Lab 8 - Web Application Security

Summary

The purpose of this assignment is to configure Jakarta EE security in our projects to be used by a JSF login process, and to gain experience with JSF templates.

Requirements

Documentation

Create a document titled **Lab 8.**

Database Setup

Use your itmd4515 database and user from Lab 1.

Project Setup

Your uid-fp repository should already be setup, and you should continue pushing your commits into GitHub for this lab. Use a prefix of Lab 8 on your commit messages so I can distinguish them from other labs.

Project Requirements

You should add these changes to your uid-fp project.

Make sure you are following standard Java naming conventions. Please review the following if you are not sure what those naming conventions are:

- Java 8 Pocket Guide by Patricia Liguori, Robert Liguori - Chapter 1. Naming Conventions
- Code Conventions for the Java Programming Language - 9 Naming Conventions (dated, but still correct)

You are welcome to follow the design and patterns I used in my examples, but if you find yourself doing so please make sure you provide a reference in your javadoc comments and/or README page explaining that you based your project on "Instructor Example" or "Example from Web." Always provide references and citations when you use ideas from other sources. Don't copy and paste my code, however, you need to adapt designs and patterns to your own projects and actually write the code yourself.

1. As demonstrated in class, Introduce Jakarta EE security to your application that incorporates both authentication and authorization. A complete example of this will be provided in class demonstration. You are welcome to follow my example, which made use of Jakarta EE Security, or to explore and introduce your own approach. If you follow my example, you will need to:
    1. Create User and Group JPA entities
    2. Associate them to each other with a ManyToMany relationship
    3. Associate User to your business domain with a OneToOne relationship
    4. Create EJB services for the new security entities
    5. Populate data using your existing Startup Singleton database loader.
    6. Create a new application scoped security configuration class using:
        1. @CustomFormAuthenticationMechanismDefinition
        2. @DatabaseIdentityStoreDefinition
        3. @DeclareRoles
    7. Define XML security-constraint elements in web.xml, containing a web-resource-collection and auth-constraint to restrict access to URL paths within your application
    8. Map your roles to users and groups using the security-role-mapping XML element in payara-web.xml
2. **Important** - you must hash your passwords to work with JSR-375 Security! In class we reviewed how to do this with the default Pbkdf2PasswordHash (PBKDF2WithHmacSHA256).
3. Build your login process with JSF using JSR-375 security. As demonstrated in class, this requires the following:
    1. Introduce a LoginController with validated username and password fields
    2. Inject the SecurityContext and FacesContext
    3. Implement doLogin and doLogout action methods in the LoginController
    4. Implement login.xhtml and error.xhtml pages using Facelets XHTML, binding the fields from LoginController as appropriate
    5. Handle errors appropriately. Log technical exceptions verbosely. Add appropriate context for the user via JSF messages, and display on the view using the JSF messages component.
4. Make use of JSF template(s) for your layout. You could also opt to use a separate login page template from Bootstrap, and I encourage you to adopt any of their examples that apply more directly to the purpose of your application. Be creative! You do not need to implement one of the Bootstrap examples this week as a template, just put a basic template in place.
5. Provide logout functionality. I expect to be able to logout from your application and click a link to login again with a different user.
6. Documentation
    1. Include screenshots and narrative to prove:
        1. Your authentication is working.
        2. You know who the authenticated user is
        3. You know what role(s) the authenticated user has

    2. Make sure you include your username/password combinations in your README page so I can test your application, as well as any steps necessary to create the security realm.
    3. Discuss your experiences, including any difficulties you had or changes you made
7. Submit to Canvas or Beacon
    1. Right your **uid-fp** project and select "Clean"
    2. Go to your NetBeans Projects directory. Create a zip file of the **uid-fp** folder and submit it to the Canvas or Beacon assignment.