PART 1. Reading Assignment [Ignore Annotations (@ sign indicates annotation)]

https://www.safaribooksonline.com/library/view/spring-in-action/9781617291203/kindle_split_009.html

https://www.safaribooksonline.com/library/view/spring-in-action/9781617291203/kindle_split_010.html

PART 2. Reading Assignment (ignore the annotation part, and anything we did not discuss in the class).

http://docs.jboss.org/hibernate/orm/5.4/quickstart/html_single/#tutorial-native

PART 3. Reading Assignment (ignore Maven Installation)

https://www.safaribooksonline.com/library/view/beginning-hibernate-for/9781484223192/A321250_4_En_1_Chapter.html

https://www.safaribooksonline.com/library/view/beginning-hibernate-for/9781484223192/A321250_4_En_2_Chapter.html

~~PART 4. Programming Assignment~~

Using more than one Controller is probably the last thing you want to do for a simple application, considering the maintenance involved for each Controller. In this assignment, you are to develop a simple application using only one Controller. The application developed is an online quiz similar to the one in the following URL:

http://www.javatpoint.com/directload.jsp?val=200

The UI does not need to be the same. Whichever UI you like, that is fine as soon as the functionality is the same. The problem with using one Controller is that each form will submit to the same servlet and the same service method will be invoked. How do you know which form to display next? One solution is simply done by incorporating a hidden field or a query string parameter, call it page, with a value of the form number. In the first form, the page field will have the value 1, and in the second form this field will have the value of 2. When the Controller is called, the service method is invoked to get the page number. Questions to be passed to the View page as an ArrayList with the ModelAndView. You will use JSTL/EL to display the questions in the View Page. The entire application needs to be developed with only 1 Controller, and 1 View page. Use a URL pattern in the following format

/quiz/*.htm

This request will be handled by the same Controller. When the URL is /quiz/1.htm, question 1 is to be displayed, and when it is /quiz/2.htm, question 2 is to be displayed, and so on. Keep the answers in a HttpSession object. When the last question is submitted, you should retrieve the answers from the Session object, display on the View page.

PART 5. Programming Assignment

Redo Part 4 of this assignment using AbstractWizardFormController. Add Previous, Next and Cancel Buttons to display the next or previous page. Just like the AbstractFormController, this was deprecated with Spring 3.2.5, and removed from the later versions. There is a better way of doing it (we will discuss later), but this class will help you understand how the request goes from one page to another. Also, the class does not exist as a SuperClass, but the behavior could be implemented in a plain class using Annotations.

Question1--> Question2 --> Question3 --> . . . --> ResultsPage

You need to include Spring 3.2 when creating the web application, the same way we did with AbstractFormController.

In order to use AbstractWizardFormController, just create a plain Java class, and extend to AbstractWizardFormController. Once you create the Controller class, you need to override validatePage(), processFinish() and processCancel() methods.

Please look at the Spring JavaDoc page to see the methods:

https://docs.spring.io/autorepo/docs/spring/3.2.5.RELEASE/javadoc-api/org/springframework/web/servlet/mvc/AbstractWizardFormController.html

~~**PART 6. Programming Assignment**~~

Redo HW2 Part7 using SimpleFormController and Hibernate. Searching Movies to be done by MovieId as the Hibernate Session objects's get method requires Serializable ID (which is basically the PK in the DB, but SerializableID in Java). Use a Spring Validator to validate the fields. You could have 2 different FormControllers for saving and searching. UI may be different from the previous implementation in HW2. The first page will be the form-view, and the results page will be the success-view.

Create the SessionFactory the old way from the Configuration instance.

~~**PART 7. Programming Assignment**~~

Redo Part 6 of this assignment, but create the SessionFactory the new way.