7planning.springmvcshoppingcart.entity.Account;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.transaction.annotation.Transactional;    // Transactional for Hibernate  @Transactional  public class AccountDAOImpl implements AccountDAO {        @Autowired      private SessionFactory sessionFactory;        @Override      public Account findAccount(String userName ) {          Session session = sessionFactory.getCurrentSession();          Criteria crit = session.createCriteria(Account.class);          crit.add(Restrictions.eq("userName", userName));          return (Account) crit.uniqueResult();      }    } |

**OrderDAOImpl.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137 | package org.o7planning.springmvcshoppingcart.dao.impl;    import java.util.Date;  import java.util.List;  import java.util.UUID;    import org.hibernate.Criteria;  import org.hibernate.Query;  import org.hibernate.Session;  import org.hibernate.SessionFactory;  import org.hibernate.criterion.Restrictions;  import org.o7planning.springmvcshoppingcart.dao.OrderDAO;  import org.o7planning.springmvcshoppingcart.dao.ProductDAO;  import org.o7planning.springmvcshoppingcart.entity.Order;  import org.o7planning.springmvcshoppingcart.entity.OrderDetail;  import org.o7planning.springmvcshoppingcart.entity.Product;  import org.o7planning.springmvcshoppingcart.model.CartInfo;  import org.o7planning.springmvcshoppingcart.model.CartLineInfo;  import org.o7planning.springmvcshoppingcart.model.CustomerInfo;  import org.o7planning.springmvcshoppingcart.model.OrderDetailInfo;  import org.o7planning.springmvcshoppingcart.model.OrderInfo;  import org.o7planning.springmvcshoppingcart.model.PaginationResult;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.transaction.annotation.Transactional;    //Transactional for Hibernate  @Transactional  public class OrderDAOImpl implements OrderDAO {        @Autowired      private SessionFactory sessionFactory;        @Autowired      private ProductDAO productDAO;        private int getMaxOrderNum() {          String sql = "Select max(o.orderNum) from " + Order.class.getName() + " o ";          Session session = sessionFactory.getCurrentSession();          Query query = session.createQuery(sql);          Integer value = (Integer) query.uniqueResult();          if (value == null) {              return 0;          }          return value;      }        @Override      public void saveOrder(CartInfo cartInfo) {          Session session = sessionFactory.getCurrentSession();            int orderNum = this.getMaxOrderNum() + 1;          Order order = new Order();            order.setId(UUID.randomUUID().toString());          order.setOrderNum(orderNum);          order.setOrderDate(new Date());          order.setAmount(cartInfo.getAmountTotal());            CustomerInfo customerInfo = cartInfo.getCustomerInfo();          order.setCustomerName(customerInfo.getName());          order.setCustomerEmail(customerInfo.getEmail());          order.setCustomerPhone(customerInfo.getPhone());          order.setCustomerAddress(customerInfo.getAddress());            session.persist(order);            List<CartLineInfo> lines = cartInfo.getCartLines();            for (CartLineInfo line : lines) {              OrderDetail detail = new OrderDetail();              detail.setId(UUID.randomUUID().toString());              detail.setOrder(order);              detail.setAmount(line.getAmount());              detail.setPrice(line.getProductInfo().getPrice());              detail.setQuanity(line.getQuantity());                String code = line.getProductInfo().getCode();              Product product = this.productDAO.findProduct(code);              detail.setProduct(product);                session.persist(detail);          }            // Set OrderNum for report.          // Set OrderNum để thông báo cho người dùng.          cartInfo.setOrderNum(orderNum);      }        // @page = 1, 2, ...      @Override      public PaginationResult<OrderInfo> listOrderInfo(int page, int maxResult, int maxNavigationPage) {          String sql = "Select new " + OrderInfo.class.getName()//                  + "(ord.id, ord.orderDate, ord.orderNum, ord.amount, "                  + " ord.customerName, ord.customerAddress, ord.customerEmail, ord.customerPhone) " + " from "                  + Order.class.getName() + " ord "//                  + " order by ord.orderNum desc";          Session session = this.sessionFactory.getCurrentSession();            Query query = session.createQuery(sql);            return new PaginationResult<OrderInfo>(query, page, maxResult, maxNavigationPage);      }        public Order findOrder(String orderId) {          Session session = sessionFactory.getCurrentSession();          Criteria crit = session.createCriteria(Order.class);          crit.add(Restrictions.eq("id", orderId));          return (Order) crit.uniqueResult();      }        @Override      public OrderInfo getOrderInfo(String orderId) {          Order order = this.findOrder(orderId);          if (order == null) {              return null;          }          return new OrderInfo(order.getId(), order.getOrderDate(), //                  order.getOrderNum(), order.getAmount(), order.getCustomerName(), //                  order.getCustomerAddress(), order.getCustomerEmail(), order.getCustomerPhone());      }        @Override      public List<OrderDetailInfo> listOrderDetailInfos(String orderId) {          String sql = "Select new " + OrderDetailInfo.class.getName() //                  + "(d.id, d.product.code, d.product.name , d.quanity,d.price,d.amount) "//                  + " from " + OrderDetail.class.getName() + " d "//                  + " where d.order.id = :orderId ";            Session session = this.sessionFactory.getCurrentSession();            Query query = session.createQuery(sql);          query.setParameter("orderId", orderId);            return query.list();      }    } |

**ProductDAOImpl.java**

?

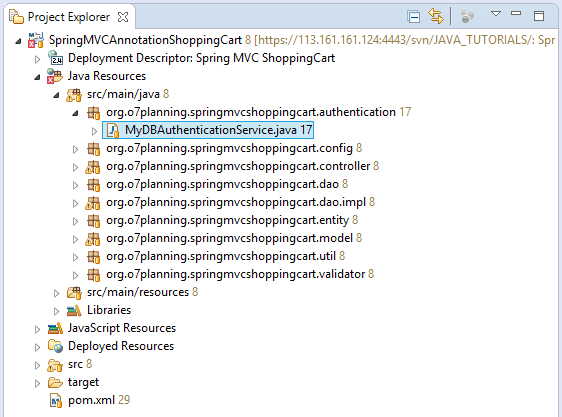
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 | package org.o7planning.springmvcshoppingcart.dao.impl;    import java.util.Date;    import org.hibernate.Criteria;  import org.hibernate.Query;  import org.hibernate.Session;  import org.hibernate.SessionFactory;  import org.hibernate.criterion.Restrictions;  import org.o7planning.springmvcshoppingcart.dao.ProductDAO;  import org.o7planning.springmvcshoppingcart.entity.Product;  import org.o7planning.springmvcshoppingcart.model.PaginationResult;  import org.o7planning.springmvcshoppingcart.model.ProductInfo;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.transaction.annotation.Transactional;    // Transactional for Hibernate  @Transactional  public class ProductDAOImpl implements ProductDAO {        @Autowired      private SessionFactory sessionFactory;        @Override      public Product findProduct(String code) {          Session session = sessionFactory.getCurrentSession();          Criteria crit = session.createCriteria(Product.class);          crit.add(Restrictions.eq("code", code));          return (Product) crit.uniqueResult();      }        @Override      public ProductInfo findProductInfo(String code) {          Product product = this.findProduct(code);          if (product == null) {              return null;          }          return new ProductInfo(product.getCode(), product.getName(), product.getPrice());      }        @Override      public void save(ProductInfo productInfo) {          String code = productInfo.getCode();            Product product = null;            boolean isNew = false;          if (code != null) {              product = this.findProduct(code);          }          if (product == null) {              isNew = true;              product = new Product();              product.setCreateDate(new Date());          }          product.setCode(code);          product.setName(productInfo.getName());          product.setPrice(productInfo.getPrice());            if (productInfo.getFileData() != null) {              byte[] image = productInfo.getFileData().getBytes();              if (image != null && image.length > 0) {                  product.setImage(image);              }          }          if (isNew) {              this.sessionFactory.getCurrentSession().persist(product);          }          // If error in DB, Exceptions will be thrown out immediately          // Nếu có lỗi tại DB, ngoại lệ sẽ ném ra ngay lập tức          this.sessionFactory.getCurrentSession().flush();      }        @Override      public PaginationResult<ProductInfo> queryProducts(int page, int maxResult, int maxNavigationPage,              String likeName) {          String sql = "Select new " + ProductInfo.class.getName() //                  + "(p.code, p.name, p.price) " + " from "//                  + Product.class.getName() + " p ";          if (likeName != null && likeName.length() > 0) {              sql += " Where lower(p.name) like :likeName ";          }          sql += " order by p.createDate desc ";          //          Session session = sessionFactory.getCurrentSession();            Query query = session.createQuery(sql);          if (likeName != null && likeName.length() > 0) {              query.setParameter("likeName", "%" + likeName.toLowerCase() + "%");          }          return new PaginationResult<ProductInfo>(query, page, maxResult, maxNavigationPage);      }        @Override      public PaginationResult<ProductInfo> queryProducts(int page, int maxResult, int maxNavigationPage) {          return queryProducts(page, maxResult, maxNavigationPage, null);      }    } |

**Utils.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38 | package org.o7planning.springmvcshoppingcart.util;    import javax.servlet.http.HttpServletRequest;    import org.o7planning.springmvcshoppingcart.model.CartInfo;    public class Utils {        // Products in Cart, stored in Session.      public static CartInfo getCartInSession(HttpServletRequest request) {            // Get Cart from Session.          CartInfo cartInfo = (CartInfo) request.getSession().getAttribute("myCart");            // If null, create it.          if (cartInfo == null) {              cartInfo = new CartInfo();                // And store to Session.              request.getSession().setAttribute("myCart", cartInfo);          }            return cartInfo;      }        public static void removeCartInSession(HttpServletRequest request) {          request.getSession().removeAttribute("myCart");      }        public static void storeLastOrderedCartInSession(HttpServletRequest request, CartInfo cartInfo) {          request.getSession().setAttribute("lastOrderedCart", cartInfo);      }        public static CartInfo getLastOrderedCartInSession(HttpServletRequest request) {          return (CartInfo) request.getSession().getAttribute("lastOrderedCart");      }    } |

## 10- Authentication classes



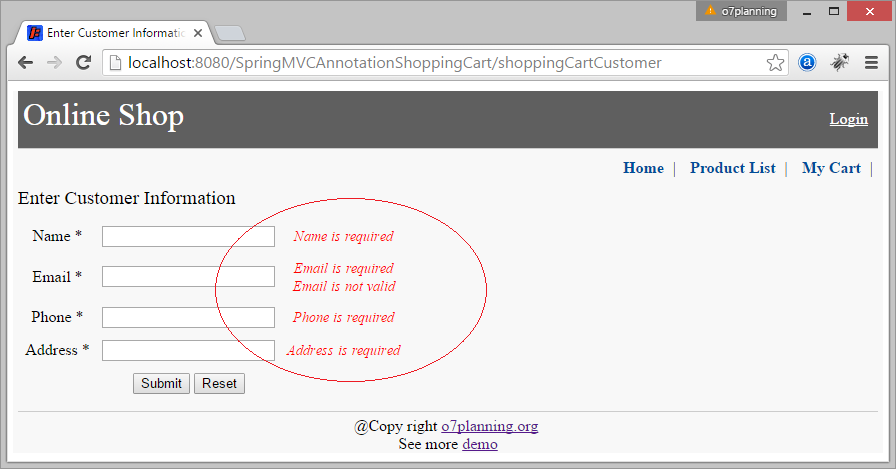
**MyDBAuthenticationService.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55 | package org.o7planning.springmvcshoppingcart.authentication;    import java.util.ArrayList;  import java.util.List;    import org.o7planning.springmvcshoppingcart.dao.AccountDAO;  import org.o7planning.springmvcshoppingcart.entity.Account;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.security.core.GrantedAuthority;  import org.springframework.security.core.authority.SimpleGrantedAuthority;  import org.springframework.security.core.userdetails.User;  import org.springframework.security.core.userdetails.UserDetails;  import org.springframework.security.core.userdetails.UserDetailsService;  import org.springframework.security.core.userdetails.UsernameNotFoundException;  import org.springframework.stereotype.Service;    @Service  public class MyDBAuthenticationService implements UserDetailsService {        @Autowired      private AccountDAO accountDAO;        @Override      public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {          Account account = accountDAO.findAccount(username);          System.out.println("Account= " + account);            if (account == null) {              throw new UsernameNotFoundException("User " //                      + username + " was not found in the database");          }            // EMPLOYEE,MANAGER,..          String role = account.getUserRole();            List<GrantedAuthority> grantList = new ArrayList<GrantedAuthority>();            // ROLE\_EMPLOYEE, ROLE\_MANAGER          GrantedAuthority authority = new SimpleGrantedAuthority("ROLE\_" + role);            grantList.add(authority);            boolean enabled = account.isActive();          boolean accountNonExpired = true;          boolean credentialsNonExpired = true;          boolean accountNonLocked = true;            UserDetails userDetails = (UserDetails) new User(account.getUserName(), //                  account.getPassword(), enabled, accountNonExpired, //                  credentialsNonExpired, accountNonLocked, grantList);            return userDetails;      }    } |

## 11- Spring MVC Validator

**CustomerValidator**& **ProductInfoValidator**are classes to check user data entered into a **form**. In case the user entered wrong, there will be notice as shown below:



**CustomerInfoValidator.java**

?

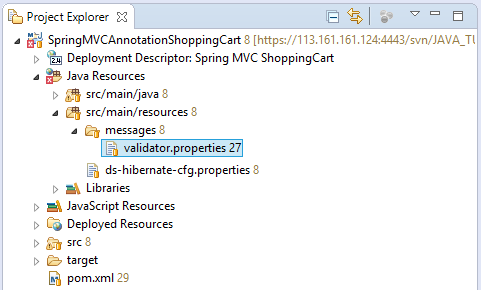
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38 | package org.o7planning.springmvcshoppingcart.validator;    import org.apache.commons.validator.routines.EmailValidator;  import org.o7planning.springmvcshoppingcart.model.CustomerInfo;  import org.springframework.stereotype.Component;  import org.springframework.validation.Errors;  import org.springframework.validation.ValidationUtils;  import org.springframework.validation.Validator;    // @Component: As a Bean.  @Component  public class CustomerInfoValidator implements Validator {        private EmailValidator emailValidator = EmailValidator.getInstance();        // This Validator support CustomerInfo class.      @Override      public boolean supports(Class<?> clazz) {          return clazz == CustomerInfo.class;      }        @Override      public void validate(Object target, Errors errors) {          CustomerInfo custInfo = (CustomerInfo) target;            // Check the fields of CustomerInfo class.          ValidationUtils.rejectIfEmptyOrWhitespace(errors, "name", "NotEmpty.customerForm.name");          ValidationUtils.rejectIfEmptyOrWhitespace(errors, "email", "NotEmpty.customerForm.email");          ValidationUtils.rejectIfEmptyOrWhitespace(errors, "address", "NotEmpty.customerForm.address");          ValidationUtils.rejectIfEmptyOrWhitespace(errors, "phone", "NotEmpty.customerForm.phone");            if (!emailValidator.isValid(custInfo.getEmail())) {              errors.rejectValue("email", "Pattern.customerForm.email");          }      }    } |

**ProductInfoValidator.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49 | package org.o7planning.springmvcshoppingcart.validator;    import org.o7planning.springmvcshoppingcart.dao.ProductDAO;  import org.o7planning.springmvcshoppingcart.entity.Product;  import org.o7planning.springmvcshoppingcart.model.ProductInfo;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.stereotype.Component;  import org.springframework.validation.Errors;  import org.springframework.validation.ValidationUtils;  import org.springframework.validation.Validator;    // @Component: As a Bean.  @Component  public class ProductInfoValidator implements Validator {        @Autowired      private ProductDAO productDAO;        // This Validator support ProductInfo class.      @Override      public boolean supports(Class<?> clazz) {          return clazz == ProductInfo.class;      }        @Override      public void validate(Object target, Errors errors) {          ProductInfo productInfo = (ProductInfo) target;            // Check the fields of ProductInfo class.          ValidationUtils.rejectIfEmptyOrWhitespace(errors, "code", "NotEmpty.productForm.code");          ValidationUtils.rejectIfEmptyOrWhitespace(errors, "name", "NotEmpty.productForm.name");          ValidationUtils.rejectIfEmptyOrWhitespace(errors, "price", "NotEmpty.productForm.price");            String code = productInfo.getCode();          if (code != null && code.length() > 0) {              if (code.matches("\\s+")) {                  errors.rejectValue("code", "Pattern.productForm.code");              } else if(productInfo.isNewProduct()) {                  Product product = productDAO.findProduct(code);                  if (product != null) {                      errors.rejectValue("code", "Duplicate.productForm.code");                  }              }          }      }    } |

The error messages are configured in the **messages/validator.properties** file:



**messages/validator.properties**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | NotEmpty.customerForm.name=Name is required  NotEmpty.customerForm.email=Email is required  NotEmpty.customerForm.address=Address is required  NotEmpty.customerForm.phone=Phone is required    Pattern.customerForm.email=Email is not valid      NotEmpty.productForm.name=Product name is required  NotEmpty.productForm.code=Product code is required  Pattern.productForm.code=Product code is not valid  Duplicate.productForm.code=Duplicate products      NotFound.loginForm.account=Account not found  Disabled.loginForm.account=Account is disabled |

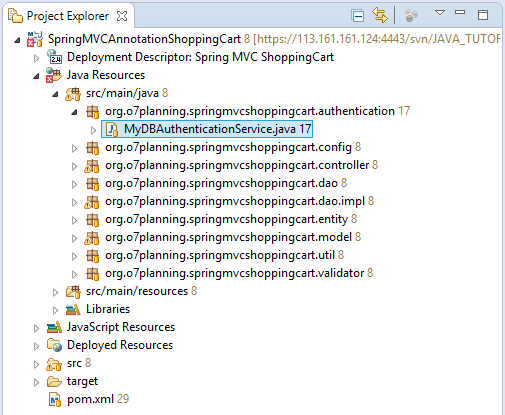
The content of validator.properties loaded by **ApplicationContextConfig**:

**\*\* ApplicationContextConfig.java \*\***

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | // Load property in message/validator.properties    @Bean  public ResourceBundleMessageSource messageSource() {     ResourceBundleMessageSource rb = new ResourceBundleMessageSource();     // Load property in message/validator.properties     rb.setBasenames(new String[] { "messages/validator"});     return rb;  } |

## 12- Classes for Login

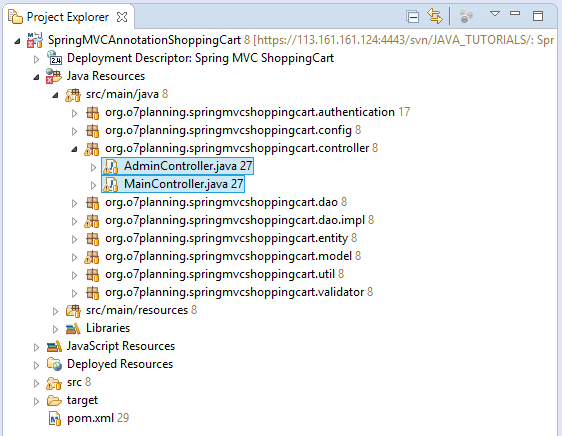


**MyDBAuthenticationService.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55 | package org.o7planning.springmvcshoppingcart.authentication;    import java.util.ArrayList;  import java.util.List;    import org.o7planning.springmvcshoppingcart.dao.AccountDAO;  import org.o7planning.springmvcshoppingcart.entity.Account;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.security.core.GrantedAuthority;  import org.springframework.security.core.authority.SimpleGrantedAuthority;  import org.springframework.security.core.userdetails.User;  import org.springframework.security.core.userdetails.UserDetails;  import org.springframework.security.core.userdetails.UserDetailsService;  import org.springframework.security.core.userdetails.UsernameNotFoundException;  import org.springframework.stereotype.Service;    @Service  public class MyDBAuthenticationService implements UserDetailsService {        @Autowired      private AccountDAO accountDAO;        @Override      public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {          Account account = accountDAO.findAccount(username);          System.out.println("Account= " + account);            if (account == null) {              throw new UsernameNotFoundException("User " //                      + username + " was not found in the database");          }            // EMPLOYEE,MANAGER,..          String role = account.getUserRole();            List<GrantedAuthority> grantList = new ArrayList<GrantedAuthority>();            // ROLE\_EMPLOYEE, ROLE\_MANAGER          GrantedAuthority authority = new SimpleGrantedAuthority("ROLE\_" + role);            grantList.add(authority);            boolean enabled = account.isActive();          boolean accountNonExpired = true;          boolean credentialsNonExpired = true;          boolean accountNonLocked = true;            UserDetails userDetails = (UserDetails) new User(account.getUserName(), //                  account.getPassword(), enabled, accountNonExpired, //                  credentialsNonExpired, accountNonLocked, grantList);            return userDetails;      }    } |

## 13- Spring Controller



**MainController.java**

?

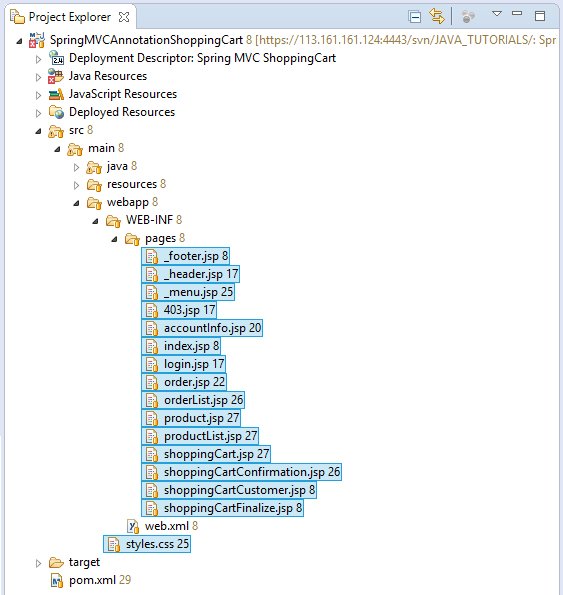
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149  150  151  152  153  154  155  156  157  158  159  160  161  162  163  164  165  166  167  168  169  170  171  172  173  174  175  176  177  178  179  180  181  182  183  184  185  186  187  188  189  190  191  192  193  194  195  196  197  198  199  200  201  202  203  204  205  206  207  208  209  210  211  212  213  214  215  216  217  218  219  220  221  222  223  224  225  226  227  228  229  230  231  232  233  234  235  236  237  238  239  240  241  242  243  244  245  246  247  248  249  250  251  252  253  254  255  256  257  258  259  260  261  262  263  264  265  266  267  268  269  270  271  272  273  274  275  276  277  278  279  280  281  282 | package org.o7planning.springmvcshoppingcart.controller;    import java.io.IOException;    import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;    import org.o7planning.springmvcshoppingcart.dao.OrderDAO;  import org.o7planning.springmvcshoppingcart.dao.ProductDAO;  import org.o7planning.springmvcshoppingcart.entity.Product;  import org.o7planning.springmvcshoppingcart.model.CartInfo;  import org.o7planning.springmvcshoppingcart.model.CustomerInfo;  import org.o7planning.springmvcshoppingcart.model.PaginationResult;  import org.o7planning.springmvcshoppingcart.model.ProductInfo;  import org.o7planning.springmvcshoppingcart.util.Utils;  import org.o7planning.springmvcshoppingcart.validator.CustomerInfoValidator;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.stereotype.Controller;  import org.springframework.transaction.annotation.Propagation;  import org.springframework.transaction.annotation.Transactional;  import org.springframework.ui.Model;  import org.springframework.validation.BindingResult;  import org.springframework.validation.annotation.Validated;  import org.springframework.web.bind.WebDataBinder;  import org.springframework.web.bind.annotation.InitBinder;  import org.springframework.web.bind.annotation.ModelAttribute;  import org.springframework.web.bind.annotation.RequestMapping;  import org.springframework.web.bind.annotation.RequestMethod;  import org.springframework.web.bind.annotation.RequestParam;  import org.springframework.web.servlet.config.annotation.EnableWebMvc;  import org.springframework.web.servlet.mvc.support.RedirectAttributes;    @Controller  // Enable Hibernate Transaction.  @Transactional  // Need to use RedirectAttributes  @EnableWebMvc  public class MainController {        @Autowired      private OrderDAO orderDAO;        @Autowired      private ProductDAO productDAO;        @Autowired      private CustomerInfoValidator customerInfoValidator;        @InitBinder      public void myInitBinder(WebDataBinder dataBinder) {          Object target = dataBinder.getTarget();          if (target == null) {              return;          }          System.out.println("Target=" + target);            // For Cart Form.          // (@ModelAttribute("cartForm") @Validated CartInfo cartForm)          if (target.getClass() == CartInfo.class) {            }          // For Customer Form.          // (@ModelAttribute("customerForm") @Validated CustomerInfo          // customerForm)          else if (target.getClass() == CustomerInfo.class) {              dataBinder.setValidator(customerInfoValidator);          }        }        @RequestMapping("/403")      public String accessDenied() {          return "/403";      }        @RequestMapping("/")      public String home() {          return "index";      }        // Product List page.      // Danh sách sản phẩm.      @RequestMapping({ "/productList" })      public String listProductHandler(Model model, //              @RequestParam(value = "name", defaultValue = "") String likeName,              @RequestParam(value = "page", defaultValue = "1") int page) {          final int maxResult = 5;          final int maxNavigationPage = 10;            PaginationResult<ProductInfo> result = productDAO.queryProducts(page, //                  maxResult, maxNavigationPage, likeName);            model.addAttribute("paginationProducts", result);          return "productList";      }        @RequestMapping({ "/buyProduct" })      public String listProductHandler(HttpServletRequest request, Model model, //              @RequestParam(value = "code", defaultValue = "") String code) {            Product product = null;          if (code != null && code.length() > 0) {              product = productDAO.findProduct(code);          }          if (product != null) {                // Cart info stored in Session.              CartInfo cartInfo = Utils.getCartInSession(request);                ProductInfo productInfo = new ProductInfo(product);                cartInfo.addProduct(productInfo, 1);          }          // Redirect to shoppingCart page.          return "redirect:/shoppingCart";      }        @RequestMapping({ "/shoppingCartRemoveProduct" })      public String removeProductHandler(HttpServletRequest request, Model model, //              @RequestParam(value = "code", defaultValue = "") String code) {          Product product = null;          if (code != null && code.length() > 0) {              product = productDAO.findProduct(code);          }          if (product != null) {                // Cart Info stored in Session.              CartInfo cartInfo = Utils.getCartInSession(request);                ProductInfo productInfo = new ProductInfo(product);                cartInfo.removeProduct(productInfo);            }          // Redirect to shoppingCart page.          return "redirect:/shoppingCart";      }        // POST: Update quantity of products in cart.      @RequestMapping(value = { "/shoppingCart" }, method = RequestMethod.POST)      public String shoppingCartUpdateQty(HttpServletRequest request, //              Model model, //              @ModelAttribute("cartForm") CartInfo cartForm) {            CartInfo cartInfo = Utils.getCartInSession(request);          cartInfo.updateQuantity(cartForm);            // Redirect to shoppingCart page.          return "redirect:/shoppingCart";      }        // GET: Show Cart      @RequestMapping(value = { "/shoppingCart" }, method = RequestMethod.GET)      public String shoppingCartHandler(HttpServletRequest request, Model model) {          CartInfo myCart = Utils.getCartInSession(request);            model.addAttribute("cartForm", myCart);          return "shoppingCart";      }        // GET: Enter customer information.      @RequestMapping(value = { "/shoppingCartCustomer" }, method = RequestMethod.GET)      public String shoppingCartCustomerForm(HttpServletRequest request, Model model) {            CartInfo cartInfo = Utils.getCartInSession(request);            // Cart is empty.          if (cartInfo.isEmpty()) {                // Redirect to shoppingCart page.              return "redirect:/shoppingCart";          }            CustomerInfo customerInfo = cartInfo.getCustomerInfo();          if (customerInfo == null) {              customerInfo = new CustomerInfo();          }            model.addAttribute("customerForm", customerInfo);            return "shoppingCartCustomer";      }        // POST: Save customer information.      @RequestMapping(value = { "/shoppingCartCustomer" }, method = RequestMethod.POST)      public String shoppingCartCustomerSave(HttpServletRequest request, //              Model model, //              @ModelAttribute("customerForm") @Validated CustomerInfo customerForm, //              BindingResult result, //              final RedirectAttributes redirectAttributes) {            // If has Errors.          if (result.hasErrors()) {              customerForm.setValid(false);              // Forward to reenter customer info.              return "shoppingCartCustomer";          }            customerForm.setValid(true);          CartInfo cartInfo = Utils.getCartInSession(request);            cartInfo.setCustomerInfo(customerForm);            // Redirect to Confirmation page.          return "redirect:/shoppingCartConfirmation";      }        // GET: Review Cart to confirm.      @RequestMapping(value = { "/shoppingCartConfirmation" }, method = RequestMethod.GET)      public String shoppingCartConfirmationReview(HttpServletRequest request, Model model) {          CartInfo cartInfo = Utils.getCartInSession(request);            // Cart have no products.          if (cartInfo.isEmpty()) {              // Redirect to shoppingCart page.              return "redirect:/shoppingCart";          } else if (!cartInfo.isValidCustomer()) {              // Enter customer info.              return "redirect:/shoppingCartCustomer";          }            return "shoppingCartConfirmation";      }        // POST: Send Cart (Save).      @RequestMapping(value = { "/shoppingCartConfirmation" }, method = RequestMethod.POST)      // Avoid UnexpectedRollbackException (See more explanations)      @Transactional(propagation = Propagation.NEVER)      public String shoppingCartConfirmationSave(HttpServletRequest request, Model model) {          CartInfo cartInfo = Utils.getCartInSession(request);            // Cart have no products.          if (cartInfo.isEmpty()) {              // Redirect to shoppingCart page.              return "redirect:/shoppingCart";          } else if (!cartInfo.isValidCustomer()) {              // Enter customer info.              return "redirect:/shoppingCartCustomer";          }          try {              orderDAO.saveOrder(cartInfo);          } catch (Exception e) {              // Need: Propagation.NEVER?              return "shoppingCartConfirmation";          }          // Remove Cart In Session.          Utils.removeCartInSession(request);            // Store Last ordered cart to Session.          Utils.storeLastOrderedCartInSession(request, cartInfo);            // Redirect to successful page.          return "redirect:/shoppingCartFinalize";      }        @RequestMapping(value = { "/shoppingCartFinalize" }, method = RequestMethod.GET)      public String shoppingCartFinalize(HttpServletRequest request, Model model) {            CartInfo lastOrderedCart = Utils.getLastOrderedCartInSession(request);            if (lastOrderedCart == null) {              return "redirect:/shoppingCart";          }            return "shoppingCartFinalize";      }        @RequestMapping(value = { "/productImage" }, method = RequestMethod.GET)      public void productImage(HttpServletRequest request, HttpServletResponse response, Model model,              @RequestParam("code") String code) throws IOException {          Product product = null;          if (code != null) {              product = this.productDAO.findProduct(code);          }          if (product != null && product.getImage() != null) {              response.setContentType("image/jpeg, image/jpg, image/png, image/gif");              response.getOutputStream().write(product.getImage());          }          response.getOutputStream().close();      }    } |

**AdminController.java**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149  150  151  152  153  154  155  156  157  158  159  160  161  162 | package org.o7planning.springmvcshoppingcart.controller;    import java.util.List;    import org.o7planning.springmvcshoppingcart.dao.OrderDAO;  import org.o7planning.springmvcshoppingcart.dao.ProductDAO;  import org.o7planning.springmvcshoppingcart.model.OrderDetailInfo;  import org.o7planning.springmvcshoppingcart.model.OrderInfo;  import org.o7planning.springmvcshoppingcart.model.PaginationResult;  import org.o7planning.springmvcshoppingcart.model.ProductInfo;  import org.o7planning.springmvcshoppingcart.validator.ProductInfoValidator;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.context.support.ResourceBundleMessageSource;  import org.springframework.security.core.context.SecurityContextHolder;  import org.springframework.security.core.userdetails.UserDetails;  import org.springframework.stereotype.Controller;  import org.springframework.transaction.annotation.Propagation;  import org.springframework.transaction.annotation.Transactional;  import org.springframework.ui.Model;  import org.springframework.validation.BindingResult;  import org.springframework.validation.annotation.Validated;  import org.springframework.web.bind.WebDataBinder;  import org.springframework.web.bind.annotation.InitBinder;  import org.springframework.web.bind.annotation.ModelAttribute;  import org.springframework.web.bind.annotation.RequestMapping;  import org.springframework.web.bind.annotation.RequestMethod;  import org.springframework.web.bind.annotation.RequestParam;  import org.springframework.web.multipart.support.ByteArrayMultipartFileEditor;  import org.springframework.web.servlet.config.annotation.EnableWebMvc;  import org.springframework.web.servlet.mvc.support.RedirectAttributes;    @Controller  // Enable Hibernate Transaction.  @Transactional  // Need to use RedirectAttributes  @EnableWebMvc  public class AdminController {        @Autowired      private OrderDAO orderDAO;        @Autowired      private ProductDAO productDAO;        @Autowired      private ProductInfoValidator productInfoValidator;        // Configurated In ApplicationContextConfig.      @Autowired      private ResourceBundleMessageSource messageSource;        @InitBinder      public void myInitBinder(WebDataBinder dataBinder) {          Object target = dataBinder.getTarget();          if (target == null) {              return;          }          System.out.println("Target=" + target);            if (target.getClass() == ProductInfo.class) {              dataBinder.setValidator(productInfoValidator);              // For upload Image.              dataBinder.registerCustomEditor(byte[].class, new ByteArrayMultipartFileEditor());          }      }        // GET: Show Login Page      @RequestMapping(value = { "/login" }, method = RequestMethod.GET)      public String login(Model model) {            return "login";      }        @RequestMapping(value = { "/accountInfo" }, method = RequestMethod.GET)      public String accountInfo(Model model) {            UserDetails userDetails = (UserDetails) SecurityContextHolder.getContext().getAuthentication().getPrincipal();          System.out.println(userDetails.getPassword());          System.out.println(userDetails.getUsername());          System.out.println(userDetails.isEnabled());            model.addAttribute("userDetails", userDetails);          return "accountInfo";      }        @RequestMapping(value = { "/orderList" }, method = RequestMethod.GET)      public String orderList(Model model, //              @RequestParam(value = "page", defaultValue = "1") String pageStr) {          int page = 1;          try {              page = Integer.parseInt(pageStr);          } catch (Exception e) {          }          final int MAX\_RESULT = 5;          final int MAX\_NAVIGATION\_PAGE = 10;            PaginationResult<OrderInfo> paginationResult //          = orderDAO.listOrderInfo(page, MAX\_RESULT, MAX\_NAVIGATION\_PAGE);            model.addAttribute("paginationResult", paginationResult);          return "orderList";      }        // GET: Show product.      @RequestMapping(value = { "/product" }, method = RequestMethod.GET)      public String product(Model model, @RequestParam(value = "code", defaultValue = "") String code) {          ProductInfo productInfo = null;            if (code != null && code.length() > 0) {              productInfo = productDAO.findProductInfo(code);          }          if (productInfo == null) {              productInfo = new ProductInfo();              productInfo.setNewProduct(true);          }          model.addAttribute("productForm", productInfo);          return "product";      }        // POST: Save product      @RequestMapping(value = { "/product" }, method = RequestMethod.POST)      // Avoid UnexpectedRollbackException (See more explanations)      @Transactional(propagation = Propagation.NEVER)      public String productSave(Model model, //              @ModelAttribute("productForm") @Validated ProductInfo productInfo, //              BindingResult result, //              final RedirectAttributes redirectAttributes) {            if (result.hasErrors()) {              return "product";          }          try {              productDAO.save(productInfo);          } catch (Exception e) {              // Need: Propagation.NEVER?              String message = e.getMessage();              model.addAttribute("message", message);              // Show product form.              return "product";            }          return "redirect:/productList";      }        @RequestMapping(value = { "/order" }, method = RequestMethod.GET)      public String orderView(Model model, @RequestParam("orderId") String orderId) {          OrderInfo orderInfo = null;          if (orderId != null) {              orderInfo = this.orderDAO.getOrderInfo(orderId);          }          if (orderInfo == null) {              return "redirect:/orderList";          }          List<OrderDetailInfo> details = this.orderDAO.listOrderDetailInfos(orderId);          orderInfo.setDetails(details);            model.addAttribute("orderInfo", orderInfo);            return "order";      }    } |

## 14- JSP files



**styles.css**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139  140  141  142 | html {      background: white;  }  h3 {      margin: 0px;      padding: 0px;  }  body {      max-width: 860px;      min-width: 360px;      margin: 0px auto;      background: #F8F8F8;      padding:0px 5px;      text-align:center;  }    .page-title  {      font-size:120%;      text-align: left;      margin:10px 0px;  }  .header-container {      text-align: left;      border-bottom: 1px solid #ccc;      position: relative;      background: #5f5f5f;      color: white;  }  .header-container .header-bar  {      position: absolute;      right: 10px;      bottom: 20px;  }  .header-container .header-bar  a  {      color: white;      font-size: bold;  }    .footer-container {      text-align: center;      margin-top: 10px;      padding: 5px 0px 0px 0px;      border-top: 1px solid #ccc;  }  .menu-container {      text-align: right;      margin: 10px 5px;  }  .menu-container a {      margin: 5px 5px 5px 10px;      color: #004d99;      font-weight: bold;      text-decoration: none;  }    .site-name {      font-size:200%;      margin:10px 10px 10px 0px;      padding: 5px;  }    .container  {      margin: 5px 0px;  }    .demo-container, .login-container, .account-container {      padding: 5px;      border: 1px solid #ccc;      text-align:left;      margin:20px 0px;  }    .customer-info-container {      text-align: left;      border: 1px solid #ccc;      padding: 0px 5px;  }  .product-preview-container {      border: 1px solid #ccc;      padding: 5px;      width: 250px;      margin: 10px ;      display: inline-block;      text-align:left;  }    .product-preview-container input {      width: 50px;  }      .product-image {      width: 120px;      height: 80px;  }    ul {      list-style-type: none;      padding-left: 5px;      margin:5px;  }      .navi-item {      margin: 5px 5px 5px 20px;  }    .button-update-sc {      color: red;      margin: 5px 5px 5px 20px;  }    .button-send-sc {      color: red;      margin: 5px 5px 5px 20px;  }    .error-message {      font-size: 90%;      color: red;      font-style: italic;  }    .price {      color: blue;      font-weight: bold;  }    .subtotal {      color: red;      font-weight: bold;  }    .total {      color: red;      font-weight: bold;      font-size: 120%;  }    table td {      padding: 5px;  } |

**\_footer.jsp**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | <%@ page language="java" contentType="text/html; charset=UTF-8"      pageEncoding="UTF-8"%>      <div class="footer-container">       @Copy right <a href="[http://o7planning.org](http://o7planning.org/)" target="\_blank">o7planning.org</a>     <br>     See more <a>demo</a>    </div> |

**\_header.jsp**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22 | <%@ page language="java" contentType="text/html; charset=UTF-8"      pageEncoding="UTF-8"%>  <%@ taglib uri="<http://java.sun.com/jsp/jstl/core>" prefix="c"%>    <div class="header-container">        <div class="site-name">Online Shop</div>        <div class="header-bar">          <c:if test="${pageContext.request.userPrincipal.name != null}">          Hello             <a href="${pageContext.request.contextPath}/accountInfo">                  ${pageContext.request.userPrincipal.name} </a>           &nbsp;|&nbsp;             <a href="${pageContext.request.contextPath}/logout">Logout</a>            </c:if>          <c:if test="${pageContext.request.userPrincipal.name == null}">              <a href="${pageContext.request.contextPath}/login">Login</a>          </c:if>      </div>  </div> |

**\_menu.jsp**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33 | <%@ page language="java" contentType="text/html; charset=UTF-8"      pageEncoding="UTF-8"%>    <%@ taglib uri="<http://www.springframework.org/security/tags>" prefix="security" %>      <div class="menu-container">       <a href="${pageContext.request.contextPath}/">Home</a>     |     <a href="${pageContext.request.contextPath}/productList">        Product List     </a>     |     <a href="${pageContext.request.contextPath}/shoppingCart">        My Cart     </a>     |     <security:authorize  access="hasAnyRole('ROLE\_MANAGER','ROLE\_EMPLOYEE')">       <a href="${pageContext.request.contextPath}/orderList">           Order List       </a>       |     </security:authorize>       <security:authorize  access="hasRole('ROLE\_MANAGER')">           <a href="${pageContext.request.contextPath}/product">                          Create Product           </a>           |     </security:authorize>    </div> |

**403.jsp**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">    <title>Access Denied</title>    <link rel="stylesheet" type="text/css" href="${pageContext.request.contextPath}/styles.css">    </head>  <body>         <jsp:include page="\_header.jsp" />     <jsp:include page="\_menu.jsp" />       <div class="page-title">Access Denied!</div>       <h3 style="color:red;">Sorry, you can not access this page!</h3>         <jsp:include page="\_footer.jsp" />    </body>  </html> |

**accountInfo.jsp**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41 | <%@ taglib uri="<http://java.sun.com/jsp/jstl/core>" prefix="c"%>  <%@ taglib uri="<http://www.springframework.org/security/tags>" prefix="security" %>    <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">    <title>Account Info</title>    <link rel="stylesheet" type="text/css" href="${pageContext.request.contextPath}/styles.css">    </head>  <body>         <jsp:include page="\_header.jsp" />     <jsp:include page="\_menu.jsp" />       <div class="page-title">Account Info</div>       <div class="account-container">             <ul>             <li>User Name: ${pageContext.request.userPrincipal.name}</li>             <li>Role:                 <ul>                     <c:forEach items="${userDetails.authorities}" var="auth">                         <li>${auth.authority }</li>                     </c:forEach>                 </ul>             </li>         </ul>     </div>         <jsp:include page="\_footer.jsp" />    </body>  </html> |

**index.jsp**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33 | <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">    <title>Books Shop Online</title>    <link rel="stylesheet" type="text/css" href="${pageContext.request.contextPath}/styles.css">    </head>  <body>         <jsp:include page="\_header.jsp" />     <jsp:include page="\_menu.jsp" />       <div class="page-title">Shopping Cart Demo</div>       <div class="demo-container">     <h3>Demo content</h3>       <ul>        <li>Buy online</li>        <li>Admin pages</li>        <li>Reports</li>     </ul>     </div>         <jsp:include page="\_footer.jsp" />    </body>  </html> |

**login.jsp**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66 | <%@ taglib uri="<http://java.sun.com/jsp/jstl/core>" prefix="c"%>    <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">    <title>Login</title>    <link rel="stylesheet" type="text/css" href="${pageContext.request.contextPath}/styles.css">    </head>  <body>         <jsp:include page="\_header.jsp" />     <jsp:include page="\_menu.jsp" />           <div class="page-title">Login (For Employee, Manager)</div>       <div class="login-container">           <h3>Enter username and password</h3>         <br>         <!-- /login?error=true -->         <c:if test="${param.error == 'true'}">             <div style="color: red; margin: 10px 0px;">                   Login Failed!!!<br /> Reason :                 ${sessionScope["SPRING\_SECURITY\_LAST\_EXCEPTION"].message}               </div>         </c:if>           <form method="POST"             action="${pageContext.request.contextPath}/j\_spring\_security\_check">             <table>                 <tr>                     <td>User Name \*</td>                     <td><input name="userName" /></td>                 </tr>                   <tr>                     <td>Password \*</td>                     <td><input type="password" name="password" /></td>                 </tr>                   <tr>                     <td>&nbsp;</td>                     <td><input type="submit" value="Login" /> <input type="reset"                         value="Reset" /></td>                 </tr>             </table>         </form>           <span class="error-message">${error }</span>       </div>         <jsp:include page="\_footer.jsp" />    </body>  </html> |

**order.jsp**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83 | <%@ taglib uri="<http://java.sun.com/jsp/jstl/core>" prefix="c"%>  <%@ taglib prefix="fmt" uri="<http://java.sun.com/jsp/jstl/fmt>" %>    <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">  <title>Product List</title>    <link rel="stylesheet" type="text/css" href="${pageContext.request.contextPath}/styles.css">    </head>  <body>       <jsp:include page="\_header.jsp" />     <jsp:include page="\_menu.jsp" />       <fmt:setLocale value="en\_US" scope="session"/>       <div class="page-title">Order Info</div>       <div class="customer-info-container">         <h3>Customer Information:</h3>         <ul>             <li>Name: ${orderInfo.customerName}</li>             <li>Email: ${orderInfo.customerEmail}</li>             <li>Phone: ${orderInfo.customerPhone}</li>             <li>Address: ${orderInfo.customerAddress}</li>         </ul>         <h3>Order Summary:</h3>         <ul>             <li>Total:             <span class="total">             <fmt:formatNumber value="${orderInfo.amount}" type="currency"/>             </span></li>         </ul>     </div>       <br/>       <table border="1" style="width:100%">         <tr>             <th>Product Code</th>             <th>Product Name</th>             <th>Quantity</th>             <th>Price</th>             <th>Amount</th>         </tr>         <c:forEach items="${orderInfo.details}" var="orderDetailInfo">             <tr>                 <td>${orderDetailInfo.productCode}</td>                 <td>${orderDetailInfo.productName}</td>                 <td>${orderDetailInfo.quanity}</td>                 <td>                  <fmt:formatNumber value="${orderDetailInfo.price}" type="currency"/>                 </td>                 <td>                  <fmt:formatNumber value="${orderDetailInfo.amount}" type="currency"/>                 </td>             </tr>         </c:forEach>     </table>     <c:if test="${paginationResult.totalPages > 1}">         <div class="page-navigator">            <c:forEach items="${paginationResult.navigationPages}" var = "page">                <c:if test="${page != -1 }">                  <a href="orderList?page=${page}" class="nav-item">${page}</a>                </c:if>                <c:if test="${page == -1 }">                  <span class="nav-item"> ... </span>                </c:if>            </c:forEach>           </div>     </c:if>             <jsp:include page="\_footer.jsp" />    </body>  </html> |

**orderList.jsp**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71 | <%@ taglib uri="<http://java.sun.com/jsp/jstl/core>" prefix="c"%>  <%@ taglib prefix="fmt" uri="<http://java.sun.com/jsp/jstl/fmt>" %>    <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">  <title>Product List</title>    <link rel="stylesheet" type="text/css" href="${pageContext.request.contextPath}/styles.css">    </head>  <body>       <jsp:include page="\_header.jsp" />     <jsp:include page="\_menu.jsp" />       <fmt:setLocale value="en\_US" scope="session"/>       <div class="page-title">Order List</div>       <div>Total Order Count: ${paginationResult.totalRecords}</div>       <table border="1" style="width:100%">         <tr>             <th>Order Num</th>             <th>Order Date</th>             <th>Customer Name</th>             <th>Customer Address</th>             <th>Customer Email</th>             <th>Amount</th>             <th>View</th>         </tr>         <c:forEach items="${paginationResult.list}" var="orderInfo">             <tr>                 <td>${orderInfo.orderNum}</td>                 <td>                    <fmt:formatDate value="${orderInfo.orderDate}" pattern="dd-MM-yyyy HH:mm"/>                 </td>                 <td>${orderInfo.customerName}</td>                 <td>${orderInfo.customerAddress}</td>                 <td>${orderInfo.customerEmail}</td>                 <td style="color:red;">                    <fmt:formatNumber value="${orderInfo.amount}" type="currency"/>                 </td>                 <td><a href="${pageContext.request.contextPath}/order?orderId=${orderInfo.id}">                    View</a></td>             </tr>         </c:forEach>     </table>     <c:if test="${paginationResult.totalPages > 1}">         <div class="page-navigator">            <c:forEach items="${paginationResult.navigationPages}" var = "page">                <c:if test="${page != -1 }">                  <a href="orderList?page=${page}" class="nav-item">${page}</a>                </c:if>                <c:if test="${page == -1 }">                  <span class="nav-item"> ... </span>                </c:if>            </c:forEach>           </div>     </c:if>             <jsp:include page="\_footer.jsp" />    </body>  </html> |

**product.jsp**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81 | <%@ taglib uri="<http://java.sun.com/jsp/jstl/core>" prefix="c"%>  <%@taglib uri="<http://www.springframework.org/tags/form>" prefix="form"%>    <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">  <title>Product</title>    <link rel="stylesheet" type="text/css" href="${pageContext.request.contextPath}/styles.css">    </head>  <body>       <jsp:include page="\_header.jsp" />     <jsp:include page="\_menu.jsp" />       <div class="page-title">Product</div>       <c:if test="${not empty errorMessage }">       <div class="error-message">           ${errorMessage}       </div>     </c:if>       <form:form modelAttribute="productForm" method="POST" enctype="multipart/form-data">         <table style="text-align:left;">             <tr>                 <td>Code \*</td>                 <td style="color:red;">                    <c:if test="${not empty productForm.code}">                         <form:hidden path="code"/>                         ${productForm.code}                    </c:if>                    <c:if test="${empty productForm.code}">                         <form:input path="code" />                         <form:hidden path="newProduct" />                    </c:if>                 </td>                 <td><form:errors path="code" class="error-message" /></td>             </tr>               <tr>                 <td>Name \*</td>                 <td><form:input path="name" /></td>                 <td><form:errors path="name" class="error-message" /></td>             </tr>               <tr>                 <td>Price \*</td>                 <td><form:input path="price" /></td>                 <td><form:errors path="price" class="error-message" /></td>             </tr>             <tr>                 <td>Image</td>                 <td>                 <img src="${pageContext.request.contextPath}/productImage?code=${productForm.code}" width="100"/></td>                 <td> </td>             </tr>             <tr>                 <td>Upload Image</td>                 <td><form:input type="file" path="fileData"/></td>                 <td> </td>             </tr>                 <tr>                 <td>&nbsp;</td>                 <td><input type="submit" value="Submit" /> <input type="reset"                     value="Reset" /></td>             </tr>         </table>     </form:form>             <jsp:include page="\_footer.jsp" />    </body>  </html> |

**productList.jsp**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67 | <%@ taglib uri="<http://java.sun.com/jsp/jstl/core>" prefix="c"%>  <%@ taglib prefix="fmt" uri="<http://java.sun.com/jsp/jstl/fmt>" %>    <%@ taglib uri="<http://www.springframework.org/security/tags>" prefix="security" %>    <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">  <title>Product List</title>    <link rel="stylesheet" type="text/css" href="${pageContext.request.contextPath}/styles.css">    </head>  <body>       <jsp:include page="\_header.jsp" />     <jsp:include page="\_menu.jsp" />       <fmt:setLocale value="en\_US" scope="session"/>       <div class="page-title">Product List</div>           <c:forEach items="${paginationProducts.list}" var="prodInfo">         <div class="product-preview-container">             <ul>                 <li><img class="product-image"                     src="${pageContext.request.contextPath}/productImage?code=${prodInfo.code}" /></li>                 <li>Code: ${prodInfo.code}</li>                 <li>Name: ${prodInfo.name}</li>                 <li>Price: <fmt:formatNumber value="${prodInfo.price}" type="currency"/></li>                 <li><a                     href="${pageContext.request.contextPath}/buyProduct?code=${prodInfo.code}">                         Buy Now</a></li>                 <!-- For Manager edit Product -->                 <security:authorize  access="hasRole('ROLE\_MANAGER')">                   <li><a style="color:red;"                       href="${pageContext.request.contextPath}/product?code=${prodInfo.code}">                         Edit Product</a></li>                 </security:authorize>             </ul>         </div>       </c:forEach>     <br/>         <c:if test="${paginationProducts.totalPages > 1}">         <div class="page-navigator">            <c:forEach items="${paginationProducts.navigationPages}" var = "page">                <c:if test="${page != -1 }">                  <a href="productList?page=${page}" class="nav-item">${page}</a>                </c:if>                <c:if test="${page == -1 }">                  <span class="nav-item"> ... </span>                </c:if>            </c:forEach>           </div>     </c:if>       <jsp:include page="\_footer.jsp" />    </body>  </html> |

**shoppingCart.jsp**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83 | <%@ taglib uri="<http://java.sun.com/jsp/jstl/core>" prefix="c"%>  <%@ taglib prefix="fmt" uri="<http://java.sun.com/jsp/jstl/fmt>" %>  <%@taglib uri="<http://www.springframework.org/tags/form>" prefix="form"%>    <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">    <title>Shopping Cart</title>    <link rel="stylesheet" type="text/css" href="${pageContext.request.contextPath}/styles.css">    </head>  <body>     <jsp:include page="\_header.jsp" />       <jsp:include page="\_menu.jsp" />       <fmt:setLocale value="en\_US" scope="session"/>       <div class="page-title">My Cart</div>       <c:if test="${empty cartForm or empty cartForm.cartLines}">         <h2>There is no items in Cart</h2>         <a href="${pageContext.request.contextPath}/productList">Show             Product List</a>     </c:if>       <c:if test="${not empty cartForm and not empty cartForm.cartLines   }">         <form:form method="POST" modelAttribute="cartForm"             action="${pageContext.request.contextPath}/shoppingCart">               <c:forEach items="${cartForm.cartLines}" var="cartLineInfo"                 varStatus="varStatus">                 <div class="product-preview-container">                     <ul>                         <li><img class="product-image"                             src="${pageContext.request.contextPath}/productImage?code=${cartLineInfo.productInfo.code}" />                         </li>                         <li>Code: ${cartLineInfo.productInfo.code} <form:hidden                                 path="cartLines[${varStatus.index}].productInfo.code" />                           </li>                         <li>Name: ${cartLineInfo.productInfo.name}</li>                         <li>Price: <span class="price">                             <fmt:formatNumber value="${cartLineInfo.productInfo.price}" type="currency"/>                           </span></li>                         <li>Quantity: <form:input                                 path="cartLines[${varStatus.index}].quantity" /></li>                         <li>Subtotal:                           <span class="subtotal">                                <fmt:formatNumber value="${cartLineInfo.amount}" type="currency"/>                             </span>                         </li>                         <li><a                             href="${pageContext.request.contextPath}/shoppingCartRemoveProduct?code=${cartLineInfo.productInfo.code}">                                 Delete </a></li>                     </ul>                 </div>             </c:forEach>             <div style="clear: both"></div>             <input class="button-update-sc" type="submit" value="Update Quantity" />             <a class="navi-item"                 href="${pageContext.request.contextPath}/shoppingCartCustomer">Enter                 Customer Info</a>             <a class="navi-item"                 href="${pageContext.request.contextPath}/productList">Continue                 Buy</a>         </form:form>         </c:if>         <jsp:include page="\_footer.jsp" />    </body>  </html> |

**shoppingCartConfirmation.jsp**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89 | <%@ taglib uri="<http://java.sun.com/jsp/jstl/core>" prefix="c"%>  <%@ taglib prefix="fmt" uri="<http://java.sun.com/jsp/jstl/fmt>" %>    <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">    <title>Shopping Cart Confirmation</title>    <link rel="stylesheet" type="text/css" href="${pageContext.request.contextPath}/styles.css">    </head>  <body>    <jsp:include page="\_header.jsp" />      <jsp:include page="\_menu.jsp" />      <fmt:setLocale value="en\_US" scope="session"/>      <div class="page-title">Confirmation</div>          <div class="customer-info-container">        <h3>Customer Information:</h3>        <ul>            <li>Name: ${myCart.customerInfo.name}</li>            <li>Email: ${myCart.customerInfo.email}</li>            <li>Phone: ${myCart.customerInfo.phone}</li>            <li>Address: ${myCart.customerInfo.address}</li>        </ul>        <h3>Cart Summary:</h3>        <ul>            <li>Quantity: ${myCart.quantityTotal}</li>            <li>Total:            <span class="total">              <fmt:formatNumber value="${myCart.amountTotal}" type="currency"/>            </span></li>        </ul>    </div>      <form method="POST"        action="${pageContext.request.contextPath}/shoppingCartConfirmation">          <!-- Edit Cart -->        <a class="navi-item"            href="${pageContext.request.contextPath}/shoppingCart">Edit Cart</a>          <!-- Edit Customer Info -->        <a class="navi-item"            href="${pageContext.request.contextPath}/shoppingCartCustomer">Edit            Customer Info</a>          <!-- Send/Save -->        <input type="submit" value="Send" class="button-send-sc" />    </form>      <div class="container">          <c:forEach items="${myCart.cartLines}" var="cartLineInfo">            <div class="product-preview-container">                <ul>                    <li><img class="product-image"                        src="${pageContext.request.contextPath}/productImage?code=${cartLineInfo.productInfo.code}" /></li>                    <li>Code: ${cartLineInfo.productInfo.code} <input                        type="hidden" name="code" value="${cartLineInfo.productInfo.code}" />                    </li>                    <li>Name: ${cartLineInfo.productInfo.name}</li>                    <li>Price: <span class="price">                       <fmt:formatNumber value="${cartLineInfo.productInfo.price}" type="currency"/>                    </span>                    </li>                    <li>Quantity: ${cartLineInfo.quantity}</li>                    <li>Subtotal:                      <span class="subtotal">                         <fmt:formatNumber value="${cartLineInfo.amount}" type="currency"/>                      </span>                    </li>                </ul>            </div>        </c:forEach>      </div>      <jsp:include page="\_footer.jsp" />    </body>  </html> |

**shoppingCartCustomer.jsp**

?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64 | <%@ taglib uri="<http://java.sun.com/jsp/jstl/core>" prefix="c"%>  <%@taglib uri="<http://www.springframework.org/tags/form>" prefix="form"%>  <%@ taglib prefix="spring" uri="<http://www.springframework.org/tags>"%>      <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">    <title>Enter Customer Information</title>    <link rel="stylesheet" type="text/css" href="${pageContext.request.contextPath}/styles.css">    </head>  <body>  <jsp:include page="\_header.jsp" />  <jsp:include page="\_menu.jsp" />    <div class="page-title">Enter Customer Information</div>       <form:form method="POST" modelAttribute="customerForm"         action="${pageContext.request.contextPath}/shoppingCartCustomer">           <table>             <tr>                 <td>Name \*</td>                 <td><form:input path="name" /></td>                 <td><form:errors path="name" class="error-message" /></td>             </tr>               <tr>                 <td>Email \*</td>                 <td><form:input path="email" /></td>                 <td><form:errors path="email" class="error-message" /></td>             </tr>               <tr>                 <td>Phone \*</td>                 <td><form:input path="phone" /></td>                 <td><form:errors path="phone" class="error-message" /></td>             </tr>               <tr>                 <td>Address \*</td>                 <td><form:input path="address" /></td>                 <td><form:errors path="address" class="error-message" /></td>             </tr>               <tr>                 <td>&nbsp;</td>                 <td><input type="submit" value="Submit" /> <input type="reset"                     value="Reset" /></td>             </tr>         </table>       </form:form>         <jsp:include page="\_footer.jsp" />      </body>  </html> |

**shoppingCartFinalize.jsp**

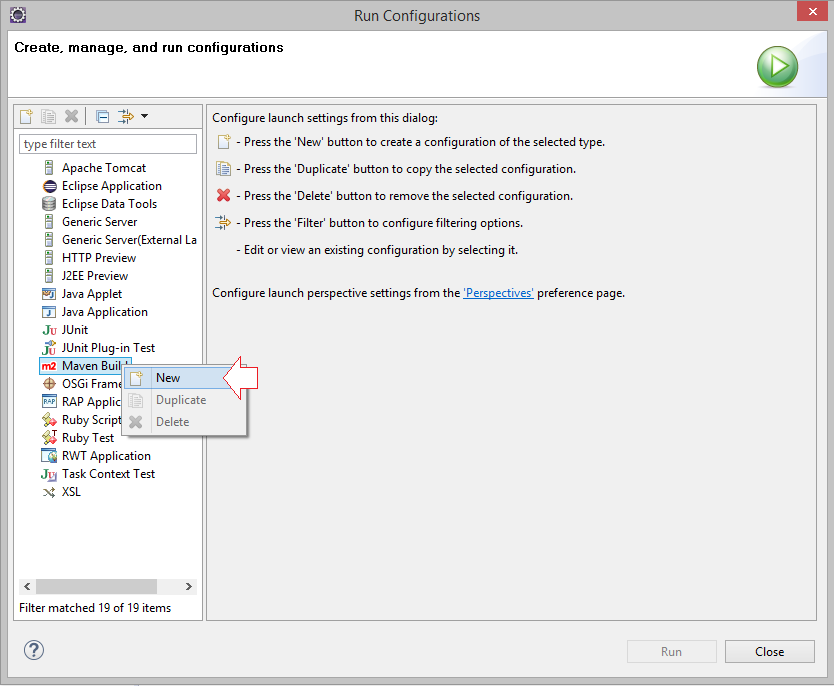
?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | <%@ taglib uri="<http://java.sun.com/jsp/jstl/core>" prefix="c"%>    <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">    <title>Shopping Cart Finalize</title>    <link rel="stylesheet" type="text/css" href="${pageContext.request.contextPath}/styles.css">    </head>  <body>     <jsp:include page="\_header.jsp" />       <jsp:include page="\_menu.jsp" />       <div class="page-title">Finalize</div>       <div class="container">         <h3>Thank you for Order</h3>         Your order number is: ${lastOrderedCart.orderNum}     </div>       <jsp:include page="\_footer.jsp" />    </body>  </html> |

15- Running application

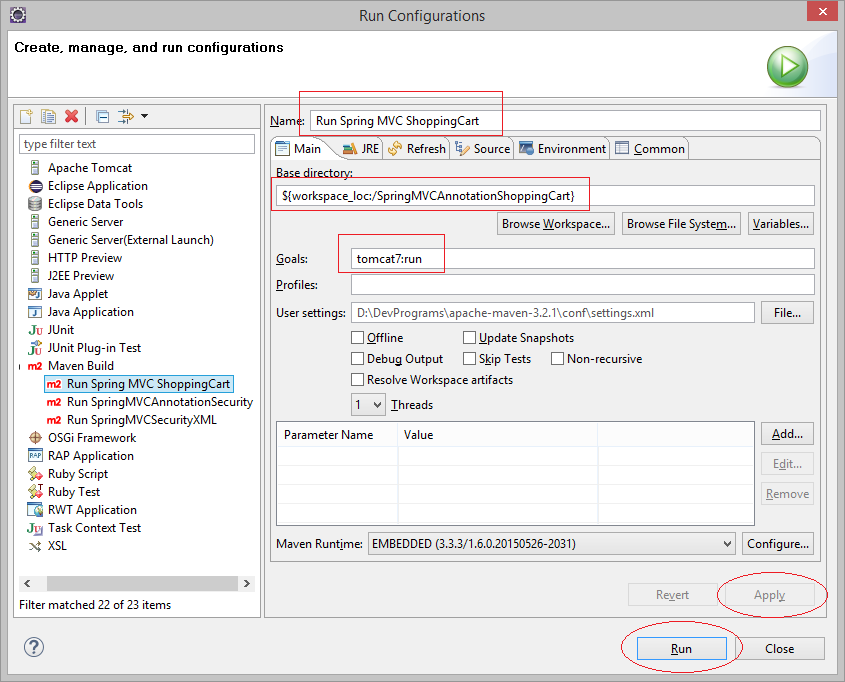
Right click the project, select:

* **Run As/Run Applications:**



Enter:

* **Name:** Run Spring MVC Shopping Cart
* **Base directory:** ${workspace\_loc:/SpringMVCAnnotationShoppingCart}
* **Goals:** tomcat7:run



* <http://localhost:8080/SpringMVCAnnotationShoppingCart>

