Design Document Guidelines:
Date Feb 19, 2024

How to write a design document:
The audience of your design document is the TA/instructor. The rationale behind a design document is that it helps you plan your design so you can surgically make the requisite changes/additions to the code and avoid designing on the fly. A design document should reflect project research, planning, and brainstorming before coding.

Suggested Layout:
Software engineers, computer scientists, and theorists often take a top-down approach. Start with high-level goals/objectives, then explain how you will achieve these goals.

The recommended format is as follows:
1) Overview
2) In-depth Analysis and Implementation
3) Risk Analysis
4) Assignment Questions

1. Overview
Start with a few sentences on the goals of your assignment. Then, decompose the assignment into the major parts you must complete. For example, in assignment 1, a good topic breakdown might be:
• Identifying processes
• fork
• execv
• waitpid/exit
• Other system calls

For each section, start with a few sentences describing the goals. For example, the goal of the system call implementation might be to simplify kernel calls, prevent user code from corrupting the kernel, and allow the core system call implementation to be called from within the kernel without protection checks. List the critical challenges to accomplishing these goals, e.g., translating addresses or releasing locks correctly on error conditions.
In addition, you should describe (where applicable):
• How the different parts of the design interact together. Think modules.
• Major data structures. Heaps, stacks, queues, and more.
• Synchronization. How is this achieved? Challenges?
• Major algorithms.

2. In-depth analysis and implementation
Be specific. Decompose the design into:
1) The functions to be implemented
2) The existing functions to be modified

3) Corner cases that need to be handled (list all possible error conditions returned from each system call)
4) Test plan
You must analyze the subtitles and hidden problems that you may encounter. Psuedo-code can help

It might be helpful to write down which files must be modified for each part. Read through the Minix code to determine where the code should be placed.

The objective of this section is to convince the TA that the goals and challenges laid out can be addressed.

3. Risks
Engineering is not risk-free. Thus, it becomes essential to manage risks. The design document should convince the reader that you have explained every problematic detail. Write down every detail you still need to figure out. Then, refine this as you better understand the code; you can move items out of the risk category.

Provide time estimates for how long the project should take. If everything goes well (best case), if things turn out badly (worst case), and the expected time (average case). For each case, prioritize the features and indicate which ones you will skip if you cannot implement a particular feature. What are some workarounds if a feature is not implemented?

4. Assignment questions
Include any embedded questions from the assignment.

5. Final
Turn in the design document, a readme file indicating how to compile and run your code, and your source code. Do not submit executables, as your code will be re-compiled and tested.