CS304 Project: Progress Report *(Total: 60 points + 28 points for code reviews)*

Deadline: May 8 **(Friday), 11.59pm**

You have specified your schedule in the project proposal. The goal of this progress report is for you to discuss your progress since the project proposal. (Invitation link: https://classroom.github.com/g/4qN8liNK. **When you are ready to submit your project, you need to create a tag called "version-1.0" in this repository**. Read https://stackoverflow.com/questions/18216991/create-a-tag-in-a-github-repository if you don't know how to create a tag in GitHub.)

What to submit?

- All source code and code comments
- All tests
- README.md. **All answers for the question (except for Javadoc comments and JUnit test). Include the group name and name and id of each team member.**
- .gitignore

1. Choose issues **to implement for this iteration** based on two criteria: (1) at least 1-2 GitHub issues per person, and (2) the **number of selected issues should be more than 50% of all scheduled issues**. Note that we want you to do planning for the issues again for this iteration to get your most up-to-date schedule.

a) List all the issues that your group have chosen. Explain the reason for choosing this issue? *Hint: Your answer can be based on either (1) importance of the issue; or (3) dependencies between issues; or (2) the ease of constructing test cases but you need to explain in more details (for example, I choose issue A because A is the issue with the highest priority among the selected issues).* (Total: 6 points, 2 points for each issue & reason)

b) Write 2 test scenario for each issue. The test scenario could either be **JUnit test cases (better to have JUnit test cases) or test scripts that help you to reproduce the bug** These test cases serves as the specification for each issue (Example positive test scenario: "When user login successfully to her account, a "Welcome" dialog box is shown". Example negative test scenario: "When user login with an incorrect userid/password, the user will get

an error message saying “Incorrect message”). **You should commit the test scenario in the README.md**. *(Total: 12 point, 2 points for each test scenario)*

c) Implement all the user stories of the issues that you have chosen. *(Maximum Total: 12 points, 4 points for each issue implemented)*

d) Run Checkstyle, FindBugs, and PMD for all the submitted code. Make sure that you follow the coding standard specified in your selected project (you can use the Google coding standard if your project don’t have any coding standard. Read the “Project Resources and Coding Standard folder” in Blackboard for getting a copy of the Google coding standard) . For Checkstyle, change your code so that there is no serious errors (severity=”error”). For FindBugs, change your code so that there is no error with priority=”Medium” and priority=”High”. For PMD, change your code so that there is no error left. *Note that the TA will randomly choose a team member to run these tools so if the team member doesn’t how to run these plugins, 5 points will be deducted for the whole team so please help your team member in the installation and the usage of the tool. (Total: 6 points, 1 point will be deducted for each violation in Checkstyle, FindBugs and PMD.).*

e) [The person who was in charge of Documentation should check this requirement] Write meaningful Javadoc comments for each public method. Read the link at https://google.github.io/styleguide/javaguide.html#s7-javadoc for information about the coding standard for Javadoc. *(Total: 5 points, 1 point will be deducted for each public method that are not documented.).*

f) [The person who was in charge of testing should check this requirement] Write >=2 JUnit tests for each modified/newly added public method for your implementation for GitHub issues. *(Total: 14 points, 1 point will be deducted for each public method that are not tested.).*

g) Include a schedule for each week after May 1 to plan for your remaining issues. **It is recommended to plan for two subparts (two user stories) of the GitHub issues implemented per week and one small subpart (one user story) for Week 15 (Need to**

prepare final release of the project). *(5 points)*

Example Table:

Week 13	“Login …”, “Machine-learning…”
Week 14	
Week 15	

h) Which lab session will you attend for the Code Review? **Choose the lab session that most of your group members have registered for (If you choose the wrong session, 5 points will be deducted).** The leader of the team should put the choice in GitHub discussion according to the sample format in https://github.com/orgs/cs304-fall2020/teams/all-students/discussions/3. Those who comment earlier will get the slot that they want. For example, For TA A who is in charge of 4 groups, student could choose TA A, Slot 1, Slot 2, …Slot 4.

A) Lab Session 1 实验 1 班(Instructor: Prof. Shin Hwei Tan)

TA A (4 groups), TA B (4 groups), TA C (3 groups)

B) Lab Session 2 实验 2 班(Instructor: Hu Chun Feng)

TA D (4 groups), TA E (4 groups)

C) Lab Session 3 实验 3 班(Instructor: Hu Chun Feng)

TA F (4 groups), TA G (4 groups)

D) Lab Session 4 实验 4 班(Instructor: Hu Chun Feng)

TA H (4 groups), TA I (4 groups)

Online Code Review with TA/Student Helper (Total: 28 points)

We will schedule a code review during the lab session on May 9 -May 15. Please attend your lab session according to the schedule. The code review will be conducted in the following way:

1. The leader will first introduce their selected projects. Then, he will summarize the progress of the team in a few sentences and introduce the role of each team member.
2. The designer will explain the design of their implementation and describe the design plan for the next iteration. The designer will also show a demo of the implemented issues.
3. The TA will pick one person randomly to run the static analysis tool.
4. The person (developer) in charge of each issue will describe how it works.
5. The person in charge of documentation will explain the part that he has implemented and the Javadoc comment for each method.
6. The testers will run all tests and shows the tests results and code coverage results.
7. The team leader will end the code review by showing and explaining their plan for next iteration. The TA will give some suggestions for future improvement.

Points for the Code Review

- **Demonstration of the implemented issues.** (5 points)
- **Demonstration of running static analysis tools.** *(The TA will randomly choose a team member to run these tools so if the team member doesn't how to run these plugins, points will be deducted for the whole team so please help your team member in the installation and the usage of the tool).* (5 points)
- **Quality of the code written.** *Points will be deducted if the TA read your code and see that have non-descriptive names in your variables, method or classes (for example,*

method called "methodA") or putting everything into one class, or long methods.
(5 points, 1 points deducted for each mistake)

- **Quality of the Javadoc comments written.** *If the Javadoc comments do not explain the parameter well (for example, "@param a"* only have the name of the parameter without any description) (5 points)

- **Quality of the Junit test written. Show the tests written to the TA and running code coverage tool.** (5 points)

- **Others (Design).** 1 point will be deducted for not including ".gitignore" for your project. 2 point will be deducted for not well-structured/ messy code. (3 points)