Assignment 5

This assignment consists of two parts, both team efforts:

- A brief presentation to your discussion section of your web server's current API, **happening on Friday, May 2.**
- A written proposal, based on what you've learned, of what you think would be a good API for your web server, due by **noon on Monday, May 5.** This should be submitted by the TL in a first-draft version of your assignment submission form by noon on Monday, May 5 so we can use your submissions during lecture that day.

Each student should submit this assignment by 11:59PM on May 6, 2025 into the submission form. The TL should re-submit the submission form after any cleanup code changes have been submitted for this assignment so they are properly counted.

TABLE OF CONTENTS

API proposal

In the course of developing your web server, you and your team have (perhaps unintentionally) developed and specified multiple APIs that can be used to extend the functionality of your web server:

1. The request handler interface
2. The format of the configuration file

Think about these APIs. *Are they reusable? Are they easy to extend? Is there anything you'd like to change?*

Prepare a brief presentation on your web server's current (or proposed) API. The presentation should be 5 minutes or less, so all teams have time to present.

Your presentation should explain:

- Your request handler interface
  - *What are the important methods?*
  - *What do these methods do?*
  - *When are these methods called?*
  - *Who calls them?*
  - *How could one test the request handlers?*
- Your configuration file format, and an example configuration file
  - *How are server-level parameters (like the port number) specified?*
  - *How are URLs associated with request handlers?*
  - *How are request handlers configured?*
- The mechanism for connecting the config file and request handlers (i.e. the dispatch mechanism)
  - *How is the config file used to construct request handler objects?*
  - *What is the lifetime of request handler objects?*
  - *How are HTTP requests parsed and dispatched to the correct request handler?*

Your API does not need to be exactly what is currently implemented. You can propose improvements based on what you've learned so far.

Your presentation will be given this Friday, May 2, in discussion section.

API design document

After you've presented your API, and seen the presentations from other teams, your team will write a design document (*design doc*) for the APIs discussed above. Your final design doc may differ from your API proposal if you would like to incorporate good ideas from other teams.

Your design doc should cover the same things as the presentation, but in more depth, and based on what you've learned from other teams' proposals. You should include:

- A description of the request handler interface. Basically, write and explain a `request_handler.h` header spec
- A description, with multiple examples, of the configuration file format.
- A description of the web server's dispatch mechanism.

For each API choice you make, you should provide some justification. Also discuss any alternatives you considered, and why you rejected them.

For a good background on the purpose and scope of a design doc, read this blog post, which only coincidentally was written by a Google engineer. 🤷‍♂️

Please use the template below to frame your design doc. Italicized text is intended only for guidance and should be removed.

API Design Proposal

Team: *Your team name here*

Authors: *Your person names here*

Objective

What are you doing and why? What problem are you trying to solve? Include major goals and non-goals. Stay high-level, maximum 1 paragraph. Leave the details for later sections.

Background

What is the background of the problem? Where is this design intended to be implemented? Consider including links to your code repository or other documentation.

Requirements

What are the requirements that your design should meet? What are the detailed goals of the solution? What are the explicit non-goals that you are choosing to ignore? The guideposts of your problem as stated here should help inform the ***Detailed design*** *below.*

Detailed design

Here is where you describe your design in detail. Include any code snippets, example configurations, and other details here. Create sub-sections for each major component you are discussing.

Alternatives considered

List any alternative design choices you considered, and justify why you ultimately did not choose them. Justifications may reference points in the ***Requirements*** *section (e.g. failing to meet goals, or unnecessarily meeting non-goals).*

Code cleanup

No new functionality is required for this assignment, but you should still be writing at least a few changelists on the project. Use the time to clean up code, refactor to make your code quality better, add comments, improve readability, improve test coverage, repair flaky tests, improve logging, or make any other changes we've talked about.

Individual Contributor criteria includes:

- Code submitted for review (follows existing style, readable, testable)
- Addressed and resolved all comments from TL

Tech Lead criteria includes:

- Kept assignment tracker complete and up to date
- Maintained comprehensive meeting notes
- Gave multiple thoughtful (more than just an LGTM) reviews in Gerrit

General criteria includes how well your team follows the class project protocol. Additional criteria may be considered at the discretion of graders or instructors.

Submit your assignment

To facilitate the API discussion, your team should add 1 or 2 slides to your lab section's slide decks outlining the API your web server currently has:

- Lab 1A1

- Lab 1A2
- Lab 1B1
- Lab 1B2
- Lab 1C1
- Lab 1C2
- Lab 1D1
- Lab 1D2

Everyone should fill out the submission form before 11:59PM on the due date. We will only review code that was submitted to the `main` branch of the team repository before the time of the TL's submission. You may update your submission by re-submitting the form, which will update the submission time considered for grading purposes if you are the TL.

Late hours will acrue based on the submission time, so TL's should avoid re-submitting the form after the due date unless they want to use late hours.

"You may think using Google's great, but I still think it's terrible." —Larry Page