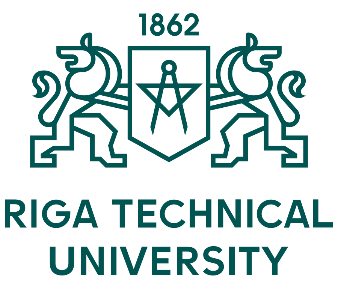
**Telecommunication Software**

Fourth practical exercise

Full name: **Ergashev Nurmukhammad**

ID: **241AEM009**

**Riga 2024**

Contents

[Example 1: Please test requests Python library 3](#_Toc182788097)

[Example 2: Search engine keyword submission interface with requests python library. 4](#_Toc182788098)

[Example 3: Image crawling. 4](#_Toc182788099)

[Example 4: University ranking print 6](#_Toc182788100)

[Example 5: Crawling of goods web pages and printing relevant numbers, goods names, and prices with requests from the Python library. 8](#_Toc182788101)

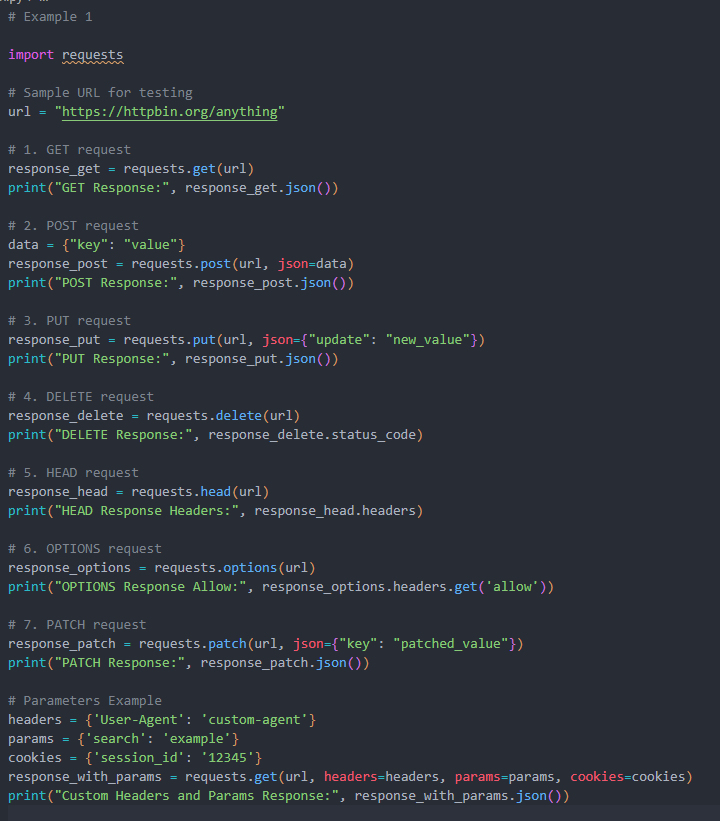
[Example 6: Please reference two public projects finishing your Scrapy project 8](#_Toc182788102)

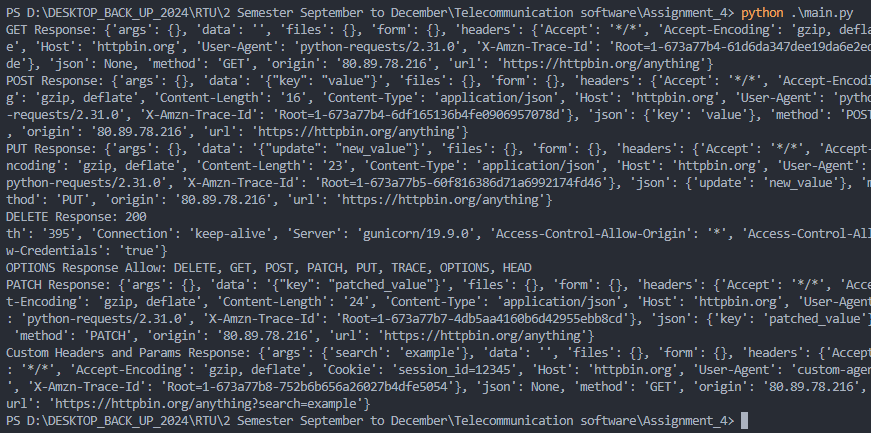
[Conclusion 9](#_Toc182788103)

[Source Code Repository 10](#_Toc182788104)

# Example 1: Please test requests Python library

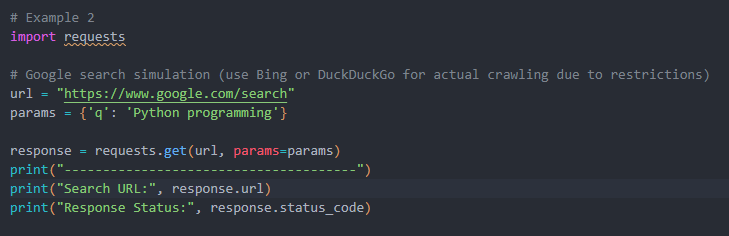
The requests library in Python is a powerful tool for making HTTP requests. It supports several methods (GET, POST, PUT, DELETE, HEAD, OPTIONS, PATCH) and various parameters like headers, cookies, auth, etc.

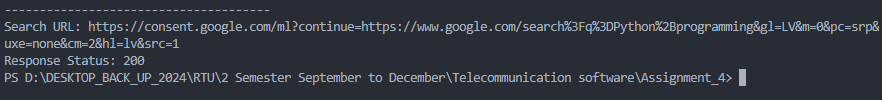




# Example 2: Search engine keyword submission interface with requests python library.

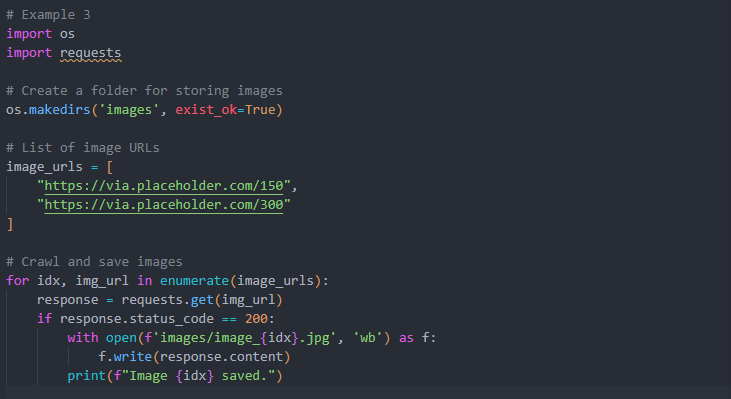
Simulating keyword submission to a search engine using requests.

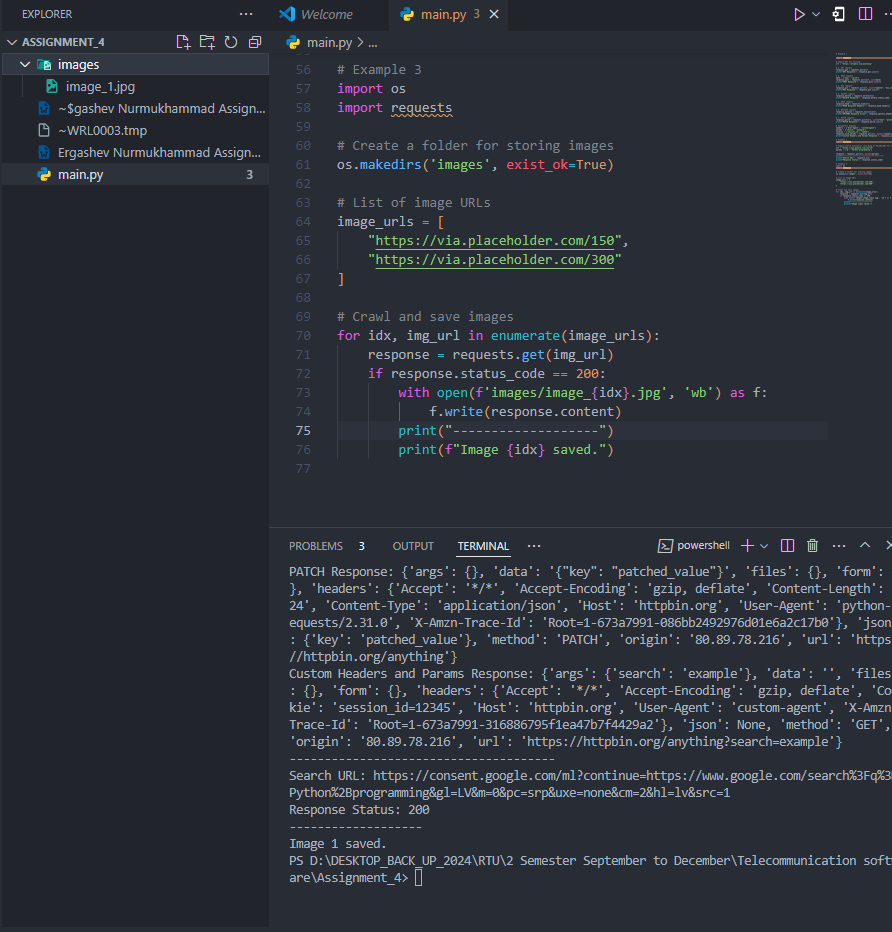




# Example 3: Image crawling.

Downloading images using requests.

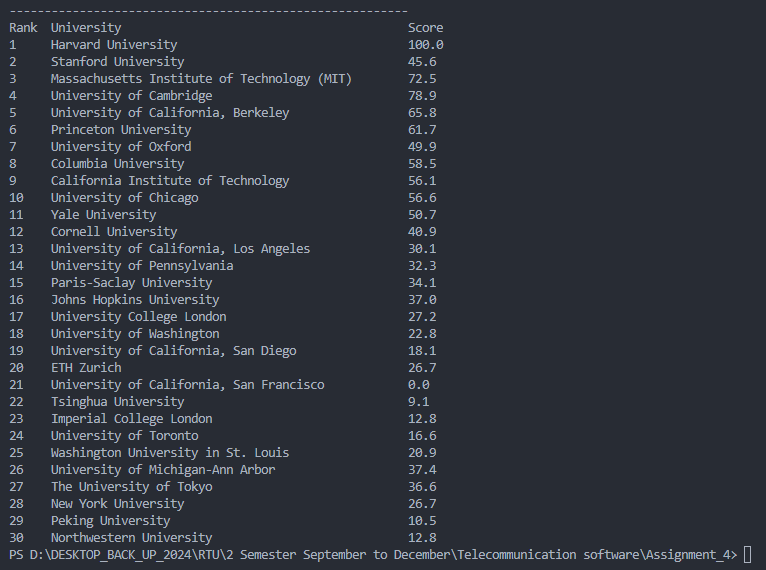




# Example 4: University ranking print

Using requests and BeautifulSoup to extract and display university rankings.

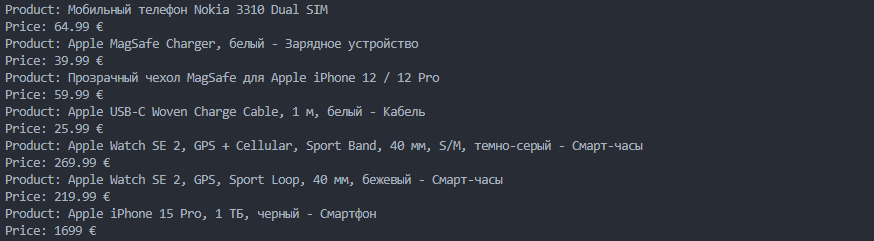




# Example 5: Crawling of goods web pages and printing relevant numbers, goods names, and prices with requests from the Python library.

Fetching product details (numbers, names, prices) from a webpage.



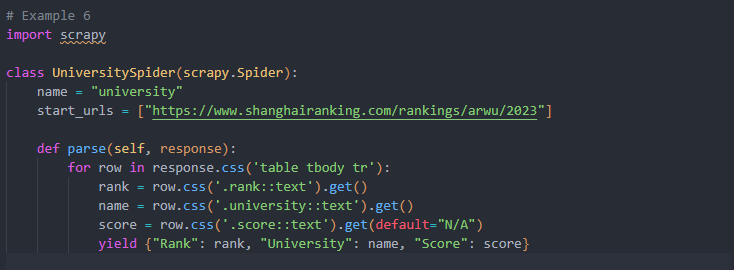


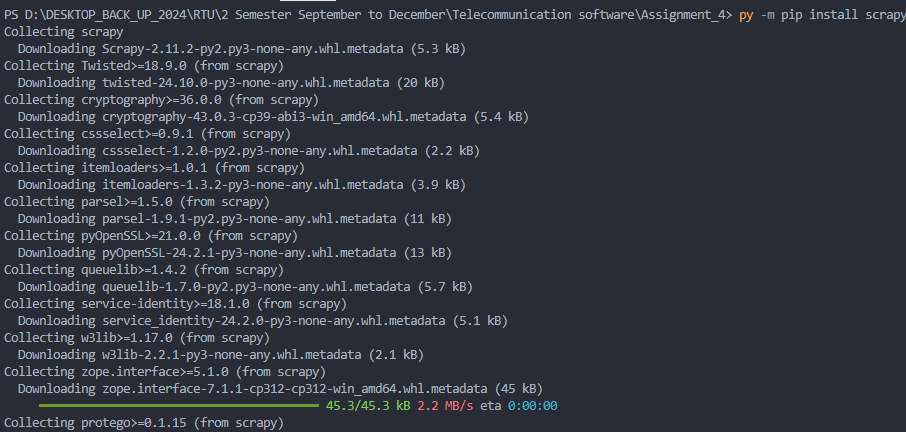
# Example 6: Please reference two public projects finishing your Scrapy project

Building a Scrapy project to extract university rankings and save to a TXT file.

**Steps**

1. Install Scrapy: pip install scrapy.
2. Generate a Scrapy project: scrapy startproject rankings.
3. Create a Spider:
   * scrapy genspider university shanghairanking.com.
4. Write a Scrapy script to crawl rankings.



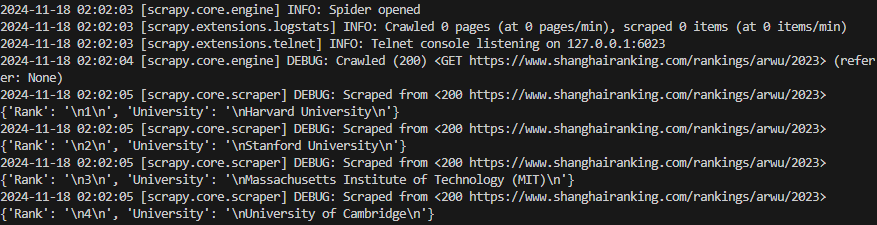




Run the spider to save results

scrapy crawl university -o rankings.txt

or can also use scrapy runspider



# Conclusion

In this project, we explored various web scraping techniques and tools to extract meaningful data from web pages using Python. The examples demonstrated included:

1. **HTTP Requests with requests**: Showcased how to perform different HTTP methods and manipulate parameters for effective data retrieval.
2. **Keyword Submission Interface**: Simulated search engine keyword submissions to demonstrate the practical use of query parameters.
3. **Image Crawling**: Highlighted methods to download and save images from URLs using requests.
4. **University Rankings Extraction**: Parsed a structured webpage to extract and display rankings using requests and BeautifulSoup.
5. **Product Crawling**: Scraped e-commerce data (product numbers, names, and prices) with real-world applications.
6. **Scrapy for Large-Scale Web Crawling**: Implemented a scalable scraping solution using Scrapy, with outputs stored in a structured file format.

Each example reflects practical web scraping applications, from extracting structured information to handling dynamic web content efficiently.

# Source Code Repository

All the source code for the examples discussed in this report has been uploaded to GitHub for reference and further development. The repository contains:

* Python scripts for each example.
* Dependencies and setup instructions.
* Sample outputs and data files.

Access the full repository here: [LINK HERE](https://github.com/ergashevnurik/ts_assignment4)