MIRRORED FROM: https://git.blob42.xyz/sp4ke/hugobot

HUGOBOT

hugobot is a bot that automates the fetching and aggregation of content for Hugo data-driven websites. It has the following features:

Data fetch

- Use the `feeds` table to register feeds that will be fetched periodically.
- Currently, it can handle these types of feeds: `RSS`, `Github Releases`, `Newsletters`
- To define your own feed types, implement the `JobHandler` interface (see `handlers/handlers.go`).
- Hugobot automatically fetches new posts from the registered feeds.
- The database uses Sqlite for storage. It has `feeds` and `posts` tables.
- The scheduler can handle an unlimited number of tasks and uses leveldb for caching and resuming jobs.

Hugo export

- Data is automatically exported to the configured Hugo website path.
- It can export data as `markdown` files or `json/toml` data files.
- You can customize all fields in the exported files.
- You can define custom output formats by using the `FormatHandler` interface.
- You can register custom filters and post-processing for exported posts to prevent altering the raw data stored in the database.
- You can force data export using the CLI.

API

- It uses `gin-gonic` as the web framework.
- *hugobot* also includes a webserver API that can be used with Hugo Data Driven Mode.
- You can insert and query data from the database. This feature is still a work in progress, but you can easily add the missing code on the API side to automate inserting and querying data from the database.
- For example, it can be used to automate the generation of Bitcoin addresses for new articles on bitcointechweekly.com.

Other

- Some commands are available through the CLI (`github.com/urfave/cli`), you can add your own custom commands.

Sqliteweb interface

- See the Docker files for more information.

First time usage

- The first time you run the program, it will automatically generate the database. You can add your feeds to the Sqlite database using your preferred Sqlite GUI.

Contribution

- We welcome pull requests. Our current priority is adding tests.
- Check the TODO section.

TODO:

- Add tests.
- Handle more feed formats: `tweets`, `mailing-list emails` ...
- TLS support in the API (not a priority, can be done with a reverse proxy).