

****

**Midterm Project Report**

**Advanced Computer Programming**

**Web Scraping with Python**

**Student Name : MOON JUSUNG**

**Student ID : 112021188**

**Teacher : DINH-TRUNG VU**

**2024-04**

# Introduction

## Github

1. **Personal Github Account**: https://github.com/FluffyFrostBean
2. **Group Github Repository**: https://github.com/112021179/1
3. **Group Project Repository**: https://github.com/112021179/1/tree/main/midterm
4. **List of submitted file**s:
   * **104jobscraper**.py

## Topic

Build various web scraper of any topic to extract or simplify information proving its usefulness in our daily lives.(Web scraper for searching for jobs in biggest job searching website in Taiwan,” 104 Job Bank”)

## Project Overview

The requests, bs4, pandas and googletrans library has been used. My program prints all job listings from the constructed/modified URL which is “https://www.104.com.tw/jobs/search/?ro=0&kwop=7&keyword={keyword}&jobsource=2018indexpoc&city=6001001000&area={Location}”.Rather simply and concisely, it prints the job information like the job title ,salary/other descriptions if not specified, experience required, location and finally the job link for further information if the job satisfies the user. It provides a more friendly interface in English for those who do not read or understand Chinese well enough to navigate through the job searching portal. It should be useful for those who had trouble navigating the job searching site before due to language barrier and complexity of the website.

# Implementation

## Function translate\_to\_english(text):

This function takes a piece of text as input and translates it from Chinese (zh-tw) to English (en) using the Google Translate API.

## Function scrape\_104\_jobs(Location, keyword):

The scrape\_104\_jobs function efficiently extracts job listings from the 104 Job Bank website based on user-specified location and keyword parameters. It begins by constructing the appropriate URL using the provided location code and keyword. An HTTP GET request is then sent to retrieve the webpage content. Upon successful retrieval, BeautifulSoup is employed to parse the HTML structure, identifying and iterating through each job listing on the page. For every listing, key details such as title, location, experience, salary, and job link are extracted. These details are then translated into English using the translate\_to\_english function for better accessibility. The extracted and translated job information is stored in a structured format, specifically a list of dictionaries. Finally, this list is transformed into a Pandas DataFrame, enabling convenient analysis and manipulation of the collected job data. The DataFrame is then printed, providing a clear overview of the scraped job listings. In the event of a failed request, an informative error message is displayed, indicating the inability to retrieve data from the 104 Job Bank.

## “area\_data” Dictionary:

Contains city names and their corresponding location codes.

## 2.4“Area\_df” DataFrame:

Converts the area\_data dictionary into a pandas DataFrame for better organization and visualization.

Prints the DataFrame to display the city names and location codes.

## 2.5 Main Execution:

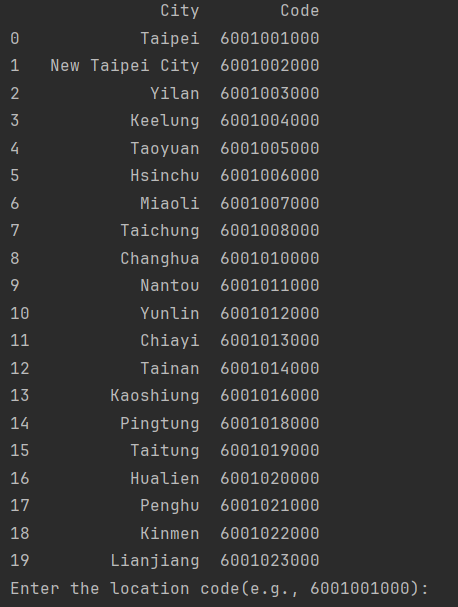
Prompts the user to enter the location code and keyword for job search.

Calls the scrape\_104\_jobs function with the provided location code and keyword.

# Results

## Result 1

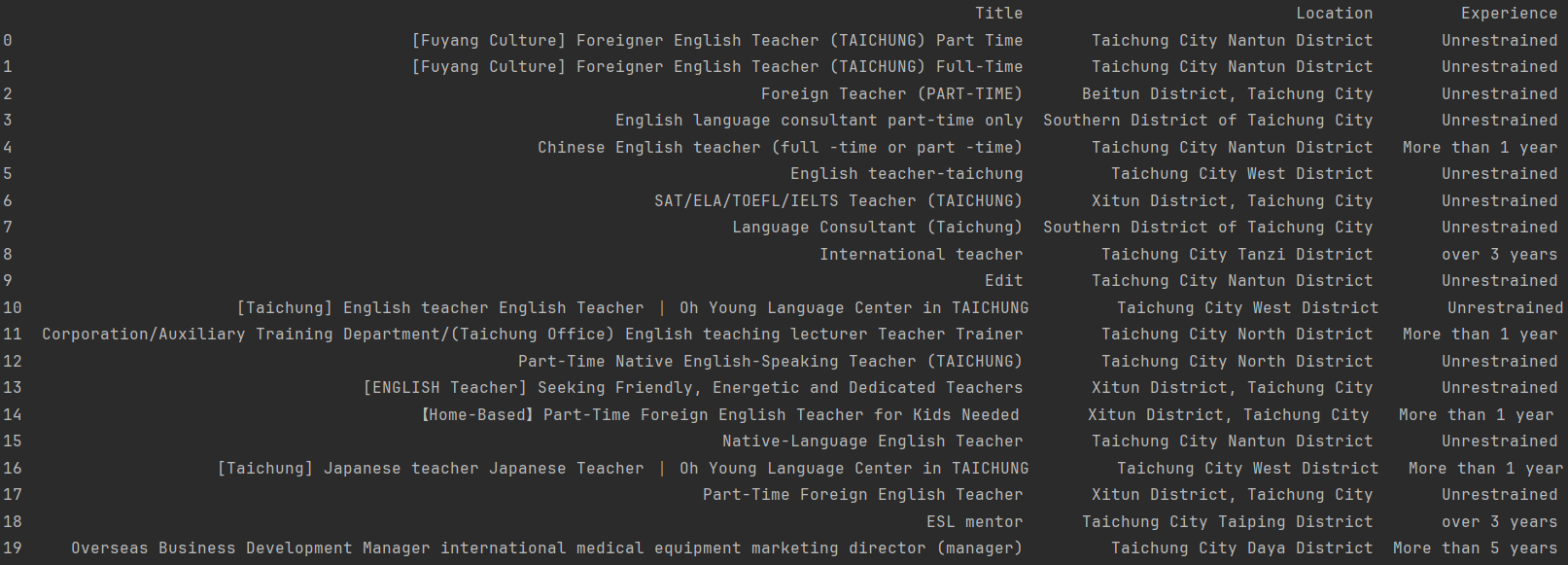
The script first prints a DataFrame (area\_df) containing the city names and their corresponding location codes.



The user inputs the desired location code and keyword after being prompt.



The code then constructs the URL for the search query on the 104 Job Bank website using the provided location code and keyword. It sends a GET request to the constructed URL and checks if the request was successful. If the request was successful (status code 200), the code parses the HTML content of the page using BeautifulSoup. It will take some time to print all the listings and translating the job content from Chinese to English. It will take around 1 to 3 minutes. Finally, it finds all job listings on the page and extracts relevant information such as job title, location, experience,



(scrolling right)salary/other descriptions, and job link for each job listing.



# Conclusions

The implemented code successfully scrapes job listings from the 104 Job Bank website based on user-provided location and keyword inputs. It effectively utilizes web scraping techniques with the requests and BeautifulSoup libraries to extract relevant information from HTML content. Additionally, it provides a user-friendly interface for entering search parameters and viewing job listings.