DAT152: Lab – Exercises #2

Extending the Library Services

Part A

For this lab, you will extend the library service to include two additional features.

1. Add Author
2. Delete Book

Preambles: Download and unzip **Lab2a.zip** (same as *demo-library-flowmanager.zip*). Then, import the maven project into your preferred IDE. Add the project to your Apache Tomcat server 10.1.x. Start your server and point your url to "http://localhost:8080/library" (default server port=8080)

Task #1: **Add Author**: (You can use the AuthorDAO (no.hvl.dat152.dao) for database access)
You need to update/create:

a) index.jsp
   - include the href command for the **GET** requests (e.g., *Add Author*)

b) View (addauthor.jsp)
   - Include the form action for the **POST** requests (e.g., *<form action="addauthor" method="post">*)

c) Create the actions (**GET** and **POST**) in the "no.hvl.dat152.action" package
   - AddAuthorFormAction.java
   - AddAuthorAction.java

d) Update the Map in FlowManager.java in the "no.hvl.dat152.controller" package to map the commands with their corresponding pages (views)

e) Update the Map in the createActionsMap() method of the FrontController.java class to map the commands with their corresponding actions.

Task #2: **Delete Book**
You need to update/create

a) viewbooks.jsp
   - include hyperlink (href) to the command (deletebookform) for the GET requests.

b) deletebook.jsp
   - include the href to the command (deletebook) for the delete action

c) Create the action (**GET**) in the "no.hvl.dat152.action" package
   - DeleteBookAction.java

d) Update the Map in FlowManager.java in the "no.hvl.dat152.controller" package to map the commands with their corresponding pages (views). You can, for example map the "deletebook" command to "viewbooks.jsp". It means that when the delete action is completed, the viewbooks 'view' will be displayed.

e) Update the Map in the createActionsMap() method of the FrontController.java class to map the commands with their corresponding actions.

Part B – Security features (Optional)

Secure admin pages (e.g., add Book, add Author, update Book, delete Book) from non-admin users.

Preambles: Download and unzip **Lab2b.zip**. Then, import the maven project into your preferred IDE. Add the project to your Apache Tomcat server 10.1.x. Start your server and point your url to "http://localhost:8080/library"

Two users are created during initialization:

- username=admin, password=password, role=ADMIN
- username=user, password=password, role=USER

We use two filters (AuthenticationFilter and AuthorizationFilter) to intercept requests and check for authentication and authorization permissions before forwarding.

Test

Run and test that the normal user can view books but cannot add book.
Run and test that the admin user can both add and view books.

Task #1: Update the project with your additional features (copy your working features from part A)

Task #2: Configure the Authorization filter in the web.xml to include all commands that should be verified for authorization.
e.g., addauthor,addauthorform,deletebook, etc

Run and test your newly added authorization configurations.