# SOEN 487 – Winter 2020
# Assignment-I

6% of the Final Grade

Due Date: February 14, 2020 – 11:59 PM
Group Assignment: A maximum of three per team

## REST Hello World

### Objectives

The purpose of this assignment is to create a RESTful web service using java.

### Overview

In this assignment you are implementing a RESTful Web Service for a library system. The library system maintains the information for books. The system maintains the following information for each book:

- title
- description
- isbn
- author
- publisher

Assume each book has one author and one publisher. Use an integer id as the primary key for the book entries.

### Project Outline

This assignment consists of five tasks as listed in the following sections to be implemented in four projects:

- The Library Core
  A java class library containing the common classes / data structures that are shared between the client and the service
- The Library System
  A java class library that that implements the library system (The Business)
- The Library Service
  A java Web Application implementing RESTful web service that uses the above library system.
- The Library Client
  A java Console application that enables user to call the web service and display the information.

### Task 1. The Library Core

Create the library core project and include common data structures (i.e. Entity Class(es)) that are shared between client and the service in here.
No business classes to be implemented here.

## Task 2. The Library System

Design and implement a class called Library that takes care of storing and modifying the library data. In this assignment, the data is held in memory. As a result, once the service is shutdown, the data is lost. Beware of concurrency.
Implement standard CRUD operations. Reference Library Core in task 1.

## Task 3. The Library Service

Using the class library created in task 2, implement a RESTful service to provide the following functionalities:

- list (shows the list of current books, id and title)
- get book (returns the book info by the id)
- add book (adds a new book to the system)
- update book (updates specific book, id and the data to be given as arguments)
- delete book (deletes a book by id)

Create and implement the Library RESTful Web Service that implements all four verbs:

**GET**, **POST**, **PUT**, and **DELETE**.

Use appropriate REST method for CRUD operations. Use annotation for parameters.
You may use String data type for all the attributes, and integer for the book id.
Use **plain text** MIME type to return any data to the user. Do not use xml or json.
You may implement toString() method for the entity class(es) in task 1.

## Task 4. The Library Client

Create a Console Application that implements the REST client and lets the user to perform all four verbs on the service. The console application displays a menu to the user and repeats this until the user wishes to quit. The menu options are as follows:

- help / about (displays the generic information about your program
- list (shows the list of current books, id and title)
- display book (displays the book info by getting the id from the user)
- add book (adds a new book to the system, data to be given in the console by the user)
- update book (updates specific book, id and the data to be given in the console by the user)
- delete book (deletes a book, id to be given in the console by the user)
- quit

Make sure you handle errors properly. Your code must catch all runtime exceptions.
Bonus: In case of error, you may give a retry option to the user by remembering the data.

Classes to be created:

- The REST Client
  The REST client class may be implemented manually or be generated automatically using tools.

- The Console Class
  This class implements the client application by interacting with the user and calling the web service using the above REST client class.

**Task 5. The readme.md file**

Create an MD file. Provide a short description of the project, including the tasks that are done by the individual team members. Include any notes on project setup, libraries you used, etc. here as well.

**Deliverables**

IMPORTANT: You are allowed to work on a team of 3 students at most (including yourself). You and your teammate must be in the same section. Any teams of 4 or more students will result in 0 marks for all team members. If your work on a team, ONLY one copy of the assignment is to be submitted for both members. You must make sure that you upload the assignment to the correct directory of **Assignment 1** on Moodle. Assignments uploaded to the wrong directory will be discarded and no resubmission will be allowed.

Naming convention for uploaded file: Create one zip file, containing all needed files for your assignment using the following naming convention:
The zip file should be called *a#_studentID*, where # is the number of the assignment *studentID* is your student ID(s) number. For example, for the first assignment, student 12345678 would submit a zip file named *a1_12345678.zip*. If you work on a team and your IDs are 12345678 and 34567890, you would submit a zip file named *a1_12345678_34567890.zip.*

Submit your assignment electronically via Moodle based on the instruction given by your instructor as indicated above. **Please see course outline for submission rules and format, as well as for the required demo of the assignment.** A working copy of the code and a sample output should be submitted for the tasks that require them. A text file with answers to the different tasks should be provided. Put it all in a file layout as explained below, archive it with any archiving and compressing utility, such as WinZip, WinRAR, tar, gzip, bzip2, or others. **You must keep a record of your submission confirmation.** This is your proof of submission, which you may need should a submission problem arises.

### Grading Scheme

| T# | MX | MK | |
|---|---|---|---|
| 1 | /1 | | *common structure (lightweight, no business)* |
| 2 | /3 | | *library business, concurrency* |
| 3 | /2 | | *REST and annotations, no xml or json* |
| 4 | /3 | | *REST Client, console app, error handling* |
| 5 | /1 | | |
| Total: | /10 | | |

(T# - task number, MX - max (out of), MK - your mark)

### References

1. https://www.vogella.com/tutorials/REST/article.html
2. https://www.markdownguide.org/getting-started/