TL;DR

- What: Each team member must document the developer interfaces for the modules for which he or she is responsible
- Due: See Piazza or calendar
- Where: Google folder and GitHub

Background

For efficiency, large development teams often need to work in parallel on different components of a software product, even when these components interact with one another. This semi-independent development can occur when there is a well-defined interface for the various components that other developers will use. While in many commercial development environments your team would be small enough that communicating these interface details can occur bilaterally and informally, you will be required to demonstrate that you have given focused, thoughtful consideration to interface design and to how your team has decided to divide up the work and tackle it in parallel. And as the cartoonist Richard Guindon is purported to have said:

> *"Writing is nature's way of letting you know how sloppy your thinking is."*

While Agile development practices do not encourage the completion of big design work before starting to code, high-performing development teams do start with a broad understanding of how the major sub-systems and components will be organized and how they will interact. Different team members will tend to focus on different areas based on a combination of their current skill and experience, where they are trying to grow their knowledge, and on what the team needs to ensure appropriate redundancy in system understanding.

Assignment

You will come an agreement with your teammates regarding how to divide up responsibility for various components or sub-systems. Each member of the team will document a specification of the programmatic, or developer-centric, interface to the components for which he or she is responsible. You should not document the user- or customer-interface. While you may (and probably should) use automatically-generated documentation to assist you, you should not rely solely upon such practices. Your documentation must be detailed enough so that, notionally, a new developer could be added to your team and could start using your modules using your documentation alone. It may include diagrams or other visual representations. Your document should include a small section describing the overall project and how responsibility for components is divided among team members; each member can use the same content for this section, but beyond that the rest of the document should be yours.

For example, the following is a list of items that should be documented, if they apply to your project:

- Network service invocations (e.g. RESTful services) to include formats (e.g. JSON) for passed data
- APIs for libraries your team develops (and ABIs if applicable)
- Data serialization formats (e.g. Avro, Google Protocol Buffers, JSON)
- File formats (if using files to pass data between components)

- What authentication, if any, is required to access the interface
- Other Remote Procedure Call details

There are several examples of well-documented interfaces available online. Many of these are for very robust and mature systems, and there is no expectation that your documentation will be as extensive as these, but they are provided anyway for inspiration:

- REST calls:
  - Stripe API https://stripe.com/docs/api
- Descriptions of errors:
  - Mailchimp https://developer.mailchimp.com/documentation/mailchimp/guides/error-glossary/
- Software API:
  - BeautifulSoup https://www.crummy.com/software/BeautifulSoup/bs4/doc/

By the end of the year, it is likely that the interfaces may change some from what is documented here. This is an inevitable and acceptable evolution of the software engineering process and will not necessarily count against your ultimate grade. This assignment is intended to motivate you to consider critical organization and development decisions early, and to practice your ability to capture and communicate them to others.

Grading

This assignment will be evaluated for the following:

- Design: does the interface represent good software engineering practices?
- Completeness: does the documentation completely describe the interface and how to use it?
- Clarity: is the documentation well-written, precise, and easy to understand?
- Structure: Is the document structured and organized in a logical manner?
- Responsibility: Are you adequately contributing to the team with the responsibilities you have accepted?

Form & Submission

Create a document with the content described herein in your Google folder. Post PDF version of the document to your web page in the gh-pages branch of your repository. The document should use 11pt. font, single-spaced. Please see Piazza or the calendar for the deadline.