亞洲大學
ASIA UNIVERSITY

Midterm Project Report

Advanced Computer Programming

Student Name : Amarsanaa Burenbaatar

Student ID : 113021188

Teacher : DINH-TRUNG VU

2024-04

Chapter 1 Introduction

1.1 Github

1) **Personal Github Account**: amaworldpeace (Amarsanaa Burenbaatar)
2) **Group Project Repository**: jnomin/GroupAdvancedComputerProgramming at amarsanaaburenbaatar_1205220751

1.2 Overview

In this project, I used **Scrapy**, a powerful Python web scraping framework, to extract information about public repositories from a GitHub profile page (https://github.com/amaworldpeace?tab=repositories). This project is part of a broader study in advanced computer programming, showcasing practical knowledge of web scraping techniques, data parsing, and structured data extraction.

The scraper focuses on collecting key metadata from each repository, including: Repository URL, Description, Last updated date, Programming languages used, Number of commits.

By implementing a two-step parsing approach, the scraper first identifies and gathers basic repository information from the user's repositories list page, then follows each repository link to extract additional data such as programming languages and commit count from individual repository pages.

Implementation

1.1 Class 1 - GithubScraperItem

This class defines the data structure used to store information scraped from GitHub. Each field in this class corresponds to a piece of data extracted from the target web pages.

1.1.1 Fields

- url: The full URL of the GitHub repository.
- about: The description of the repository, or the repository name if a description is not provided.
- last_updated: The timestamp of the last update to the repository.
- languages: A list of programming languages used in the repository.
- commits: The number of commits in the repository.

1.1.2 Methods

This class does not contain any custom methods, as it is solely used for data storage. All fields are defined as instances of scrapy.Field.

1.1.3 Functions

No functions are defined in this class since it is purely a data container.

1.2 Class 2 - GithubReposSpider

This is the main spider class that handles crawling and parsing. It inherits from scrapy.Spider and defines the logic for navigating the GitHub repositories page and individual repository details pages.

1.3 Method/Function 1 - parse

Parse extracts the list of repositories from the GitHub profile and gathers primary information such as URL, description, and last update time. The method begins by selecting all repository items from the page using the CSS selector **li[itemprop="owns"]**. For each repository, it extracts the relative URL and constructs the complete URL by joining it with the domain name. It then attempts to retrieve the repository description; if the description is not available, it defaults to using the repository name. Next, it captures the date and time the repository was last updated by accessing the relative-time attribute. Finally, the method sends a new Scrapy request to the individual repository page in order to extract further details, passing the current data item through the meta parameter for use in the next parsing stage.

1.4 Method/Function 2 - parse_repo_details

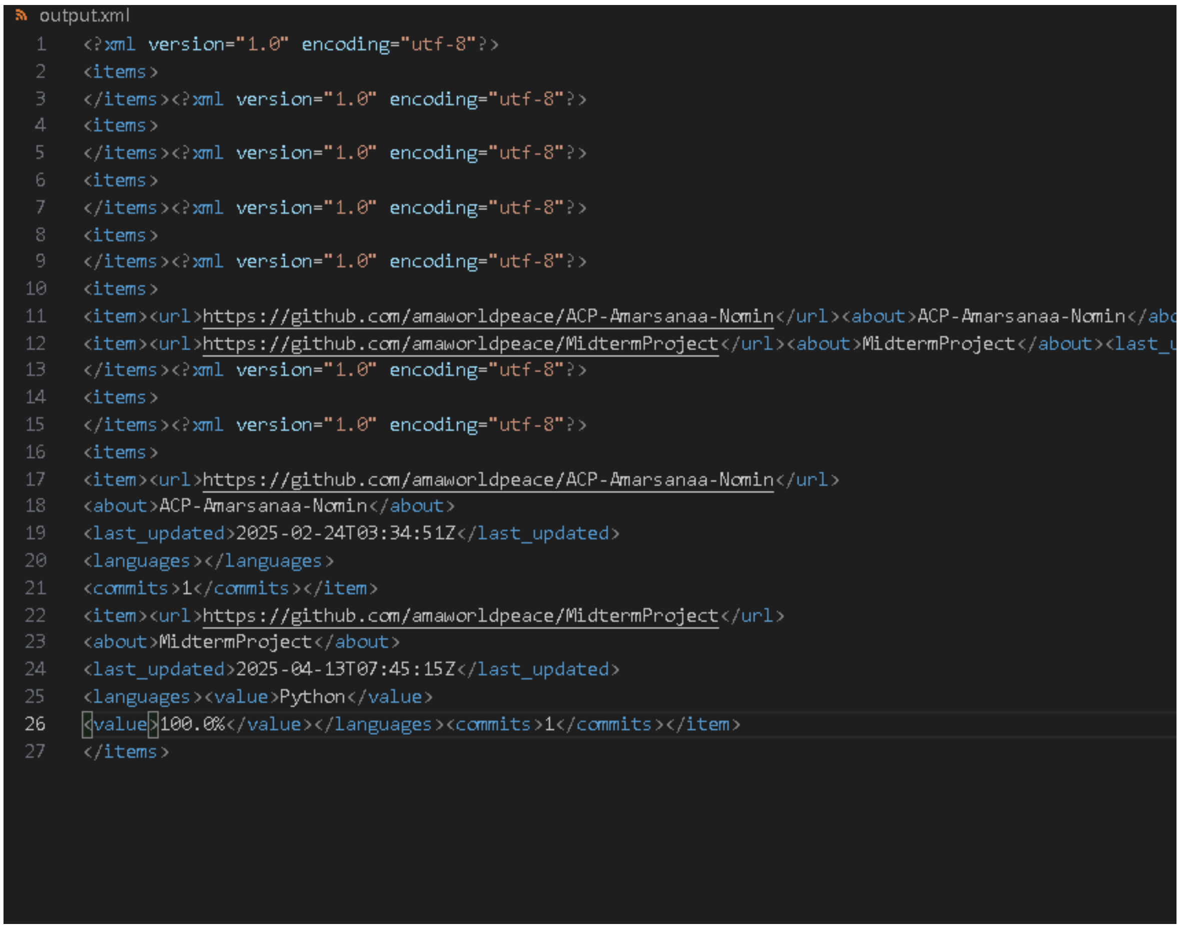
This method is designed to extract additional information from each individual repository page, specifically focusing on the programming languages used and the total number of commits.

Upon receiving the response from the repository's detail page, the method first retrieves the corresponding item passed through the meta attribute. It then checks for the presence of the main content container, identified by the div.Layout-main element, to confirm that the page structure is valid. The method also attempts to retrieve the total number of commits by locating the commits section in the page and parsing the associated number.

Chapter 2 Results

1.1 Result 1

.



```
output.xml
<?xml version="1.0" encoding="utf-8"?>
<items>
</items><?xml version="1.0" encoding="utf-8"?>
<items>
</items><?xml version="1.0" encoding="utf-8"?>
<items>
</items><?xml version="1.0" encoding="utf-8"?>
<items>
</items><?xml version="1.0" encoding="utf-8"?>
<items>
<item><url>https://github.com/amaworldpeace/ACP-Amarsanaa-Nomin</url><about>ACP-Amarsanaa-Nomin</abo
<item><url>https://github.com/amaworldpeace/MidtermProject</url><about>MidtermProject</about><last_u
</items><?xml version="1.0" encoding="utf-8"?>
<items>
</items><?xml version="1.0" encoding="utf-8"?>
<items>
<item><url>https://github.com/amaworldpeace/ACP-Amarsanaa-Nomin</url>
<about>ACP-Amarsanaa-Nomin</about>
<last_updated>2025-02-24T03:34:51Z</last_updated>
<languages></languages>
<commits>1</commits></item>
<item><url>https://github.com/amaworldpeace/MidtermProject</url>
<about>MidtermProject</about>
<last_updated>2025-04-13T07:45:15Z</last_updated>
<languages><value>Python</value>
<value>100.0%</value></languages><commits>1</commits></item>
</items>
```

.

Chapter 2 Conclusions

The project demonstrates how advanced web scraping techniques can be used to automate data collection from GitHub. Using Scrapy, I learned how to handle asynchronous requests, parse HTML with precision using selectors, and export structured data efficiently. The spider was able to handle optional or missing data and fall back to alternatives, ensuring consistent results across varied repo pages.