# 爬虫期中作业报告

目录

[爬虫期中作业报告 1](#_Toc7186349)

[一、项目需求 2](#_Toc7186350)

[二、技术简介 2](#_Toc7186351)

[三、项目实现： 3](#_Toc7186352)

[1、静态页面爬虫程序（豆瓣电影top250） 3](#_Toc7186353)

[2、动态页面爬虫程序（京东手机） 5](#_Toc7186354)

[3、新建Django项目来展示数据： 7](#_Toc7186355)

[4、完整项目目录 11](#_Toc7186356)

## 一、项目需求

编写爬虫程序爬取网页数据并存入数据库，再将数据利用Django展示在网页上。

## 二、技术简介

爬虫：请求网站并提取数据的自动化程序。

Mysql：最流行的关系型数据库管理系统。

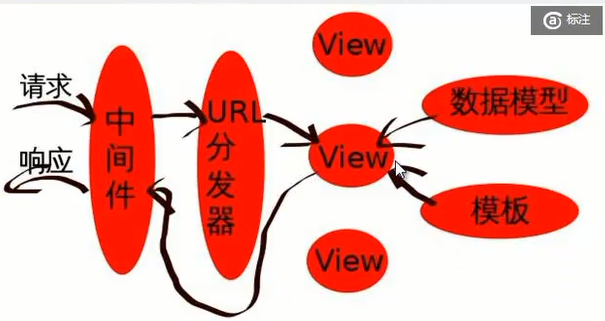
Django：一个开放源代码的Web应用框架，由Python写成。采用了MVC框架。适用于快速高效的上手开发一个网站。

Django MTV模式：

m-models：用于业务对象和数据库之间的映射

t-templates：用于展示业务的模板页面

v-views：负责业务逻辑。



## 三、项目实现：

### 1、静态页面爬虫程序（豆瓣电影top250）

#### 1.1、实现思路：

用request库请求豆瓣页面，得到第一页的html代码；用正则表达式re库分析每条电影的正则，得到电影名、主演等信息放入字典。这样就得到第一页的25条电影信息；再点击下一页观察页面url的变化<https://movie.douban.com/top250?start=2>，发现只有start的值改变，因此只需改变每次传入的start的值并循环调用main（start）方法10次即可。最后用多进程来提高爬取速度。

#### 1.2、主要代码：

#请求网页

def get\_one\_page(url):  
 try:  
 respone=requests.get(url)  
 if respone.status\_code==200:  
 return respone.text  
 return None  
 except RequestException:  
 return None

#解析单个页面  
def parse\_one\_page(html):  
 pattern = re.compile('<li>.\*?<em.\*?>(\d+)</em>.\*?src="(.\*?)".\*?"title">(.\*?)</span>'  
 +'.\*?<p class="">(.\*?)&nbsp;&nbsp;&nbsp;(.\*?)<br>(.\*?)&nbsp;/&nbsp;'  
 +'(.\*?)&nbsp;/&nbsp;(.\*?)</p>.\*?"v:average">(.\*?)</span>.\*?</li>',re.S)  
 items = re.findall(pattern, html)  
 for item in items:  
 data = {  
 'index': item[0],  
 'title': item[2],  
 'director': item[3].strip()[3:],  
 'actor': item[4][3:],  
 'time': item[5].strip(),  
 'region': item[6],  
 'type': item[7].strip(),  
 'score': item[8],  
 'images': item[1]  
 }  
 write\_to\_mysql(data)  
 return items

#存入mysql  
def write\_to\_mysql(data):  
 db = pymysql.connect(host='localhost', user='root', password='123456', db='myblog', charset='utf8')  
 cur = db.cursor()  
 sqlc = '''  
 insert into mymovie\_movies  
 values(null,%s,%s,%s,%s,%s,%s,%s,%s)  
 '''  
 try:  
 if cur.execute(sqlc, (data["title"], data["director"], data["actor"], data["time"], data["region"], data["type"], data["score"], data["images"])):  
 print('Successful')  
 db.commit()  
 except Exception as e:  
 print(e)  
 print('Failed')  
 db.rollback()  
 cur.close()  
 db.close()  
  
def main(start):  
 url = 'https://movie.douban.com/top250?start=' + str(start)  
 html = get\_one\_page(url)  
 for item in parse\_one\_page(html):  
 print(item)  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 pool = Pool()  
 pool.map(main, [i\*25 for i in range(10)])

#### 1.3、效果截图;





### 2、动态页面爬虫程序（京东手机）

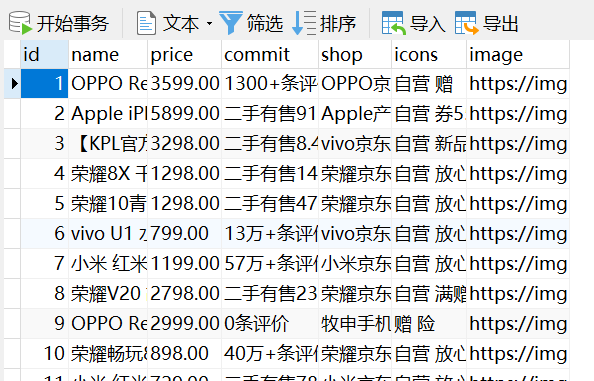
#### 2.1、实现思路：

用selenium库模拟浏览器来爬取京东手机商品信息。首先下载对应的chromedriver，并设置为无头浏览；通过css选择器来找到商品的输入框并输入“手机”且提交；同样的模拟翻页的情况，总共有100页；引入pyquery库，通过分析节点来获取到商品的不同信息；最后存入到数据库。

#### 2.2、主要代码：

def search():  
 try:  
 browser.get("https://www.jd.com/")  
 input = wait.until(EC.presence\_of\_element\_located((By.ID, 'key')))  
 submit = wait.until(EC.presence\_of\_element\_located((By.CSS\_SELECTOR, '#search > div > div.form > button')))  
 input.send\_keys(keyword)  
 submit.click()  
 total = wait.until(EC.presence\_of\_element\_located((By.CSS\_SELECTOR, '#J\_bottomPage > span.p-skip > em:nth-child(1) > b')))  
 get\_products()  
 return total.text  
 except TimeoutException:  
 return search()  
  
def next\_page(page\_number):  
 print('正在翻页', page\_number)  
 try:  
 # 滑动到底部，加载出后三十个货物信息  
 browser.execute\_script("window.scrollTo(0, document.body.scrollHeight);")  
 time.sleep(5)  
  
 input = wait.until(EC.presence\_of\_element\_located((By.CSS\_SELECTOR, '#J\_bottomPage > span.p-skip > input')))  
 submit = wait.until(EC.element\_to\_be\_clickable((By.CSS\_SELECTOR, '#J\_bottomPage > span.p-skip > a')))  
 input.clear()  
 input.send\_keys(page\_number)  
 submit.click()  
 wait.until(EC.text\_to\_be\_present\_in\_element((By.CSS\_SELECTOR, '#J\_bottomPage > span.p-num > a.curr'), str(page\_number)))  
 get\_products()  
 except TimeoutException:  
 next\_page(page\_number)  
  
def get\_products():  
 wait.until(EC.presence\_of\_element\_located((By.CSS\_SELECTOR, '#J\_goodsList > ul')))  
 html = browser.page\_source  
 doc = pq(html)  
 items = doc('#J\_goodsList .gl-warp .gl-item .gl-i-wrap').items()  
 images = browser.find\_element\_by\_xpath('//div[@class="gl-i-wrap"]/div[1]/a/img')  
 #获取商品信息列表  
 for item in items:  
 product = {  
 'name': re.search('.\*?\n', item.find('.p-name').text()).group(0)[:-1],  
 'price': item.find('.p-price').text()[2:],  
 'commit': re.sub('\n', '', item.find('.p-commit').text()),  
 'shop': item.find('.p-shop').text(),  
 'icons': re.sub('\n', ' ', item.find('.p-icons').text()),  
 'image': images.get\_attribute('src')  
 }  
 print(product)  
 write\_to\_mysql(product)

#### 2.3、效果截图：

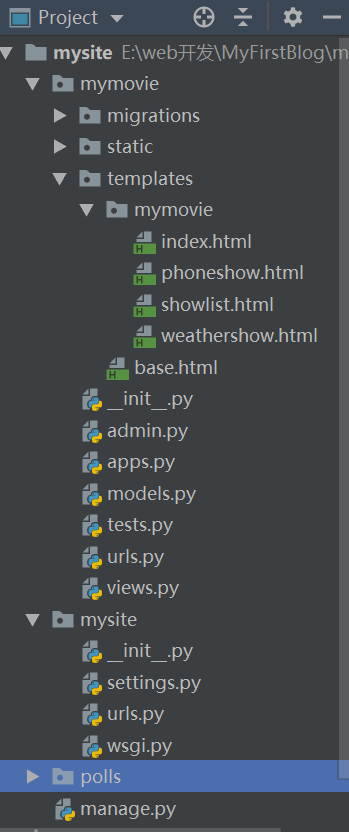


### 3、新建Django项目来展示数据：

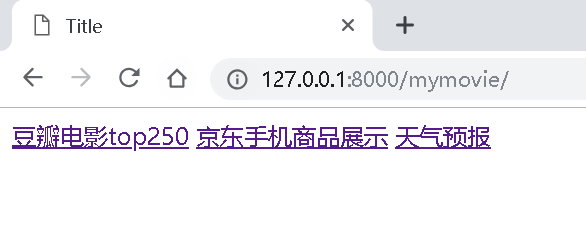
#### 3.1、实现步骤：

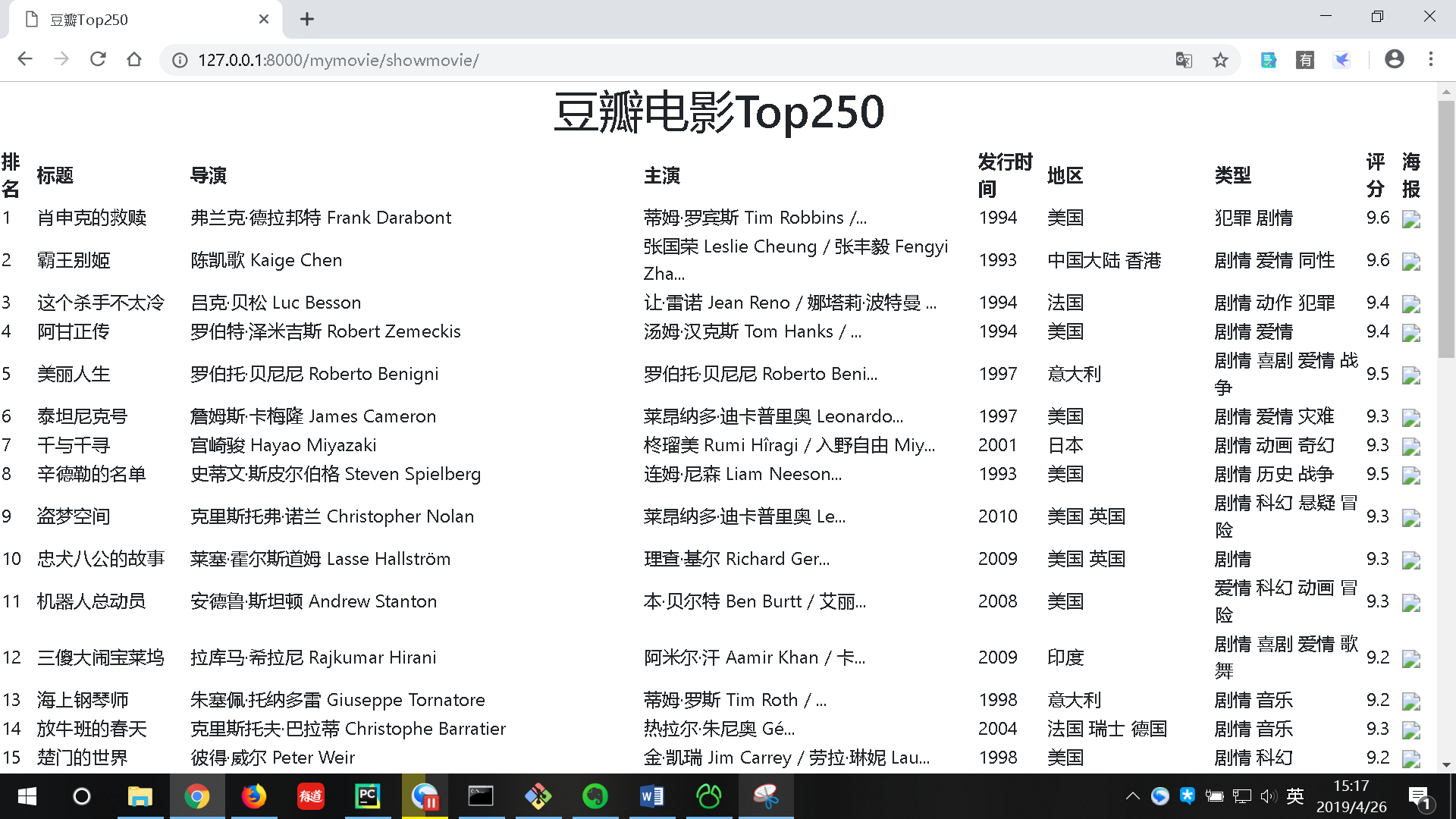
首先创建models，映射到mysql；创建templates，编写要展示的html页面；在views里编写功能并调用templates；编写urls。

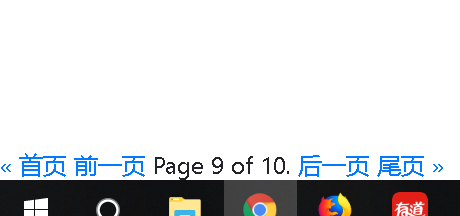
#### 3.2、项目目录结构：



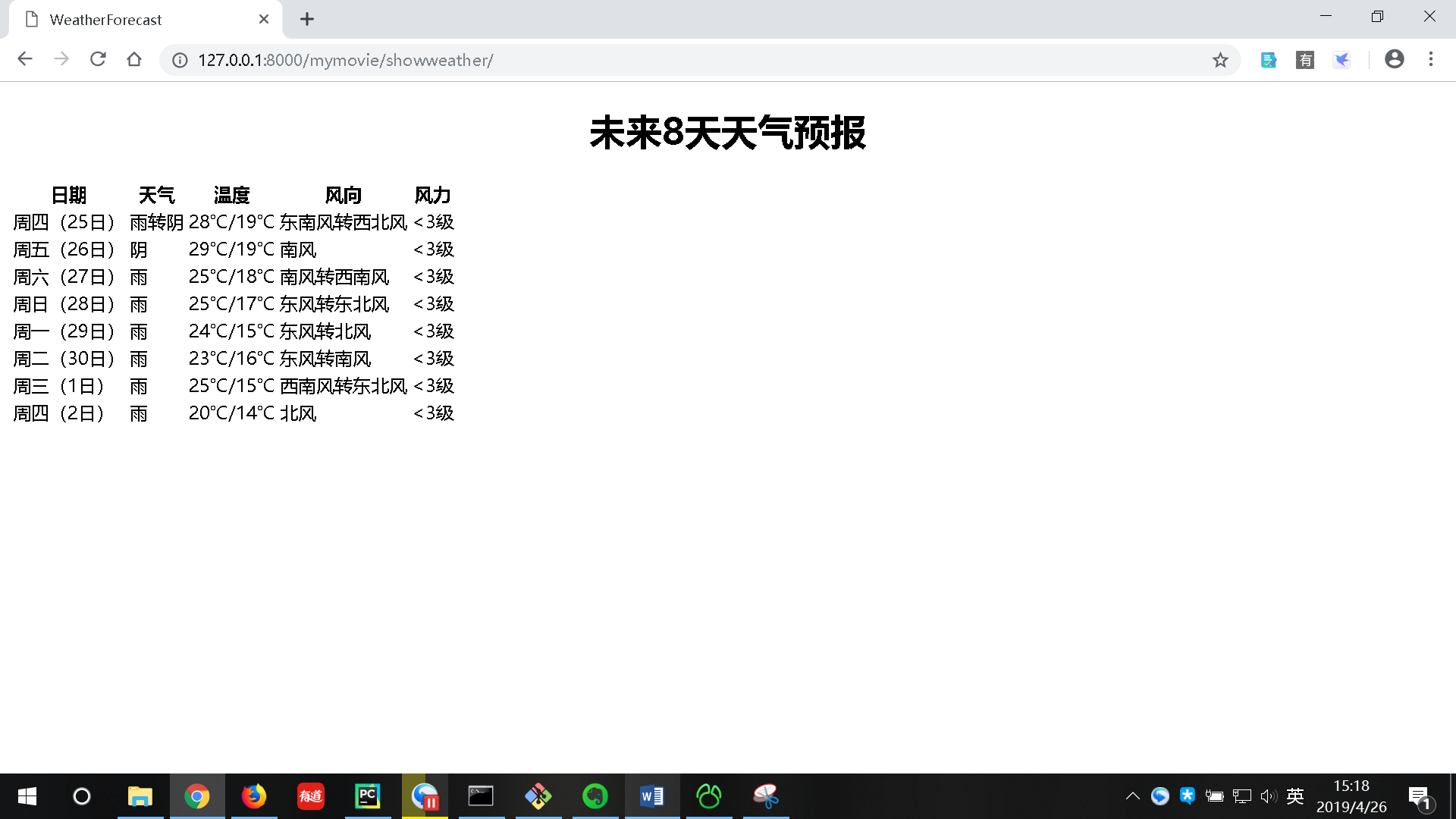
#### 3.3、效果截图：



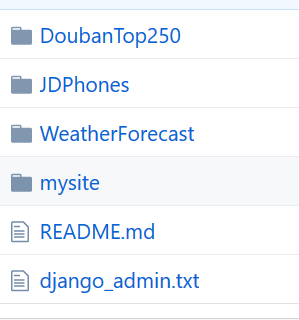








### 4、完整项目目录



DouanTop250为静态页面爬虫程序，JDPhones为动态页面爬虫程序，mysite为Django项目。