Midterm Project Report

Advanced Computer Programming

Student Name : Kenneth Bryan Theodore

Student ID : 113021130

Teacher : DINH-TRUNG VU

2024-04

Chapter 1 Introduction

1.1 Github

1) **Personal Github Account**: 113021130
2) **Group Project Repository**: acp-1132

1.2 Overview

My program features the usage of scrapy to scrape the web for information . In this case to extract structured information from a GitHub profile page. It targets repository data including the url of each of the repositories, the about, last updated timestamp, programming languages used, and also the number of commits on the page https://github.com/113021130?tab=repositories.

I used a few advanced language features such as regular expressions(re) to detect if the repository is empty or not. I also used the data class **RepositoryItem** from **scrapy.Item** to structure the data that is scraped.

The libraries that I used include Scrapy, the main web scraping framework to crawl and extract data from the web. Other than that I used CSS selector to target html elements and also Parsel for parsing the HTML.

Implementation

2.1 Class 1: RepositoryItem

This class defines the data structure for the scraped repository data it inherits from scrapy.item.

2.1.1 Fields

The fields I used in my program is url which is the full url to the github repository. Next I used about which is the short description of the repository (repo name if about is empty). After that is last_updated which is the timestamp of the last update on the repository. There's also languages which is basically a list of the programming languages used in the repository. Lastly is the number of commits on the repository.

2.1.2 Methods & Functions

This class uses default methods from **scrapy.Item** to store and retrieve field data.

```
class RepositoryItem(scrapy.Item):
    url = scrapy.Field()
    about = scrapy.Field()
    last_updated = scrapy.Field()
    languages = scrapy.Field()
    number_of_commits = scrapy.Field()

    pass
```

2.2 Class 2 : GithubSpider

This class is the main spider that is performing the crawling and extracting data from the website.

2.2.1 Fields

This class uses the fields name, allowed_domains, and start_urls. The name is just the name of the class given to execute it on the command line. allowed_domains are the domains the spider is allowed to crawl in, in this case the domains of GitHub. start_urls is the starting page for the spider to crawl in.

2.2.2 Methods & functions

- parse method

```
def parse(self, response):
    global last_updated
    repo_links = response.css("div.d-inline-block.mb-1 a::attr(href)").getall()

    for link in repo_links:
        last_updated = response.css('relative-time::attr(datetime)').get()
        full_url = response.urljoin(link)
        yield scrapy.Request(full_url, callback=self.parse_repo)
```

This is the main method that Scrapy runs when it gets a response from a URL listed in start_urls. This function is to extract links to individual repositories from the user's repository list. It also is used to retrieve the last updated timestamp for each repository in the page. It handles the HTML content of the GitHub user's repositories page.

- parse_repo

```
def parse_repo(self, response):
    global last_updated
    item = RepositoryItem()
    item['url'] = response.url

    about = response.css('p.f4.my-3::text').get()
    if about:
        item['about'] = about.strip()
    else:
        is_empty = response.css('div.Box-body.p-6.py-3').re_first(r'This repository is empty')
        if not is_empty:
            repo_name = response.url.strip('/').split('/')[-1]
            about = repo_name
        else:
            about = None

    item['about'] = about
    item['last_updated'] = last_updated

    match = re.match(r'https://github.com/([^/]+)/([^/]+)', response.url)
    if match:
        owner = match.group(1)
        repo = match.group(2)
    else:
        owner = "unknown"
        repo = "unknown"

    commits_api = f'https://api.github.com/repos/{owner}/{repo}/commits'
    languages_api = f'https://api.github.com/repos/{owner}/{repo}/languages'

    yield scrapy.Request(
        commits_api,
        callback=self.parse_commits,
        meta={
            'repo_name': repo,
            'repo_url': response.url,
            'about': item['about'],
            'last_updated': last_updated,
            'languages_api': languages_api
        },
        headers={"Accept": "application/vnd.github+json"}
    )
```

This function is to parse the content of a single repository page. It also extracts the description of the repository and also constructs API URLs for commits and languages. Initiates further requests to GitHub's API to gather detailed data.

- parse_commits

```python
def parse_commits(self, response):
    repo_name = response.meta['repo_name']
    repo_url = response.meta['repo_url']
    about = response.meta['about']
    last_updated = response.meta['last_updated']
    languages_api = response.meta['languages_api']

    try:
        commits = json.loads(response.text)
        num_commits = len(commits)
    except Exception:
        num_commits = None

    yield scrapy.Request(
        languages_api,
        callback=self.parse_languages,
        meta={
            'repo_name': repo_name,
            'repo_url': repo_url,
            'about': about,
            'last_updated': last_updated,
            'num_commits': num_commits
        },
        headers={"Accept": "application/vnd.github+json"}
    )
```

This function processes the data returned from the GitHub commits API and extracts and counts the number of commits for the repository.

- parse_languages

```
def parse_languages(self, response):
    repo_name = response.meta['repo_name']
    repo_url = response.meta['repo_url']
    about = response.meta['about']
    last_updated = response.meta['last_updated']
    num_commits = response.meta['num_commits']

    try:
        languages_json = json.loads(response.text)
        languages = list(languages_json.keys())
    except Exception:
        languages = None

    yield {
        'repo_name': repo_name,
        'repo_url': repo_url,
        'about': about,
        'last_updated': last_updated,
        'languages': languages,
        'num_commits': num_commits
    }
```

This function parses the response from GitHub's Languages API and retrieves the list of programming languages used in the repository. It also yields the final compiled data including all of the attributes.

Chapter 3 Results

3.1 Result 1

The spider scraps and yielded structured information about all repositories for the GitHub user https://github.com/113021130.

For each repository a valid URL, Description name (repo name if none), last updated date from the relative-time HTML tag, number of commits via the GitHub Commits API, and languages used via the GitHub Languages API.

Chapter 4 Conclusions

This project uses Scrapy's framework to scrape GitHub profile pages. Some advanced techniques and libraries such as re and json were used to scrape the web and process the data. Github public APIs were also used to collect data.