Exam Coversheet

To be completed by the student:

Student number:

_____

Name:

_____

Group/Class:

_____

2412mm132A

To be completed by the lecturer

Exam Name	Java Fundamentals
Exam Code	1915IN248Z
Exam Date	12-06-2023
Time	12:00
Classroom	Proctored online
Length (in minutes)	120
Academic Year	2022-2023
Education Period	2.4
Opportunity/Chance	1
Examiners	Wim Wiltenburg Mark de Haan
Faculty	Engineering, Design and Computing
Cluster	ICT
Programme + full time/part time	Information Technology fulltime
Location	Haarlem
Number of pages (incl. coversheet)	5
Number of questions	10
Passmark	55
Allowed tools/aids	Computer, Eclipse/IntelliJ/Internet, Pen and Paper
Answer sheet	
Additional comments/details	Online communication means not allowed

(* delete if not applicable)

In te vullen door de toetsorganisatie:

Surveillanten:

Bijzondere voorzieningen:

Voorziening:	Aantal:	Opmerking:
Dyslexie		
A3		
Overig		

Exam Java Advanced

A Course Rating API

Introduction

For this exam, you will have to develop an application using Spring Boot. Your task is to create a simplified Study Course Review System using Java Spring Boot. The system should be able to create, read, update, and delete (CRUD) course entries. allow a student to review a course and provide statistics on course ratings.

The exam has been divided into parts with parts with assignments. Individual assignments can be skipped or made in a different order than given, if the student wishes to do so. We strongly advise reading through the exam before starting the assignments.

Requirements

The code you submit needs to meet the following requirements:

1. The code adheres strictly to specifications in the assignments
2. Coding standards apply:
    a. Java Naming Conventions
    b. Each statement is on a separate line
    c. No more than 30 lines to a method
    d. Use comments to explain the what and why of your code, but keep it brief
    e. Use proper formatting (use your IDE to do so)
    f. Use packages to logically group your classes.
3. Write your own code. Copying and pasting is not allowed
4. The application must start up.

Grading

The assignments are graded according to the following specification:

Assignment	**Points**
1.1	5
1.2	10
1.3	5
1.4	10
2.1	15
2.2	15
3.1	5
3.2	15
4.1	15
5.1	5
Total	100

Part 1: Setup and Data Persistence

1. Use Spring Initializr to set up a new Spring Boot project with:

   - Project type: Maven
   - Language: Java
   - Spring Boot version: 3.1.0
   - Group name: nl.inholland.exam
   - Artifact/name: your first name (no spaces, use only lowercase letters a-z)
   - Packaging: Jar
   - Java version: 17
   - Dependencies Spring Web, Spring Data JPA, H2 database, Lombok).

2. Create a the following entity classes:

   Course

   - long id (auto generated)
   - String title
   - String description
   - List<Review> reviews (One to many, JsonManagedReference)

   Review

   - long id (auto generated)
   - int rating
   - int studentNumber
   - String comment
   - LocalDate date
   - Course course (Many to one, JsonBackReference)

   Use Lombok to generate getters, setters and constructors for these model classes.

3. Add Spring Data JPA repository interfaces (CourseRepository and ReviewRepository).

4. Add Service classes (CourseService and ReviewService). Add the repositories as injected dependencies and use them to create methods to provide these database interactions:

   - get a list of courses
   - get a single course by its id, including the list of reviews
   - add a course
   - update a course
   - add a review

Part 2: REST Functionality

1. Add controller classes for both entities to expose the CRUD operations implemented in Part 1.4 as a RESTful API with GET, POST and PUT operations.

   The endpoints for courses should be accessible under:
   - :8080/courses (GET),
   - :8080/courses/{id} (GET, POST & PUT)

   The posted and returned json fields for courses should match the field names specified in 1.2. Ensure that a course can be created without having to provide reviews.

   Use a DTO class to enable posting to :8080/reviews. The posted JSON should contain the following fields:
   - courseId
   - rating
   - studentNumber
   - comment

   The date should be filled in server side and the review should be linked to the course with the given courseId.

   Test your code, and make sure it returns the proper JSON.

2. Ensure that it is not possible to post more than one rating for a specific course, with the same student number. To achieve this, implement a **named query** in the ReviewRepository that checks if a review for a specific course and a specific student number already exists. Then use this query in the ReviewService to check before saving the rating.

Part 3: Stream API

1. Add a field averageRating of type double to the Course class. Make sure this field is not stored in the database (it should be a **@transient** field).
2. In the CourseService method where the courses are retrieved, use the **Stream API** and lambda expressions to set the averageRating value for each course. Ensure the averageRating is now returned when courses are retrieved through the endpoint at :8080/courses.

Part 4: Security

1. Implement a request filter for an API key. You can inherit your filter class from the OncePerRequestFilter class.

   Only requests with a valid API key should be able to access your API. For this exam, only the key '12345'is accepted. This value can be hardcoded in the filter.

   The key is sent with the request in the HTTP Authorization header and includes "Bearer " before the token. The exact header is: `Authorization: Bearer 12345`

   Any request that does not contain this header with the specified value, should receive a HTTP 403 response.

Part 5: Assignment completion

1. Create a zip archive from the root of the project, and make sure that it the file only contains:

- /src folder and all contents
- pom.xml

The name of your zip archive is the same as your artifactId suffixed with ".zip", e.g. john.zip. Turning in a different kind of archive (.rar, .7z) will lead to a point deduction.

Do not add files like:

- /target folder and contents
- /.idea folder and contents
- /.mvn folder and content
- .iml files
- mvnw and mvnw.cmd

Upload your zip archive to Moodle. This completes the exam.