

Assignment T1: Preliminary Project Proposal

✓ Published

 Edit





This is a team assignment. This assignment will be graded as 0 to max points. Scroll down for submission instructions at the end.

Part 0:

If your team name, team membership, planned programming language, platform and/or team github repository has changed since your team formation assignment, please explain. If not, you do not need to include part 0 in your submission.

Part 1:

Write a few paragraphs that provide an overview of the software engineering project that your team would like to do and answers all five of the numbered questions. 1. What will your project do? 2. Who or what will be its users? Your project must impose registration, authenticated login and timeout/explicit logout, so your answer should say something about this. 3. Your project must be demoable, but does not need a GUI if there's a command line console or some other way to demonstrate. (All demos must be entirely online, there will be no in-person demos.) What do you think you'll be able to show in your demo? 4. Your project must store and retrieve some application data persistently (e.g., using a database or key-value store, not just a file), such that when your application terminates and starts up again some arbitrary time later, the data is still there and used for later processing. What kind of data do you plan to store? 5. Your project must leverage some publicly available API beyond those that "come with" the platform; it is acceptable to use an API for external data retrieval instead of to call a library or service. The API does not need to be a REST API. There are many public APIs linked at <https://github.com/public-apis/public-apis>  (<https://github.com/public-apis/public-apis>) and <https://github.com/n0shake/Public-APIs>  (<https://github.com/n0shake/Public-APIs>). What API do you plan to use and what will you use it for?

Part 2:

Write three to five user stories for your proposed application, constituting a Minimal Viable Product (MVP); registration/login/logout should not be included among these user stories, nor should 'help' or other generic functionality. That is, your application should do at least three application-specific things. Use the 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 >.

Course Chat



The type of user (role) does not need to be human. You may add additional user stories to add if time permits. Keep in mind the reason (if applicable within the system), all the common cases testable and demoable.

Part 3:

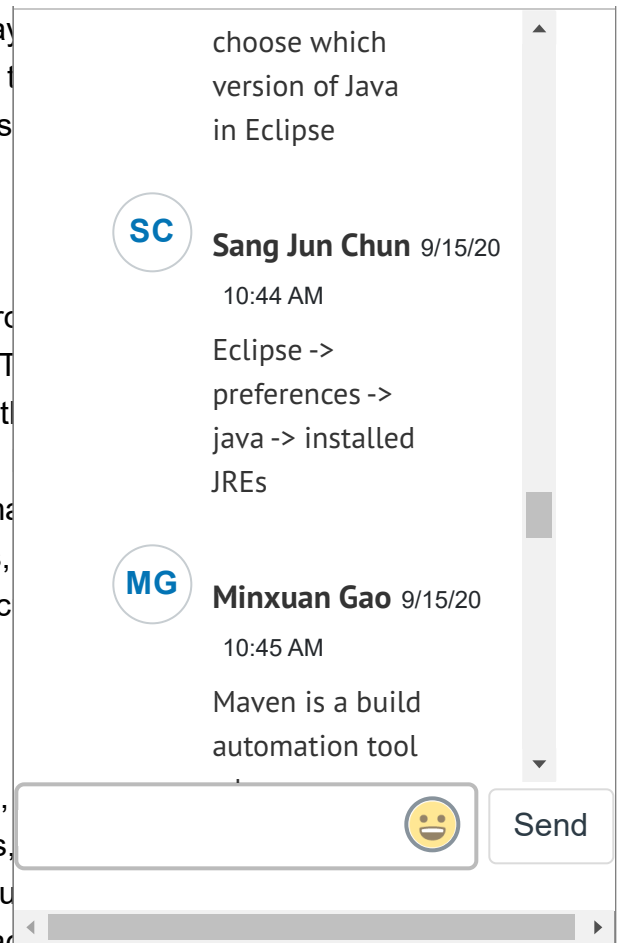
Explain how you will conduct acceptance testing on your project. Your story must be associated with a plan for user-level testing. Test common cases and special cases. Discuss sample inputs to the system, results expected for the corresponding test to pass vs. fail. Test on different network, devices, etc., not necessarily from a GUI or command line. Discuss changes in application state, files, outgoing network traffics, etc. Test outputs via a GUI or command line. You may optionally discuss additional user stories, if any.

Part 4:

Identify the specific facilities corresponding to JDK, Eclipse, Spotbugs, and SQLite that your team plans to use. That is, for each tool, a compiler/runtime (or equivalent), an IDE or code editor, a build tool (if not applicable), a style checker, a unit testing tool, a coverage tracking tool, a bug finder (that's not just a style checker), and a persistent data store appropriate for your chosen language(s) and platform(s). If different members of the team plan to use different tools, please explain. It is ok to change your choice of tools later after you start developing your application.

You can discuss this assignment in this course's team_project folder on [piazza](https://piazza.com/columbia.edu/spring16/cs4335), (https://courseworks2.columbia.edu/courses/104335/external_tools/1456) on relevant community forums, on stackoverflow, etc. However, do not ask for someone else outside your team to define your project, user stories, acceptance tests, etc. ; this is a violation of the university's academic honesty policy.

Submission instructions: One member of your team should submit a single file (preferably pdf) presenting your preliminary proposal. Your file must contain your team name and the names/unis of all team members, and may optionally include links to external resources. You may submit repeatedly until the deadline (note that if multiple team members submit, the most recent submission will override all previous submissions by the same or different team members).



Points 20

Submitting a file upload

File Types pdf, doc, docx, and txt

Due	For	Available from	Until
Oct 27, 2020	Everyone	Sep 1, 2020 at 12am	Nov 3, 2020 at 11:59pm

Preliminary Proposal

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

Criteria	Ratings		Pts
Part 1`Overview 1 point for what the project will do, does it sound reasonable. 1 point for who or what will be its users and how will authentication be handled ("what" instead of "who" comes up if the project provides a service to other software). 1 point for does demo plan sound reasonable. 1 point for explaining what data will be stored. 1 point for is there an external API and does it seem appropriate for this project.	5 to >0.0 pts Full Marks	0 pts No Marks	5 pts
Part 2 User Stories 2 points per user story up to 3 user stories and 6 points, i.e., 2*3. (If 4 or more user stories are given, grade the best 3 and give no more than 6 points total. No credit for generic capabilities.) For each of those 3 user stories, the 2 points are divided between 1 point for "< label >: As a < type of user >, I want < some goal > so that < some reason >" portion and 1 point for "My conditions of satisfaction are < list of common cases and special cases that must work >" portion. Do these all sound reasonable and is the correct format used?	6 to >0.0 pts Full Marks	0 pts No Marks	6 pts
Part 3 Acceptance Testing 2 points each for 3 best user story testing plans, no more than 6 points total (i.e., 2*3). Each user story should be associated with at least two plausible-sounding system-level tests (not unit tests), 1 point for testing a common case and 1 point for testing a special case, e.g., user/client or system error. (If more than two tests are described for the same user story, still max 2 points per user story).	6 to >0.0 pts Full Marks	0 pts No Marks	6 pts
Part 4 Technology Chosen 1/2 point each for mentioning an IDE or code editor, a build tool (or package manager if 'build' not applicable), a style checker, a unit testing tool, a test coverage tracking tool, and a bug finder (that's not just a style checker, although its ok to use a style checker that is part of the bug finder) appropriate for the team's chosen language(s) and platform(s).	3 to >0.0 pts Full Marks	0 pts No Marks	3 pts
Total Points: 20			