10\_Spring\_Assignment5

**Code snippets:**

**Model classes:**

Book.Java:

package com.mcnz.rps.smvc;

import java.time.LocalDate;

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.ManyToOne;

import javax.persistence.Table;

@Entity

@Table(name = "book")

public class Book {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

@Column(name="book\_id")

private int bookId;

private String title;

private double price;

private Integer volume;

private LocalDate publishDate;

@ManyToOne(cascade = CascadeType.ALL)

@JoinColumn(name = "subject\_id")

private Subject subjects;

public Subject getSubjects() {

return subjects;

}

public void setSubjects(Subject subjects) {

this.subjects = subjects;

}

@Override

public String toString() {

return "Book [bookId=" + bookId + ", title=" + title + ", price="

+ price + ", volume=" + volume + ", publishDate=" + publishDate

+ "]";

}

public Book(String title, double price, Integer volume,

LocalDate publishDate, Subject subjects) {

super();

this.title = title;

this.price = price;

this.volume = volume;

this.publishDate = publishDate;

this.subjects = subjects;

}

public Book(String title, double price, Integer volume,

LocalDate publishDate) {

super();

this.title = title;

this.price = price;

this.volume = volume;

this.publishDate = publishDate;

}

public Book() {

}

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 Integer getVolume() {

return volume;

}

public void setVolume(Integer volume) {

this.volume = volume;

}

public LocalDate getPublishDate() {

return publishDate;

}

public void setPublishDate(LocalDate publishDate) {

this.publishDate = publishDate;

}

@Override

public int hashCode() {

final int prime = 31;

int result = 1;

result = prime \* result + bookId;

return result;

}

@Override

public boolean equals(Object obj) {

if (this == obj)

return true;

if (obj == null)

return false;

if (getClass() != obj.getClass())

return false;

Book other = (Book) obj;

if (bookId != other.bookId)

return false;

return true;

}

}

**Subject.Java:**

**package** com.mcnz.rps.smvc;

**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.OneToMany;

**import** javax.persistence.Table;

@Entity

@Table(name = "subject")

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

@Id

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

@Column(name="subject\_id")

**long** subjectId;

@Column(name="subtitle")

String subtitle;

@Column(name="duration\_in\_hours")

**int** durationInHours;

@OneToMany(mappedBy = "subjects", cascade = CascadeType.***ALL***)

Set<Book> references;

**public** Subject() {

}

**public** Subject(String subtitle, **int** durationInHours) {

**this**.subtitle = subtitle;

**this**.durationInHours = durationInHours;

}

@Override

**public** String toString() {

**return** "Subject [subjectId=" + subjectId + ", subtitle=" + subtitle

+ ", durationInHours=" + durationInHours + ", references="

+ 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;

}

}

**Repositories:**

**BookRepository with custom query:**

package com.mcnz.rps.smvc;

import java.util.List;

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

import org.springframework.data.jpa.repository.Query;

import org.springframework.data.repository.query.Param;

import org.springframework.stereotype.Repository;

@Repository

public interface BookRepository extends JpaRepository<Book, Integer> {

@Query("SELECT b FROM Book b WHERE LOWER(b.title) = LOWER(:title)")

public List<Book> findBookByTitle(@Param("title") String title);

}

**Subject Repository with custom quey:**

**package** com.mcnz.rps.smvc;

**import** java.util.List;

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

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

**import** org.springframework.data.repository.query.Param;

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

@Repository

**public** **interface** SubjectRepository **extends** JpaRepository<Subject, Long> {

@Query("SELECT s FROM Subject s WHERE LOWER(s.subtitle) = LOWER(:subtitle)")

**public** List<Subject> findSubjectBySubtitle(@Param("subtitle") String subtitle);

}

**Controller:**

package com.mcnz.rps.smvc;

import java.time.LocalDate;

import java.util.Arrays;

import java.util.HashSet;

import java.util.Iterator;

import java.util.Optional;

import java.util.Set;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.validation.Valid;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.validation.BindingResult;

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

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

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

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.ModelAndView;

@Controller

public class WebController {

@Autowired

SubjectRepository subRepository;

@Autowired

BookRepository bookrepo;

@GetMapping ("/playagame")

public String playGame(

@RequestParam(name="choice", required=false)

String theChoice,

Model model) {

if (theChoice == null) {

return "index";

}

String theOutcome = "error";

if (theChoice.equals("rock")) {

theOutcome = "tie";

}

if (theChoice.equals("paper")) {

theOutcome = "win";

}

if (theChoice.equals("scissors")) {

theOutcome = "loss";

}

model.addAttribute("outcome", theOutcome);

return "results";

}

@GetMapping(value = "/savebook")

public ModelAndView saveBook(HttpServletRequest request, HttpServletResponse response) {

System.out.println("................. Get ..........");

ModelAndView mav = new ModelAndView("savebook");

mav.addObject("subject", new Subject());

return mav;

}

@PostMapping("/savebookprocess")

public String addUser(@Valid Subject subject, BindingResult result, Model model) {

if (result.hasErrors()) {

return "savebook";

}

Set<Book> bookA = new HashSet();

bookA.add(new Book("physics", 150, 1, LocalDate.now(),subject));

bookA.add(new Book("material physics", 250, 1, LocalDate.now(),subject));

bookA.add(new Book("solid physics", 350, 1, LocalDate.now(),subject));

subject.setReferences(bookA);

subRepository.save(subject);

model.addAttribute("subjects", subRepository.findAll());

model.addAttribute("success", "Successfully added");

return "listsubjects";

}

@GetMapping ("/getall")

public String getAllSubjects(Model model) {

for (Subject subject : subRepository.findAll()) {

System.out.println("Subject-->" + subject);

}

model.addAttribute("subjects", subRepository.findAll());

//Optional<Subject> subject=subRepository.findById(2L);

//System.out.println("subject........................->"+subject.toString());

//model.addAttribute("subject", subject);

return "listsubjects";

}

@GetMapping ("/getsubjectbyid/{subjectId}")

public String getSubjectById(@PathVariable("subjectId") Long subjectId,Model model) {

Optional<Subject> subject=subRepository.findById(subjectId);

System.out.println("subject........................->"+subject.toString());

model.addAttribute("subject", subject);

return "subject";

}

@GetMapping ("/getbookbyid/{bookId}")

public String getBookById(@PathVariable("bookId") Integer bookId,Model model) {

Optional<Book> book=bookrepo.findById(bookId);

System.out.println("book........................->"+book.toString());

model.addAttribute("book", book);

return "subject";

}

@GetMapping ("/getbooksbysubjectid/{subjectId}")

public String getBooksBySubjectId(@PathVariable("subjectId") Long subjectId,Model model) {

Optional<Subject> subject=subRepository.findById(subjectId);

System.out.println("subject->"+subject.toString());

Set<Book> bookset=subject.get().getReferences();

Iterator<Book> bookitr=bookset.iterator();

while(bookitr.hasNext()){

Book b1=bookitr.next();

System.out.println("Book------------>"+b1.toString());

}

model.addAttribute("books", bookset);

return "listbooks";

}

@GetMapping ("/getsubjectbysubtitle/{subtitle}")

**public** String getfindSubjectBySubtitle(@PathVariable("subtitle")String subtitle,Model model){

System.***out***.println("...........getfindSubjectBySubtitle..................................");

List<Subject> subjects=subRepository.findSubjectBySubtitle("chemistry");

System.***out***.println("subjectList"+subjects);

model.addAttribute("subjects", subjects);

**return** "listsubjects";

}

@GetMapping ("/getbookbytitle/{title}")

**public** String getfindBookByTitle(@PathVariable("title")String title,Model model){

System.***out***.println("...........getfindBookByTitle..................................");

List<Book> bookList=bookrepo.findBookByTitle("physics");

System.***out***.println("bookList"+bookList);

model.addAttribute("books", bookList);

**return** "listbooks";

}

@GetMapping ("/deletesubject/{subjectId}")

public String deleteSubjectById(@PathVariable("subjectId") Long subjectId,Model model) {

Optional<Subject> subject=subRepository.findById(subjectId);

System.out.println("subject->"+subject.toString());

Set<Book> bookset=subject.get().getReferences();

Iterator<Book> bookitr=bookset.iterator();

while(bookitr.hasNext()){

Book b1=bookitr.next();

System.out.println("Book------------>"+b1.toString());

bookrepo.delete(b1);

}

return "redirect:/getall";

}

}

Spring boot test class:

**package** com.mcnz.rps.smvc;

**import** java.time.LocalDate;

**import** java.util.Arrays;

**import** java.util.HashSet;

**import** java.util.Iterator;

**import** java.util.Optional;

**import** java.util.Set;

**import** javax.transaction.Transactional;

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

**import** org.springframework.boot.CommandLineRunner;

**import** org.springframework.boot.SpringApplication;

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

@SpringBootApplication

**public** **class** SpringMvcRpsApplication {//implements CommandLineRunner{

@Autowired

SubjectRepository subRepository;

@Autowired

BookRepository bookrepo;

**public** **static** **void** main(String[] args) {

SpringApplication.*run*(SpringMvcRpsApplication.**class**, args);

}

}

Database connection:---

**Application.properties:**

# DATASOURCE (DataSourceAutoConfiguration & DataSourceProperties)

spring.datasource.url=jdbc:mysql://localhost:3306/demoabc?useSSL=false

spring.datasource.username=root

spring.datasource.password=password-123

# Hibernate

# The SQL dialect makes Hibernate generate better SQL for the chosen database

spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5InnoDBDialect

# Hibernate ddl auto (create, create-drop, validate, update)

spring.jpa.hibernate.ddl-auto = update

logging.level.org.hibernate.SQL=DEBUG

logging.level.org.hibernate.type=TRACE

pom.xml:

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

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

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

<modelVersion>4.0.0</modelVersion>

<groupId>com.mcnz.rps.spring</groupId>

<artifactId>roshambo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>spring-mvc-rps</name>

<description>rps</description>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.0.3.RELEASE</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<properties>

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

<project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-thymeleaf</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

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

</dependency>

<dependency>

<groupId>mysql</groupId>

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

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

</dependency>

<!-- Need this to compile JSP -->

<dependency>

<groupId>org.apache.tomcat.embed</groupId>

<artifactId>tomcat-embed-jasper</artifactId>

<scope>provided</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

**Templates:**

listbooks.html:

<html xmlns:th=*"http://www.thymeleaf.org"*>

<head>

<link rel=*"stylesheet"* href=*"test.css"*>

</head>

<h2>Subjects</h2>

<table>

<thead>

<tr>

<th>Id</th>

<th>Title</th>

<th>Volume</th>

<th>price</th>

<th>Publish Date</th>

<th>Edit</th>

<th>Delete</th>

</tr>

</thead>

<tbody>

<tr th:each=*"book : ${books}"*>

<td th:text=*"${book.bookId}"*></td>

<td th:text=*"${book.title}"*></td>

<td th:text=*"${book.volume}"*></td>

<td th:text=*"${book.price}"*></td>

<td th:text=*"${book.publishDate}"*></td>

<td><a th:href=*"@{/editbook/{id}(id=${book.bookId})}"*>Edit</a></td>

<td><a th:href=*"@{/deletebook/{id}(id=${book.bookId})}"*>Delete</a></td>

</tr>

</tbody>

</table>

</div>

<p><a href=*"/savebook"*>Add a new subject</a></p>

<p><a href=*"/getall"*>All subjects</a></p>

</div>

</html>

**listsubjects.html:**

<html xmlns:th=*"http://www.thymeleaf.org"*>

<head>

<link rel=*"stylesheet"* href=*"test.css"*>

</head>

<h2>Subjects</h2>

<table>

<thead>

<tr>

<th>Subtitle</th>

<th>Duration In Hours</th>

<th>Edit</th>

<th>Delete</th>

</tr>

</thead>

<tbody>

<tr th:each=*"subject : ${subjects}"*>

<td th:text=*"${subject.subtitle}"*></td>

<td th:text=*"${subject.durationInHours}"*></td>

<td><a th:href=*"@{/editsubject/{id}(id=${subject.subjectId})}"*>Edit</a></td>

<td><a th:href=*"@{/deletesubject/{id}(id=${subject.subjectId})}"*>Delete</a></td>

<td><a th:href=*"@{/getbooksbysubjectid/{id}(id=${subject.subjectId})}"*>Get All Books</a></td>

</tr>

</tbody>

</table>

</div>

<p><a href=*"/savebook"*>Add a new subject</a></p>

</div>

</html>

Savebook.html:

<html lang=*"en"* xmlns:th=*"http://www.thymeleaf.org"*>

<body>

<form action=*"#"* th:action=*"@{/savebookprocess}"* th:object=*"${subject}"* method=*"post"*>

<label for=*"subtitle"*>Subtitle</label>

<input type=*"text"* th:field=*"\*{subtitle}"* id=*"subtitle"* placeholder=*"Name"*/>

<span th:if=*"${#fields.hasErrors('subtitle')}"* th:errors=*"\*{subtitle}"*></span>

<label for=*"durationInHours"*>Duration In Hours</label>

<input type=*"text"* th:field=*"\*{durationInHours}"* id=*"durationInHours"* placeholder=*"durationInHours"*/>

<span th:if=*"${#fields.hasErrors('durationInHours')}"* th:errors=*"\*{durationInHours}"*></span>

<input type=*"Submit"* value=*"Add Subject"*/>

</form>

</body>

</html>>

**Subject.html:**

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

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

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

<%@ page session="false" %>

<html>

<head>

<title>Subject Details</title>

<style type=*"text/css"*>

*.tg* {border-collapse:*collapse*;border-spacing:*0*;border-color:*#ccc*;}

*.tg* **td**{font-family:*Arial, sans-serif*;font-size:*14px*;padding:*10px 5px*;border-style:*solid*;border-width:*1px*;overflow:*hidden*;word-break:*normal*;border-color:*#ccc*;color:*#333*;background-color:*#fff*;}

*.tg* **th**{font-family:*Arial, sans-serif*;font-size:*14px*;font-weight:*normal*;padding:*10px 5px*;border-style:*solid*;border-width:*1px*;overflow:*hidden*;word-break:*normal*;border-color:*#ccc*;color:*#333*;background-color:*#f0f0f0*;}

*.tg* *.tg-4eph*{background-color:*#f9f9f9*}

</style>

</head>

<body>

<br>

<h3>Subject List</h3>

<c:if test=*"${!empty references}"*>

<table class=*"tg"*>

<tr>

<th width=*"80"*>Subject ID</th>

<th width=*"120"*>Subject subtitle</th>

</tr>

<c:forEach items=*"${references}"* var=*"subject"*>

<tr>

<td>${subject.subjectId}</td>

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

</tr>

</c:forEach>

</table>

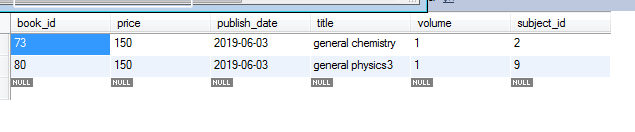
</c:if>

</body>

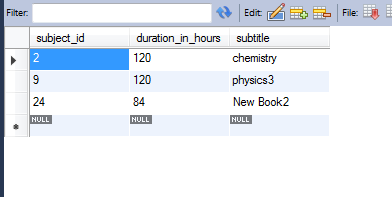
</html>

Output Screenshots:

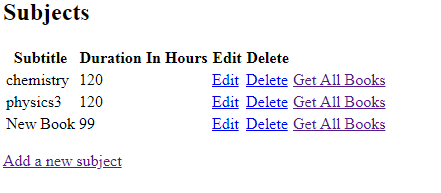
Book Table:



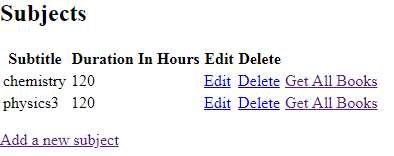
Subject Table:



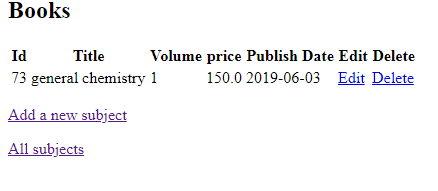
**Search a Subject:**



Delete a subject:



Get a Book:



Search subjects and books by title:

