Assignment T3: First Iteration

Published | Edit | ⋮

Implement a demoable version of your service and make sure all the implementation code has been committed to the main branch of your github repository (its ok to have some other code in branches but it will not be considered part of your submission). This version of your service does not need to include all the functionality you proposed, but some API entry points must be operational in order to demo.

Implement and run system tests that exercise your service's initial API, e.g., using Postman or some other API testing tool. If you use Postman, please sign up for a free account using **this link** (https://docs.google.com/forms/d/e/1FAIpQLSdWrdayMvsD1PssUA3Za6l5dLpbwprW7UpAQwTdXqkQdDW
usp=pp_url&entry.1767743879=kaiser%40cs.columbia.edu&entry.1788256813=Columbia%20University&er
, which takes you through "training", and then use **this other link** (https://app.getpostman.com/join-team?invite_code=1694c264c4364da325b2176faa0b3cb9) to join the class space. Your tests should exercise the multiple clients and persistent data aspects of your service. Try to fix most of the bugs found by the tests.

Implement unit tests for all the major "units" (methods, functions, procedures, subroutines, etc.) in your code. Unit tests should be grouped so related code (e.g., classes, modules, files, etc.) share setup, teardown and mocks when appropriate. Your entire collection of unit tests should run "push-button" via some test runner appropriate for your language and platform. Try to fix most of the bugs found by the tests.

Make sure all your test code has been committed to the main branch of your repository (its ok to have some other test code in branches but it will not be considered part of your submission). Your tests do not need to achieve high coverage, yet, that will be part of the second iteration. You will also use a static analysis bug finder during the second iteration. Your can certainly use coverage and/or bug finder tools during the first iteration if you want, but this is not required.

All commit messages, except for the initial repo-setup commit(s), should say something meaningful about what was added/removed/changed.

All non-trivial code, including test cases, should be self-documenting with mnemonic identifiers and header comments and, where appropriate, in-line comments. (Trivial means things like getters, setters, small helper functions, and other self-evident code. There is no point in commenting **int addone (x) { return x+1 }**.)

Your entire codebase, including test cases and even trivial code, should be compliant with a style checker appropriate for your language/platform. Include some reports from your style checker in your repository. If it does not generate files, use redirection of output to a file, screenshots, or some other means to produce files.

Include in your repository any configuration files needed to build, run and test your codebase.

Write a README.md that describes in simple terms how to build, run and test your initial service.

The README.md should also describe all the operational entry points to your service. As noted above, this might be a subset of your proposed entry points. This is your "API documentation". If certain entry points must be called in certain orders or never called in certain orders, make sure to say so.

If any third-party code is included in your codebase, also document exactly which code this is, where it resides in your codebase, and where you got it from (e.g., download url).

Submission instructions: One member of your team should submit a url pointing to your codebase repository. Make sure your IA mentor has collaborator access! This should normally be the same repository from your revised proposal, but we ask you to submit again for this assignment to make it easier to use Coursework's "SpeedGrader" tool.

Points	50
Submitting	a website url

Due	For	Available from	Until
Nov 15, 2021	Everyone	Oct 1, 2021 at 12:01am	Jan 31, 2022 at 11:59pm

T3 First Iteration

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

Criteria	Ratings	Pts
Ignore imported code for all concerns. There do not need to be any test cases, comments, style compliance, etc. for any code documented in the README as third-party code.	This area will be used by the assessor to leave comments related to this criterion.	0 pts
Documented API Is their API documented in the README? Is it really a code-calls-code API, not a user-facing UI? Its fine to link to an API document hosted elsewhere.	This area will be used by the assessor to leave comments related to this criterion.	10 pts
System Tests Corresponding to API Are there code-calls-code system-level tests exercising every API entry point?	This area will be used by the assessor to leave comments related to this criterion.	10 pts
Unit Tests Are there unit tests for most major methods and are they organized to all run automatically from a "push-button" unit testing tool?	This area will be used by the assessor to leave comments related to this criterion.	10 pts
Commit Messages Check a few of the commit messages, do they say something meaningful about what changed?	This area will be used by the assessor to leave comments related to this criterion.	5 pts
Self-Documenting Code Check a few of the longer functions (or methods, whatever), do they have reasonable comments and meaningful function names and parameter names?	This area will be used by the assessor to leave comments related to this criterion.	5 pts
Style Compliant Can you find style checker reports in their repository and does it look like most of the problems were fixed?	This area will be used by the assessor to leave comments related to this criterion.	5 pts
Build, run, test instructions Does the README include instructions explaining how to build, run and test their service and does the repository contain all configuration files needed for build, run and test?	This area will be used by the assessor to leave comments related to this criterion.	5 pts
Partial credit Need to determine how to handle partial credit for all relevant items.	This area will be used by the assessor to leave comments related to this criterion.	0 pts
		Total Points: 50