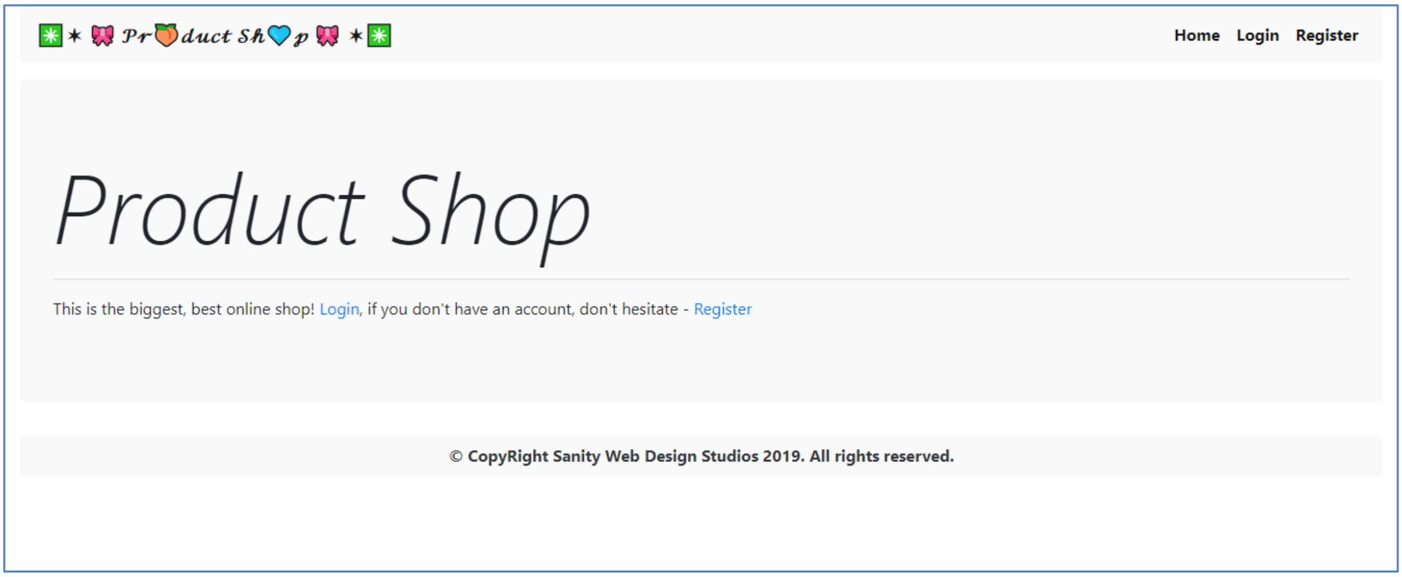
Project: Product Shop

`Product Shop` is a system that registers users, categories, products and orders. It is a significantly big project, and as such it will have several parts. In this exercise you will land the basics of the application, in other words - the user functionality.



Product Shop

This is the biggest, best online shop! Login, if you don't have an account, don't hesitate - Register

© CopyRight Sanity Web Design Studios 2019. All rights reserved.

Workshop: Part 4.5 – Error Handling

1. Error Action & Error Page

By default, upon encountering an exception, Spring tries to find an action mapping at URL "**`/error`**". If it finds the mapping, Spring sends the error information there. If there is no such mapping, the **`Whitelabel`** page is visualized instead.

Your task is to implement an Error action mapping on "**`/error`**". It would be better if the action is in a separate controller. The action should return a simple **`Thymeleaf`** view, which you should also implement.

Visualize basic error data – **HTTP Status Code**, **Exception message** etc. Format it in a User-friendly way.

NOTE: The mapping type of the error action is a [**GET**] mapping.

2. Basic Error Handling

Implement **global exception handlers**, for exceptions which may occur during **Framework processes**. For example, database exceptions, Spring request chain exceptions, etc.

The global exception handler should always return a **500 Internal Server Error**.

NOTE: If you implement everything correctly, it should result in the Error page, you've implemented in the previous task.

3. Custom Error Handling

Implement **Custom Exceptions**, for every Application error you can think of, like requesting details about a non-existent Product, Ordering with invalid data etc.

Annotate the exceptions with the appropriate **`@ResponseStatus`** annotation.

Attach appropriate messages, which should be constants, for easy code maintenance.

NOTE: If you implement everything correctly, it should result in the Error page, you've implemented in the previous task.

4. ** Setting up application environments

Do a little research on how you can setup environments in the Spring application. Setup a **Production environment** and **Development environment**.

In **Production environment** you should keep public error information to an **absolute minimum**. Only a User-friendly message and a status.

In **Development environment** you should present maximum information about the error. You should present error message, stack trace etc.