First Individual Assessment

Published Edit ⋮

Maximum total 100 points. Part 1 is 50 points, part 2 is 20 points and part 3 is 30 points. There is no credit for part 0, but 0A is required in the sense that we will not grade parts 1 and 2 if part 0A is missing or incomplete and you will receive zero credit for the assessment. Part 0B is 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 other than the teaching staff. 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 and will receive zero credit.

Ungraded questions: part 0A is required and part 0B is optional.

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 thus far, and explain your scores.** Make sure the total is exactly 100. Remember to include yourself! For allocating points, consider level of effort and contributions for anything relevant to the team project. Note equal participation is not the same thing as equal lines of code produced. Please do not predict the future.

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 team project effort thus far. Small distinctions like one team member did 21 while another did 19 are not of interest.

In a team with a substantial disparity in contributions, however, the scores might be: Iron Man 35, Captain America 35, Hulk 25, Thor 5 (this team has 4 members), and your explanation might describe Iron Man and Captain America as doing most of the work, and Hulk helped, 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, which in the instructor’s opinion do not ask the right questions. Is there anything that you think should have been covered already but wasn't? (Various topics will be covered later in the semester.) Is there anything that was covered unnecessarily or should have been put off until later? 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 the remainder of this course will be greatly appreciated. The instructor does not, however, have any control over classroom assignment and layout or anything CVN does or does not do.

Graded questions: There are three graded questions, part 1, part 2 and part 3.

Part 1 (50 points): Consider the service your team has proposed as your team project. Imagine the US federal government has imposed a new rule that all software services operating in the US provide "special access" for the NSA to monitor all activities of all client services and applications using the service. Your service operates in the US and your small company has no means to fight this new rule; however, a large company named "Orange" is fighting the rule in court. Removing your service from the US market is not an option for the purposes of this assessment. Devise features that you would need to add to your service to support the "special access" rule. Assume your development team will implement these features, and the NSA will not modify your codebase or its dependencies in any way. Include a *kill switch* that can be instantly activated to turn off all these features temporarily, even when their use is in progress, should Orange obtain a temporary suspension of the rule from a federal court. Keep in mind you may need to turn all these features back on again at any time, and back off again at any time, as Orange's case works its way through the courts. Include in your answer a link to your team project github repository and make sure the instructor and the IAs have collaborator access.

Part 1 has two graded subparts, A and B.

part 1A (25 points): First briefly state the gist of what your service does. Then write one or more user stories for the new "special access" functionality to be added to your service. These user stories should reflect what "monitoring" the use of your service could mean in the context of the features provided by your specific service. Maybe the NSA wants to know whether foreign nationals are using your service to infiltrate art schools or to trade photos of kittens (if information about art schools or photo sharing is supported by your service, respectively). Finally, write one or more user stories about the kill switch.

The article at **https://www.reuters.com/article/us-usa-security-congress-insight/spy-agency-ducks-questions-about-back-doors-in-tech-products-idUSKBN27D1CS** **(https://www.reuters.com/article/us-usa-security-congress-insight/spy-agency-ducks-questions-about-back-doors-in-tech-products-idUSKBN27D1CS)** explains that such a feature is not as far-fetched as it sounds.

Write at least two user stories, not a single huge user story, at least one about the NSA monitoring and at least one about the kill switch. User stories that do not address the NSA monitoring or the kill switch will not count towards the two. Your user stories must use this format:

< label >: As a < type of user >, I want < some goal > so that < some reason >

My conditions of satisfaction are < list of common cases and special cases that must work >

part 1B (25 points): First write one or more use cases for the new “special access” functionality to be added to your service. These use cases should reflect what “monitoring” the use of your service could mean in the context of the features provided by your specific service. Then write one or more use cases about the kill switch. Your use cases do not need to match your user stories.

You can assume that the NSA will provide a special API for authenticating itself. The article at **https://tutanota.com/blog/posts/encryption-backdoor-fails/** **(https://tutanota.com/blog/posts/encryption-backdoor-fails/)** gives an idea of how well you should expect that API to work, but use it anyway.

Write at least two use cases, not a single huge use case, at least one about the NSA monitoring and at least one about the kill switch. Use cases that do not address the NSA monitoring or the kill switch will not count towards the two. Each of your use cases should include at least a title, primary actor, preconditions, postconditions, basic flow, and one or more alternative flows. Make sure to label every step in each flow, including the branch points for alternative flows.

Part 2 (20 points): Recall our in-class exercises discussing the APIs needed to implement a CONTAIN app (**CON**tact **T**racing for instruction **A**ss**I**sta**N**ts). If you do not attend class regularly, watch the videos and read the lecture slides; CONTAIN has been discussed during at least five different class sessions.

Write a set of CRC cards for the functionality of the CONTAIN app as discussed in class (put the content of your “cards” in the document to submit on courseworks, do not use physical cards). All of the CONTAIN app functionalities discussed in class should appear as responsibilities of some component. All the APIs to external services discussed in class as being useful for CONTAIN should appear as collaborators of one or more components. Make sure you write a *set* of CRC cards corresponding to multiple components of the CONTAIN app, not a single huge card.

Part 3 (30 points, 5 points per principle): Consider **your own implementation** of the Connect-4 game mini-project from the viewpoint of design principles. Use the latest version you submitted; do not change your code. Discuss each of the following six design principles. If your implementation – including the skeleton code - violates the principle, explain how and in what way it does so. If it does not violate the principle, explain how it might be modified to intentionally violate the principle. Your suggested modifications need to be realistic for the Connect-4 game and its implementation based on the skeleton code, do not add arbitrary features. Include a link to your mini-project repository and make sure the instructor and the IAs have collaborator access.

1. Single Responsibility Principle
2. Open-Closed Principle
3. Liskov Substitution Principle
4. Interface Segregation Principle
5. Dependency Inversion Principle

6. Don’t Repeat Yourself Principle

This is the end of the assessment.

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

Due	For	Available from	Until
Oct 30, 2021	Everyone	Oct 27, 2021 at 12:01am	Oct 31, 2021 at 11:59am

First Assessment

You've already rated students with this rubric. Any major changes could affect their assessment results.

Criteria	Ratings			Pts
1a. presenting a reasonable description of what their service does	**3 pts** **Full Marks**	**0 pts** **No Marks**		3 pts
1a. one user story about an NSA agent (or multiple agents) “monitoring” use of the service	**12 pts** **Full Marks**	**6 pts** **baseline user story but no meaningful conditions of satisfaction about the queries/filters**	**0 pts** **No Marks**	12 pts
1a. one user story about a service administrator “kill switch” that turns off and on NSA’s monitoring	**10 pts** **Full Marks**	**5 pts** **baseline user story but no meaningful conditions of satisfaction that should include the status of NSA access**	**0 pts** **No Marks**	10 pts
1b. presenting a reasonable description of what their service does	**3 pts** **Full Marks**	**0 pts** **No Marks**		3 pts
1b. use case (or multiple use cases) about an NSA agent (or multiple agents) “monitoring” use of the service in the sense above for part 1A.	**12 pts** **Full Marks** 2 of these points for using the NSA-provided authentication API somehow, either in a precondition/trigger or an early step, to make sure the access is really from the NSA. 2 of these points for at least one precondition that sounds plausible such as a previously flagged user doing something with the service (a trigger instead of or in addition to a precondition is fine) 2 points for at least one postcondition that seems plausible, 3 points for a basic flow with multiple steps that look reasonable such as running queries, aggregating queries, converting data among formats, etc. 3 points for at least one alternative flow with a plausible branch point from the basic flow such as when the NSA authentication fails, no user or data matching an NSA query/filter can be found, or there’s some glitch like database or some dependency “down”, “times out”, etc.	**0 pts** **No Marks**		12 pts
1b. use case (or multiple use cases) about a service administrator “kill switch” that turns off and on NSA’s monitoring in the	**10 pts** **Full Marks** 2 points of this for at least one precondition concerned with Orange’s court case (a trigger instead of or in addition to a precondition is fine) 2 points for at least one postcondition such as status of NSA access or non-interruption of regular user access 3 points for a basic flow 3 points for at least one alternative flow with a branch point from the	**0 pts** **No Marks**		10 pts

Criteria	Ratings			Pts
sense above for part 1B.	basic flow – turning off might be basic flow and turning on might be alternative flow, or vice versa, or these might be separate stories, alternative flows might also involve some glitch like database or some dependency "down", "times out", etc.			
2. correct format	**2 pts** **Full Marks**	**0 pts** **No Marks**		2 pts
2. using (somewhere) the CANVAS, SSOL and drone APIs as collaborators.	**3 pts** **Full Marks**	**0 pts** **No Marks**		3 pts
2.covering the baseline functionality of using CANVAS to get list of IAs and students in a class and then including IAs as well as students in contact tracing for that class.	**5 pts** **Full Marks**	**0 pts** **No Marks**		5 pts
2. using CANVAS or SSOL to trace through all the different classes taken by a student including IAs.	**5 pts** **Full Marks**	**0 pts** **No Marks**		5 pts
2. filtering to only consider students and IAs who actually attended the same class session as recorded by the drone	**5 pts** **Full Marks**	**0 pts** **No Marks**		5 pts
3. Single Responsibility Principle	**5 pts** **Full Marks**	**2 pts** **Understands principle but doesn't correctly apply**	**0 pts** **No Marks**	5 pts
3.Open-Closed Principle	**5 pts** **Full Marks**	**2 pts** **Understands principle but doesn't correctly apply**	**0 pts** **No Marks**	5 pts

Criteria	Ratings			Pts
3. Liskov Substitution Principle	**5 pts Full Marks**	**2 pts Understands principle but doesn't correctly apply**	**0 pts No Marks**	5 pts
3.Interface Segregation Principle	**5 pts Full Marks**	**2 pts Understands principle but doesn't correctly apply**	**0 pts No Marks**	5 pts
3.Dependency Inversion Principle	**5 pts Full Marks**	**2 pts Understands principle but doesn't correctly apply**	**0 pts No Marks**	5 pts
3.Don’t Repeat Yourself Principle	**5 pts Full Marks**	**2 pts Understands principle but doesn't correctly apply**	**0 pts No Marks**	5 pts
				Total Points: 100