爬取大数据网站详细招聘信息

# 1．前言

## 1.1 课题描述

爬取51job（[https://search.51job.com](https://search.51job.com/)）中的大数据职位信息：**招聘链接（job\_href）, 工作名称（job\_name）, 发布日期（updatedate）, 公司名称（company\_name）, 预期工资（providesalary\_text）, 工资规模（companysize\_text）, 公司营业范围（companyind\_text）, 公司待遇（jobwelf）, 基本要求（attribute\_text）**，并将其保存在MongoDB**数据库（job数据库，information表）**中。

## 1.2 课题背景及意义

如今，人类社会已经进入了大数据时代，数据已经成为必不可少的部分，可见数据的获取非常重要。而爬虫作为获取数据的一大利器，可以让我们获取足够的数据并用于实际分析。

练习爬虫能力，获取51job（[https://search.51job.com](https://search.51job.com/)）中的大数据职位详细信息。

## 1.3 相关技术介绍

**request**：一个Python第三方库，处理URL资源特别方便。Requests 继承了urllib2的所有特性。Requests支持HTTP连接保持和连接池，支持使用cookie保持会话，支持文件上传，支持自动确定响应内容的编码，支持国际化的 URL 和 POST 数据自动编码。

**re**：Python库，用于正则匹配。

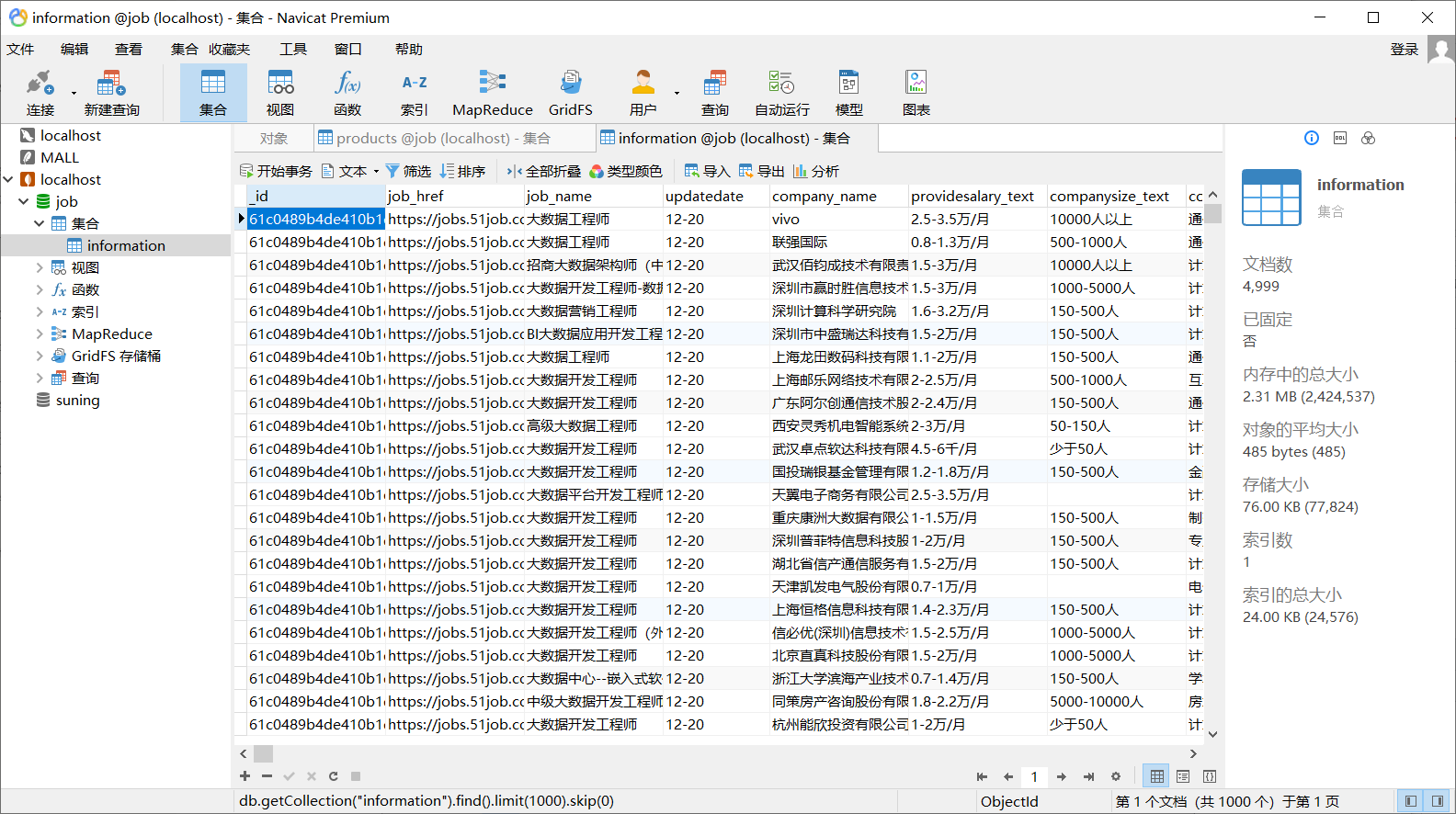
**json**：Python库，用于json的解析。

**pymongo**：一个Python第三方库，用于python操纵MongoDB。

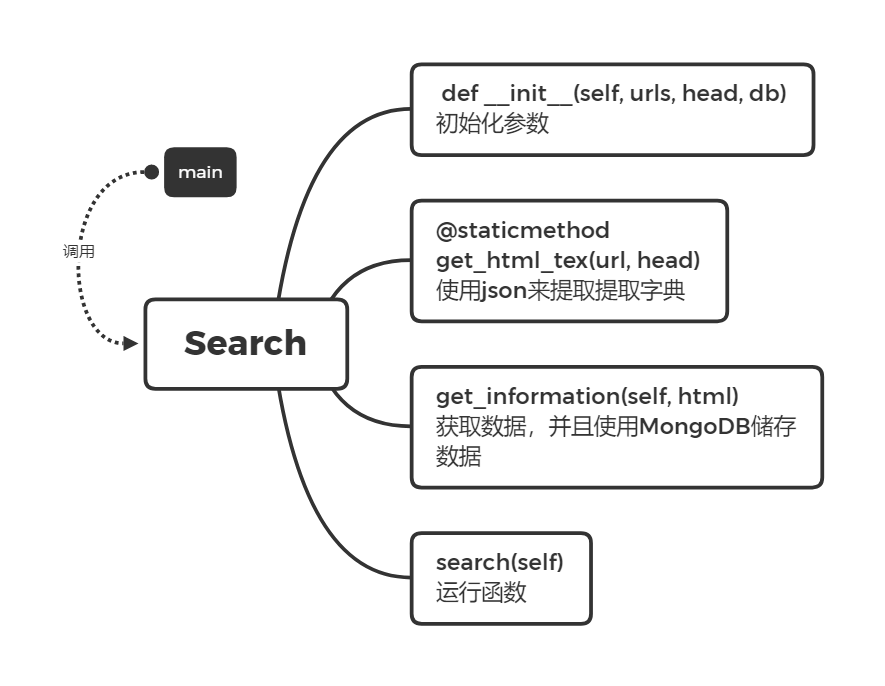
# 2.系统分析

爬取51job（[https://search.51job.com](https://search.51job.com/)）中的大数据职位信息：**招聘链接（job\_href）, 工作名称（job\_name）, 发布日期（updatedate）, 公司名称（company\_name）, 预期工资（providesalary\_text）, 工资规模（companysize\_text）, 公司营业范围（companyind\_text）, 公司待遇（jobwelf）, 基本要求（attribute\_text）**，并将其保存在MongoDB**数据库（job数据库，information表）**中。

**保存的MongoDB的图片（部分）**：

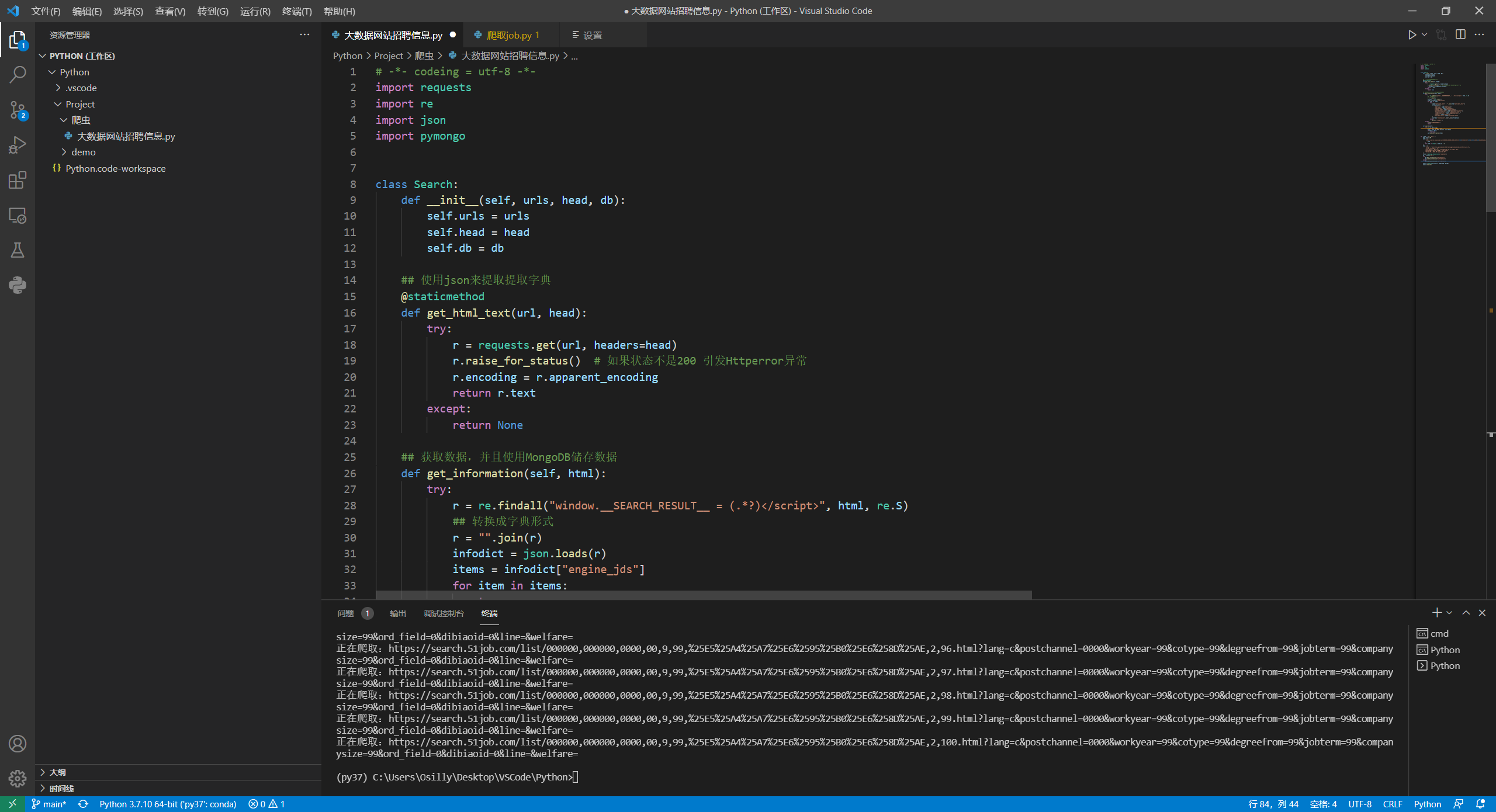


# 3.系统设计



# 4.系统实现

## 4.1 运行界面



## 4.2 代码实现

# -\*- codeing = utf-8 -\*-   
import requests   
import re   
import json   
import pymongo   
   
   
class Search:   
 def \_\_init\_\_(self, urls, head, db):   
 self.urls = urls   
 self.head = head   
 self.db = db   
   
 ## 使用json来提取提取字典   
 @staticmethod   
 def get\_html\_text(url, head):   
 try:   
 r = requests.get(url, headers=head)   
 r.raise\_for\_status()   
 r.encoding = r.apparent\_encoding   
 return r.text   
 except:   
 return None   
   
 ## 获取数据，并且使用MongoDB储存数据   
 def get\_information(self, html):   
 try:   
 r = re.findall("window.\_\_SEARCH\_RESULT\_\_ = (.\*?)</script>", html, re.S)   
 ## 转换成字典形式   
 r = "".join(r)   
 infodict = json.loads(r)   
 items = infodict["engine\_jds"]   
 for item in items:   
 try:   
 item["attribute\_text"] = "".join(item["attribute\_text"])   
 information = {   
 "job\_href": item["job\_href"],   
 "job\_name": item["job\_name"],   
 "updatedate": item["updatedate"],   
 "company\_name": item["company\_name"],   
 "providesalary\_text": item["providesalary\_text"],   
 "companysize\_text": item["companysize\_text"],   
 "companyind\_text": item["companyind\_text"],   
 "jobwelf": item["jobwelf"],   
 "attribute\_text": item["attribute\_text"],   
 }   
 self.db["information"].insert\_one(information)   
 except:   
 print("爬取失败!")   
 except:   
 print("获取html失败!")   
 return   
   
 def search(self):   
 for url in self.urls:   
 print("正在爬取：" + url)   
 html = self.get\_html\_tex(url, self.head)   
 if html is None:   
 continue   
 self.get\_information(html)   
   
   
if \_\_name\_\_ == "\_\_main\_\_":   
 page\_num = 100   
 urls = [   
 "https://search.51job.com/list/000000,000000,0000,00,9,99,%25E5%25A4%25A7%25E6%2595%25B0%25E6%258D%25AE,2,{}.html?lang=c&postchannel=0000&workyear=99&cotype=99&degreefrom=99&jobterm=99&companysize=99&ord\_field=0&dibiaoid=0&line=&welfare=".format(   
 page   
 )   
 for page in range(1, page\_num + 1)   
 ]   
 head = {   
 "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8",   
 "Accept-Language": "en",   
 "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) "   
 "AppleWebKit/537.36 (KHTML, like Gecko) "   
 "Chrome/95.0.4638.69 Safari/537.36",   
 }   
 client = pymongo.MongoClient("localhost")   
 db = client["job"]   
 try:   
 db.drop\_collection("information")   
 db.create\_collection("information")   
 except:   
 db.create\_collection("information")   
   
 search = Search(urls=urls, head=head, db=db)   
 search.search()

# 5. 结束语

对爬虫有了新的了解，学会了一些基本爬虫操作。