Overview of business logic:

- Uses latest version of MVC and Ozark (milestone 2 snapshots), because of the validation
- entities package contains the JPA entities
- rest package contains the JAX-RS bootstrap class
- If you want to do anything with sessions, inject and use SessionManager
- If you want to get the current logged in user, create the following field:
  @Inject
  @LoggedIn
  private User currentUser
- To change the current user, inject UserContext and call its setter
- UserManager takes care of users
- TestDataInserter inserts some test data from past JUG sessions. The initially logged in user is nayden

Note:

Make sure you have started the database. If you prefer not to change anything in your persistence.xml, just go to your <glassfish-dir>/javadb/bin directory and run the startNetworkServer script for you operating system

Task 1: Display all the sessions as well as the session by the currently logged in user

Hints:

- Create a controller that puts into the model the sessions and returns sessions.jsp
- Maybe you need two separate methods in the controller that are listening to two different paths
- Inject the current user as described in the previous section

Task 2: Switch user

Hints:

- Create a controller that shows the login page upon GET and accepts the parameter from the login form upon POST
- Form parameters are mapped to @PathParam(<parameter-name>) annotated method parameters
- As a result the POST method should redirect either to the home page upon a successful login and back to the login page upon login failure

Task 3: Submit a proposal and validate the input

Hints:

- Chapter 3 of the EDR 1 spec is outdated
- There is a special @Model bean with bean validation annotations on JugSubmission. Take a look at it

- You should create a controller (or reuse existing) again with a couple of methods: one for showing the form (GET) and another one for handling its submission (POST)
- Consider creating another @Model bean that holds the validation error messages and can be accessed from the JSP via EL as we showed in Part 1
- Look at https://github.com/spericas/ozark/tree/master/test/validation/src/main/java/org/glassfish/ozark/test/validation for bundling bean validation with MVC