Assignment T1: Preliminary Project Proposal

Published | Assign To | Edit | ⋮

This is a team assignment. See submission instructions at the end.

Invent an idea for a useful server (or service) that 1. maintains some relevant data persistently during its operation, 2. supports multiple clients, and 3. has no GUI.

Maintaining data persistently means that when your server terminates and starts up again some arbitrary time later, the data still exists and can be used and modified by later processing. The datastore should be read/write, not append-only. Your server is not required to support crash recovery, in the sense of the machine suddenly losing power or rebooting, only intentional termination (shut-down).

Multiple clients means some of the data may be associated with one client and some with another. There may also be some aggregate or shared data. Your server needs to be able to tell the clients apart and it should provide basic protection for the integrity and confidentiality of client-specific data, protecting it from other clients. You do not need to consider carefully crafted hacker exploits or protecting client data from the server-provider itself. It is strongly recommended that you use some "free" widely-used authentication package, e.g., **Auth0** (https://auth0.com/docs/), not develop the security code yourself. Clients can be other servers and services with their own clients, not just end-user apps.

Instead of a GUI (graphical user interface), your server should provide an API (Application Programming Interface). The API does not need to be a REST API. The API inputs might be intended to ultimately come from some application that does have a GUI (or multiple different GUI apps), and you can optionally built a sample GUI client as part of the second iteration, but for the first iteration you need to develop a server with no GUI (and no pseudo-GUI like **ascii art** (https://www.asciiart.eu/)) and devise some non-GUI way to demonstrate that your server "works", e.g., using **Postman** (https://www.postman.com/) or an administrative CLI (command line interface). There should not be any way for end-users to access the server except via a separate client over an internet connection.

Web servers, application servers, content servers, database servers, etc. are not acceptable; your server must "do something" other than host other applications and services. You can look at **https://github.com/public-apis/public-apis** (https://github.com/public-apis/public-apis) and/or **https://github.com/n0shake/Public-APIs** (https://github.com/n0shake/Public-APIs) for inspiration but you need to do something at least margically different, not reimplement verbatim one of the APIs listed on those pages.

Part 1:

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, simply state that nothing has changed.

Part 2:

Write a few paragraphs that provide an overview of the server that your team would like to develop and answers these three sets of questions: 1. What will your server do? What kind of functionality or features will it provide? 2. Who or what will be its clients? What might they use the functionality for? 3. What kind of data will your server create or accumulate? What will the data be used for? For this assignment, you do not need to present your features as user stories or use cases.

Part 3:

Write a few paragraphs that describe, at a high level, how you plan to test the functionality of your server without any clients, GUI or otherwise. (It's ok to use testing tools that have GUIs.) You will expand testing in the second iteration to include sample clients, but need to test stand-alone during the first iteration. Think in terms of testing your server as a whole, in addition to unit testing of individual subroutines, and answer these three questions: 1. How will you test that your server does what it is supposed to do and provides the intended functionality? 2. How will you check that your server does not behave badly if its clients use it in unintended ways or provide invalid inputs? 3. How will you test that your server handles its data the way its supposed to?

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. 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
Sep 26, 2022	Everyone	Sep 1, 2022 at 12am	Sep 27, 2022 at 11:59pm

Team Assignment 1

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

Criteria	Ratings	Pts
Part 1 1 point each for 1. team name, 2. names and unis of all team members, 3. programming language 4. platform 5.Github repo	This area will be used by the assessor to leave comments related to this criterion.	5 pts
Part 2 2 points each for answering the 3 questions: 1. What will your service do? 2. Who or what will be its users? 3. What kind of data will your service create or accumulate? What will the data be used for?	This area will be used by the assessor to leave comments related to this criterion.	6 pts
Part 3 2 points each for answering 3 questions: 1. How will you test that your service does what it is supposed to do and provides the intended functionality? 2. How will you check that your service does not behave badly if its clients use it in unintended ways or provide invalid inputs? 3. How will you test that your service handles its data the way its supposed to?	This area will be used by the assessor to leave comments related to this criterion.	6 pts
Overview Project ideas meet the 3 requirements: 1. Maintain relevant data persistently 2. Supports multiple clients 3. No GUI	This area will be used by the assessor to leave comments related to this criterion.	3 pts
		Total Points: 20