THE UNIVERSITY OF WESTERN ONTARIO

DEPARTMENT OF COMPUTER SCIENCE
LONDON, ONTARIO, CANADA

Computer Science 2212b
Introduction to Software Engineering - Winter 2014

Team Assignment 3
Due March 14 at 11:59 PM

1 Overview

In this assignment, your team will complete a second iteration, releasing another prototype with additional functionality. You will also ensure that your team has a Jenkins server running, which automatically builds your code, runs your tests, and generates a code coverage report, each time a change is pushed to your GitHub repository. Finally, you will write a set of acceptance tests for a few requirements of the project specification.

The TA must be able to test your prototype, so the code in your repository that is tagged `asn3` must compile using `mvn package` and the TA must be able to run the resulting JAR file.

In addition to the code, you will need to include instructions for the TA on which user stories you completed, and how he/she can test the completed stories.

2 PDF Report

You must submit a PDF report containing the following:

- Title page, in the same format as that required for Team Assignment 2
- Section 1: Revised UML Class Diagram
- Section 2: Revised User Stories
- Section 3: Acceptance Tests
- Section 4: Jenkins CI
- Section 5: Project Plan Documentation

2.1 Revised UML Class Diagram

This section should include a revised and updated version of the UML class diagram submitted for team assignment 2. Your diagram should capture ALL the system's functionality, as specified in the project specification document.

In addition, your diagram should include details such as attribute types, method parameters, and return types, and should identify which attributes and methods are public, private, etc.

GUI classes should be present in the diagram, but do not need to have any attributes or methods. The diagram should, however, show the associations between your GUI and non-GUI classes.

In order to make it easier for your teaching assistant to evaluate your new UML class diagram, **make sure to use a different colour and/or font and/or weight in your diagram** to distinguish between the updates you have made to the new class diagram versus the class diagram you submitted for team assignment 2.

2.2 Revised User Stories

You must update your PivotalTracker and indicate which stories are complete. Next, you must take a screenshot of your tracker so that we can see:

- Done
- Current
- Backlog
- Icebox
- Epics

Include this screenshot in this section, along with a textual link to your tracker. Furthermore, this section must list the user stories that have been completed since the last assignment, and provide instructions on how the TA can test each user story in your prototype (e.g. what to click on, how to navigate to the implemented functionality, etc.).

2.3 Acceptance Tests

For this section, to give you practice thinking about acceptance testing and writing comprehensive test suites, you are required to create an acceptance test plan consisting of a set of test cases needed to completely test the system in response to users performing the following:

- A user creating a new course
- A user editing an existing student

Each of the above tasks will likely require more than one test case in order to adequately test the task. Each test case must have the following layout:

- A unique ID
- A descriptive title that indicates its purpose
- A reference to the related requirement(s) in the requirements specification
- A description of steps need to carry out the test, including any necessary sample inputs.
- Expected output telling the tester how the system should behave in response to the instructions.

For example, see the acceptance test script below that *partially* shows the acceptance tests once might create for adding a new user to a system:

ID	1
Testing	Admin User adding a new user; invalid user ID
References	Requirement 4.c.1.1
Test Case	1. At the UNIX prompt, enter `java UWOBoggle -admin` 2. Select **Add User** 3. Enter `abcde` for the user ID
Expected Output	**Error**. **Message:** *Invalid user ID: user ID is too long*

ID	2
Testing	Admin User adding a new user; invalid user ID
References	Requirement 4.c.1.1
Test Case	1. Enter `ab` for the user ID
Expected Output	**Error**. **Message:** *Invalid user ID: user ID is too short*

ID	3
Testing	Admin User adding a new user; invalid user ID
References	Requirement 4.c.1.1
Test Case	1. Enter `Pabc` for the user ID
Expected Output	**Error**. **Message:** *Invalid user ID: user ID should start with a U*

⋮

Make sure that your test plan is clear, easy to read, and well-organized. If there is a test case that is not possible to test because of the way you designed your GUI (e.g. if a name is limited to 20 characters and you have designed your GUI so that the user cannot enter more than 20 characters), you should still include that test case but indicate that the GUI will not allow this to happen.

2.4 Jenkins CI

At this point, one of your team members should have completed lab 5. Your team is required to have a Jenkins server running that, upon changes being pushed to GitHub, automatically:

- Checks out your project code
- Builds the code
- Runs your unit tests
- Generates a JaCoCo coverage report, which is subsequently available through your Jenkins server

Create a Jenkins user `ta` and set a password for this user. Include the URL to your Jenkins server, along with the password for the `ta` account in your PDF file.

2.5 Project Plan

This section should be an update of the project plan from the second report. In particular, your team should provide a list of tasks to be accomplished between milestones 3 and 4.

Indicate all the information for each task that was required in team assignment 2. Remember to update the percentage of completion of any previous tasks and any upcoming tasks that you have already started. Make sure your Gantt chart correctly indicates who is/was assigned each task, and who actually completed each task, so that we know that the work is being fairly distributed.

3 Prototype

For this assignment, your team must have **66%** of its user stories complete. That is, if you have 18 user stories in total, then you must have 12 user stories fully implemented in the GitHub commit tagged `asn3` (but will have

completed six of those for assignment 2).

The stories chosen should be listed in the **Revised User Stories** section of your PDF report, as discussed in section 2.2. Feel free to implement more than **66%** of your user stories – you will thank yourself later at the end of the course as the deadline nears.

The TA must be able to compile the prototype by cloning your repository and using the `mvn package` command. Additionally, the JAR produced should be named `teamN-gradebook-1.0-SNAPSHOT-jar-with-dependencies.jar` (or `teamN-gradebook-1.0-SNAPSHOT.jar` if you have no dependencies), where `N` is your team number. The TA should be able to run this JAR by typing `java -jar JAR_FILENAME` (that is, it must be an executable JAR and must contain any and all dependencies within it).

Note: the TA must be able to see that when he/she starts your program, enters some data, exits your program, and then restarts it, that the data he/she entered is persistent (i.e. it is not lost when the system is shut down). Not all the data must be stored for this assignment, but the TA must be able to see that your team has figured out how to perform data persistence for the system.

And yes, you need a GUI.

3.1 Unit Tests

We will not be marking your unit tests until the end of the course. However, it is **strongly recommended** that you write your unit tests for your code as you complete it (or *before* you complete it, if you're doing test-driven development). You don't want to be in a situation at the end of the course where you're scrambling to write all your unit tests.

4 Marking

The marks will be tentatively assigned as follows:

- Title page and general formatting of hand-in: 2 marks
- Revised UML class diagram: 3 marks
- Revised user stories: 5 marks
- Acceptance tests: 10 marks
- Jenkins CI: 2 marks
- Revised project plan: 3 marks
- Prototype: 20 marks
- Spelling and grammar: 2 marks

5 Assignment Submission

- In the **root** of your team repository, create a directory `deliverables/asn3`
- In the `deliverables/asn3` directory, place **one** file named `teamN-asn3.pdf`, where `N` is your team number
- In the root of your repository, there should be a `pom.xml` file
- Your code should be present in the `src` directory, which should be found in the root of your repository
- Do **not** submit rendered JavaDoc files (e.g. `.html`, `.css`, etc.) in your project repository. Use your `.gitignore` file to ignore them.

- Commit your code and PDF to GitHub and tag the commit with the tag `asn3`
  - Do not forget to push the tag to GitHub – your assignment is not submitted unless the tag is pushed
  - You can verify that the assignment was submitted by clicking the **Releases** link on your team repository page on GitHub. You should see an `asn3` release if you submitted successfully
- Do not submit any other files in the `deliverables/asn3` directory
- Remember that part of your mark is based on the clarity and organization of your report, so make sure it is neat and well organized
- Additionally, remember that poor spelling/grammar will lose you marks, so ensure that one or two people go over the entire assignment once it is finished, prior to submission, to proofread it and correct any errors found

6 Peer Evaluation (to be done individually)

Within four days immediately after this assignment's due date, complete a peer evaluation for each individual (including yourself) on your team. The private course site will soon have a link to complete the peer evaluations.

For each peer evaluation that you fail to submit within four days of the due date, you will individually lose 0.5% off of your final course mark (up to a total of 2.5% for the 5 peer evaluations you must submit).

Note: Peer evaluations can adversely affect the final mark given to an individual, so please complete these peer evaluations constructively and fairly.

Peer evaluations without comments are next to useless. Please try to provide a short comment about each of your team members (and yourself). Only your instructor will be reading the evaluations and comments.