\*\*\*\*\*\*\*\*\*\*\*\*

## Project Overview

This project is a Flask-based web application that provides several endpoints to scrape data from different websites and fetch JSON data from an API. The project structure is as follows:

```

scapper\_api\_project

│

├── venv

│ ├── Include

│ ├── Lib

│ ├── Scripts

│ └── pyvenv.cfg

│

├── app.py

├── books2scrape.py

├── jsonRest.py

└── quotes2scrape.py

```

### Files and Directories

- `venv/`: The virtual environment directory containing all the dependencies for the project.

-

app.py

: The main Flask application file that defines the routes and their corresponding handlers.

-

books2scrape.py

: A module that scrapes book data from a website.

-

jsonRest.py

: A module that fetches JSON data from an API.

-

quotes2scrape.py

: A module that scrapes quotes from a website.

##

app.py

The

app.py

file is the main entry point of the Flask application. It defines several routes and their corresponding handlers.

### Imports

```python

from flask import Flask, jsonify

from books2scrape import scrape\_data as scrape\_books

from quotes2scrape import scrape\_data as scrape\_quotes

from jsonRest import fetch\_json\_data

```

-

Flask

and

jsonify

are imported from the

flask

package to create the Flask app and return JSON responses.

-

scrape\_data

functions are imported from

books2scrape

and

quotes2scrape

modules and aliased as

scrape\_books

and

scrape\_quotes

respectively.

-

fetch\_json\_data

is imported from the

jsonRest

module.

### Flask App Initialization

```python

app = Flask(\_\_name\_\_)

app.routes\_printed = False

```

- A Flask app instance is created.

- A custom attribute

routes\_printed

is added to the app instance to ensure routes are printed only once.

### Print Registered Routes

```python

@app.before\_request

def print\_routes():

if not app.routes\_printed:

print("Registered Routes:")

for rule in app.url\_map.iter\_rules():

print(rule)

app.routes\_printed = True

```

- This function prints all registered routes before handling any request. It ensures that the routes are printed only once by checking the

routes\_printed

attribute.

### Routes

#### Root Route

```python

@app.route("/", methods=["GET"])

def index():

print("Endpoint / called")

return "Flask app is running"

```

- This route returns a simple message indicating that the Flask app is running.

#### Scrape Books Route

```python

@app.route("/scrape/books", methods=["GET"])

def get\_books():

print("Endpoint /scrape/books called")

data = scrape\_books()

return jsonify(data)

```

- This route calls the

scrape\_books

function from the

books2scrape

module and returns the scraped book data as a JSON response.

#### Scrape Quotes Route

```python

@app.route("/scrape/quotes", methods=["GET"])

def get\_quotes():

print("Endpoint /scrape/quotes called")

data = scrape\_quotes()

return jsonify(data)

```

- This route calls the

scrape\_quotes

function from the

quotes2scrape

module and returns the scraped quotes data as a JSON response.

#### Fetch JSON Data Route

```python

@app.route("/fetch/json", methods=["GET"])

def get\_json\_data():

print("Endpoint /fetch/json called")

data = fetch\_json\_data()

return jsonify(data)

```

- This route calls the

fetch\_json\_data

function from the

jsonRest

module and returns the fetched JSON data as a JSON response.

#### Get Routes Route

```python

@app.route("/routes", methods=["GET"])

def get\_routes():

print("Endpoint /routes called")

return jsonify([str(rule) for rule in app.url\_map.iter\_rules()])

```

- This route returns a list of all registered routes in the Flask app.

### Running the Flask App

```python

if \_\_name\_\_ == "\_\_main\_\_":

print("Starting Flask app...")

app.run(debug=True, port=5001)

```

- This block runs the Flask app in debug mode on port 5001.

## books2scrape.py

This module scrapes book data from a website.

```python

import requests

from bs4 import BeautifulSoup

def scrape\_data():

url = "http://books.toscrape.com/"

response = requests.get(url)

if response.status\_code == 200:

soup = BeautifulSoup(response.content, "html.parser")

books = [book.get\_text() for book in soup.find\_all("h3")]

return {"books": books}

else:

return {"error": "Failed to retrieve data"}

```

- `requests` is used to make HTTP requests.

- `BeautifulSoup` is used to parse HTML content.

-

scrape\_data

function fetches the webpage, parses it, and extracts book titles.

## quotes2scrape.py

This module scrapes quotes from a website.

```python

import requests

from bs4 import BeautifulSoup

def scrape\_data():

url = "http://quotes.toscrape.com/"

response = requests.get(url)

if response.status\_code == 200:

soup = BeautifulSoup(response.content, "html.parser")

quotes = [

{

"text": quote.find("span", class\_="text").get\_text(),

"author": quote.find("small", class\_="author").get\_text(),

}

for quote in soup.find\_all("div", class\_="quote")

]

return {"quotes": quotes}

else:

return {"error": "Failed to retrieve data"}

```

-

scrape\_data

function fetches the webpage, parses it, and extracts quotes and their authors.

## jsonRest.py

This module fetches JSON data from an API.

```python

import requests

def fetch\_json\_data():

url = "https://jsonplaceholder.typicode.com/posts"

response = requests.get(url)

if response.status\_code == 200:

return response.json()

else:

return {"error": "Failed to retrieve data"}

```

-

fetch\_json\_data

function makes an HTTP GET request to the API and returns the JSON response.

## Running the Project

1. \*\*Set up the virtual environment:\*\*

```sh

python -m venv venv

```

2. \*\*Activate the virtual environment:\*\*

- On Windows:

```sh

venv\Scripts\activate

```

- On macOS/Linux:

```sh

source venv/bin/activate

```

3. \*\*Install dependencies:\*\*

```sh

pip install flask requests beautifulsoup4

```

4. \*\*Run the Flask app:\*\*

```sh

python app.py

```

5. \*\*Access the endpoints:\*\*

- Root: `http://127.0.0.1:5001/`

- Scrape Books: `http://127.0.0.1:5001/scrape/books`

- Scrape Quotes: `http://127.0.0.1:5001/scrape/quotes`

- Fetch JSON: `http://127.0.0.1:5001/fetch/json`

- Get Routes: `http://127.0.0.1:5001/routes`

This documentation provides a comprehensive overview of the project, its structure, and how to run it.   
  
\*\*\*\*\*\*\*\*\*\*\*\*