Dear Diary!

Hmmm… A lookup app. How could that work?

- I need a frontend obviously. And a backend, that does the heavy work.
- Frontend and backend should communicate through REST. What else?
- The backend should index the Terraform documentation
- Everything should be well tested of course

First things first. The absolute basis for the whole project is the documentation index. Without that, I could trash the whole thing.

So, how is the terraform documentation built actually?

The Terraform developers do that using a massive repository with multiple submodules for all providers:

hashicorp

/

terraform-website

Prototype of Terraform website being assembled from multiple repositories

Terraform Website

This repository contains the build infrastructure and some of the content for **terraform.io**. Pull requests from the community are welcomed!

Where the Docs Live

terraform.io is a static site built from Markdown source files using **Middleman**. Unlike most such sites, it draws content from a lot of different Git repositories, which can make it

challenging to contribute to.

To find a page the easy way: view it on **terraform.io** and click the „Edit this page“ link at the bottom. (As of Spring 2019, those links get routed to the correct repo for everything except the Google Cloud Platform provider.)

If you’d rather just remember where to look:

- This repository has the Terraform Enterprise docs, the Terraform GitHub Actions docs, and the Extending Terraform section
  - Those files can be found at `content/source/docs/`. The `master` branch is the „live“ content that gets deployed to terraform.io.
- The **hashicorp/terraform**…

View on GitHub

They write all documentation in Markdown with Frontmatter included and cramp everything together with **Middleman**

The Frontmatter includes titles, layouts, description and the information where the documentation lives in the documentation tree on the website.

So that’s what I need.

I „submoduled“ the terraform-website repository as well and wrote an **Indexer**, that walks through every datasource and resource documentation available for all providers in the terraform website.

That worked quite well (aside from some minor errors)[**https://github.com/dploeger/tflookup/blob/master/indexErrors.txt**], where people forgot how to write proper Frontmatter.

However, I needed to make some assumptions:

- Every vendor has a „website/docs“ subfolder where its documentation lives
- There’s only a datasource or a resource
- The datasources are living in a „d“ directory, the resources in an „r“ directory
- The titles are designed as „: „

I didn’t go through all documentation files, but the documentation seems to be quite structured like this.

Using all this information, I could create my documentation index, which included the title, the name, wether it's a datasource or a resource, the description and a link to the real documentation page.

I included ways to store the indexed documentation, so the future frontend server wouldn't need to initialize all the required submodules for that. I also wrote a **script which updates the documentation** every night, which is running on my private server.

Okay, index done.

Yours
Dennis

*This post is one of five posts from the **tflookup developer diary series***

*Cover Image: **„diary writing" by Fredrik Rubensson***

*Originally published to **dev.to***