**SPRING SECURITY ASSIGNMENT 1**

**pom.xml**

<project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4\_0\_0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>books.app</groupId>

<artifactId>springmvcsecurity</artifactId>

<packaging>war</packaging>

<version>0.0.1-SNAPSHOT</version>

<name>springmvcsecurity Maven Webapp</name>

<url>http://maven.apache.org</url>

<properties>

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

<spring.version>5.0.7.RELEASE</spring.version>

<spring.data>2.0.1.RELEASE</spring.data>

<springsecurity.version>5.0.7.RELEASE</springsecurity.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-web</artifactId>

<version>${springsecurity.version}</version>

</dependency>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-config</artifactId>

<version>${springsecurity.version}</version>

</dependency>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-taglibs</artifactId>

<version>${springsecurity.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-orm</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework.data</groupId>

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

<version>${spring.data}</version>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

<version>1.2</version>

</dependency>

<dependency>

<groupId>mysql</groupId>

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

<version>5.1.44</version>

</dependency> <!-- Hibernate 5.2.6 Final -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>5.2.6.Final</version>

</dependency>

<dependency>

<groupId>org.apache.commons</groupId>

<artifactId>commons-dbcp2</artifactId>

<version>2.1.1</version>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>3.8.1</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<finalName>springmvcsecurity</finalName>

</build>

</project>

**web.xml**

<web-app xmlns=*"http://java.sun.com/xml/ns/javaee"*

version=*"2.5"*

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

xsi:schemaLocation=*"http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd"*>

<display-name>Spring MVC Spring Security Application</display-name>

<servlet>

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

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

<init-param>

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

<param-value>/WEB-INF/dispatcher-servlet.xml</param-value>

</init-param>

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

</servlet>

<servlet-mapping>

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

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

</servlet-mapping>

<listener>

<listener-class>

org.springframework.web.context.ContextLoaderListener

</listener-class>

</listener>

<context-param>

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

<param-value>/WEB-INF/spring.xml</param-value>

</context-param>

</web-app>

**spring.xml**

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

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:context=*"http://www.springframework.org/schema/context"*

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

xmlns:jpa=*"http://www.springframework.org/schema/data/jpa"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans-3.0.xsd*

*http://www.springframework.org/schema/context*

*http://www.springframework.org/schema/context/spring-context-3.0.xsd*

*http://www.springframework.org/schema/tx*

*http://www.springframework.org/schema/tx/spring-tx.xsd*

*http://www.springframework.org/schema/data/jpa*

*http://www.springframework.org/schema/data/jpa/spring-jpa-1.8.xsd"*>

<context:component-scan base-package=*"com.books.app"* />

<bean id=*"userService"* class=*"com.books.app.service.UserServiceImpl"*></bean>

<bean id=*"bookAppService"* class=*"com.books.app.service.BookAppServiceImpl"*></bean>

<bean id=*"bCryptPasswordEncoder"* class=*"org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder"*></bean>

<context:property-placeholder location=*"classpath:database.properties"* />

<bean id=*"basicDataSource"* class=*"org.apache.commons.dbcp2.BasicDataSource"*

destroy-method=*"close"*>

<property name=*"driverClassName"* value=*"${database.driverClassName}"* />

<property name=*"url"* value=*"${database.url}"* />

<property name=*"username"* value=*"${database.username}"* />

<property name=*"password"* value=*"${database.password}"* />

</bean>

<bean name=*"hibernateJpaVendorAdapter"*

class=*"org.springframework.orm.jpa.vendor.HibernateJpaVendorAdapter"* />

<bean id=*"entityManagerFactory"*

class=*"org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean"*>

<property name=*"jpaVendorAdapter"* ref=*"hibernateJpaVendorAdapter"* />

<property name=*"dataSource"* ref=*"basicDataSource"* />

<property name=*"persistenceUnitName"* value=*"myJpaPersistenceUnit"* />

<property name=*"packagesToScan"*>

<list>

<value>com.books.app.model</value>

</list>

</property>

<property name=*"jpaProperties"*>

<props>

<prop key=*"hibernate.dialect"*>${hibernate.dialect}</prop>

<prop key=*"hibernate.show\_sql"*>${hibernate.show\_sql}</prop>

<prop key=*"hibernate.format\_sql"*>${hibernate.format\_sql}</prop>

<prop key=*"hibernate.id.new\_generator\_mappings"*>${hibernate.id.new\_generator\_mappings}</prop>

</props>

</property>

</bean>

<bean id=*"transactionManager"* class=*"org.springframework.orm.jpa.JpaTransactionManager"*>

<property name=*"entityManagerFactory"* ref=*"entityManagerFactory"* />

</bean>

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

<jpa:repositories base-package=*"com.books.app.dao"*

entity-manager-factory-ref=*"entityManagerFactory"* />

</beans>

**dispatcher-servlet.xml**

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

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

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

xsi:schemaLocation=*" http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.0.xsd http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc.xsd http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-3.0.xsd"*>

<mvc:annotation-driven />

<context:component-scan base-package=*"com.books.app"* />

<bean

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

<property name=*"prefix"* value=*"/pages/"* />

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

</bean>

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

</beans>

**database.properties**

database.driverClassName=com.mysql.jdbc.Driver

database.url=jdbc:mysql://localhost:3306/bookshop

database.username=root

database.password=root

hibernate.dialect = org.hibernate.dialect.MySQLDialect

hibernate.show\_sql = true

hibernate.format\_sql = true

hibernate.id.new\_generator\_mappings = false

**SecurityConfig.java**

**package** com.books.app.util;

**import** javax.sql.DataSource;

**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.builders.WebSecurity;

**import** org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;

**import** org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;

**import** org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

**import** org.springframework.security.web.util.matcher.AntPathRequestMatcher;

**import** com.books.app.dao.IQueryMapper;

@Configuration

@EnableWebSecurity

**public** **class** SecurityConfig **extends** WebSecurityConfigurerAdapter{

@Autowired

**private** BCryptPasswordEncoder bCryptPasswordEncoder;

@Autowired

**private** DataSource dataSource;

@Override

**protected** **void** configure(AuthenticationManagerBuilder auth) **throws** Exception {

auth.jdbcAuthentication().usersByUsernameQuery(IQueryMapper.***USERS\_QRY***)

.authoritiesByUsernameQuery(IQueryMapper.***ROLES\_QRY***).dataSource(dataSource)

.passwordEncoder(bCryptPasswordEncoder);

}

@Override

**protected** **void** configure(HttpSecurity http) **throws** Exception {

http.authorizeRequests()

.antMatchers("/").permitAll()

.antMatchers("/login").permitAll()

.antMatchers("/registration").permitAll()

.antMatchers("/manageBooks/\*\*").hasAuthority("LIBRARIAN")

.antMatchers("/manageSubjects/\*\*").hasAuthority("PRINCIPAL")

.anyRequest().authenticated().and().csrf().disable()

.formLogin().loginPage("/login").failureUrl("/login?error=true")

.defaultSuccessUrl("/home")

.usernameParameter("email")

.passwordParameter("password")

.and().logout()

.logoutRequestMatcher(**new** AntPathRequestMatcher("/logout"))

.logoutSuccessUrl("/").and().exceptionHandling()

.accessDeniedPage("/accessDenied");

}

@Override

**public** **void** configure(WebSecurity web) **throws** Exception {

web.ignoring()

.antMatchers("/resources/\*\*","/static/\*\*","/css/\*\*","/js/\*\*","/images/\*\*");

}

}

**BookController.java**

**package** com.books.app.controller;

**import** java.util.ArrayList;

**import** java.util.Arrays;

**import** java.util.HashSet;

**import** java.util.List;

**import** java.util.Set;

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

**import** org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

**import** org.springframework.stereotype.Controller;

**import** org.springframework.ui.Model;

**import** org.springframework.validation.BindingResult;

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

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

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

**import** org.springframework.web.servlet.ModelAndView;

**import** com.books.app.dao.IRoleRepository;

**import** com.books.app.exception.BookAppException;

**import** com.books.app.model.Book;

**import** com.books.app.model.Role;

**import** com.books.app.model.Subject;

**import** com.books.app.model.User;

**import** com.books.app.service.IBookAppService;

**import** com.books.app.service.IUserService;

@Controller

**public** **class** BookController {

@Autowired

IBookAppService bookAppService;

@Autowired

IUserService userService;

@Autowired

IRoleRepository roleRepository;

@Autowired

**private** BCryptPasswordEncoder bCryptPasswordEncoder;

@RequestMapping(value ="/", method = RequestMethod.***GET***)

**public** ModelAndView showLogin() {

**return** **new** ModelAndView("login", "command", **new** User());

}

@RequestMapping(value = "/home", method = RequestMethod.***POST***)

**public** ModelAndView showHome() {

ModelAndView modelAndView = **new** ModelAndView();

modelAndView.setViewName("home");

**return** modelAndView;

}

@RequestMapping("/registration")

**public** ModelAndView showRegForm() {

**return** **new** ModelAndView("registration", "command", **new** User());

}

@RequestMapping(value = "/registration", method = RequestMethod.***POST***)

**public** ModelAndView registerUser(@ModelAttribute("user") User user, BindingResult result, Model model) {

**try** {

user.setPassword(bCryptPasswordEncoder.encode(user.getPassword()));

user.setActive(1);

Role userRole = roleRepository.findByRolename(user.getUserrole());

user.setRoles(**new** HashSet<Role>(Arrays.*asList*(userRole)));

userService.saveUser(user);

model.addAttribute("successMsg", "User has been registered successfully");

model.addAttribute("user", **new** User());

} **catch** (BookAppException e) {

e.printStackTrace();

}

**return** **new** ModelAndView("redirect:/registration");

}

@RequestMapping(value = "/manageBooks", method = RequestMethod.***GET***)

**public** ModelAndView manageBooks() **throws** BookAppException {

ModelAndView modelAndView = **new** ModelAndView();

modelAndView.setViewName("mngBooks");

**return** modelAndView;

}

@RequestMapping(value = "/manageSubjects", method = RequestMethod.***GET***)

**public** ModelAndView manageSubjects() **throws** BookAppException {

ModelAndView modelAndView = **new** ModelAndView();

modelAndView.setViewName("mngSubjects");

**return** modelAndView;

}

@RequestMapping("/exit")

**public** String exitApp() {

**return** "exit";

}

@RequestMapping("/logout")

**public** String logoutApp() {

**return** "logout";

}

@RequestMapping("/accessDenied")

**public** String accessDenied() {

**return** "accessDenied";

}

@RequestMapping("/view")

**public** ModelAndView showAllBooks() {

List<Book> booklist = **null**;

**try** {

booklist = bookAppService.view();

} **catch** (BookAppException e) {

e.printStackTrace();

}

**return** **new** ModelAndView("view", "booklist", booklist);

}

@RequestMapping("/addBook")

**public** ModelAndView showBookForm() {

**return** **new** ModelAndView("addBook", "command", **new** Book());

}

@RequestMapping(value = "/saveBook", method = RequestMethod.***POST***)

**public** ModelAndView saveBook(@ModelAttribute("book") Book book, BindingResult result, Model model) {

**try** {

bookAppService.add(book, book.getSubject().getSubtitle());

} **catch** (BookAppException e) {

e.printStackTrace();

}

**return** **new** ModelAndView("redirect:/view");

}

@RequestMapping("/searchBookForm")

**public** ModelAndView searchBookForm() {

**return** **new** ModelAndView("searchBookForm", "command", **new** Book());

}

@RequestMapping("/searchBook")

**public** ModelAndView searchBook(@ModelAttribute("book") Book book, BindingResult result, Model model) {

**try** {

Book sbook = bookAppService.search(book.getBookId());

model.addAttribute("stitle", sbook.getSubject().getSubtitle());

model.addAttribute("bookId", sbook.getBookId());

model.addAttribute("title", sbook.getTitle());

model.addAttribute("price", sbook.getPrice());

model.addAttribute("volume", sbook.getVolume());

} **catch** (BookAppException e) {

e.printStackTrace();

}

**return** **new** ModelAndView("searchBook");

}

@RequestMapping("/searchSubjectForm")

**public** ModelAndView searchSubjectForm() {

**return** **new** ModelAndView("searchSubjectForm", "command", **new** Subject());

}

@RequestMapping("/searchSubject")

**public** ModelAndView searchSubject(@ModelAttribute("subject") Subject subject, BindingResult result, Model model) {

List<Book> booksbysubject = **new** ArrayList<Book>();

**try** {

Subject subj = bookAppService.searchSubject(subject.getSubtitle());

Set<Book> books = subj.getReferences();

**for** (Book sbook : books) {

booksbysubject.add(sbook);

}

} **catch** (BookAppException e) {

e.printStackTrace();

}

**return** **new** ModelAndView("searchSubject", "booksbysubject", booksbysubject);

}

@RequestMapping("/deleteBook")

**public** ModelAndView deleteBookForm() {

**return** **new** ModelAndView("deleteBook", "command", **new** Book());

}

@RequestMapping("/delBook")

**public** ModelAndView delBook(@ModelAttribute("book") Book book, BindingResult result, Model model) {

List<Book> booklist = **null**;

**try** {

bookAppService.delete(book.getBookId());

booklist = bookAppService.view();

} **catch** (BookAppException e) {

e.printStackTrace();

}

**return** **new** ModelAndView("view", "booklist", booklist);

}

@RequestMapping("/deleteSubject")

**public** ModelAndView deleteSubjectForm() {

**return** **new** ModelAndView("deleteSubject", "command", **new** Subject());

}

@RequestMapping("/delSubject")

**public** ModelAndView delSubject(@ModelAttribute("subject") Subject subject, BindingResult result, Model model) {

List<Book> booklist = **null**;

**try** {

bookAppService.deleteSubject(subject.getSubtitle());

booklist = bookAppService.view();

} **catch** (BookAppException e) {

e.printStackTrace();

}

**return** **new** ModelAndView("view", "booklist", booklist);

}

}

**IUserService.java**

**package** com.books.app.service;

**import** com.books.app.exception.BookAppException;

**import** com.books.app.model.User;

**public** **interface** IUserService {

**public** User findUserByEmail(String email) **throws** BookAppException;

**public** **void** saveUser(User user) **throws** BookAppException;

}

**UserServiceImpl.java**

**package** com.books.app.service;

**import** java.util.Arrays;

**import** java.util.HashSet;

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

**import** org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

**import** org.springframework.transaction.annotation.Transactional;

**import** com.books.app.dao.IRoleRepository;

**import** com.books.app.dao.IUserRepository;

**import** com.books.app.model.Role;

**import** com.books.app.model.User;

**public** **class** UserServiceImpl **implements** IUserService{

@Autowired

**private** IUserRepository userRepository;

@Autowired

**private** IRoleRepository roleRepository;

@Autowired

**private** BCryptPasswordEncoder bCryptPasswordEncoder;

@Override

**public** User findUserByEmail(String email) {

**return** userRepository.findByEmail(email);

}

@Transactional

@Override

**public** **void** saveUser(User user) {

user.setPassword(bCryptPasswordEncoder.encode(user.getPassword()));

user.setActive(1);

Role userRole = roleRepository.findByRolename(user.getUserrole());

user.setRoles(**new** HashSet<Role>(Arrays.*asList*(userRole)));

userRepository.save(user);

}

}

**IBookAppService.java**

**package** com.books.app.service;

**import** java.util.List;

**import** com.books.app.exception.BookAppException;

**import** com.books.app.model.Book;

**import** com.books.app.model.Subject;

**public** **interface** IBookAppService {

String add(Book book, String subtitle) **throws** BookAppException;

**void** delete(**int** bookid) **throws** BookAppException;

**void** deleteSubject(String subtitle) **throws** BookAppException;

Book search(**int** bookid) **throws** BookAppException;

Subject searchSubject(String subtitle) **throws** BookAppException;

List<Book> view() **throws** BookAppException;

}

**BookAppServiceImpl.java**

**package** com.books.app.service;

**import** java.util.List;

**import** java.util.Optional;

**import** java.util.Set;

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

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

**import** org.springframework.transaction.annotation.Transactional;

**import** com.books.app.dao.IBookRepository;

**import** com.books.app.dao.ISubjectRepository;

**import** com.books.app.exception.BookAppException;

**import** com.books.app.model.Book;

**import** com.books.app.model.Subject;

@Service

**public** **class** BookAppServiceImpl **implements** IBookAppService {

@Autowired

**private** ISubjectRepository subjRepo;

@Autowired

**private** IBookRepository bookRepo;

@Transactional

@Override

**public** String add(Book book, String subtitle) **throws** BookAppException {

Subject subject = **null**;

**if** (book.getSubject() != **null**) {

List<Subject> subjectList = subjRepo.findAll();

**for** (Subject subj : subjectList) {

**if** (subj.getSubtitle() != **null** && subj.getSubtitle().equalsIgnoreCase(subtitle)) {

subject = subj;

}

}

**if** (subject == **null**) {

subject = **new** Subject();

subject.setSubtitle(subtitle);

subjRepo.save(subject);

}

}

book.setSubject(subject);

Book sbook = bookRepo.save(book);

**return** sbook.getTitle();

}

@Transactional

@Override

**public** **void** delete(**int** bookid) **throws** BookAppException {

bookRepo.deleteById(bookid);

}

@Transactional

@Override

**public** **void** deleteSubject(String subtitle) **throws** BookAppException {

Subject subj = searchSubject(subtitle);

Set<Book> books = subj.getReferences();

**for** (Book book : books) {

**if**(book != **null**) {

delete(book.getBookId());

}

}

subjRepo.delete(subj);

}

@Override

**public** Book search(**int** bookid) **throws** BookAppException {

Optional<Book> optBook = bookRepo.findById(bookid);

Book book = optBook.isPresent() ? optBook.get() : **null**;

**return** book;

}

@Override

**public** Subject searchSubject(String subtitle) **throws** BookAppException {

Subject subject = **null**;

List<Subject> subjectList = subjRepo.findAll();

**for** (Subject subj : subjectList) {

**if** (subj.getSubtitle() != **null** && subj.getSubtitle().equalsIgnoreCase(subtitle)) {

subject = subj;

}

}

**return** subject;

}

@Override

**public** List<Book> view() **throws** BookAppException{

**return** bookRepo.findAll();

}

}

**IQueryMapper.java**

**package** com.books.app.dao;

**public** **interface** IQueryMapper {

**public** **static** **final** String ***GET\_ALL\_BOOKS\_QRY*** = "FROM Book";

**public** **static** **final** String ***GET\_ALL\_SUBJECTS\_QRY*** = "FROM Subject";

**public** **static** **final** String ***USERS\_QRY*** = "SELECT email, password, active FROM Users WHERE email=?";

**public** **static** **final** String ***ROLES\_QRY*** = "SELECT u.email, r.role\_name FROM Users u INNER JOIN users\_roles ur on(u.user\_id=ur.user\_id) INNER JOIN Roles r on(ur.role\_id=r.role\_id) WHERE u.email=?";

}

**IBookRepository.java**

**package** com.books.app.dao;

**import** org.springframework.data.jpa.repository.JpaRepository;

**import** com.books.app.model.Book;

**public** **interface** IBookRepository **extends** JpaRepository<Book,Integer>{

}

**ISubjectRepository.java**

**package** com.books.app.dao;

**import** org.springframework.data.jpa.repository.JpaRepository;

**import** com.books.app.model.Subject;

**public** **interface** ISubjectRepository **extends** JpaRepository<Subject,String>{

}

**IUserRepository.java**

**package** com.books.app.dao;

**import** org.springframework.data.jpa.repository.JpaRepository;

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

**import** com.books.app.model.User;

@Repository("userRepository")

**public** **interface** IUserRepository **extends** JpaRepository<User, Integer> {

User findByEmail(String email);

}

**IRoleRepository.java**

**package** com.books.app.dao;

**import** org.springframework.data.jpa.repository.JpaRepository;

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

**import** com.books.app.model.Role;

@Repository("roleRepository")

**public** **interface** IRoleRepository **extends** JpaRepository<Role, Integer> {

Role findByRolename(String rolename);

}

**BookAppException.java**

**package** com.books.app.exception;

@SuppressWarnings("serial")

**public** **class** BookAppException **extends** Exception {

**public** BookAppException(String errMsg) {

**super**(errMsg);

}

}

**Subject.java**

**package** com.books.app.model;

**import** java.util.Set;

**import** javax.persistence.Column;

**import** javax.persistence.Entity;

**import** javax.persistence.FetchType;

**import** javax.persistence.Id;

**import** javax.persistence.OneToMany;

**import** javax.persistence.Table;

@Entity

@Table(name="subjects")

**public** **class** Subject {

@Id

@Column(name="subtitle")

**private** String subtitle;

@Column(name="subjectid")

**private** **long** subjectId;

@Column(name="durationInHours")

**private** **int** durationInHours;

@OneToMany(fetch = FetchType.***EAGER***,mappedBy="subject",orphanRemoval=**true**)

**private** Set<Book> references;

**public** **long** getSubjectId() {

**return** subjectId;

}

**public** **void** setSubjectId(**long** subjectId) {

**this**.subjectId = subjectId;

}

**public** String getSubtitle() {

**return** subtitle;

}

**public** **void** setSubtitle(String subtitle) {

**this**.subtitle = subtitle;

}

**public** **int** getDurationInHours() {

**return** durationInHours;

}

**public** **void** setDurationInHours(**int** durationInHours) {

**this**.durationInHours = durationInHours;

}

**public** Set<Book> getReferences() {

**return** references;

}

**public** **void** setReferences(Set<Book> references) {

**this**.references = references;

}

}

**Book.java**

**package** com.books.app.model;

**import** java.time.LocalDate;

**import** javax.persistence.Column;

**import** javax.persistence.Entity;

**import** javax.persistence.FetchType;

**import** javax.persistence.Id;

**import** javax.persistence.JoinColumn;

**import** javax.persistence.ManyToOne;

**import** javax.persistence.Table;

**import** org.springframework.format.annotation.DateTimeFormat;

@Entity

@Table(name="books")

**public** **class** Book {

@Id

@Column(name="bookid")

**private** **int** bookId;

@Column(name="title")

**private** String title;

@Column(name="price")

**private** **double** price;

@Column(name="vol")

**private** **int** volume;

@Column(name="pdate")

@DateTimeFormat(pattern ="dd-MM-yyyy")

**private** LocalDate publistDt;

@ManyToOne(fetch=FetchType.***EAGER***)

@JoinColumn(name="stitle" , nullable=**false**)

**private** Subject subject;

**public** Subject getSubject() {

**return** subject;

}

**public** **void** setSubject(Subject subject) {

**this**.subject = subject;

}

**public** **int** getBookId() {

**return** bookId;

}

**public** **void** setBookId(**int** bookId) {

**this**.bookId = bookId;

}

**public** String getTitle() {

**return** title;

}

**public** **void** setTitle(String title) {

**this**.title = title;

}

**public** **double** getPrice() {

**return** price;

}

**public** **void** setPrice(**double** price) {

**this**.price = price;

}

**public** **int** getVolume() {

**return** volume;

}

**public** **void** setVolume(**int** volume) {

**this**.volume = volume;

}

**public** LocalDate getPublistDt() {

**return** publistDt;

}

**public** **void** setPublistDt(LocalDate publistDt) {

**this**.publistDt = publistDt;

}

}

**Role.java**

**package** com.books.app.model;

**import** javax.persistence.Column;

**import** javax.persistence.Entity;

**import** javax.persistence.Id;

**import** javax.persistence.GeneratedValue;

**import** javax.persistence.GenerationType;

**import** javax.persistence.Table;

@Entity

@Table(name="roles")

**public** **class** Role {

@Id

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

@Column(name="role\_id")

**private** **int** id;

@Column(name="role\_name")

**private** String rolename;

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getRolename() {

**return** rolename;

}

**public** **void** setRolename(String rolename) {

**this**.rolename = rolename;

}

}

**User.java**

**package** com.books.app.model;

**import** java.util.Set;

**import** javax.persistence.CascadeType;

**import** javax.persistence.Column;

**import** javax.persistence.Entity;

**import** javax.persistence.GeneratedValue;

**import** javax.persistence.GenerationType;

**import** javax.persistence.Id;

**import** javax.persistence.JoinColumn;

**import** javax.persistence.JoinTable;

**import** javax.persistence.ManyToMany;

**import** javax.persistence.Table;

@Entity

@Table(name="users")

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

@Id

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

@Column(name="user\_id")

**private** **int** id;

@Column(name="email")

**private** String email;

@Column(name="password")

**private** String password;

@Column(name="first\_name")

**private** String firstname;

@Column(name="last\_name")

**private** String lastname;

@Column(name="active")

**private** **int** active;

@Column(name="user\_role")

**private** String userrole;

@ManyToMany(cascade=CascadeType.***ALL***)

@JoinTable(name="users\_roles",

joinColumns = @JoinColumn(name = "user\_id"),

inverseJoinColumns = @JoinColumn(name = "role\_id"))

**private** Set<Role> roles;

**public** String getUserrole() {

**return** userrole;

}

**public** **void** setUserrole(String userrole) {

**this**.userrole = userrole;

}

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getEmail() {

**return** email;

}

**public** **void** setEmail(String email) {

**this**.email = email;

}

**public** String getPassword() {

**return** password;

}

**public** **void** setPassword(String password) {

**this**.password = password;

}

**public** String getFirstname() {

**return** firstname;

}

**public** **void** setFirstname(String firstname) {

**this**.firstname = firstname;

}

**public** String getLastname() {

**return** lastname;

}

**public** **void** setLastname(String lastname) {

**this**.lastname = lastname;

}

**public** **int** getActive() {

**return** active;

}

**public** **void** setActive(**int** active) {

**this**.active = active;

}

**public** Set<Role> getRoles() {

**return** roles;

}

**public** **void** setRoles(Set<Role> roles) {

**this**.roles = roles;

}

}

**registration.jsp**

<%@page contentType=*"text/html;charset=UTF-8"* language=*"java"* %>

<%@page isELIgnored=*"false"* %>

<%@ taglib prefix=*"form"* uri=*"http://www.springframework.org/tags/form"*%>

<html>

<body>

<h2>Registration Page</h2>

<form:form method=*"POST"* action=*"registration"*>

<br>${successMsg}<br>

First name : <form:input type=*"text"* path=*"firstname"*></form:input><br>

Last name : <form:input type=*"text"* path=*"lastname"*></form:input><br>

Email : <form:input type=*"text"* path=*"email"*></form:input><br>

Password : <form:input type=*"password"* path=*"password"*></form:input><br>

Role : <form:input type=*"text"* path=*"userrole"*></form:input><br>

<input type=*"submit"* value=*"Register user"*></input><br>

</form:form>

<a href=*""*>Go back to Login page</a><br>

</body>

</html>

**home.jsp**

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<%@ taglib prefix=*"form"* uri=*"http://www.springframework.org/tags/form"*%>

<html>

<body>

<h2>Welcome to Book Shop application</h2>

<a href=*"manageBooks"*>Manage Books</a><br>

<a href=*"manageSubjects"*>Manage Subjects</a><br><br>

<a href=*"logout"*>Logout</a><br>

</body>

</html>

**login.jsp**

<%@page contentType=*"text/html;charset=UTF-8"* language=*"java"* %>

<%@page isELIgnored=*"false"* %>

<%@ taglib prefix=*"form"* uri=*"http://www.springframework.org/tags/form"*%>

<html>

<body>

<h2>Login Page</h2>

<form:form method=*"POST"* action=*"home"*>

<br>${errMsg}<br>

Email : <form:input type=*"text"* path=*"email"*></form:input><br>

Password : <form:input type=*"password"* path=*"password"*></form:input><br>

<input type=*"submit"* value=*"Login"*></input>

<br><br>

<a href=*"registration"*>Click here to Register</a><br>

</form:form>

</body>

</html>

**logout.jsp**

<%@ taglib prefix=*"form"* uri=*"http://www.springframework.org/tags/form"*%>

<html>

<body>

<h2>You have logged out successfully!!</h2>

<br><br>

<a href=*""*>Go back to Login page</a><br>

</body>

</html>

**mngBooks.jsp**

<%@page contentType=*"text/html;charset=UTF-8"* language=*"java"* %>

<%@page isELIgnored=*"false"* %>

<%@ taglib uri=*"http://www.springframework.org/tags/form"* prefix=*"form"*%>

<%@ taglib uri=*"http://java.sun.com/jsp/jstl/core"* prefix=*"c"*%>

<%@ taglib uri=*"http://java.sun.com/jsp/jstl/fmt"* prefix=*"fmt"* %>

<html>

<body>

<br>${logUser}<br>

<h2>Book Shop Application</h2>

<a href=*"view"*>View all books</a><br>

<a href=*"addBook"*>Add Book</a><br>

<a href=*"searchBookForm"*>Search Book</a><br>

<a href=*"deleteBook"*>Delete Book</a><br>

<a href=*"exit"*>Exit</a><br><br><br>

<a href=*"logout"*>Logout</a><br>

</body>

</html>

**mngSubjects.jsp**

<%@page contentType=*"text/html;charset=UTF-8"* language=*"java"* %>

<%@page isELIgnored=*"false"* %>

<%@ taglib uri=*"http://www.springframework.org/tags/form"* prefix=*"form"*%>

<%@ taglib uri=*"http://java.sun.com/jsp/jstl/core"* prefix=*"c"*%>

<%@ taglib uri=*"http://java.sun.com/jsp/jstl/fmt"* prefix=*"fmt"* %>

<html>

<body>

<br>${logUser}<br>

<h2>Book Shop Application</h2>

<a href=*"searchSubjectForm"*>Search Subject</a><br>

<a href=*"deleteSubject"*>Delete Subject</a><br>

<a href=*"exit"*>Exit</a><br><br><br>

<a href=*"logout"*>Logout</a><br>

</body>

</html>

**accessDenied.jsp**

<%@ taglib prefix=*"form"* uri=*"http://www.springframework.org/tags/form"*%>

<html>

<body>

<h2>You do not have access to this link</h2>

<br><br>

<a href=*"home"*>Go back to Home page</a><br>

</body>

</html>

**addBook.jsp**

<%@ taglib prefix=*"form"* uri=*"http://www.springframework.org/tags/form"*%>

<html>

<body>

<h2>Add Book</h2>

<form:form method=*"POST"* action=*"saveBook"*>

Enter Book ID : <form:input type=*"text"* path=*"bookId"*></form:input>

<br>

Enter Book title : <form:input type=*"text"* path=*"title"*></form:input>

<br>

Enter Book price : <form:input type=*"text"* path=*"price"*></form:input>

<br>

Enter Book volume : <form:input type=*"text"* path=*"volume"*></form:input>

<br>

Enter Book published Date : <form:input type=*"text"* path=*"publistDt"*></form:input>

<br>

Enter Subject title : <form:input type=*"text"* path=*"subject.subtitle"*></form:input>

<br>

<input type=*"submit"* value=*"Add Book"*></input>

</form:form>

</body>

</html>

**deleteBook.jsp**

<%@ taglib prefix=*"form"* uri=*"http://www.springframework.org/tags/form"*%>

<html>

<body>

<h2>Delete Book</h2>

<form:form method=*"POST"* action=*"delBook"*>

Enter Book Id : <form:input type=*"text"* path=*"bookId"*></form:input>

<input type=*"submit"* value=*"Delete Book"*></input>

</form:form>

<br><br>

<a href=*""*>Go back to menu</a><br>

</body>

</html>

**deleteSubject.jsp**

<%@ taglib prefix=*"form"* uri=*"http://www.springframework.org/tags/form"*%>

<html>

<body>

<h2>Delete Subject</h2>

<form:form method=*"POST"* action=*"delSubject"*>

Enter Subject title : <form:input type=*"text"* path=*"subtitle"*></form:input>

<input type=*"submit"* value=*"Delete Subject"*></input>

</form:form>

<br><br>

<a href=*""*>Go back to menu</a><br>

</body>

</html>

**searchBookForm.jsp**

<%@ taglib prefix=*"form"* uri=*"http://www.springframework.org/tags/form"*%>

<html>

<body>

<h2>Search Book</h2>

<form:form method=*"POST"* action=*"searchBook"*>

Enter Book Id : <form:input type=*"text"* path=*"bookId"*></form:input>

<input type=*"submit"* value=*"Search Book"*></input>

</form:form>

<br><br>

<a href=*""*>Go back to menu</a><br>

</body>

</html>

**searchBook.jsp**

<html>

<body>

<h2>Book search results </h2>

<table border=*"1"* width=*"70%"* cellpadding=*"2"*>

<tr>

<th>Subject title</th>

<th>Book id</th>

<th>Book title</th>

<th>Book price</th>

<th>Book volume</th>

</tr>

<tr>

<td>${stitle}</td>

<td>${bookId}</td>

<td>${title}</td>

<td>${price}</td>

<td>${volume}</td>

</tr>

</table>

<br><br>

<a href=*""*>Go back to menu</a><br>

</body>

</html>

**searchSubjectForm.jsp**

<%@ taglib prefix=*"form"* uri=*"http://www.springframework.org/tags/form"*%>

<html>

<body>

<h2>Search Subject</h2>

<form:form method=*"POST"* action=*"searchSubject"*>

Enter Subject title : <form:input type=*"text"* path=*"subtitle"*></form:input>

<input type=*"submit"* value=*"Search Subject"*></input>

</form:form>

<br><br>

<a href=*""*>Go back to menu</a><br>

</body>

</html>

**searchSubject.jsp**

<%@ taglib uri=*"http://www.springframework.org/tags/form"* prefix=*"form"*%>

<%@ taglib uri=*"http://java.sun.com/jsp/jstl/core"* prefix=*"c"*%>

<%@ taglib uri=*"http://java.sun.com/jsp/jstl/fmt"* prefix=*"fmt"* %>

<a href=*""*>Go back to menu</a><br>

<h2>Subject search results</h2>

<table border=*"1"* width=*"70%"* cellpadding=*"2"*>

<tr>

<th>Subject title</th>

<th>Book id</th>

<th>Book title</th>

<th>Book price</th>

<th>Book volume</th>

</tr>

<c:forEach var=*"book"* items=*"*${booksbysubject}*"*>

<tr>

<td>${book.subject.subtitle}</td>

<td>${book.bookId}</td>

<td>${book.title}</td>

<td>${book.price}</td>

<td>${book.volume}</td>

</tr>

</c:forEach>

</table>

**view.jsp**

<%@ taglib uri=*"http://www.springframework.org/tags/form"* prefix=*"form"*%>

<%@ taglib uri=*"http://java.sun.com/jsp/jstl/core"* prefix=*"c"*%>

<%@ taglib uri=*"http://java.sun.com/jsp/jstl/fmt"* prefix=*"fmt"* %>

<a href=*""*>Go back to menu</a><br>

<h2>Books List</h2>

<table border=*"1"* width=*"70%"* cellpadding=*"2"*>

<tr>

<th>Subject title</th>

<th>Book id</th>

<th>Book title</th>

<th>Book price</th>

<th>Book volume</th>

</tr>

<c:forEach var=*"book"* items=*"*${booklist}*"*>

<tr>

<td>${book.subject.subtitle}</td>

<td>${book.bookId}</td>

<td>${book.title}</td>

<td>${book.price}</td>

<td>${book.volume}</td>

</tr>

</c:forEach>

</table>

**exit.jsp**

<html>

<h2>Thanks for Shopping !!</h2>

</html>



















