Take-home Assignment

Data Engineer (Crawling Focus) 🚀

FullName: Dang Tien Dat
Position: Data Engineer

Table of Contents

1. **Data Source**

- Linkedin: LinkedIn is a professional networking platform widely used by professionals across various industries. It provides a wealth of information about individuals' professional backgrounds, skills, education, work experience, and endorsements.
    - Link: https://www.linkedin.com/
- Jobgo: JobGo is a popular job search platform that aggregates job listings from various sources. While primarily known for job postings, JobGo also contains valuable data about candidates, particularly those actively seeking employment
    - Link: https://employer.jobsgo.vn/

2. Define Data Schema:

The data schema is implemented using Python's Scrapy framework, utilizing Scrapy items to define the structure of the data to be scraped from web sources. The PersonProfileItem class encapsulates the various attributes associated with candidate profiles. Below is a breakdown of the fields included in the schema:

- **id**: Unique identifier for the candidate profile.
- **fullname**: Full name of the candidate.
- **title**: Current job title or professional designation.
- **location**: Geographic location of the candidate.
- **industry**: Industry or sector in which the candidate operates.
- **current_company**: Name of the candidate's current employer or company.
- **past_company**: List of past employers or companies the candidate has worked for.

- **education**: Educational qualifications and institutions attended by the candidate.
- **profile_link**: URL link to the candidate's profile.
- **summary**: Summary or bio section describing the candidate's background and expertise.
- **website**: Personal or professional website associated with the candidate.
- **skill**: List of skills and expertise possessed by the candidate.
- **email**: Email address of the candidate.
- **image**: URL link to the candidate's profile picture.
- **experience**: Detailed work experience, including job roles, responsibilities, and dates.
- **birthday**: Date of birth or age of the candidate.
- **phone**: Contact phone number of the candidate.

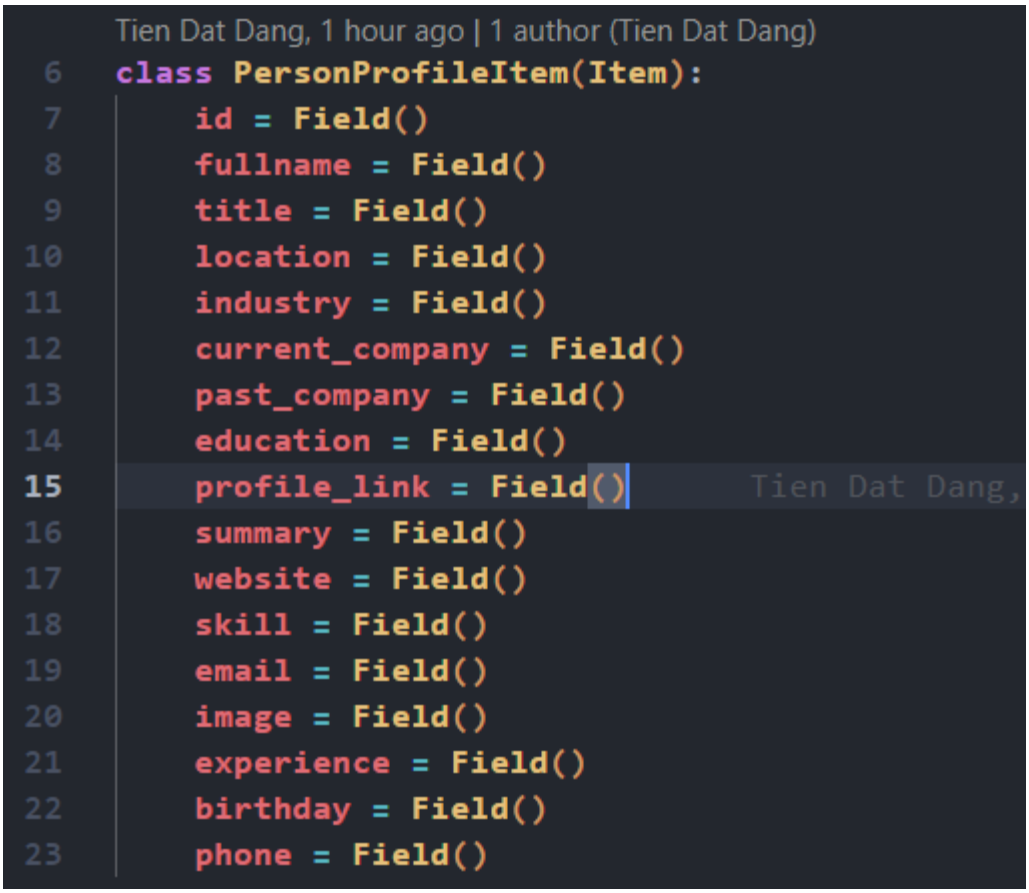


Figure 1. Schema

NOTE: Additional Schema in /schema Directory:

- The schema located in the /schema directory is described as more comprehensive and specific.
- It likely includes a broader range of attributes and may offer more detailed fields for capturing nuanced information about candidates.
- The schema may be structured to accommodate specific requirements or preferences of the organization or project, providing a tailored approach to candidate profiling.

3. Data Extraction

a. Data Extraction Techniques:

1. **Web Scraping with Scrapy:**

- Scrapy, a Python web crawling and scraping framework, was utilized for extracting data from web sources.
- Custom spiders were developed to navigate through the structure of web pages, locate relevant candidate information, and extract desired attributes.
- XPath selectors and CSS selectors were employed to target specific HTML elements containing candidate data, such as profile details, job listings, and educational backgrounds.

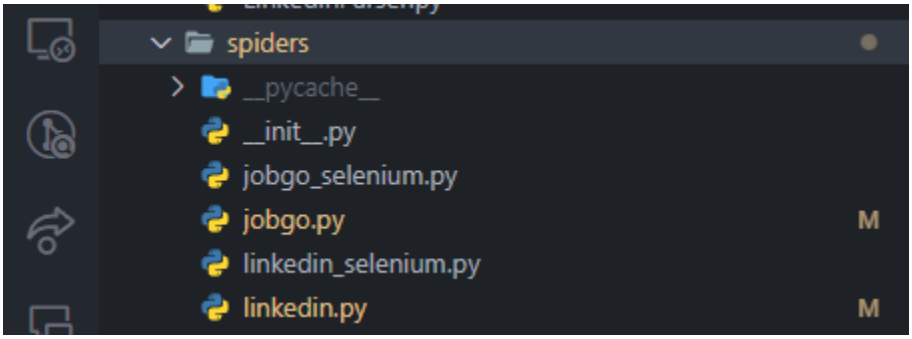


Figure 2. List spider

2. **Handling Anti-Scraping Measures:**

- Both LinkedIn and JobGo employ anti-scraping measures to prevent automated access to their platforms.
- To bypass these measures, techniques such as user-agent rotation, IP rotation, and request throttling were implemented.
- Randomized delays between requests were introduced to simulate human-like browsing behavior and avoid triggering anti-scraping mechanisms.

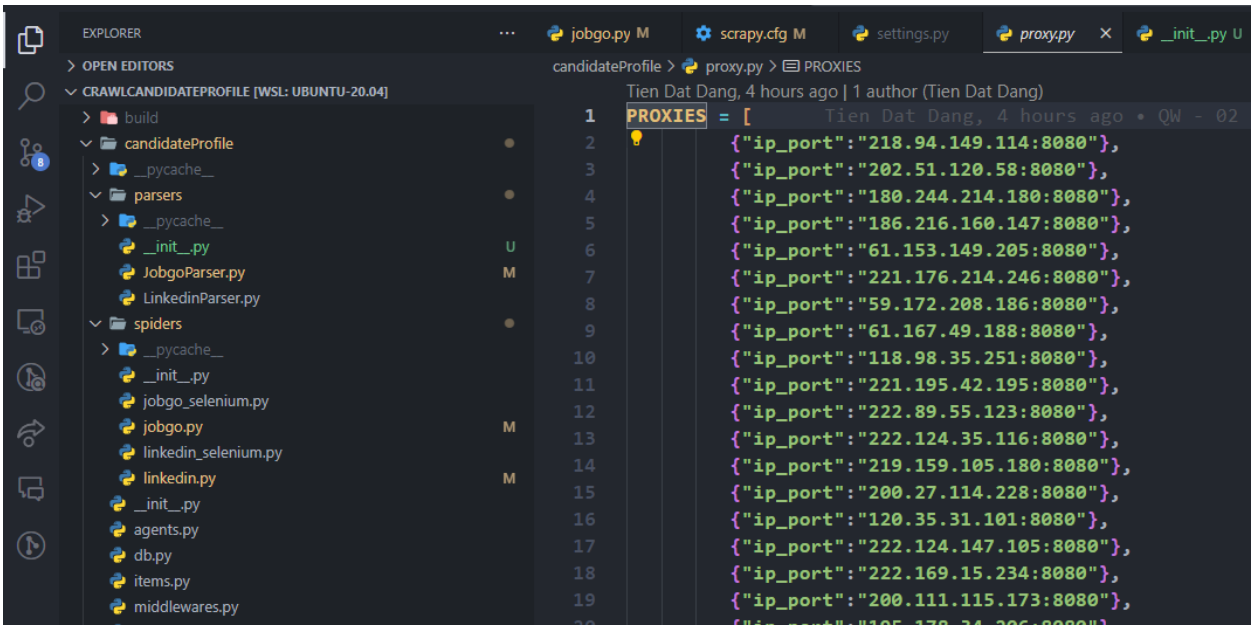


Figure 3. List Proxy

b. Data Extraction Challenges and Solutions:

CAPTCHA Challenges:

- LinkedIn and JobGo occasionally present CAPTCHA challenges to verify the legitimacy of user access, hindering automated scraping.

Solutions:

- CAPTCHA solving services or manual intervention were employed as temporary solutions to bypass CAPTCHA challenges.
- Utilizing proxy services to rotate IP addresses and avoid triggering CAPTCHA checks.

4. Data Storage in MongoDB

The candidate data stored in MongoDB is organized in collections, each representing a distinct entity or type of information. The schema design aims to maintain flexibility while accommodating the diverse attributes associated with candidate profiles. Here is an overview of the database schema:

```
Tien Dat Dang, 2 hours ago | 1 author (Tien Dat Dang)
from candidateProfile import settings
import pymongo

Tien Dat Dang, 2 hours ago | 1 author (Tien Dat Dang)
class MongoDBClient(object):
    def __init__(self, col, index=None):
        connection = pymongo.MongoClient(settings.MONGODB_URI)
        self.name_col = col
        self.db = connection[settings.MONGODB_DB]
        self.collection = self.db[col]
        if index:
            self.collection.create_index(index, unique=True)

    def refresh_collection(self):
        self.collection.drop()
        self.collection = self.db[self.name_col]

    def get_collection(self):
        return self.collection

    def _walk(self):
        skip = 0
        limit = 1000
        hasMore = True
        while hasMore:
            res = self.collection.find(skip=skip, limit=limit)
            hasMore = (res.count(with_limit_and_skip=True) == limit)
            for x in res:
                yield x
            skip += limit

    def walk(self):
        docs = []
        for doc in self._walk():
            docs.append(doc)
        return docs      Tien Dat Dang, 4 hours ago • QW - 02 | crawl
```

Figure 4. Declare connection with Mongo

5. Data Pipeline Orchestration

Zyte provides powerful tools for web scraping, data extraction, and pipeline orchestration, enabling the seamless integration of data from multiple sources into a unified workflow. The report discusses the architecture of the data pipeline, the role of Zyte in orchestrating the pipeline, and the benefits of using Zyte for managing and automating data extraction tasks.
Link: https://www.zyte.com/
Zyte's scheduling and task management features enable the automation and coordination of data extraction tasks.

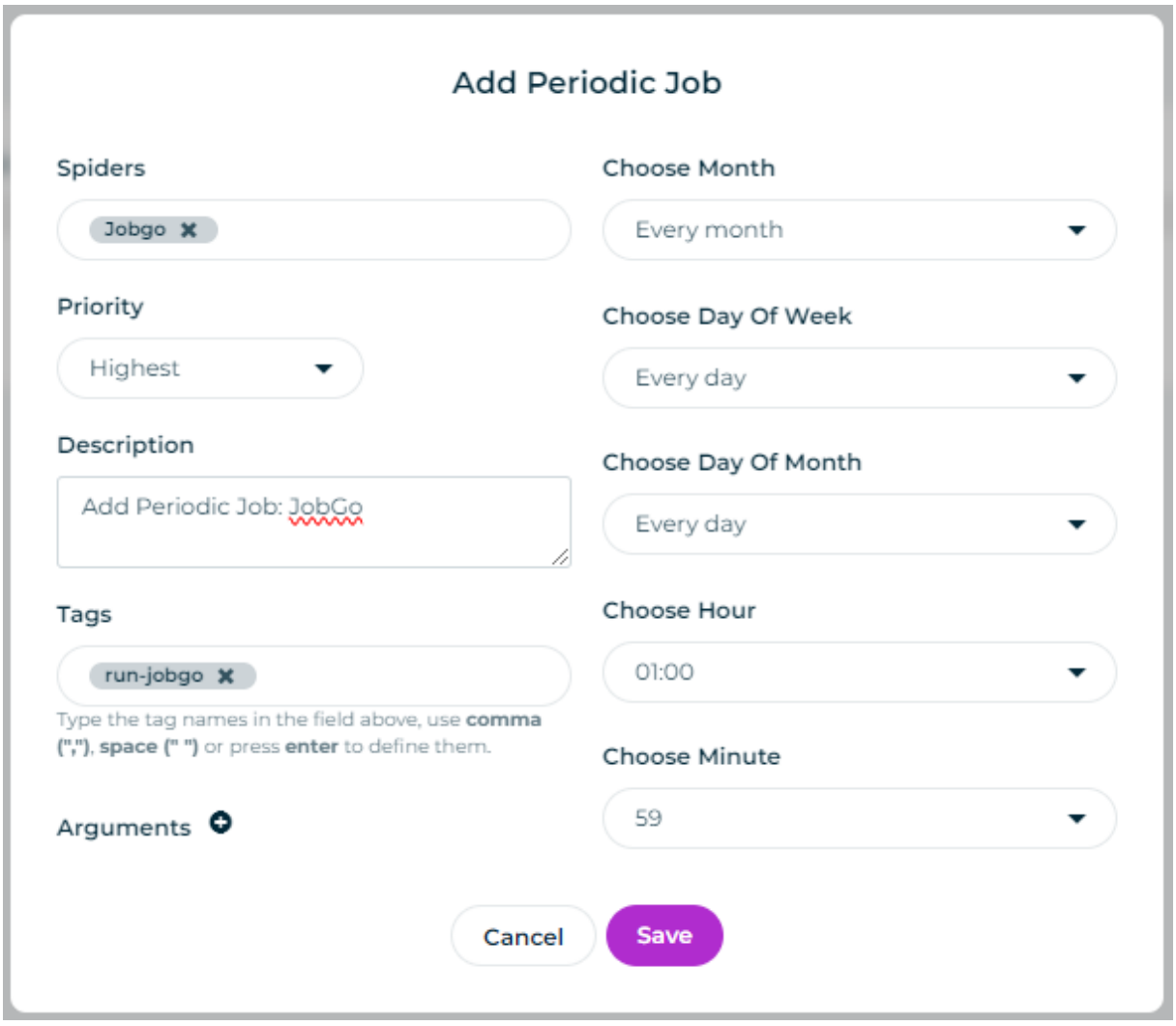


Figure 5. Add schedule Job JobGo

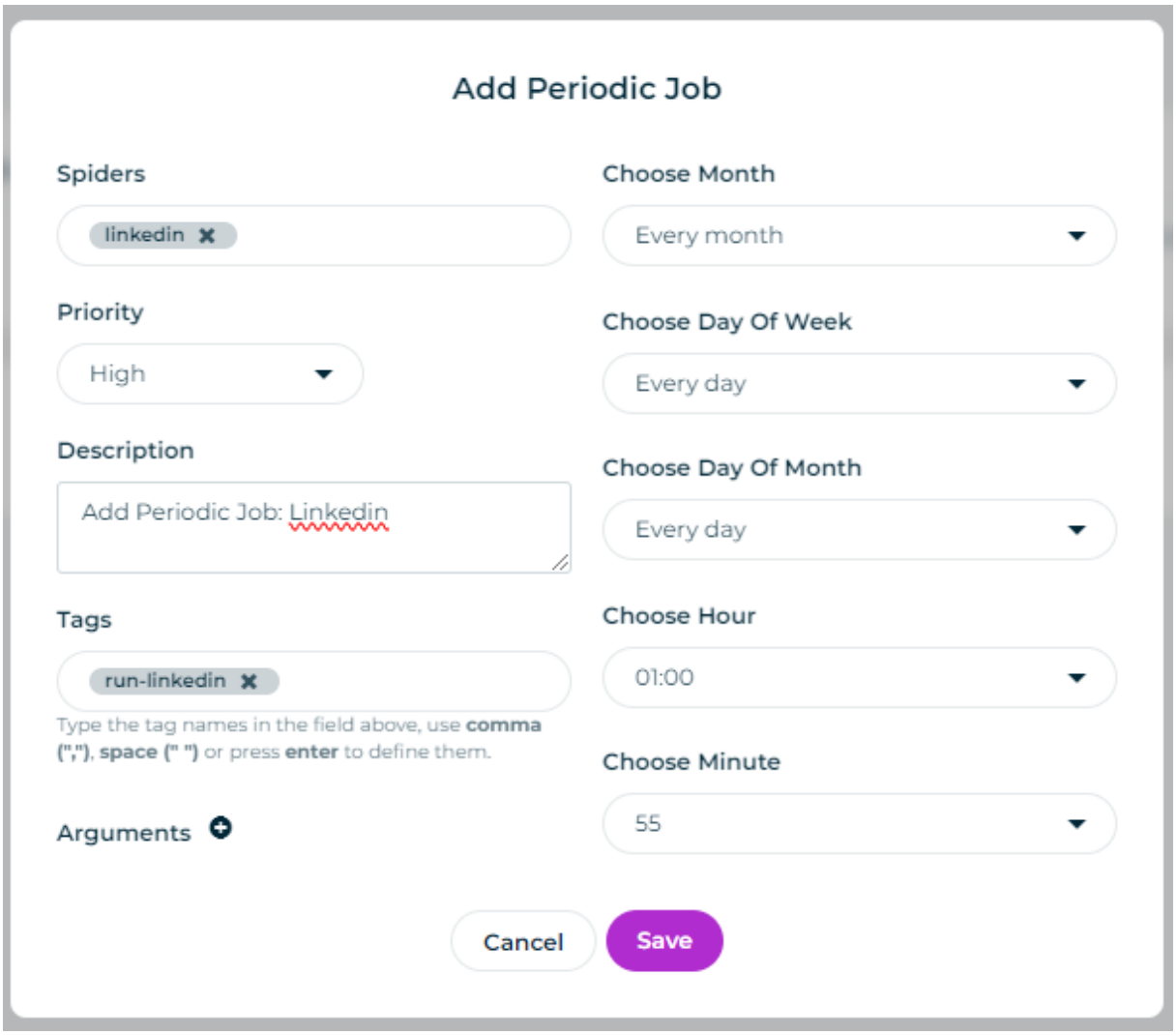


Figure 6. Add schedule Job Linkedin

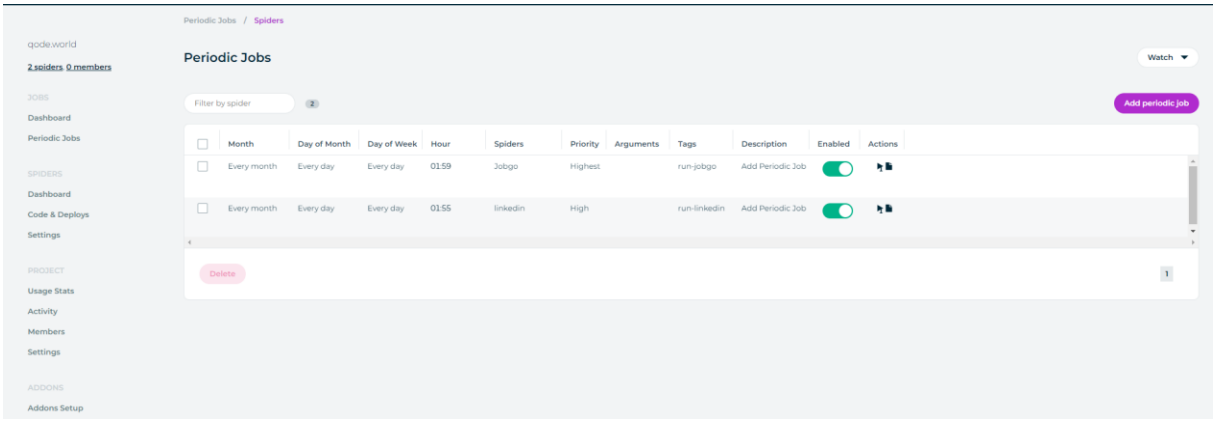


Figure 7. List Job created.

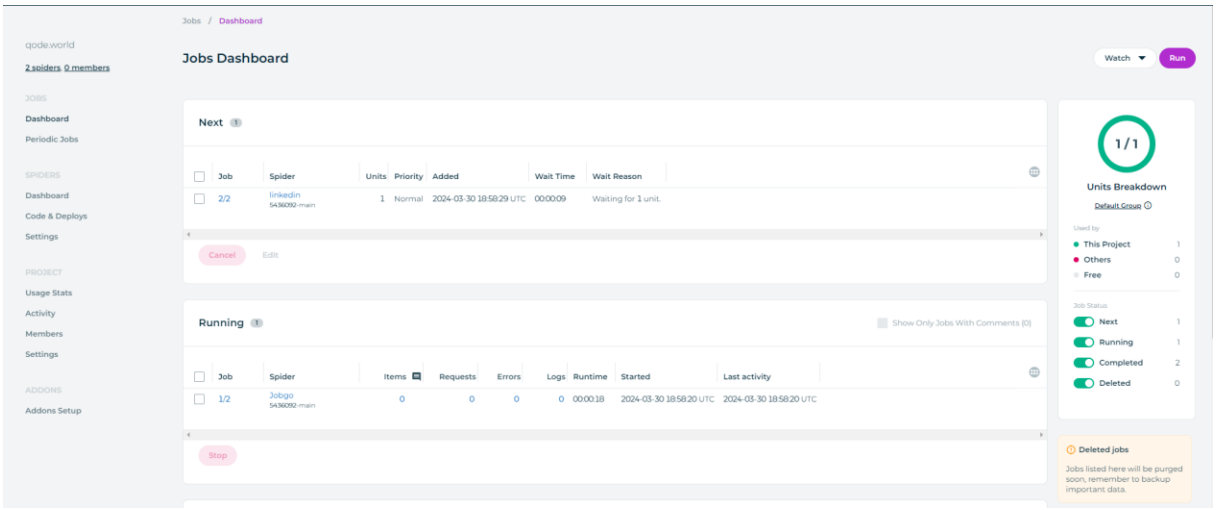


Figure 8. Job Dashboard

Figure 8, this showed job next and job running and number job is complete.

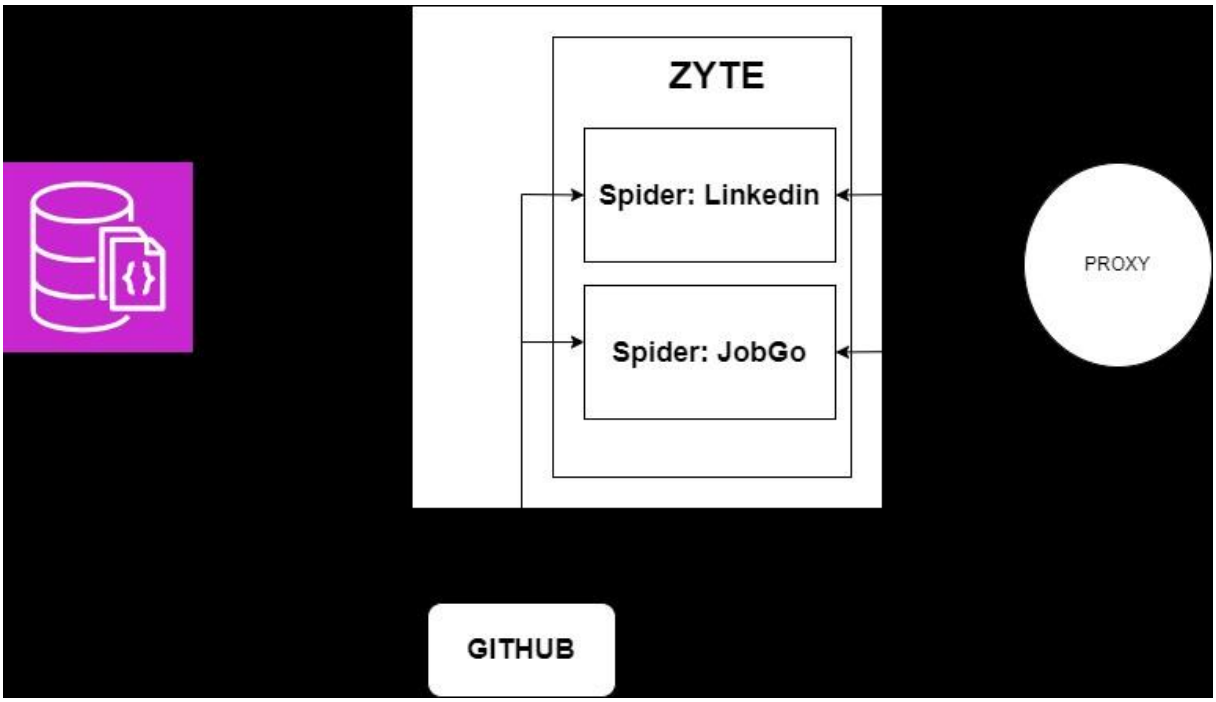
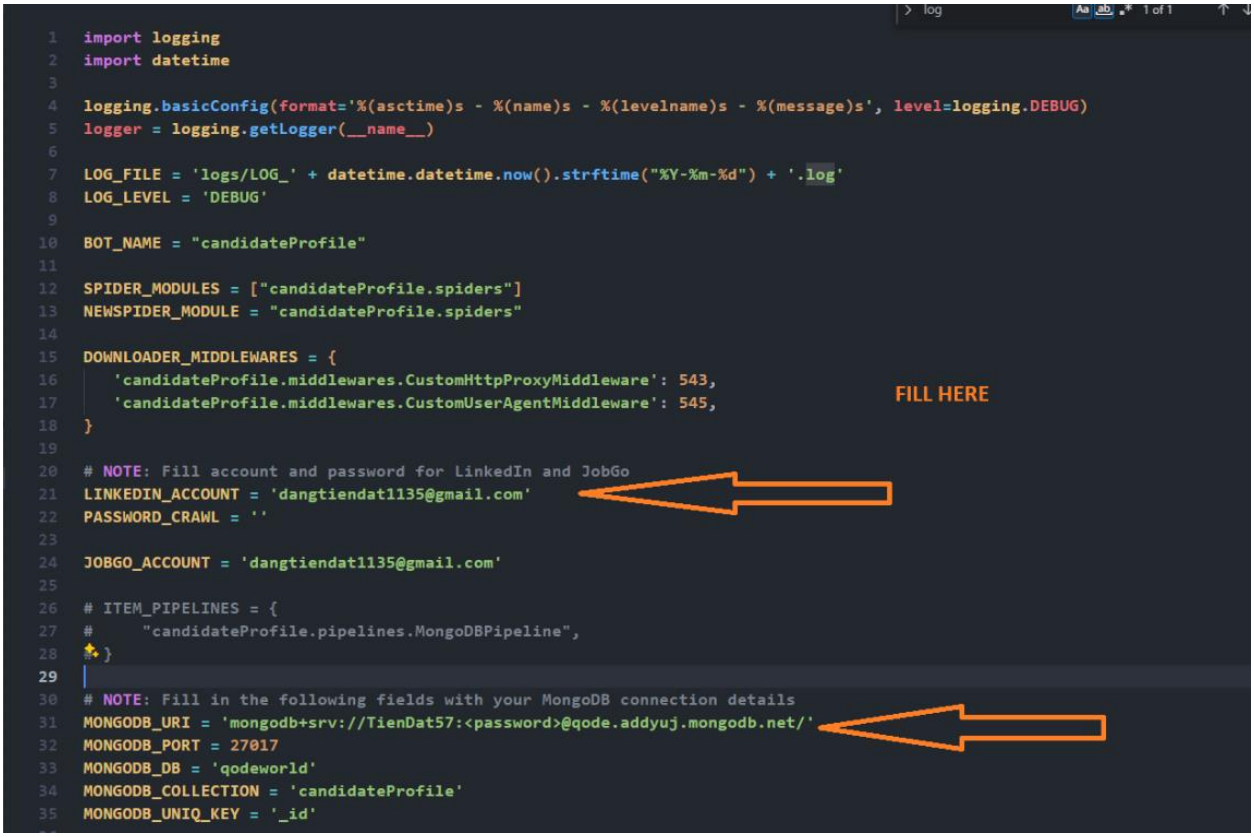


Figure 9. Architecture data pipeline

6. Instruction:

At file setttings.py, fill information necessary to here:




```
import logging
import datetime

logging.basicConfig(format='%(asctime)s - %(name)s - %(levelname)s - %(message)s', level=logging.DEBUG)
logger = logging.getLogger(__name__)

LOG_FILE = 'logs/LOG_' + datetime.datetime.now().strftime("%Y-%m-%d") + '.log'
LOG_LEVEL = 'DEBUG'

BOT_NAME = "candidateProfile"

SPIDER_MODULES = ["candidateProfile.spiders"]
NEWSPIDER_MODULE = "candidateProfile.spiders"

DOWNLOADER_MIDDLEWARES = {
   'candidateProfile.middlewares.CustomHttpProxyMiddleware': 543,
   'candidateProfile.middlewares.CustomUserAgentMiddleware': 545,
}

# NOTE: Fill account and password for LinkedIn and JobGo
LINKEDIN_ACCOUNT = 'dangtiendat1135@gmail.com'
PASSWORD_CRAWL = ''

JOBGO_ACCOUNT = 'dangtiendat1135@gmail.com'

# ITEM_PIPELINES = {
#     "candidateProfile.pipelines.MongoDBPipeline",
# }

# NOTE: Fill in the following fields with your MongoDB connection details
MONGODB_URI = 'mongodb+srv://TienDat57:<password>@qode.addyuj.mongodb.net/'
MONGODB_PORT = 27017
MONGODB_DB = 'qodeworld'
MONGODB_COLLECTION = 'candidateProfile'
MONGODB_UNIQ_KEY = '_id'
```

Run local:

Requirement:

- Installed Anaconda
- Python

Open shell:

- git clone https://github.com/TienDat57/qode-assignment

Install enviroment:

- pip install -r requirements.txt

Run spider:

- scrapy crawl linkedin
- scrapy crawl Jobgo

Link Source Code: https://github.com/TienDat57/qode-assignment