CS-321: Server-Side Web Development
Spring 2021
Northeastern Illinois University
Homework #9: Persistence, Part III
Due Date: Thursday, 04/01/21 by 6:00 p.m. CDT

IMPORTANT: Every step in this homework assignment refers to materials that was presented in the lecture videos and PowerPoints, so you should be actively using the videos, PowerPoints, and referenced materials.

Homework Goals:

1. Load initial data in your database.
2. Create a relationship between your domain models.

Part I: Creating a new domain model

- ☐ Review your non-user domain model and pick a component of that domain model that could be represented as an object. For example, if you have a domain model that represents Games, then you could create another domain model that represents game categories (with additional information about the game categories such as number of players, or other settings).
- ☐ Create this domain model in your `models` package.
- ☐ Turn this domain model into an Entity (complete with the necessary constructors and getters/setters).
- ☐ Create a repository for the model in the `data` package.
- ☐ Make sure all your code compiles and that your application runs as before. Push to GitHub.

Part II: Loading initial data

- ☐ For the model that you created previously, create an 'import.sql' file in the 'src/main/resources' directory.
- ☐ Write the SQL statements needed to load 5+ entries into the table associated with the new domain model that you created.
- ☐ Run your code and verify that you see this new table and entries in your database using phpMyAdmin.
- ☐ Push your code to GitHub.

Part III: Entity Relationships

- ☐ Associate your new domain model with the non-user entity that you created previously. What type of relationships should this be? Many to many? One to one? One to many?
- ☐ Verify your code still runs correctly and push to GitHub.
- ☐ Modify the add functionality for the non-user entity so that a user can select/add the component domain model that you created. For example, in lecture, I added a drop-down of Courses so that I could select multiple courses and add it to a Student. In your case, this depends on the type of relationship.
- ☐ Run your code and verify that this new component can be added to the model and that the join table or foreign keys (depending on relationship) show up in your database. Push to GitHub.
- ☐ Determine if this component is a required component (for example, a category would be a required component, but courses for a student may not be). If it is a required component, make sure to add the necessary validation fields as well as the error fields in the view. **Note**: If I don't agree with you, you will have to fix this!
- ☐ Run your code. Push your code to GitHub.
- ☐ Modify the view that displays all the non-user entity entries so that the component and its details are viewable.
- ☐ Push your code to GitHub.
- ☐ **Final Project Requirements:** The component itself needs to be editable/changeable (and possibly deletable depending on the component) when you are viewing the non-user entity in edit mode. While you do not have to complete this functionality as part of this homework assignment (although you are encouraged to do so), it is a required component of the final project.

Submitting your code to D2L:

1. Zip up the project folder and submit to the D2L assignment folder. I look at your code in GitHub and also use D2L as a way to know when your GitHub repo is ready to be reviewed.
2. As a comment in D2L (there's a place for students to comment when they upload files), please provide the database name, the username, and the password for your database.