COMP 31, Lab 4

Goal

The goal of this lab is to debug and document a simple Spring Web Login application. The lab will be consist of the following steps:

1. Clone the starter application (which has been modified to include several minor bugs).
2. Set up and configure a Logger in the application. (This step will be demonstrated at the beginning of the lab.)
3. Identify and document the bugs.
4. Use logging to trace the program's execution and pinpoint the errors.
5. Fix the bugs and test the corrected application.
6. Document the bugs, how you identified them, and the solutions you implemented.

1. Clone Starter Project.

The starter application is available on **github** at the following repo address

https://github.com/cpa-student/lab4.git

2. Configure the Logger

Loggers can create messages at a variety of levels, e.g. DEBUG, ERROR, INFO, depending on the severity of the message. The application can also be configured in a way that only logs above a certain severity lever will be displayed. Logger configuration is set in the **application.properties** file. Use the following settings to display all log messages and to reduce the amount of information the contained in each message:

```
logging.level.root=trace

logging.pattern.console=%level%msg%n
```

3. Identify the Bugs

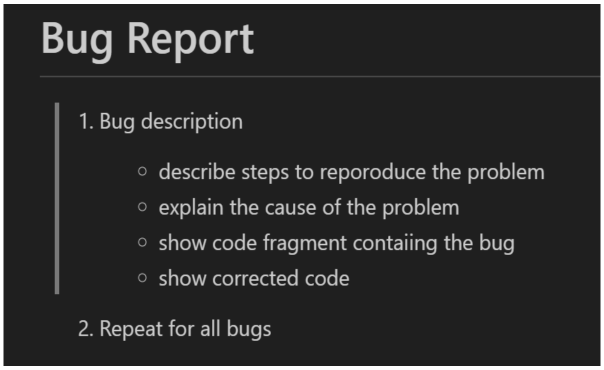
Run the application. Observe the behaviour and make note of any unexpected behaviour or results. Use log messages to trace the flow of the application and to identify where int the code the unexpected behaviour occurs. The log output should appear in the Debug window. At the TRACE log level, this window will display a lot of information coming from the framework. To focus oNote: in Week 1 you should have configured the debug output to be displayed in the Debug window. Th

4. Fix the Bugs

After identifying the location of the bugs, correct the code and rerun the program. You will find that fixing one bug may (will) introduce another, so repeat the process as many times as it takes.

5. Document the Debugging Process

The starter project includes a **doc** folder containing a **bug_report.md** file. It uses **.md** (mark down) format – a lightweight markup language that includes formatting elements added to plain text. The formatted version can be displayed by right clicking the file name in explorer window and selecting “Open Preview”. The file includes some basic formatting elements that mark up a template for reporting bugs:

# Bug Report 1. Bug description * describe steps to reproduce the problem * explain the cause of the problem * show code fragment contaiing the bug * show corrected code 2. Repeat for all bugs	 Bug Report 1. Bug description ○ describe steps to reporoduce the problem ○ explain the cause of the problem ○ show code fragment contaiing the bug ○ show corrected code 2. Repeat for all bugs

Use the template to complete your bug report.

Hand in

At the end of the lab, zip your entire project and submit it to Blackboard. Submit your project even if you did not complete all the steps. You may submit a completed version of the project by the end of the day. (Note: you must submit your work by the end of the lab to receive a lab mark.)