Second Individual Assessment

Published | Edit

You are NOT permitted to ask questions about the assessment, discuss the assessment, or work together on answering the assessment questions with your pair partner, other members of your team, other students in the class, anyone else in person or anyone else via internet or other forums besides the instructor and the IAs.

The technical questions follow at the end below.

In addition to answering the technical questions, include your response to the following requested feedback on your team and the course. This part will not be graded. However, if you do not include a meaningful response, we will not grade the technical questions - which means you will receive 0 for the assessment.

You have 100 total points to allocate among the members of your team. Please assign some number of points (possibly 0, possibly even negative) to each member of your team, including yourself, based on what you believe was that student's relative contribution to the team during the second half of the semester (since the first individual assessment). Make sure the total is exactly 100. For example, the scores might be

Jane 25

Paula 25

Georgia 30

Starr 20

Briefly explain each of your score assignments. In this example, its nice to see that your team came closer to evening out in the end. What changed? Who did or did not do what? What would your team do differently if you had to start over?

You should consider contributions to team meetings, team software development and testing, writing the team assignment submissions, setting up and presenting demos, anything else relevant to the team, in awarding the points.

This is also an opportunity (separate from the SEAS course midterm and final evaluations, which you should also submit) to provide feedback about the course. Is there anything that you think should have been covered but wasn't? Is there anything that was covered unnecessarily? Is there anything that should have been presented in a different way? Is there anything else you would change about the course, including contents, teaching methods, delivery, assignments, anything else you think relevant? Briefly explain. I'd particularly like to hear any criticisms, comments,

concerns regarding jumping right into implementation without a prior "design phase", and covering design topics only at the end of the course. Suggestions for next year's offering of this course would be greatly appreciated. You will not be penalized for saying "this course sucks", but please explain in what way it sucks and how specifically it could be improved next year.

Finally, if you have taken any other 4k or 6k Computer Science courses (COMS, CSEE, EECS, CSOR, etc.) here at Columbia during the past two years, including this semester, please tell me how the approach to testing as taught in 4156 is different from testing as addressed in any of those courses (those that covered testing at all). Thank you.

For the following questions, assume your startup (consisting solely of your team) is in discussions with a major software company, FANG, Inc., to “acquire” your product. You’ll be rich! However, FANG requires some additional work on your product before the acquisition. In particular, FANG insists on a carefully and completely documented design that can be easily understood by their engineers. Of course, in a real acquisition, the design would be documented by your team working together. But since this is an exam, you must not discuss with your teammates or anyone else other than the instructor and the IAs. Maximum 100 points.

0. List the user stories for your product. Its ok to omit conditions of satisfaction. Only include those user stories that are fully implemented, have been rigorously tested, and are really usable - FANG is not interested in purchasing vaporware. This part is just setup for the remaining questions, no credit, but if it is missing or inaccurate you will receive zero (0) on the entire exam. (If your user stories may change before the final demo and/or final report, give the date and time at which this part was written.)

1. Describe the standard architecture or architectures (may combine more than one) used by your product and why you chose it/them; this does not need to be among the architectures discussed in class. If your team seriously considered any other widely-known architecture(s), but decided against, explain why. If you did not use any standard architecture, explain what you did instead and why. 15 points.

2. Pick at least three application-specific user stories (registration, login, logout, help, and other generic features are not application-specific). Write a set of CRC cards that collectively cover those user stories. Note that three user stories will not necessarily correspond to exactly three CRC cards. If applicable, explain how and why these CRC differs from the corresponding part of your codebase, and document the discrepancies. 15 points.

3. Present the internal details of each class, from your CRC cards, as a UML class diagram (you do not need to include getters, setters, constructors, finalizers, etc.). Omit relationships among classes. Explain each diagram in prose intended to be understandable to a software engineer who will need to maintain your code. If applicable, explain how and why these class diagrams differ from the corresponding part of your codebase, and document the discrepancies. 10 points.

4. Present relationships among classes, from your CRC cards, as UML class diagrams. Each relationship must be labeled. Omit the internal details of each class. Explain each diagram in prose intended to be understandable to a software engineer who will need to maintain your code. If applicable, explain how and why these class diagrams differ from the corresponding part of your codebase, and document the discrepancies. 10 points.

5. Describe at least one design principle relevant to your project, and explain how you would modify your codebase to better fulfill this principle (or if your code already fulfills this principle, explain how). This design principle could be related to any subset of your user stories, not necessarily the three chosen for the above questions. It does not need to be one of the design principles discussed in class, but should be standard and well-known, do not invent your own. 15 points.

6. Using the information provided in your CRC cards and class diagrams, present an informal unit-testing test plan for each class (in prose or pseudo-code, do not include any actual code). Make sure your unit tests include sample inputs and comparison of expected outputs to actual outputs. If applicable, explain how and why these unit test cases differ from the corresponding part of the tests in your codebase, and document the discrepancies. 10 points.

7. Using the information provided in your CRC cards and class diagrams, present an informal integration-testing test plan covering all method-calling and/or data-sharing relationships among these classes (in prose or pseudo-code, do not include any actual code). Explain how sample inputs will be constructed to test the integration of classes and assemblies of classes (or modules, packages, etc.). If applicable, explain how and why these integration test cases differ from the corresponding part of the tests in your codebase, and document the discrepancies. 10 points.

8. FANG is also extremely concerned about security and privacy threats, since if the public finds a security or privacy flaw in your product, the CEO may have to appear before the US Congress, the European Union Parliament, and other governing bodies, and expensive sanctions may be imposed on the company. They do not expect you to solve all privacy and security problems, but you must disclose them in advance so FANG's own staff can try to remove or mitigate. Describe as many possible privacy and security threats against your product as you can think of. Privacy addresses someone else reading a user's sensitive data, and could even include someone else finding out that so-and-so is a user of your application. Security addresses someone else changing another user's data, changing other application's own data, taking control of the application, preventing others from using the application, etc. Use your imagination. 15 points.

Points	100
Submitting	a file upload
File Types	pdf, doc, docx, and txt

Due	For	Available from	Until
Dec 11, 2018	Everyone	Dec 4, 2018 at 10:10am	Dec 12, 2018 at 12:10am

+ Rubric