Maximum total 100 points. Part 0 is 0 points, parts 1 and 2 are 30 points each, and part 3 is 40 points. **Although part 0 will not be graded, we will not grade parts 1-3 unless the required part 0A has been completed - including a meaningful explanation, not just numbers; parts 0B and 0C are optional.**

You are NOT permitted to post, discuss or work together on the assessment questions with other students in the class or with anyone else. Piazza will be closed for the duration of the assessment period except for private messages to the instructor. Your answers can be as long or as short as you want, but a long meandering answer that incidentally mentions something relevant will receive less credit than a short high-quality answer. Submit your answers all in a single document in response to this assignment on courseworks. You can submit as many times as you want until the deadline. Submissions via email, piazza, or any other means besides courseworks will not be graded.

Part 0 (0 points).

Part 0A **(required**): You have 100 total points to allocate among the members of your team. **Assign some number of points (possibly 0) to each member of your team, based on what you believe was that student's relative participation in the team project, and explain your scores.** Make sure the total is exactly 100. Remember to include yourself! For allocating points, consider level of effort and contributions, whether for team meetings and discussions, software development and testing, writing the assignment submissions, setting up and presenting demos, or anything else relevant to the team. Note equal participation is not the same thing as equal lines of code produced.

For example, in a team where all team members participated more or less equally, the scores would be: Cora 20, Hannah 20, Kaedi 20, Kaylah 20, Nina 20 (this team has 5 members), and your explanation would describe who did which part(s) of the coding, testing, document writing, demo preparation, etc.

In a team with a substantial disparity in contributions, however, the scores might be: Iron Man 35, Captain America 35, Hulk 30, Thor 0 (this team has 4 members), and your explanation might describe Iron Man and Captain America as doing most of the development, Hulk has weaker programming skills so did more of the testing and document writing, but Thor rarely showed up for team meetings and did not seem interested in the project. Again, your explanation should describe who did what, or who did not do what as the case may be.

part 0B (optional): This is your opportunity to provide feedback about the course (separate from the SEAS "evaluations"). 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? Please explain. 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.

Please do not complain about being forced to take the course online using zoom, since there was no choice in this matter. However, it would be valuable for you to make suggestions about how the lectures and other aspects of the course could have been handled better, given that zoom/online was required, since online will also be mandated in the spring and possibly even next fall.

You can also post comments *anonymously* to the "instructors" or to "kaiser" on piazza.

part 0C (optional): Fill in and submit this survey about past programming experience: https://forms.gle/69Ss2t8o3mWJzr4C7

(Links to an external site.)

(Links to an external site.)

Part 1 (30 points)

Consider the final user stories for your project as reported in assignment T5 (the second iteration report).

A. (21 points) Develop a set of CRC cards corresponding to those user stories. You do not need to submit anything that visually looks like cards, the following format is fine:

Class: <name of class>

Responsibilities: <list of responsibilities>

Collaborators: <list of collaborators>

B. (9 points) Then discuss the similarities and differences between your CRC design and your actual codebase. You will not be penalized if your CRC design does not match your code; please develop the CRC cards from your user stories, not from your code.

C. (we won't grade the question if this is missing) Please include a link to your team's github repo. Do not write any new code or change your repository in any way. Remember you must do this assessment independently, not together with your teammates.

Part 2 (30 points)

Consider your team's codebase. Pick some non-trivial part of the codebase where you were not the primary author, that is, the primary author is some other member of your team. (If you wrote all the major parts of the codebase yourself, contact the instructor on piazza asap for guidance.)

Conduct an informal review of this part of the codebase: Read the code line by line and look for problems that are unlikely to be caught by a style checker or bug finder.

A. (30 points) Write a report telling us about any issues you found. For example, are there any issues with identifiers (names of methods, variables, types, etc.)? Are there any issues with comments (or lack thereof)? Is the code well-organized? Is the code readable? You must think of something to say, you will get zero credit for a content-free review like "all is good". There is no specific format required for the report.

B. (we won't grade the question if this is missing) Make sure to include the names of the specific file(s) covered in your review. In most cases your report should cover at least one entire file. (If your files are extremely tiny, e.g., one method per file, or extremely large, e.g., essentially the entire project functionality is implemented by a single file, those are organizational issues to report!)

C. (we won't grade the question if this is missing, yes give it *again*) Also please include a link to your team's github repo. Do not write any new code or change your repository in any way (except it's ok to fix any problems you found after the final version of this assessment has been submitted). Remember you must do this assessment independently, not together with your teammates.

Part 3 (40 points)

This semester started with an individual project where a program "skeleton" was provided and the assignments involved filling in the feature code, adding a database, and testing. Imagine that your own team's project was chosen to form the basis of next year's individual project.

The individual project would consist of three assignments, to implement the basic functionality (2 weeks), to add testing (1 week), and to add generic functionality like a database or authenticated login (1 week), respectively.

A, (20 points) What would you add/remove from your project implementation to provide the "skeleton"? (We'll assume you remove the database, if any, and authenticated login so you do not need to discuss these parts.)

B. (20 points) What functionality would you task the students to implement for the first assignment? Keep in mind the students will be working alone, would not start out familiar with your project idea, would only have 2 weeks, and your team (probably) won't be available to answer questions - so this might be a small subset of your project.

C. (we won't grade the question if this is missing, yes include the url one more time) Please include a link to your team's github repo. But do not write any new

code or change your repository in any way. Remember you must do this assessment independently, not together with your teammates.

Do not be concerned with what language(s) your project is implemented in, we'll assume for the purposes of this assessment that all students taking the class will already be familiar with whatever language(s) you used.

This is intentionally a very open-ended question.

This is the end of the assessment.