Form Object Exercise

Form Object

A Form Object is an object that is created in Java and is mirrored in the html form. This makes it easy to work on the same object, and there is no more any need to handle individual request parameters.

The html form refers to the form object in the form tag and refers to the variables in each input tag.

Read about form object here:

https://spring.io/guides/gs/handling-form-submission/

The Greeting object in the example is the form object. Notice how it is used in the html form in the thymeleaf template with this reference:

th:object="${greeting}"

The post mapping method get direct access to the form object with all the values the user has entered with the signature:

```
@PostMapping("/greeting")
public String greetingSubmit(@ModelAttribute Greeting greeting) {
    return "result";
}
```

Now, try to change the Login application to use a Form Object instead of individual request parameters. Look at the greeting example for inspiration.

- Create a User class with the variables username and password, add an object of this class as a **@ModelObject** to the model like in the greeting example and use it in the html form.

- Change the form in the login.html template to use the Form Object by adding **th:object="${user}"** in the form tag (this is if you gave the Form Object the name **user** in the **ModelAndView**).

- Change the input fields by adding a **th:field="*{username}"** in the username input and **th:field="*{password}"** in the password input. This is for binding the input fields to the variables in the Form Object.

- Get the Form Object back from the html form in the **@PostMapping** method as in input argument with the **@ModelAttribute** annotation.

Stretch Tasks - If you have time

1. Do the exercise in the guide mentioned above:

https://spring.io/guides/gs/handling-form-submission/

2. Try to change the forms for adding a new blog and adding a new comment in the Solution of module 8 – Java Web Jdbc Hacking Exercise.