SOEN 387 – Fall 2019
Assignment-III

15% of the Final Grade

Due Date: Dec 2, 2019 – 11:59 PM
Group Assignment: A maximum of three per team

Using Enterprise Application Patterns

Objectives

Applying web enterprise architectural patterns.

Project Outline

In this assignment, we use the project implemented in assignment 2 and refactor it by applying the following architectural patterns:

- Table Data Gateway / Active Record
- Domain Model
- Service Layer
- Template View

Part A - The Data Access Layer

In the previous assignment the repository class was implemented in the core project. In this revision we want to use a separate database layer. Create a separate project called data access layer and implement the data access using a pattern of your choice (Table Data Gateway or Active Record).
Move the database configuration that you implemented in the repository core to this layer (database connection string, user id, and password).

Patterns: Table Data Gateway or Active Record

Part B - The Repository Core

Keep the Business Core Objects, as designed in assignment 2. Remove all database implementations. Instead, use the newly Data Access Layer you created in part A.

Patterns: Domain Model (as specified in assignment 2)

Part C - The Web Presentation Layer

In the Presentation Layer, refactor the "View book details" page by applying a Template View pattern. Implement two different versions of the view for display the book information. One: with cover page, One: without the cover page, with an option to edit the book info (the link to the edit info brings the user to the edit page).

Patterns: Template View (2 implementations)

Part D - The Service Layer

In this part, you are simulating the possibility of having a separate service layer on top of the business core. For this part, create a separate web application project that hosts a servlet, namely books, for simulating the functionality of a service layer. The servlet responds to a GET request and returns book information in json format. It is specified as follows:

- *Calling servlet with no parameter:*
  servlet lists all books in json format. No cover image information is given.

- *Calling servlet with id=…*
  servlet returns the book details in json format. Since the image cover data is a byte array, it won't be included in the response. However, include a field indicating whether the book has a cover image or not (i.e. "has-image" : "yes").

In case of any errors, the error must be caught and reported in json format. Include the exception message in the response.

Note that since the repository core is a secure, the service would not work without the client being authenticated. Hence, implement two additional servlets for authentication:

- login servlet (user=…&pwd=…)
- logout servlet (no parameter)

It is assumed that the client uses cookies to store the session. Note that this is not a REST implementation.

Patterns: Service Layer

Part E - The System Model

Display the system architecture by providing updated UML diagrams and showing the above implementations.

For each project (layer), display the class diagram, as well as showing one sequence diagram for a scenario of your choice. **Total of diagrams: 8.**

For each sequence diagrams, only list classes within the project layer and only the top level class(es) from the underlying layer, if applicable.

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 3** using EAS. 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 EAS 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	/25		*Table Data Gateway or Active Record + proper configuration*
2	/15		*Domain Model + proper configuration*
3	/24		*Template View ×2 (each 12)*
4	/20		*Service Layer books: 14, login/logout (each 2)*
5	/16		*UML Package Diagram (×4) + Sequence Diagrams (×4)*
Total:	/100		

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

References

1. https://martinfowler.com/eaaCatalog/
2. https://draw.io/
3. https://www.visual-paradigm.com/VPGallery/import/Rose.html