genAI Experiment: Restructuring Text

Rebecca Ellis

Who should use this genAI Experiment?

This genAI Experiment provides instructions on how to use genAI tools to improve efficiency and quality in technical writing. Technical writing tasks genAI can be useful for automating include, but are by now way limited to:

- Generating a sitemap file in Markdown
  - Flat or nesting
- Translation
- Embed a chatbot or agent on your documentation site by integrating available LLMs for more semantic searches.
- And this one, restructuring and reorganizing text to match an existing documentation template

Want to know more? Here is how I got started with my genAI experiment to restructure text!

Imagine a task you want to automate

- Breaking up and summarizing sections of text and assigning them to table of contents items into a static site template such as Hugo or Jekyll.

Define success

Metrics:

- Time saved

If possible, measure your baseline metric for the task prior to your AI experiment so you will have a comparison point.

It is estimated that a time to port long, less structured content into a template and summarize/ restructure it around the table of contents in the template would take ¾ workday to complete. Bear in mind this includes rewriting paragraphs, changing tone and approach to fit the template for a static website.

Make a plan for quality control

Ensuring quality of genAI output means initially spending more time and reviewing the content. After this initial step, the hope is that by automating the task, time will be saved down the road. Initially, time will need to be invested to make sure the AI does not hallucinate or give output in the wrong format.

My AI tool(s)

For this experiment, I used Google generative AI chat Bard for generation. I used Github to open the files to see their structure before beginning the experiment.

Prepare your prompts and examples

Prompts

First, I made a list of what I wanted the genAI to do for me, which became a preliminary list of prompts I wanted to give it. This scratchpad helped me organize my thoughts before asking Bard. I started with the…

Objective: break up a longer less structured text into sections according to the tale of contents in the template. Then, a nice-have would be a couple paragraphs summarized for a landing page.

And it grew into 10 different steps or prompts:

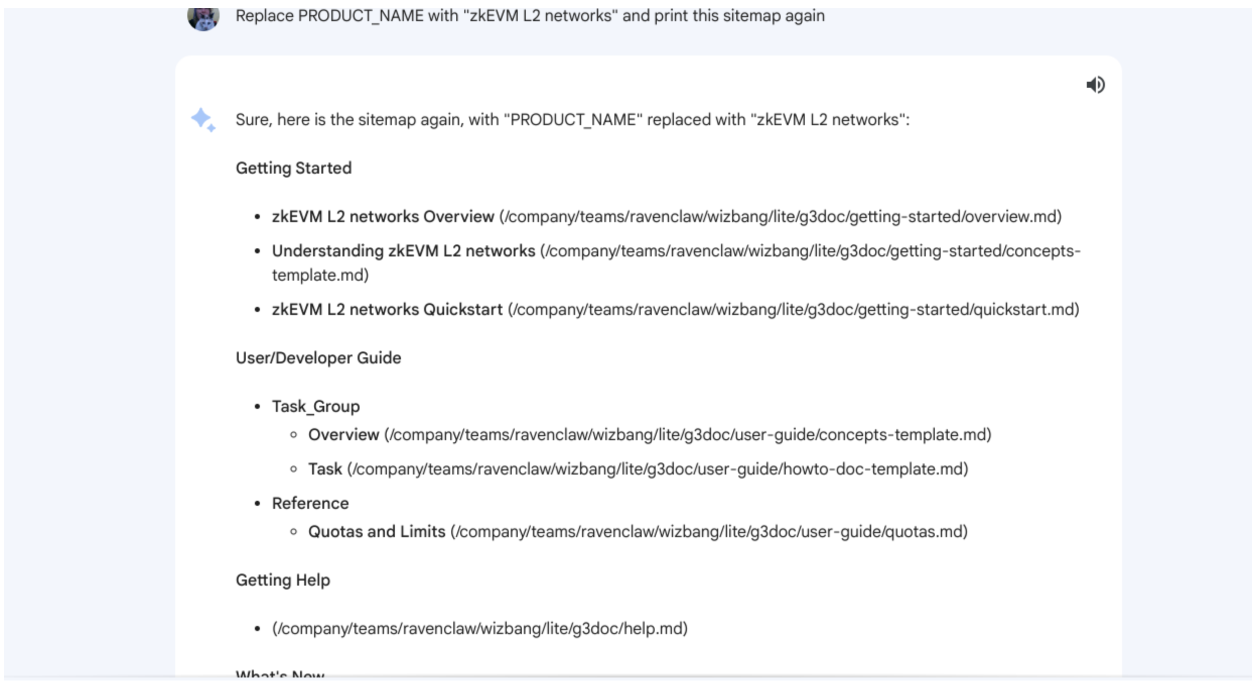
1. Read the markdown site map file such as sitemap.md or _config.md in the template.
2. Out of the headings and subheadings, make a hierarchical list like a table of contents and include the directory location for context
3. Turn the hierarchical list into a semantic tree of categories.

4. Have Bard read in the text of a longer technical description of a blockchain technology. The text was already broken into sections but had very highly technical descriptions of a technology many people do not know about. Also, the structure the original document had did not match the template at all. In the read-in prompt, I asked Bard to break up the text into the table of contents heading using the categories as a semantic tree. Knowing that behind Bard is a LLM, I specified it should use the process of semantic clustering.
5. Then I wanted Bard to put one-paragraph summaries under the category and subcategory headings.
6. Finally, I asked Bard to produce a one paragraph summary of the entire text to be put on the landing page.

In all the prompts, I tried to use examples.

Example Prompts

1. Get content from your markdown directory in Github, your cloud site, or your hard drive (give Bard the contents of the whole file)
2. Replace PRODUCT_NAME with " zkEVM L2 networks" and print this sitemap again (Assign zkEVM L2 networks to PRODUCT_NAME)



Replace PRODUCT_NAME with "zkEVM L2 networks" and print this sitemap again

Sure, here is the sitemap again, with "PRODUCT_NAME" replaced with "zkEVM L2 networks":

Getting Started

- **zkEVM L2 networks Overview** (/company/teams/ravenclaw/wizbang/lite/g3doc/getting-started/overview.md)
- **Understanding zkEVM L2 networks** (/company/teams/ravenclaw/wizbang/lite/g3doc/getting-started/concepts-template.md)
- **zkEVM L2 networks Quickstart** (/company/teams/ravenclaw/wizbang/lite/g3doc/getting-started/quickstart.md)

User/Developer Guide

- Task_Group
  - **Overview** (/company/teams/ravenclaw/wizbang/lite/g3doc/user-guide/concepts-template.md)
  - **Task** (/company/teams/ravenclaw/wizbang/lite/g3doc/user-guide/howto-doc-template.md)
- Reference
  - **Quotas and Limits** (/company/teams/ravenclaw/wizbang/lite/g3doc/user-guide/quotas.md)

Getting Help

- (/company/teams/ravenclaw/wizbang/lite/g3doc/help.md)

What's New

3. In addition, assign each of the headers, Getting Started, User/Developer Guide, Getting Help, and What's New an index.md. For example, Getting Started should have its own index.md (/company/teams/…/…/getting-started/index.md) written next to Getting Started.
4. Take the above markdown content and build a semantic tree where each header and bulleted item is a category or subcategory, respectively. Use the folder structure to guide you. For example, /company/…/…/getting-started/overview.md is zKEVM L2 networks Overview, a subcategory of the category Getting Started.

Connecting data

I have limited access to backend data, so I was not able to use internal document links. So I just had to import the text in the prompt itself. I specified in the prompt "the text between quotes."

Right-sizing your task

I broke up the prompts into as small of tasks as possible so as to not overwhelm the LLM model. I would make references to preceding chats in the same threads if necessary. Sometimes Bard would "remember" and sometimes it had trouble, in which case, I would have to repeat certain things.

Set up a tracker for your prompts and output

I kept track of my prompts in the chat thread itself. I did ask Bard if it would do that for me, but it was not able to.

Additionally, I used the prompt scratch pad in a Doc file as I went along prompting Bard. After I finished, I drafted a final detailed write-up of this experiment.

Run your prompts

ThisI ran each prompt as I went, following my prompt scratch pad. I found I had to tweak the prompts sometimes as I went along. I noticed that sometimes Bard would "forget" what it did before, even though I always stayed inside the same chat thread. I found I had to repeat information sometimes. One example is giving the folder structure as context, which Bard confused with the template itself. So when I asked it in the next step to write up a summary of the text (the long text I gave it in a preceding chat), a summary suitable for a landing page that should go into the highest level index.md file.

Assess the output

After some reworking, I found that the last results were helpful. Bard has a feature that lets you export chat results into Doc file, which I utilized to save the:

- table of contents outline with the folder structure
- semantic tree
- outline with the new text
- outline with the paragraphs
- landing page

The outputs were in formatted text. One was in markdown format which could be copied and pasted into sitemap.md.

```
# Home (/company/teams/ravenclaw/wizbang/lite/g3doc/index.md)

--------------------------------------------------------------------------------

## Getting Started
(/company/teams/ravenclaw/wizbang/lite/g3doc/getting-started/index.md)

* [zkEVM L2 networks
Overview](/company/teams/ravenclaw/wizbang/lite/g3doc/getting-started/overview.md)
* [Understanding zkEVM L2
networks](/company/teams/ravenclaw/wizbang/lite/g3doc/getting-started/concepts-templat
e.md)
* [zkEVM L2 networks
Quickstart](/company/teams/ravenclaw/wizbang/lite/g3doc/getting-started/quickstart.md)

## User/Developer Guide
(/company/teams/ravenclaw/wizbang/lite/g3doc/user-guide/index.md)

* Task Group
  *
[Overview](/company/teams/ravenclaw/wizbang/lite/g3doc/user-guide/concepts-template.md
)
```

```
  *
[Task](/company/teams/ravenclaw/wizbang/lite/g3doc/user-guide/howto-doc-template.md)
* Reference
  * [Quotas and
Limits](/company/teams/ravenclaw/wizbang/lite/g3doc/user-guide/quotas.md)

## Getting Help (/company/teams/ravenclaw/wizbang/lite/g3doc/help.md)

## What's New (/company/teams/ravenclaw/wizbang/lite/g3doc/release-notes.md)
```

I found both the formats and most of the outputs useful.

Human review was necessary, as would be additional work to copy and paste the text into the different markdown files in their respective folders in the case this experiment were to be implemented to generate a real documentation site.

No sensitive data was used, but is proprietary public information belonging to another company in the blockchain space.

Decide whether to roll out the AI workflow (and write a report!)

In future iterations, this use of genAI would save time in the initial creation of new technical documentation projects using the template.

It has the promise of saving approximately 4 work hours for a 3-page text document.

Productionize

I wrote prompts by hand for this lightweight use case. It would be worth automating the prompts in AppScript and using this workflow again on a proprietary Google document, and to try longer text documents as well.