亞洲大學
ASIA UNIVERSITY

Midterm Project Report

Advanced Computer Programming

Student Name	:	Joanes Don Bosco B.
Student ID	:	113021218
Teacher	:	DINH-TRUNG VU

2024-04

Chapter 1 Introduction

1.1 Github

1) **Personal Github Account**: https://github.com/wonderhorse90
2) **Group Project Repository**: https://github.com/wonderhorse90/ACP

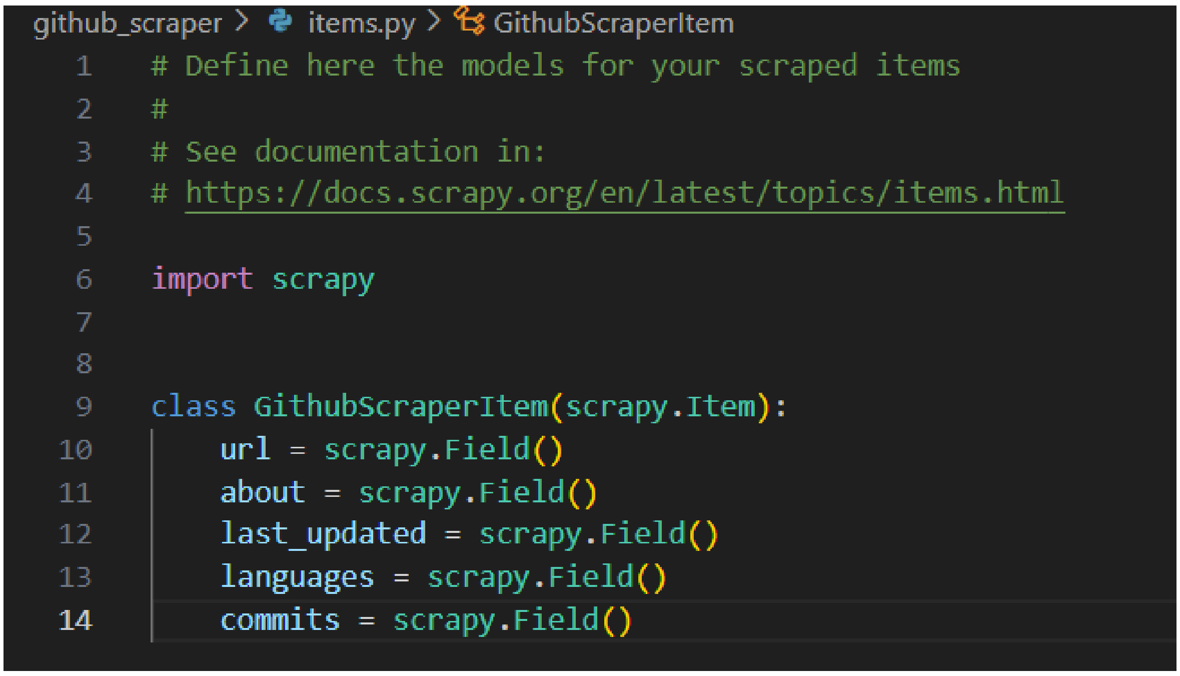
1.2 Overview

This project uses the Scrapy framework to build a web crawler that extracts repository data from a GitHub profile. It makes use of data classes from Scrapy (`Item`), CSS selectors for HTML parsing, pagination logic, and conditional handling of missing or empty data. The program successfully scrapes and stores information about each repository, including URL, About section, Last Updated date, programming languages used, and the number of commits.

Chapter 2 Implementation

2.1 Class : GithubRepoItem

This is a data structure used to store extracted information from each GitHub repository.


github_scraper > items.py > GithubScraperItem

```python
# Define here the models for your scraped items
#
# See documentation in:
# https://docs.scrapy.org/en/latest/topics/items.html

import scrapy


class GithubScraperItem(scrapy.Item):
    url = scrapy.Field()
    about = scrapy.Field()
    last_updated = scrapy.Field()
    languages = scrapy.Field()
    commits = scrapy.Field()
```

2.1.1 Fields

- url
- about
- last_updated
- languages
- commits

2.1.2 Methods

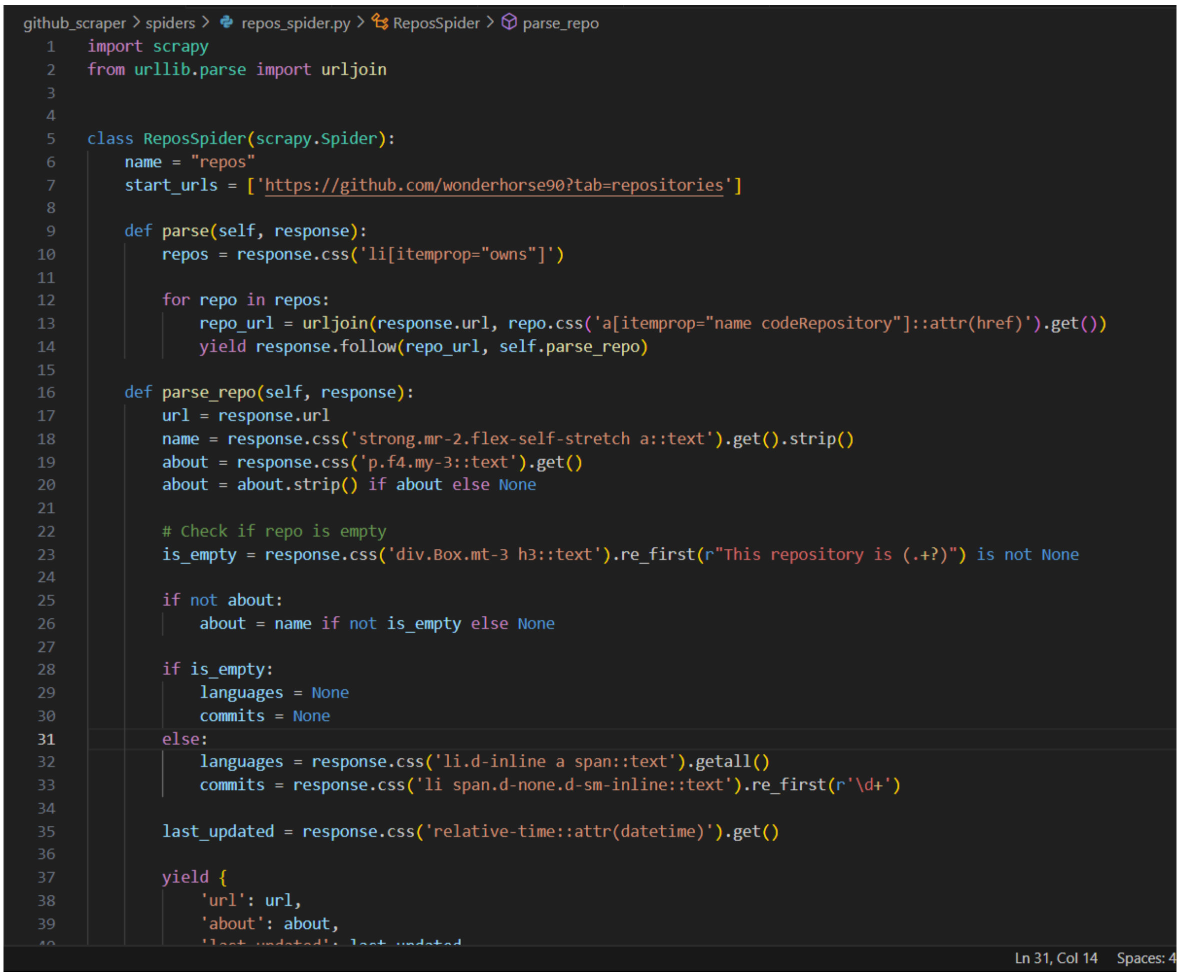
Inherited from Scrapy Item class.

2.1.3 Functions

-

2.2 Class : ReposSpider

This is the main spider class that handles crawling GitHub pages and parsing the required data


github_scraper > spiders > repos_spider.py > ReposSpider > parse_repo
Ln 31, Col 14   Spaces: 4

```python
import scrapy
from urllib.parse import urljoin


class ReposSpider(scrapy.Spider):
    name = "repos"
    start_urls = ['https://github.com/wonderhorse90?tab=repositories']

    def parse(self, response):
        repos = response.css('li[itemprop="owns"]')

        for repo in repos:
            repo_url = urljoin(response.url, repo.css('a[itemprop="name codeRepository"]::attr(href)').get())
            yield response.follow(repo_url, self.parse_repo)

    def parse_repo(self, response):
        url = response.url
        name = response.css('strong.mr-2.flex-self-stretch a::text').get().strip()
        about = response.css('p.f4.my-3::text').get()
        about = about.strip() if about else None

        # Check if repo is empty
        is_empty = response.css('div.Box.mt-3 h3::text').re_first(r"This repository is (.+?)") is not None

        if not about:
            about = name if not is_empty else None

        if is_empty:
            languages = None
            commits = None
        else:
            languages = response.css('li.d-inline a span::text').getall()
            commits = response.css('li span.d-none.d-sm-inline::text').re_first(r'\d+')

        last_updated = response.css('relative-time::attr(datetime)').get()

        yield {
            'url': url,
            'about': about,
            'last_updated': last_updated
```

2.2.1 Fields

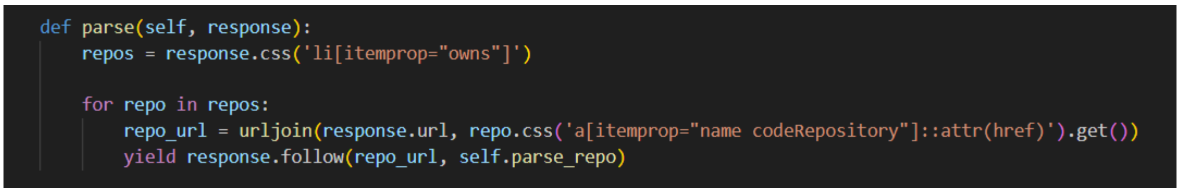
- name
- allowed_domains
- start_urls

2.2.2 Methods

- parse: Extracts repository links and follows them.
- parse_repo: Extracts information from individual repository pages.

2.3 Function : parse

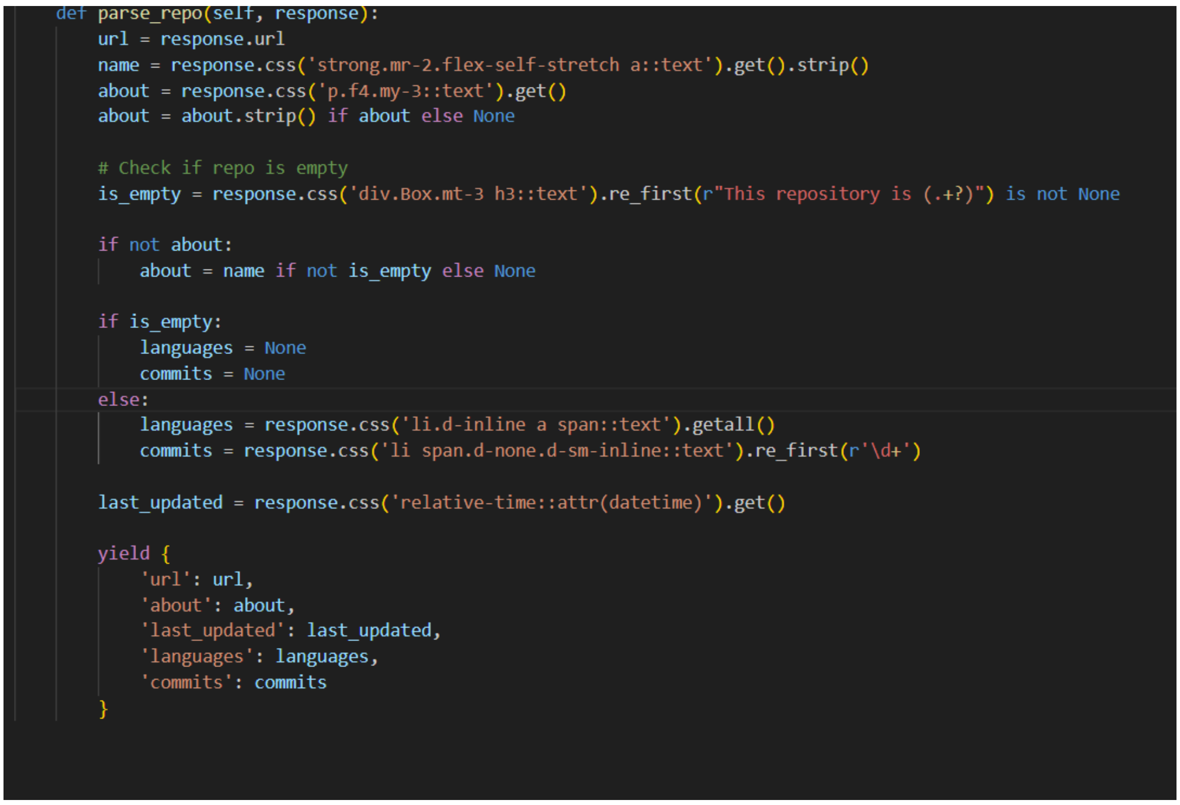
This function is called when the start URL is fetched. It locates all repositories on the page, and initiates parsing each repository individually.



```
def parse(self, response):
    repos = response.css('li[itemprop="owns"]')

    for repo in repos:
        repo_url = urljoin(response.url, repo.css('a[itemprop="name codeRepository"]::attr(href)').get())
        yield response.follow(repo_url, self.parse_repo)
```

2.4 Function : parse_repo

. This function handles parsing a single repository page to extract detailed information.

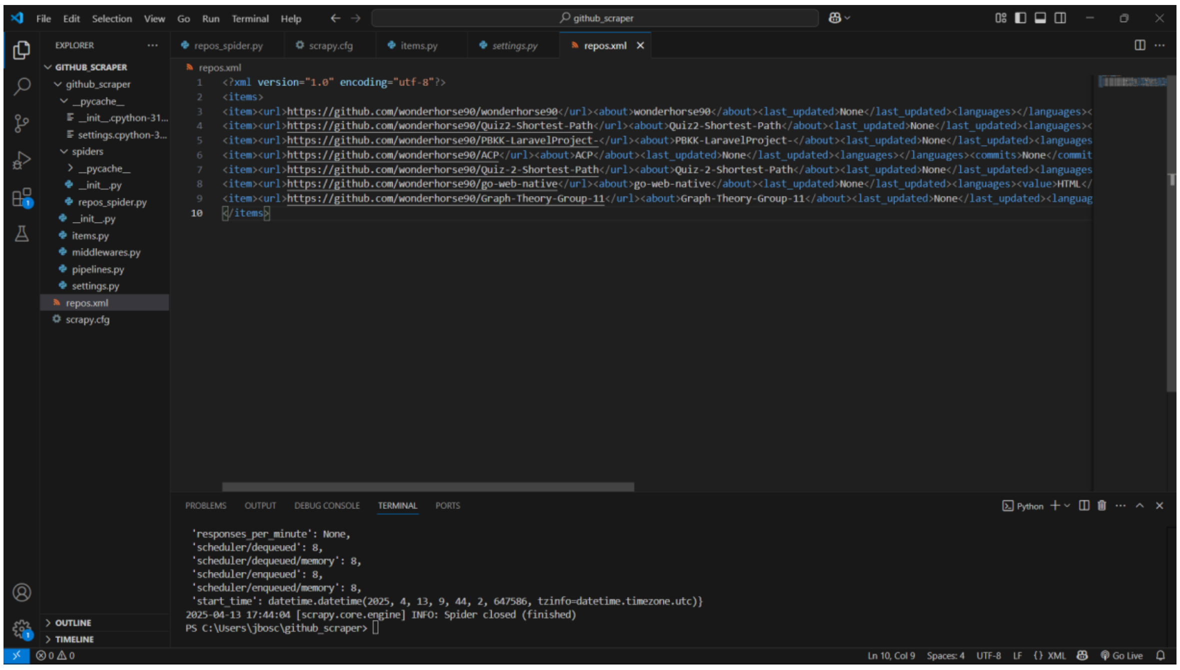


```
def parse_repo(self, response):
    url = response.url
    name = response.css('strong.mr-2.flex-self-stretch a::text').get().strip()
    about = response.css('p.f4.my-3::text').get()
    about = about.strip() if about else None

    # Check if repo is empty
    is_empty = response.css('div.Box.mt-3 h3::text').re_first(r"This repository is (.+?)") is not None

    if not about:
        about = name if not is_empty else None

    if is_empty:
        languages = None
        commits = None
    else:
        languages = response.css('li.d-inline a span::text').getall()
        commits = response.css('li span.d-none.d-sm-inline::text').re_first(r'\d+')

    last_updated = response.css('relative-time::attr(datetime)').get()

    yield {
        'url': url,
        'about': about,
        'last_updated': last_updated,
        'languages': languages,
        'commits': commits
    }
```

Chapter 3 Results

3.1 Result 1

.



```
<?xml version="1.0" encoding="utf-8"?>
<items>
<item><url>https://github.com/wonderhorse90/wonderhorse90</url><about>wonderhorse90</about><last_updated>None</last_updated><languages></languages><
<item><url>https://github.com/wonderhorse90/Quiz2-Shortest-Path</url><about>Quiz2-Shortest-Path</about><last_updated>None</last_updated><languages><
<item><url>https://github.com/wonderhorse90/PBKK-LaravelProject-</url><about>PBKK-LaravelProject-</about><last_updated>None</last_updated><languages
<item><url>https://github.com/wonderhorse90/ACP</url><about>ACP</about><last_updated>None</last_updated><languages></languages><commits>None</commit
<item><url>https://github.com/wonderhorse90/Quiz-2-Shortest-Path</url><about>Quiz-2-Shortest-Path</about><last_updated>None</last_updated><languages
<item><url>https://github.com/wonderhorse90/go-web-native</url><about>go-web-native</about><last_updated>None</last_updated><languages><value>HTML</
<item><url>https://github.com/wonderhorse90/Graph-Theory-Group-11</url><about>Graph-Theory-Group-11</about><last_updated>None</last_updated><languag
</items>
```

```
 'responses_per_minute': None,
 'scheduler/dequeued': 8,
 'scheduler/dequeued/memory': 8,
 'scheduler/enqueued': 8,
 'scheduler/enqueued/memory': 8,
 'start_time': datetime.datetime(2025, 4, 13, 9, 44, 2, 647586, tzinfo=datetime.timezone.utc)}
2025-04-13 17:44:04 [scrapy.core.engine] INFO: Spider closed (finished)
PS C:\Users\jbosc\github_scraper>
```

.

Chapter 4 Conclusions

This project demonstrates effective usage of the Scrapy framework to automate the collection of GitHub repository data. It handles edge cases such as empty repositories, missing descriptions, and paginated results. This setup can be extended to gather more metrics or be adapted for similar data extraction tasks.