CS-321: Server-Side Web Development
Spring 2021
Northeastern Illinois University
Homework #5: Thymeleaf and Basic Controllers
Due Date: Thursday, 02/25/21 by 6:00 p.m. CDT

Homework Goals:

1. Create your first model (a POJO).
2. Modify your existing Spring Boot project to have several controllers and a Thymeleaf form.
3. Push your code changes to GitHub using GitHub Desktop.

Part I: Creating a model

- This is where your homework starts to become more inventive! You're going to start building your final proejct.
- Pick a concept in your final project idea that would require some type of form for a user to fill out. For example, in the lecture, I created a very basic form (*like super basic*) to add a student. This form was represented behind the scenes by a POJO object named `Student`.
- Create a directory in your base package named `models`.
- In the `models` directory, create the POJO object complete with the necessary constructors, getters and setters. Your POJO should have at least 3 instance variables - if not more. I would go for more...
- Because you are all working on unique projects, you can get help from your peers and post your code in Slack because you will all have different code!

Part II: Creating controllers and the views

- In the controllers directory, create a controller that has a GET method handler that will render the view for the form.
- Don't forget to add an "empty" version of your POJO as a model attribute.
- In the correct directory for views, create an HTML file. This HTML file should have a form that has fields for the instance variables in your POJO as well as a submit button.
- Use CSS to center the form in the middle of the page. Where do CSS files go in a Spring Boot project? `Do not use an internal style sheet`. The majority of your CSS should be in external style sheets with only a few inline styles (once we get to Bootstrap).
- The name of your application should be at the top of the page - centered as a header.

- Run your application to make sure that the controller is handling the request and rendering your view and that you have no syntax errors.
- In your controller, create a POST method handler so that your submit button in the form works correctly.
- Your POST method should redirect to another controller and pass the properties from the form model to the other controller (Hint: `RedirectAttributes`!).
- This new controller should have a GET method handler that accepts the redirected values and adds them to the model for displaying in a new view.
- Create a new HTML file for this new view and display the properties that were entered into the form. Using CSS, these values should be easy to read and displayed nicely. Again, the name of your application should be at the top of the page - centered as a header.
- Run your application to make sure that the controller is handling the request and rendering your view and that you have no syntax errors.
- Make sure that you are following Java naming conventions, HTML naming conventions, and path naming conventions. Your code should be formatted correctly. You will be asked to re-do/fix any code that does not follow these conventions. Hint: There's an IntelliJ shortcut that can reformat your code to make sure your indentation is correct!!

Part III: GitHub!!

- Commit and push your code changes to GitHub.
- A good rule of thumb is to commit and push your code every time you get a small amount of it working.
- Do not create a new or separate project for this - the whole point of GitHub is that it provides version control - you can go back to older versions of your code and you can see the whole history of your code changes (this is why committing and pushing frequently is a good idea).

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.