Data Migration - Round 3

Automatically upload the `RST file` from Task 2 to `GitHub`

Requirements

Write a java/python command line application push the RST file to GitHub and overwrite if the file is existed.

> Input: `RST file` of task 2

> Output: File got `pushed` to GitHub branch

Evaluation

- `[x]` Push to GitHub
- `[x]` Overwrite
- `[x]` Clean code

Installation

Install the required packages by using `pip`

```
pip install -r requirements.txt
```

Getting Started

```
$ python main.py -i sample.rst -s config.yaml
```

Technical Overview

- We use the GitHub REST API to upload files.
  - `First`, use a GET method to retrieve the current SHA of a file. If file does not exist on that branch, the value of SHA is None.
  - `Second`, use a PUT method to upload the file. The file is encoded, together with its SHA (if overwrite) and commit message are pushed to the branch
- Of course, you must provide your personal `access token` and this token must be authorized to access the resources.
- For more detail, please refer to GitHub REST API documentation

Usage

The program have `2 arguments`. You can check the documentation by using the command

```
$ python main.py -h
```

```
usage: main.py [-h] [-i INPUT_FILE] [-s SETTINGS]

options:
  -h, --help            show this help message and exit
  -i INPUT_FILE, --input_file INPUT_FILE
                        Directory to input file need to push to Github.
Accepts file *.rst only
  -s SETTINGS, --settings SETTINGS
                        Directory to configure config yaml settings.
Accepts file *.yaml only
```

Example command:

```
$ python main.py -i sample.rst -s config.yaml
```

- **Step 1:** Modify the `/Task3/config.yaml`

```
AUTHENTICATION:
  TOKEN: <GitHub-Access-Token>
  USERNAME: <username>
MESSAGE: <commit-message>
REPOSITORY:
  BRANCH: <name-of-branch>
  NAME: <name-of-repo>
  OWNER: <repo-owner>
```

- **Step 2:** run the `main.py` to start uploading

```
$ python main.py -i sample.rst -s config.yaml
```