爬取苏宁易购商品信息

## 1．前言

### 1.1 课题描述

爬取苏宁易购商品（apple）信息（<https://www.suning.com/>）：**商品图片链接（image），商品价格（price），商品标题（title），商品被评论数（comment\_num），商品所属店家（shop）**，并将其保存在MongoDB（连接：localhost，数据库：suning，集合：products）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，获取苏宁易购商品信息的详细信息，并用于之后的处理与分析。

### 1.3 相关技术介绍

**selenium**：一个Python第三方库。Selenium是最广泛使用的开源Web UI（用户界面）自动化测试套件之一。它最初由杰森·哈金斯（Jason Huggins）于2004年开发，作为[Thought Works](https://baike.baidu.com/item/ThoughtWorks/5236600)的内部工具。Selenium支持跨不同浏览器，平台和编程语言的自动化。

本次项目主要使用selenium来模拟人工访问。

**pyquery**：Python第三方库，pyquery相当于[jQuery](https://so.csdn.net/so/search?from=pc_blog_highlight&q=jQuery)的[python](https://so.csdn.net/so/search?from=pc_blog_highlight&q=python)实现，可以用于解析HTML网页等。

本次项目主要使用pyquery.PyQuery来解析HTML网页。

**webdriver\_manager.chrome**：一个Python第三方库，使用它可以避免安装driver插件，使用便捷。

本次项目主要使用webdriver\_manager.chrome.ChromeDriverManager来模拟运行chrome浏览器。

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

**time**：Python库，可以用于格式化日期和时间。

**re**：Python库，用于正则匹配。

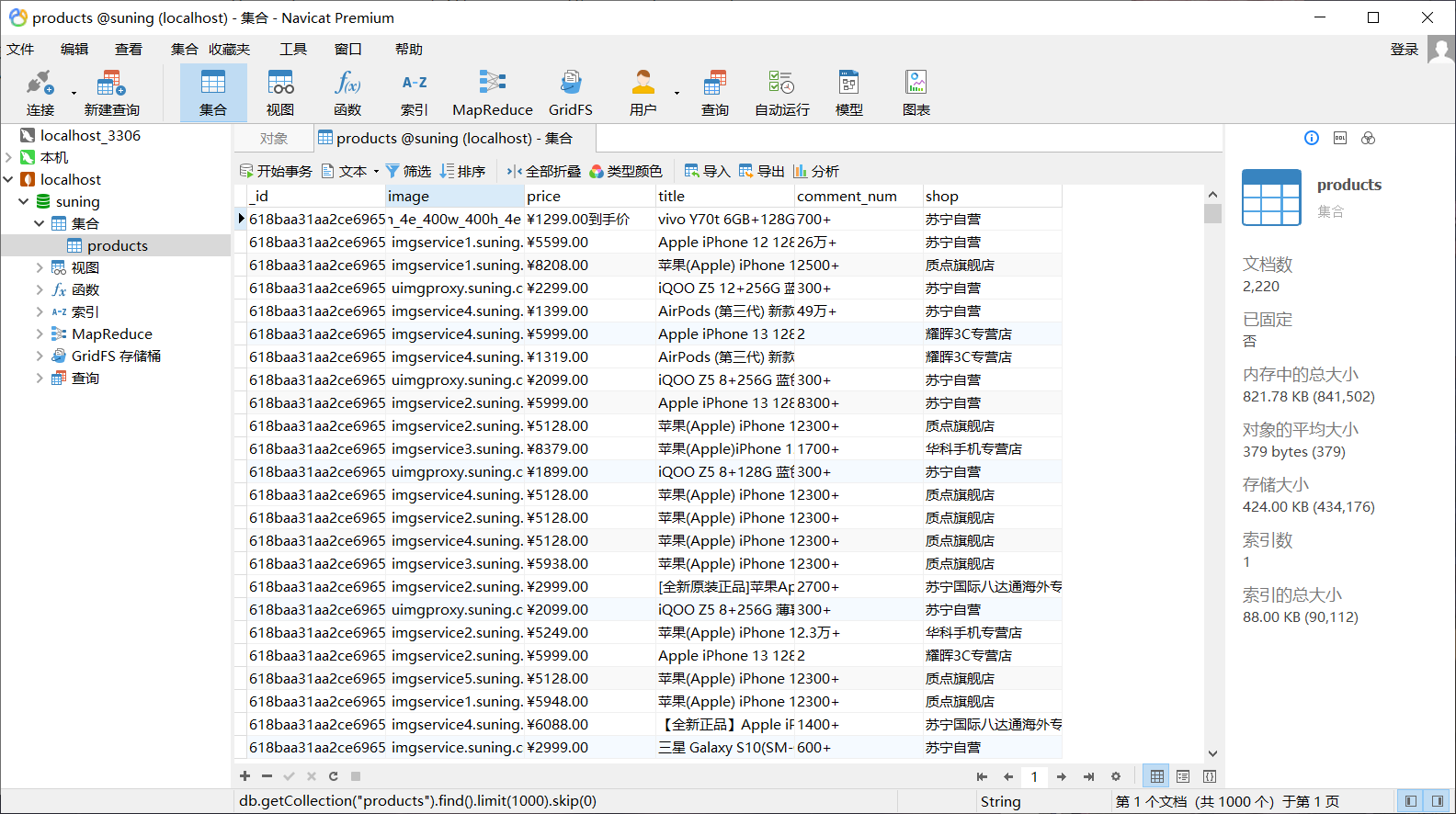
## 2.系统分析

爬取苏宁易购商品（apple）信息（<https://www.suning.com/>）：**商品图片链接（image），商品价格（price），商品标题（title），商品被评论数（comment\_num），商品所属店家（shop）**，并将其保存在MongoDB（连接：localhost，数据库：suning，集合：products）中。

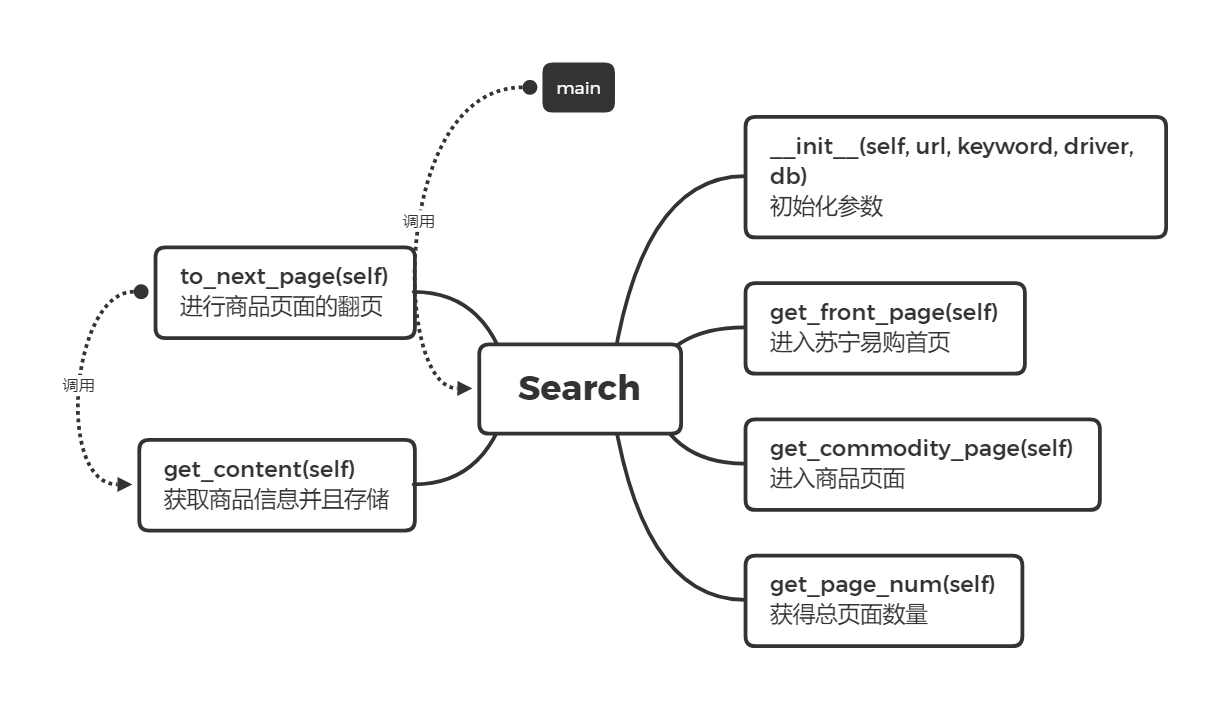
**输出结果的图片（部分）**：



**保存的MongoDB的图片（部分）**：

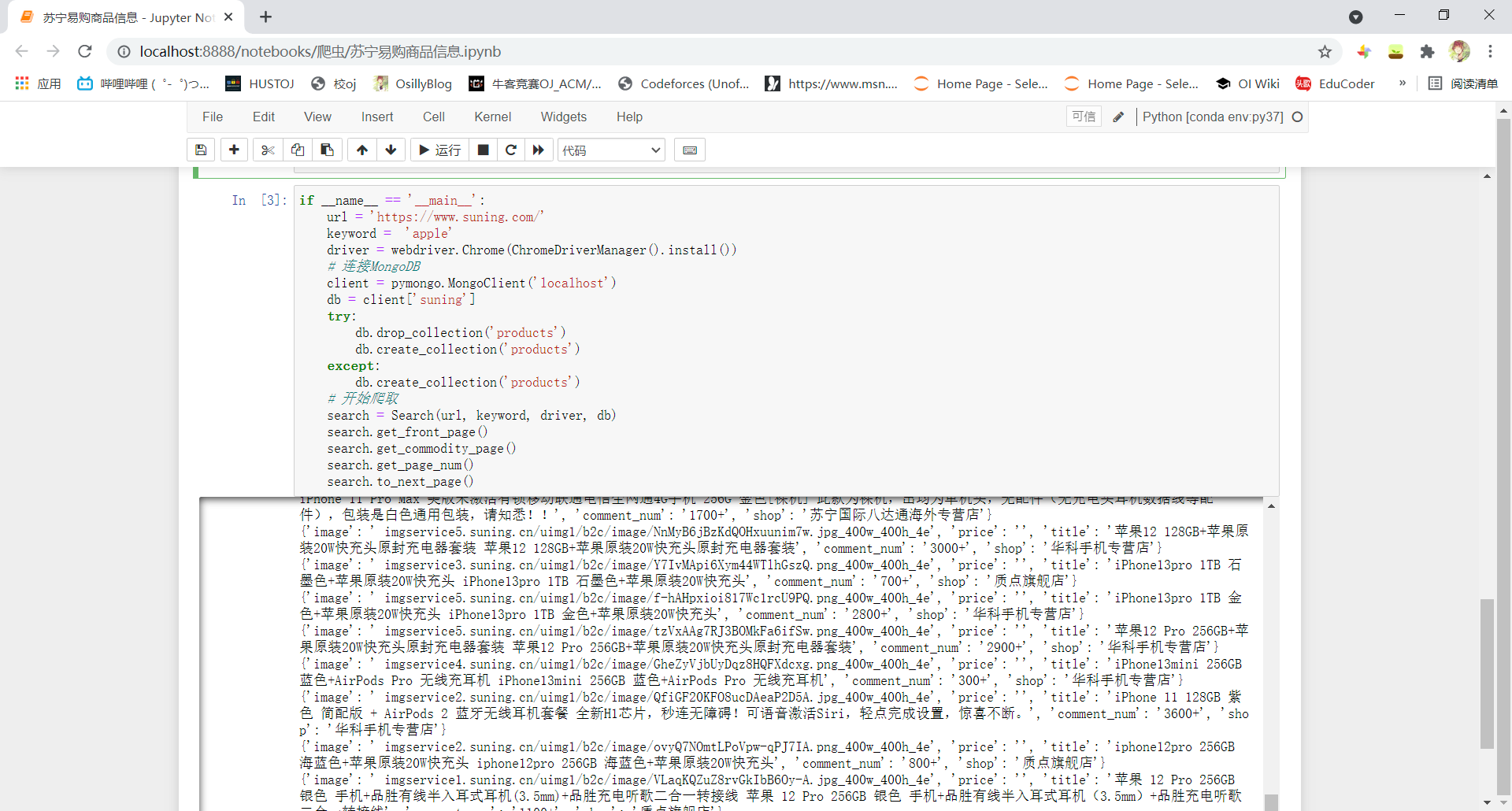


## 3.系统设计



## 4.系统实现

### 4.1 运行界面



### 4.2 代码实现

#!/usr/bin/env python   
# coding: utf-8   
   
# In[1]:   
   
   
from selenium import webdriver   
from selenium.webdriver.common.by import By   
from selenium.webdriver.support.ui import WebDriverWait   
from selenium.webdriver.support import expected\_conditions as EC   
import re   
import time   
import pymongo   
from selenium.webdriver.common.keys import Keys   
from pyquery import PyQuery as pq   
from webdriver\_manager.chrome import ChromeDriverManager   
   
   
# In[2]:   
   
# 爬取商品信息   
class Search:   
 def \_\_init\_\_(self, url, keyword, driver, db):   
 self.url = url   
 self.keyword = keyword   
 self.driver = driver   
 self.db = db   
   
 # 进入苏宁易购首页   
 def get\_front\_page(self):   
 self.driver.get(self.url)   
   
 # 进入商品页面   
 def get\_commodity\_page(self):   
 # 输入的搜索框   
 input = WebDriverWait(self.driver, 10).until(   
 EC.presence\_of\_element\_located((By.CSS\_SELECTOR, '#searchKeywords'))   
 )   
 # 搜索按钮   
 submit = WebDriverWait(self.driver, 10).until(   
 EC.element\_to\_be\_clickable((By.CSS\_SELECTOR, "#searchSubmit")))   
 input.send\_keys(self.keyword)   
 submit.click()   
   
 # 获得总页面数量   
 def get\_page\_num(self):   
 target = self.driver.find\_element\_by\_css\_selector('#bottom\_pager > div > span.page-more')   
 self.driver.execute\_script("arguments[0].scrollIntoView();", target)   
 time.sleep(3)   
 total = WebDriverWait(self.driver, 10).until(   
 EC.presence\_of\_element\_located((By.CSS\_SELECTOR, '#bottom\_pager > div > span.page-more')))   
 total\_text = total.text   
 # 正则处理   
 pattern = re.compile('\S\S(\d+).\*?')   
 result = re.search(pattern, total\_text)   
 self.total = int(result.group(0)[1:])   
   
 # 获取商品信息并且存储   
 def get\_content(self):   
 try:   
 # 选择展示框   
 WebDriverWait(self.driver, 20).until(   
 EC.presence\_of\_element\_located((By.CSS\_SELECTOR, '#product-list .item-wrap')))   
   
 html = self.driver.page\_source   
 doc = pq(html)   
 # 获得所有商品的数据   
 items = doc('#product-list .item-wrap').items()   
 for item in items:   
 proucts = {   
 'image': item.find(".img-block")('a')('img').attr('src').strip().replace('//', ' '),   
 'price': item.find('.def-price').text().replace('\n', ' '),   
 'title': item.find('.title-selling-point').text().strip().replace('\n', ' '),   
 'comment\_num': item.find('.info-evaluate')('a')('i').text(),   
 'shop': item.find('.store-stock').text(),   
 }   
 print(proucts)   
 # 保存在products集合中   
 try:   
 self.db['products'].insert\_one(proucts)   
 except:   
 continue   
 except:   
 return   
   
 # 进行商品页面的翻页   
 def to\_next\_page(self):   
 for page in range(1, self.total + 1):   
 print('当前页:%d/%d' % (page, self.total))   
 try:   
 # 找到输入框   
 time.sleep(1)   
 inputs = WebDriverWait(self.driver, 20).until(   
 EC.presence\_of\_element\_located((By.CSS\_SELECTOR, '#bottomPage')))   
 time.sleep(1)   
 inputs.clear()   
 # 找到确定按钮   
 submit = WebDriverWait(self.driver, 10).until(   
 EC.element\_to\_be\_clickable(   
 (By.CSS\_SELECTOR, "#bottom\_pager > div > a.page-more.ensure")))   
 inputs.send\_keys(page)   
 submit.send\_keys(Keys.ENTER)   
 target = self.driver.find\_element\_by\_css\_selector('#bottom\_pager > div > span.page-more')   
 # 将页面下拉至底部   
 self.driver.execute\_script("arguments[0].scrollIntoView();", target)   
 time.sleep(3)   
 self.get\_content()   
 # 进行判定   
 WebDriverWait(self.driver, 10).until(   
 EC.element\_to\_be\_clickable(   
 (By.CSS\_SELECTOR, '#bottom\_pager > div > a.cur')), str(page))   
 except:   
 print("wrong page!")   
 return   
 self.driver.close()   
   
   
# In[3]:   
   
   
if \_\_name\_\_ == '\_\_main\_\_':   
 url = 'https://www.suning.com/'   
 keyword = 'apple'   
 driver = webdriver.Chrome(ChromeDriverManager().install())   
 # 连接MongoDB   
 client = pymongo.MongoClient('localhost')   
 db = client['suning']   
 try:   
 db.drop\_collection('products')   
 db.create\_collection('products')   
 except:   
 db.create\_collection('products')   
 # 开始爬取   
 search = Search(url, keyword, driver, db)   
 search.get\_front\_page()   
 search.get\_commodity\_page()   
 search.get\_page\_num()   
 search.to\_next\_page()

## 5.结束语

对爬虫有了新的了解，学会了使用selenium来爬取信息。