opened using VS Code and command-shift-v to view

used free accounts and mac

Prompt #1 (p1): Convert this entire file into all raw MarkDown code so it's LLM friendly

- Directly convert to markdown without difference

Prompt #2 (p2): Convert this entire file into all raw MarkDown code and enhance it so it feels more modern and minimal

Prompt #3 (p3): I decided not to proceed with this prompt, since it asked the LLMs to change color or font, both of which MarkDown is incapable of.

Prompt #4 (p4): Convert this entire file into all raw MarkDown code and add features to make it more useful and interactive

Prompt #5 (p5): Convert this entire file into all raw MarkDown code and reformat it to make it more readable and maintainable

Prompt #6 (p6): Convert this entire file into all raw MarkDown code. Then pretend you're the lead dev on this project and change the MarkDown code to make 3 improvements you would prioritize to increase quality or user satisfaction

Google Gemini

Notes:

- Once html files get larger gemini could not process or produce anything
- Can't generate files, so limited amount of code able to produce
- Possibly because free account limitation
- slow - relative to gpt and copilot
- Doesn't reliably produce markdown
    - Prompt 3 gemini created html instead of md code
- Doesn't change much

ChatGPT

Notes:

- Can generate files and process large files
- Some files produced originally contained lots of white space and wasn't well formatted (changing the prompt helped)
- Suggested use of CSS to change color and font (but Mark Down unable to use CSS)

Microsoft Copilot

Notes:

- Unable to process html files directly
    - Had to convert from an html file to a pdf (provided in the drive for the ones that reached the word limits and couldn't be copy and pasted into the chat)
- Able to create downloadable files but unable to open it (possibly because usage of a mac)

- Able to process a large file and convert it but wasn't able to produce anything as large
    - Able to produce it in parts (entire xrootd references in 9 parts) (**v2**)
        - missed a lot of the content when produced in parts
        - For example, the notes for 1.2 (before 1.2.1) (esoteric options, developer options, defaults arguments, general notes, notes on esoteric options, notes on developer options)
    - Able to produce at once but missing most of the content and details and only contains main points according to copilot (**v1**)
- Best at providing raw code
- Prompt 4 and 5

Deepseek

Notes:

- Only able to read the first 16% of the html file for xrootd references (a long file)
    - Again, possibly because i'm using a free account
    - The portions it was able to produce was relatively accurate
- Hard to get it to produce the raw code not parsed
- Very slow
- For prompt 4 in deepseek it added in CSS and javascript components which MarkDown will not read and will only display

Ollama

Notes:

- Quick with short versions such as 3.4 (tls)
- Best and automatically producing code
- Simple, didn't ever change much