温州大學瓯江學院

WENZHOU UNIVERSITY OUJIANG COLLEGE

《爬虫期末作业》

题　　目：网络爬虫

分　　院：数学与信息工程学院

班　　级：16 计算机科学与技术二班

姓　　名：邱朝海

学　　号：16219111201

完成日期：2019 年 6 月 12 日

温州大学瓯江学院教务部

二〇一二年十一月制

目录

一、实验要求介绍

1、实验要求：

（1）模仿书本 P36 “Request 爬虫实践：TOP250 电影数据”爬取相关数据，并数入据存入数据库中。（数据库软件自选）

（2）建立 Django 项目：

基本要求：可运行 Django 项目；

进阶要求：展示爬虫信息的 Django 项目

高阶要求：基于 BeautifulSoup 或其他网页模拟的爬虫的 Django 项目

（3）使用 selenium 爬取动态网站—京东手机

（4）上传代码到 Github

以上均为期中作业要求，以下为期末作业要求：

（5）实现 12306 网站的验证码识别登录

2、开发工具与编程语言：

这里只是简单介绍：

使用软件：**Visual Studio code ， MySQL**

Python 语言中的相关模块和类库：

Request：Request 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，支持自动响应内容的编码，支持国际化的 URL 和 POST 数据自动编码。在 python 内置模块的基础上进行了高度的封装，从而使得 python 进行网络请求时，变得人性化，使用 Requests 可以轻而易举的完成浏览器可有的任何操作。

BeautifulSoup：Beautiful Soup 提供一些简单的、python 式的函数用来处理导航、搜索、修改分析树等功能。它是一个工具箱，通过解析文档为用户提供需要抓取的数据，因为简单，所以不需要多少代码就可以写出一个完整的应

用程序。简单来说，Beautiful Soup 是 python 的一个库，最主要的功能是从网页抓取数据。

Pymysql： 在 python3.x 中，可以使用 pymysql 来 MySQL 数据库的连接，并实现数据库的各种操作。

Selenium: selenium 是一套完整 web 应用程序测试系统，包含了测试的录制（selenium IDE），编写及运行（Selenium Remote Control）和测试的并行处理（Selenium Grid）。Selenium 的核心 Selenium Core 基于 JsUnit，完全由 JavaScript 编写，因此可以用于任何支持 JavaScript 的浏览器上。selenium 可以模拟真实浏览器，自动化测试工具，支持多种浏览器，爬虫中主要用来解决 JavaScript 渲染问题。

Re: 正则表达式（Regular Expression）是字符串处理的常用工具，通常被用来检索、替换那些符合某个模式（Pattern）的文本。很多程序设计语言都支持正则表达式，像 Perl、Java、C/C++。在 Python 中是通过标准库中的 re 模块 提供对正则的支持。

其余的还有 **time，os** 两个类库就不介绍。

二、设计过程简单介绍

1、静态爬虫：（只提供部分代码）

用 Request 读取网页源代码：

```
def getHTMLText(url):
    try:
        res=requests.get(url,timeout=60)
        res.raise_for_status()
        res.encoding=res.apparent_encoding
        return res.text
    except:
        return "产生异常！"
```

用 BeautifulSoup 分析源代码，爬取需要的数据：

```
def fillList(ulist,html):
    soup=BeautifulSoup(html,"html.parser")
    movies=soup.find('ol',class_="grid_view")
    if movies:
        for mv in movies.find_all('li'):
            href=mv.find('div',class_="hd").a.attrs['href']
            title=mv.find('div',class_="hd").a.span.get_text()
            quote=mv.find('p',class_="quote").span.get_text()
            ulist.append([href,title,quote])
        return ulist
    else:
        print("找不到参数！")
```

设计相关代码，传入数据库（Mysql）：

```
def inputMysql(ulist):    #存入数据库
    conn=pymysql.connect(host='localhost',user='root',passwd='admin',db='douban',charset="utf8",cursorclass=pymysql.cursors.DictCursor)
    cur=conn.cursor()
    try:
        for i in ulist:
            cur.execute("insert into movies (链接,标题,简介) values (%s,%s,%s)",(i[0],i[1],i[2]))
        cur.close()
        conn.commit()
        conn.close()
        print("数据已传入数据库！")
    except:
        print("数据传输失败！")
```

2、动态爬虫：（只提供部分代码）

因为用的是 Chrome 浏览器，下载对应版本的插件，这里因为找不到当初装 Python 的安装目录，所以只好自己在代码中指定路径：

```
url="https://www.jd.com"
phone=JDphone()
phone.Spider(url,"手机",tlist)
```

```
class JDphone():
    def Open_web(self,url,key):      #找到查询板块，模拟浏览器输入“手机”搜索手机页面
        chromedriver="D:/平时文件/爬虫/京东手机/chromedriver.exe"
        os.environ["webdriver.chrome.driver"]=chromedriver
        self.driver=webdriver.Chrome(chromedriver)
        print("正在打开网页...")
        self.driver.get(url)
```

首先模拟的是进入京东首页，然后在**搜索框中输入“手机”进行查询**，跳转到相应页面：

```
keyput=self.driver.find_element_by_id("key")
keyput.send_keys(key)
keyput.send_keys(Keys.ENTER)
```

再模拟**下拉网页**，保证所有手机的数据能够读取到：

```
print("网站正在响应...")
self.driver.implicitly_wait(3)
for i in range(10):
    self.driver.execute_script("var q=document.documentElement.scrollTop={0}".format(i*1000))
    time.sleep(1)
```

网页响应完，获取源代码后，即可定位数据（如何定位的就不多说了，唯一注意的是**放置图片链接的位置有 2 个**，其中一个找不到的话，肯定会在另外一个位置，所以定位了 2 次，再判断哪个不为空）：

```
try:
    phone=self.driver.find_elements_by_xpath("//div[@id='J_goodsList']//li[@class='gl-item']")
    for p in phone:
        try:
            price=p.find_element_by_xpath(".//div[@class='p-price']//i").text
            introduce=p.find_element_by_xpath(".//div[@class='p-name p-name-type-2']//em").text
            brand=introduce.split(" ")[0]
            brand=brand.replace(",","")#品牌
            quato=introduce.replace(",","")#简介
            evaluate_num=p.find_element_by_xpath(".//div[@class='p-commit']//strong//a").text#评价条数
            try:#图片链接
                img1=p.find_element_by_xpath(".//div[@class='p-img']//a//img").get_attribute("src")
            except:
                img1=""
            try:
                img2=p.find_element_by_xpath(".//div[@class='p-img']//a//img").get_attribute("data-lazy-img")
            except:
                img2=""
            if img1:
                img=img1
            else:
                img=img2
            tlist.append([img,brand,price,quato,evaluate_num])
```

```
                        img=img2
                    tlist.append([img,brand,price,quato,evaluate_num])
                except:
                    print("error")
            return tlist
        except:
            print("出错")
```

此次爬取，选择爬取**两页**数据，所以定义了**翻页**的方法：

```
def nextPage(self):#翻页
    try:
        previous=self.driver.find_element_by_xpath("//span[@class='p-num']//a[@class='pn-prev disabled']")
    except:
        previous=""
    #只有第一页的上一页按钮的class为“pn-prev disabled”，所以第一页后，previous均为空
    #第一页时，previous不为空，执行点击事件
    if previous:
        next=self.driver.find_element_by_xpath("//span[@class='p-num']//a[@class='pn-next']")
        next.click()
```

如图，只有在第一页能找到 class=’pn-prev disabled’ 的“上一页”属性，找到后，即可进行“下一页”的点击事件。

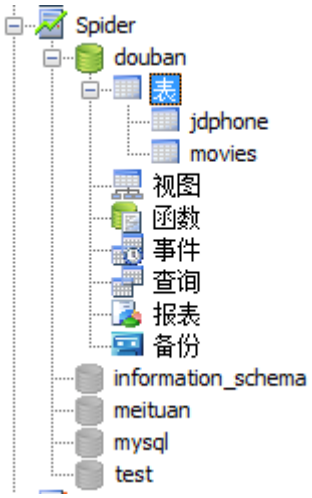
同样将爬取的数据存入数据库相应的表中：

```python
def inputMysql(self,tlist): #数据库先手动建好
    print("正在传入数据库...")
    conn=pymysql.connect(host='localhost',user='root',passwd='admin',db='douban',charset="utf8",cursorclass=pymysql.cursors.DictCursor)
    cur=conn.cursor()
    try:
        for i in tlist:
            cur.execute("insert into jdphone (图片,品牌,价格,简介,评价条数) values (%s,%s,%s,%s,%s)",(i[0],i[1],i[2],i[3],i[4]))
        cur.close()
        conn.commit()
        conn.close()
        print("数据已传入数据库！")
    except:
        print("数据传输失败！")
```

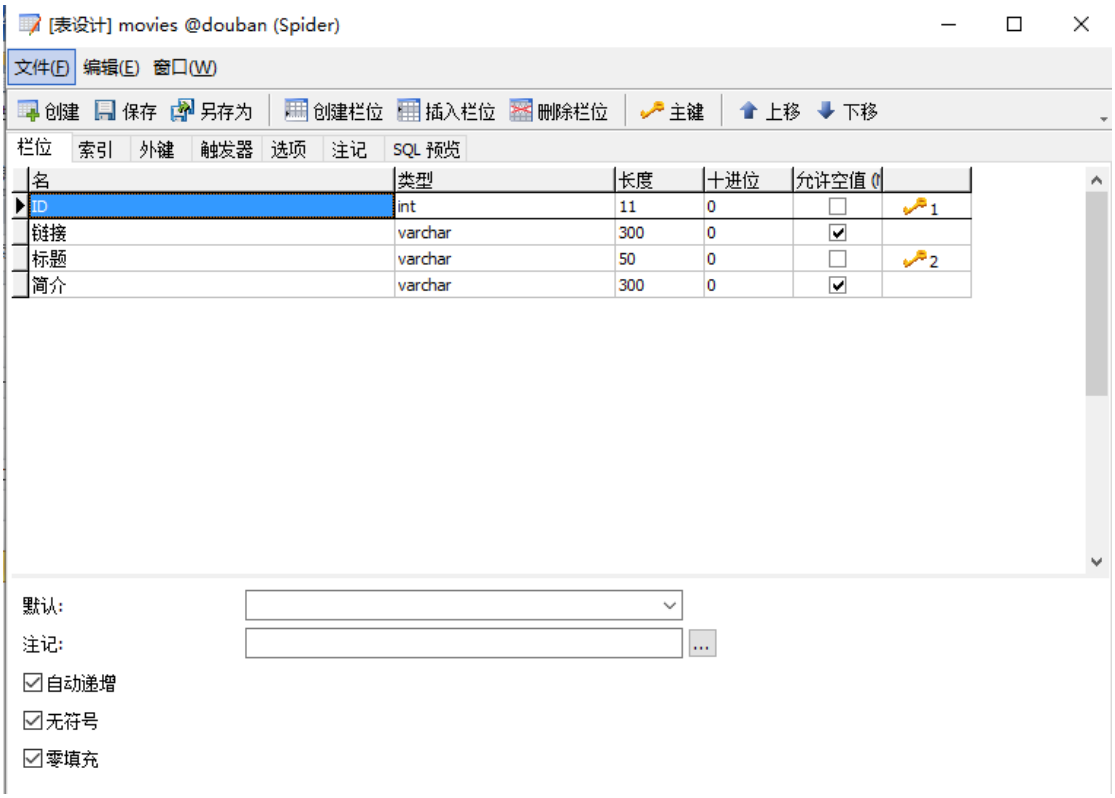
3、数据库：（数据库中的两表都是事先创建好的）

```
conn=pymysql.connect(host='localhost',user='root',passwd='admin',db='douban',charset="utf8",cursorclass=pymysql.cursors.DictCursor)
```

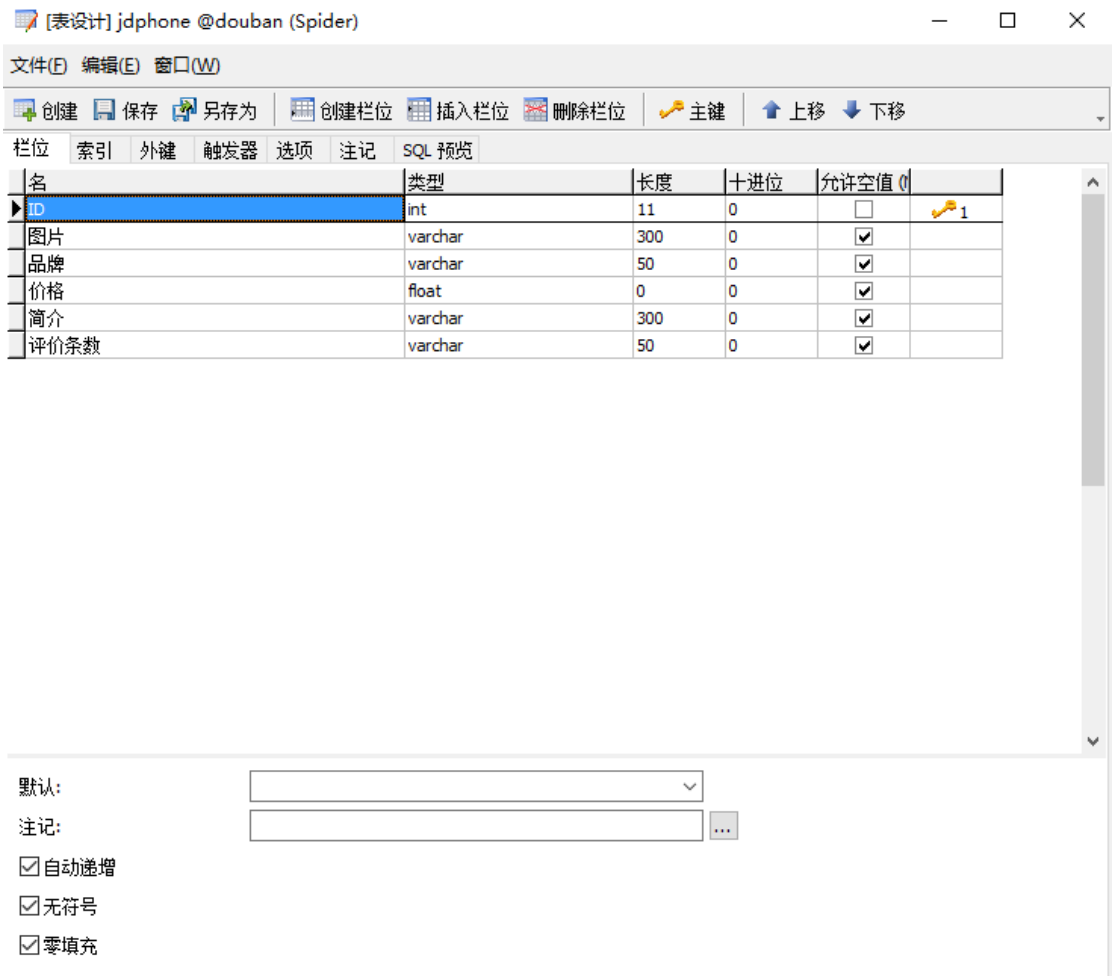
静态爬取数据在 movies，动态爬取数据在 jdphone：



Movies：


[表设计] movies @douban (Spider)
文件(F) 编辑(E) 窗口(W)
创建 保存 另存为 创建栏位 插入栏位 删除栏位 主键 上移 下移
栏位 索引 外键 触发器 选项 注记 SQL 预览
名 类型 长度 十进位 允许空值 (
ID int 11 0 1
链接 varchar 300 0
标题 varchar 50 0 2
简介 varchar 300 0
默认:
注记:
自动递增
无符号
零填充

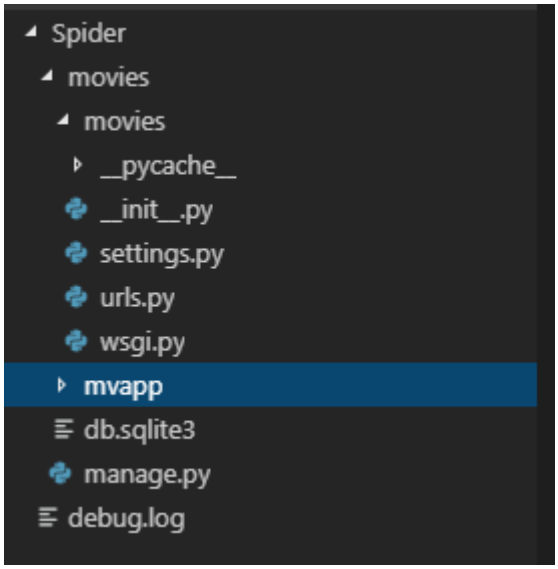
Jdphone：



4、Django:（我将静态和动态的代码均放在同一个框架内，所以文件命名可以无视）

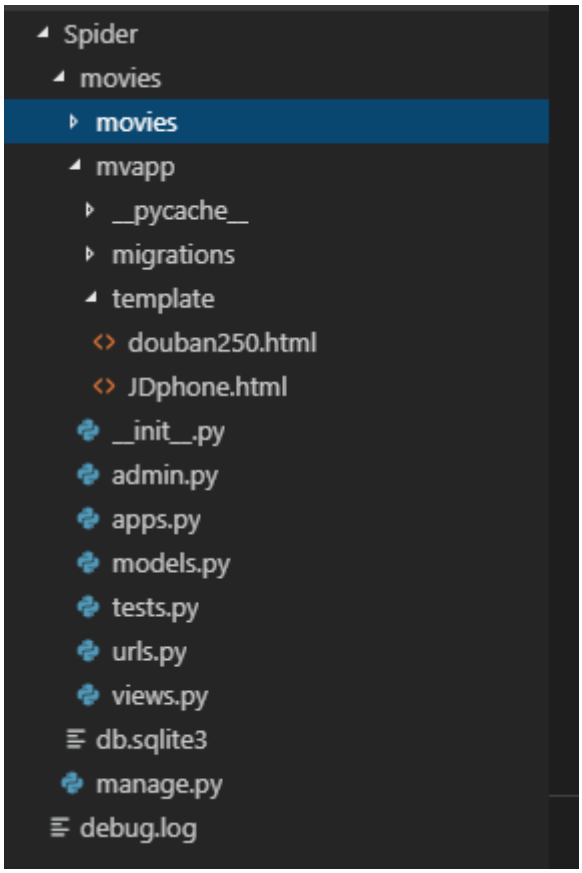
使用 django-admin.py 来创建 movies 项目：

```
django-admin.py startproject movies
```

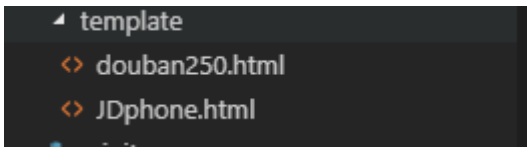


创建一个 app，来使用模型：

python manage.py startapp mvapp



在 mvapp 中新建 template 文件夹，在其中新建 2 个 html 文件分别展示静态和动态爬取的数据：（部分代码）




```
<body>
    <h1 align="center">爬取数据展示区（静态）</h1>
    <br>
    <table class="douban">
        <tr>
            <th>ID</th>
            <th>链接</th>
            <th>标题</th>
            <th>简介</th>
        </tr>
            {% for line in Show%}
                <tr>
                <td>{{forloop.counter}}</td>
                <td>{{line.链接}}</td>
                <td>{{line.标题}}</td>
                <td>{{line.简介}}</td>
                </tr>
            {% endfor %}
    </table>
</body>
```

```
<body>
    <h1 align="center">爬取数据展示区（动态）</h1>
    <br>
    <table class="JD">
        <tr>
            <th>ID</th>
            <th>图片</th>
            <th>品牌</th>
            <th>价格</th>
            <th>简介</th>
            <th>评价条数</th>
        </tr>
            {% for line in JD%}
                <tr>
                <td>{{forloop.counter}}</td>
                <td><img src="{{line.图片}}"></td>
                <td>{{line.品牌}}</td>
                <td>￥{{line.价格}}</td>
                <td>{{line.简介}}</td>
                <td>{{line.评价条数}}</td>
                </tr>
            {% endfor %}
    </table>
</body>
```

开始修改相应文件：

Settings.py:

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'mvapp'  #添加相应应用
]
```

```
# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = ['*']
```

Movies 中的 urls.py:

```python
from django.contrib import admin
from django.urls import path
from mvapp import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('Show',views.Show),
    path('JD',views.JD),
]
```

Mvapp 中的__init__.py:

views.py × __init__.py ×

```python
import pymysql
pymysql.install_as_MySQLdb()
```

Views.py 中放置了爬取静态、爬取动态、存入数据库、从数据库展示到页面上的所有代码，因代码过多，就不过多展示了：

views.py

```python
from django.shortcuts import render
import pymysql
import requests
from bs4 import BeautifulSoup

from selenium import webdriver
from selenium.webdriver.chrome.options import Options
from selenium.webdriver.common.keys import Keys
import time
import os

# Create your views here.

#静态爬取豆瓣top250
#爬取数据，并存入数据库
def getHTMLText(url):
    try:
        res=requests.get(url,timeout=60)
        res.raise_for_status()
        res.encoding=res.apparent_encoding
        return res.text
    except:
        return "产生异常！"

def fillList(ulist,html):
    soup=BeautifulSoup(html,"html.parser")
    movies=soup.find('ol',class_="grid_view")
    if movies:
        for mv in movies.find_all('li'):
            href=mv.find('div',class_="hd").a.attrs['href']
            title=mv.find('div',class_="hd").a.span.get_text()
            quote=mv.find('p',class_="quote").span.get_text()
            ulist.append([href,title,quote])
        return ulist
```

运行的总函数如下：

```python
def _main_():
    mv=[]
    movies=[]
    start_num=0
    for i in range(2):
        url='https://movie.douban.com/top250?start=%s&filter='%(start_num+25*i)
        html=getHTMLText(url)
        movies=fillList(mv,html)
    inputMysql(movies)

#读取数据库，展示在页面上
def get_data(sql):
    conn=pymysql.connect('localhost','root','admin','douban',port=3306,charset="utf8",cursorclass=pymysql.curs
    cur=conn.cursor()
    cur.execute(sql)
    results=cur.fetchall()
    cur.close()
    conn.close()
    return results

def Show(request):
    _main_()
    sql="select * from movies"
    movie_list=get_data(sql)
    return render(request,'douban250.html',{'Show':movie_list})
#静态爬取结束
```

```python
    def Spider(self,url,key,tlist):#运行总过程
        print("开始爬取")
        self.Open_web(url,key)
        phone1=self.fillList(tlist)
        tlist=[]
        self.nextPage()
        print("正在爬取第二页...")
        phone2=self.fillList(tlist)
        self.driver.close()
        self.inputMysql(phone1+phone2)

def JD(request):
    tlist=[]
    url="https://www.jd.com"
    phone=JDphone()
    phone.Spider(url,"手机",tlist)
    sql="select * from jdphone"
    phone_list=get_data(sql)
    return render(request,'JDphone.html',{'JD':phone_list})
#动态爬取结束
```

5、12306 验证码识别登录：（此处只展示 12306_post.py 的部分代码，用 selenium 模拟的代码还有问题，没成功）

代码中设计的函数分别有 getImg(self)、YZ_result(self)、login(self)。

```python
def load(self):
    self.getImg()
    time.sleep(3)
    self.YZ_result()
    time.sleep(10)
    self.login()
    time.sleep(10)
```

getImg():从12306的登录网页获取验证码的图片并下载下来保存为**Yz_img.jpg**,其中需解码。

```python
def getImg(self):
    #构造请求头
    headers = {'User-Agent':
    'Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86 Safari/537.36'
    }

    Xq_url = 'https://kyfw.12306.cn/passport/captcha/captcha-image64'    #生成验证码的url

    #构造请求表单
    Xq_parmas = {
        "login_site": "E",
        "module": "login",
        "rand": "sjrand",
        "1559721131663":"",
        "callback": "jQuery191090876281268888729_1559721117999",
        "_": "1559721118001",
    }

    #创建sess对象 保持会话一至
    sess=requests.session()

    #对图片页发送请求
    response=sess.get(url=Xq_url,params=Xq_parmas,headers=headers).text
    #获取图片数据
    image_bs64=re.findall('"image":"(.*?)",',response)[0]
    #解码数据
    image=base64.b64decode(image_bs64)
    #保存图片
    with open('Yz_image.jpg','wb') as f:
        f.write(image)

    self.sess=sess
```

YZ_result()：此处验证码识别我选择交给http://littlebigluo.qicp.net:47720/处理，如图，上传之前getImg()函数已下载的 Yz_img.jpg，获取相应图片的位置。再构造像素表单，用获取到的图片位置，来判断验证码识别所需要的图片点击坐标，即**answer**，下个函数会讲到。

不安全 | littlebigluo.qicp.net:47720

应用 哔哩哔哩 (゜- ゜)つ... AcFun弹幕视频网... 嘀哩嘀哩，这里是

请上传一张12306验证码图片

选择文件 未选择任何文件 上传

请点击下图中**所有的**风铃

经过仔细揣摩-图片貌似选: **2**

第一排图片从左到右编号依次为:1 2 3 4

第二排图片从左到右编号依次为:5 6 7 8

耗时:951毫秒!觉得俺bigluo眼力如何??

如果不确定结果是否正确，不妨登陆一下12306试试！！

有意见或建议？？欢迎交流:3490699170@qq.com或者点击这里

```python
def YZ_result(self):
    #验证部分交给了http://littlebigluo.qicp.net:47720/
    Xq_url1='http://littlebigluo.qicp.net:47720/'
    sess1=requests.session()
    response1=sess1.post(url=Xq_url1,data={"type":"1"},files={'pic_xxfile':open('Yz_image.jpg','rb')})
    result=[]
    #print(response.text)
    try:
        for i in re.findall("<B>(.*)</B>",response1.text)[0].split(" "):
            result.append(int(i))
    except:
        print("该验证网站在开小差...(系统繁忙)")

    #构建像素表单
    coord_data ={
        "1":"40,40",
        "2":"120,40",
        "3":"180,40",
        "4":"250,40",
        "5":"40,100",
        "6":"120,100",
        "7":"180,100",
        "8":"250,100",
    }
    answerlist = []
    print('选中图片为:',result)
    #循环输入的值取出字典相应的坐标
    for i in result:
        answerlist.append(coord_data[str(i)])
    print('坐标为: ' + ';'.join(answerlist))
    answer = ','.join(answerlist)
    self.answer=answer
```

login()：此处涉及到了验证码识别通过和登录申请两方面。

```python
Yz_url="https://kyfw.12306.cn/passport/captcha/captcha-check"    #验证验证码的url
Sy_url="https://kyfw.12306.cn/passport/web/login"    #登录的url
```

Get() 和 **post()** 中的相关参数可以在 https://kyfw.12306.cn/otn/resources/login.html 网页中右击检查等操作找到参数值，如图：（之前函数里的 answer 就是其中一个参数）

```python
#发送图片验证请求
response2=self.sess.get(url=Yz_url,params=log_parmas,headers=Yz_headers).text
#获得图片验证信息
print(re.findall('"result_message":"(.*?)"',response2))
```

```python
response3=self.sess.post(url=Sy_url,data=log_data,headers=Sy_headers)
#返回编码后的数据
response3.encoding='utf-8'
print(re.findall('"result_message":"(.*?)"',response3.text))
```

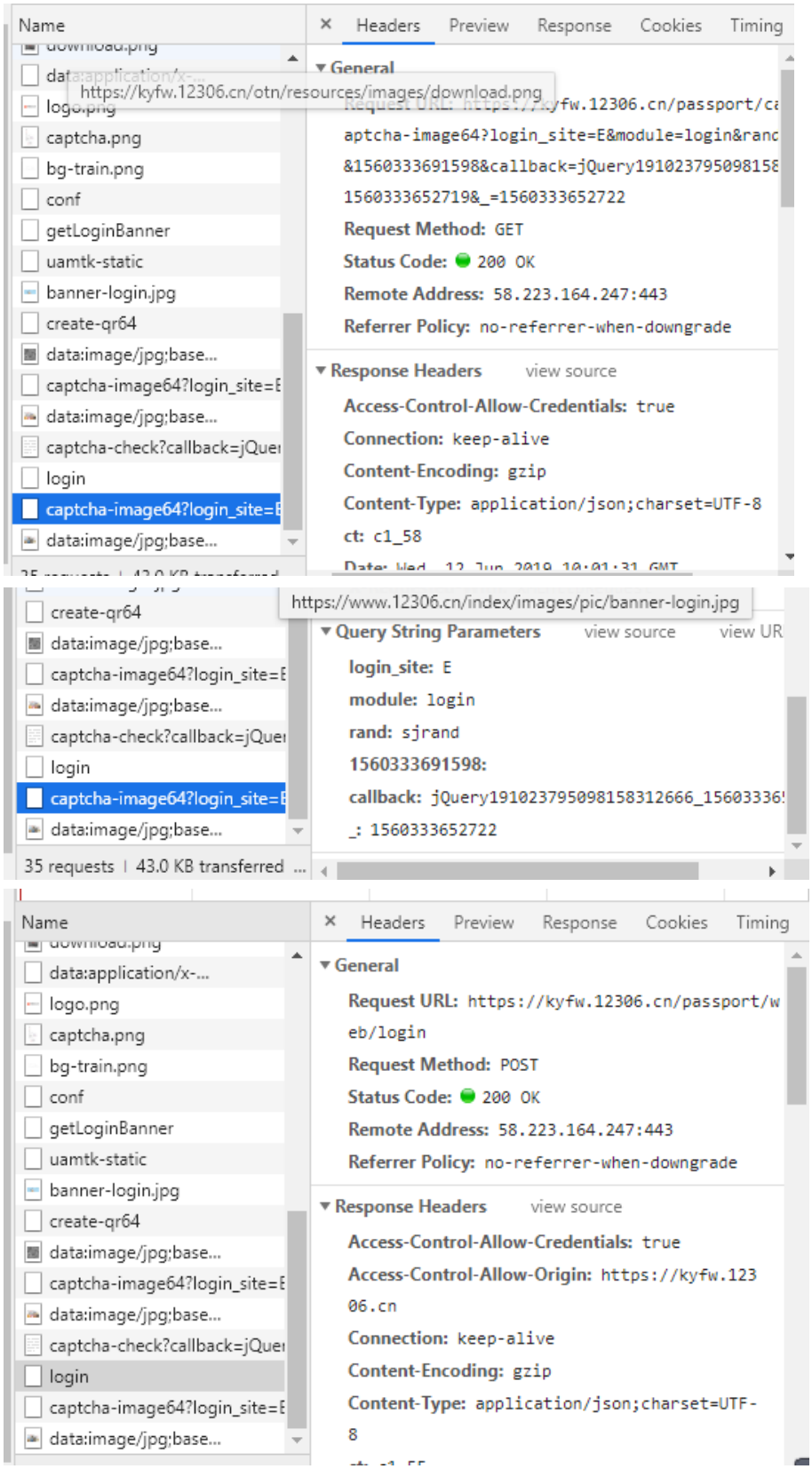
```
#构造data表单
log_data = {
    "username": username,
    "password": password,
    "appid": "otn",
    #"answer": self.answer,
}
#Yz_url
log_parmas = {
    "callback": "jQuery191050103005287633 58_1559733968819",
    "answer":self.answer,
    "rand": "sjrand",
    "login_site": "E",
    "_": "1559733968821",
}
```

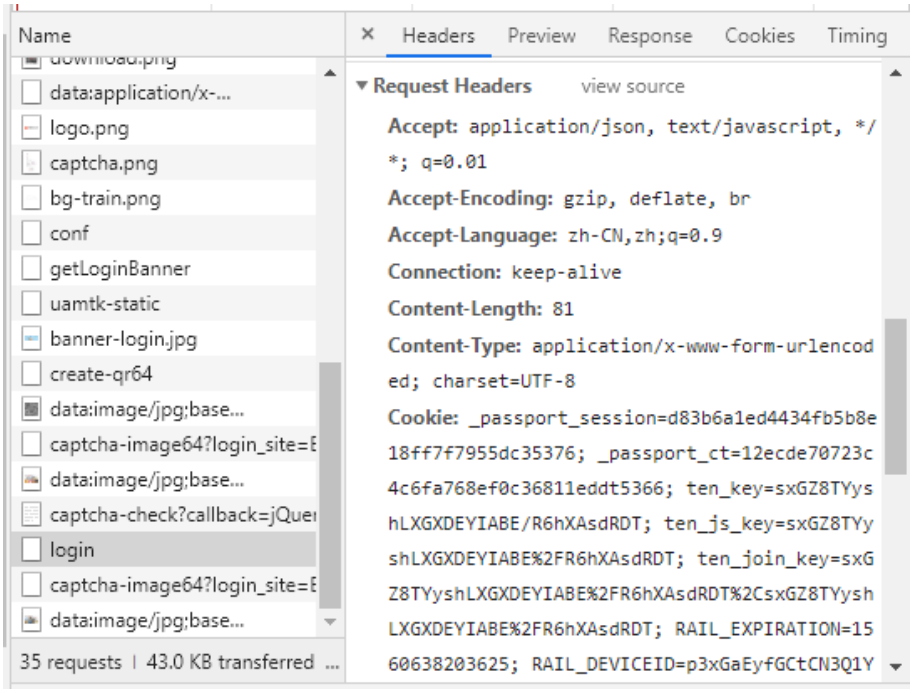
其中登录申请时，也需要**增加 cookies**（参数也可以找到）：

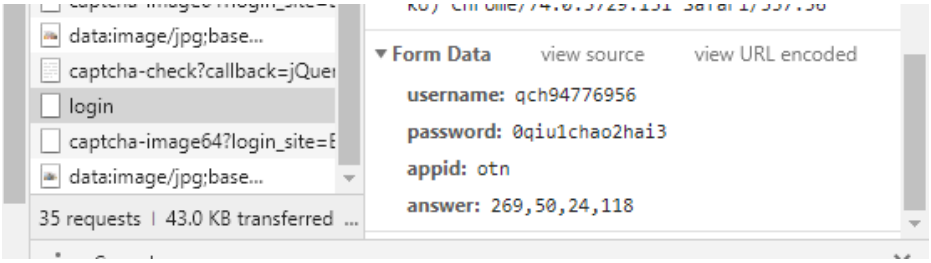
```
#增加cookies
self.sess.cookies.update({
    'RAIL_EXPIRATION':'1560638203625',
    'RAIL_DEVICEID':'p3xGaEyfGCtCN3Q1YYWImAt9vwLj1LB4oZltVN07EBp_UARZbREL1HzlVDd2B1_HL980K1Pl__t1bGWfXfNGt6zbaoppKN
})
```

相比起验证码识别，登录申请的 **headers 所涉及到的参数**就需要很多（原因我不知道，我只知道不加参数，登录别想通过）：

```
Yz_headers = {
    'User-Agent':'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729
}
Sy_headers = {
    'User-Agent':'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729
    'Accept-Encoding':'gzip, deflate, br',
    'Accept-Language':'zh-CN,zh;q=0.9',
    'Content-Type':'application/x-www-form-urlencoded; charset=UTF-8',
    'Origin':'https://kyfw.12306.cn',
    'Referer':'https://kyfw.12306.cn/otn/resources/login.html',
    'Accept':'application/json, text/javascript, */*; q=0.01',
}
```


Name
Headers
Preview
Response
Cookies
Timing
https://kyfw.12306.cn/otn/resources/images/download.png
captcha.png
bg-train.png
conf
getLoginBanner
uamtk-static
banner-login.jpg
create-qr64
login
General
Request Method: GET
Status Code: 200 OK
Remote Address: 58.223.164.247:443
Referrer Policy: no-referrer-when-downgrade
Response Headers
view source
Access-Control-Allow-Credentials: true
Connection: keep-alive
Content-Encoding: gzip
Content-Type: application/json;charset=UTF-8
ct: c1_58
https://www.12306.cn/index/images/pic/banner-login.jpg
Query String Parameters
view source
login_site: E
module: login
rand: sjrand
1560333691598:
_: 1560333652722
35 requests | 43.0 KB transferred
Request URL: https://kyfw.12306.cn/passport/web/login
Request Method: POST
Status Code: 200 OK
Remote Address: 58.223.164.247:443
Referrer Policy: no-referrer-when-downgrade
Access-Control-Allow-Origin: https://kyfw.12306.cn






```python
def login(self):
    Yz_url="https://kyfw.12306.cn/passport/captcha/captcha-check"    #验证验证码的url
    Sy_url="https://kyfw.12306.cn/passport/web/login"    #登录的url

    Yz_headers = {
        'User-Agent':'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729
    }
    Sy_headers = {
        'User-Agent':'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729
        'Accept-Encoding':'gzip, deflate, br',
        'Accept-Language':'zh-CN,zh;q=0.9',
        'Content-Type':'application/x-www-form-urlencoded; charset=UTF-8',
        'Origin':'https://kyfw.12306.cn',
        'Referer':'https://kyfw.12306.cn/otn/resources/login.html',
        'Accept':'application/json, text/javascript, */*; q=0.01',
    }

    username = input('请输入用户名: ')
    password = input('请输入密码: ')

    #构造data表单
    log_data = {
        "username": username,
        "password": password,
        "appid": "otn",
        #"answer": self.answer,
    }
```

```
#构造data表单
log_data = {
    "username": username,
    "password": password,
    "appid": "otn",
    #"answer": self.answer,
}
#Yz_url
log_parmas = {
    "callback": "jQuery191050103005287633358_1559733968819",
    "answer":self.answer,
    "rand": "sjrand",
    "login_site": "E",
    "_": "1559733968821",
}
#发送图片验证请求
response2=self.sess.get(url=Yz_url,params=log_parmas,headers=Yz_headers).text
#获得图片验证信息
print(re.findall('"result_message":"(.*?)"',response2))

#增加cookies
self.sess.cookies.update({
    'RAIL_EXPIRATION':'1560638203625',
    'RAIL_DEVICEID':'p3xGaEyfGCtCN3Q1YYWImAt9vwLj1LB4oZltVN07EBp_UARZbREL1HzlVDd2B1_HL980K1Pl__t1bGWfXfNGt6zbaoppKN
})

response3=self.sess.post(url=Sy_url,data=log_data,headers=Sy_headers)
#返回编码后的数据
response3.encoding='utf-8'
print(re.findall('"result_message":"(.*?)"',response3.text))
```

三、运行截图

1、运行前的注意事项：

(1)因代码中并**未设计创建数据库的代码**，所以数据库必须事先自己建好，上传文件中应该有相应的 sql 运行文件，**请先导入**；

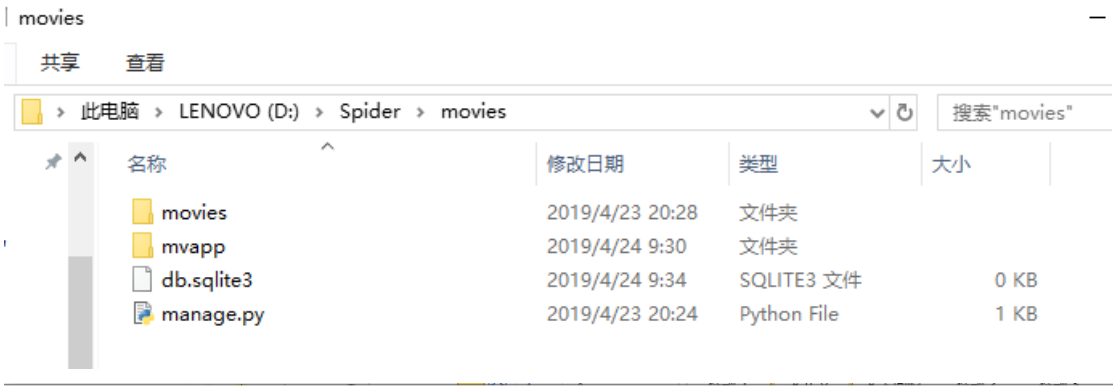
(2)2 个插入数据库的代码中，会有

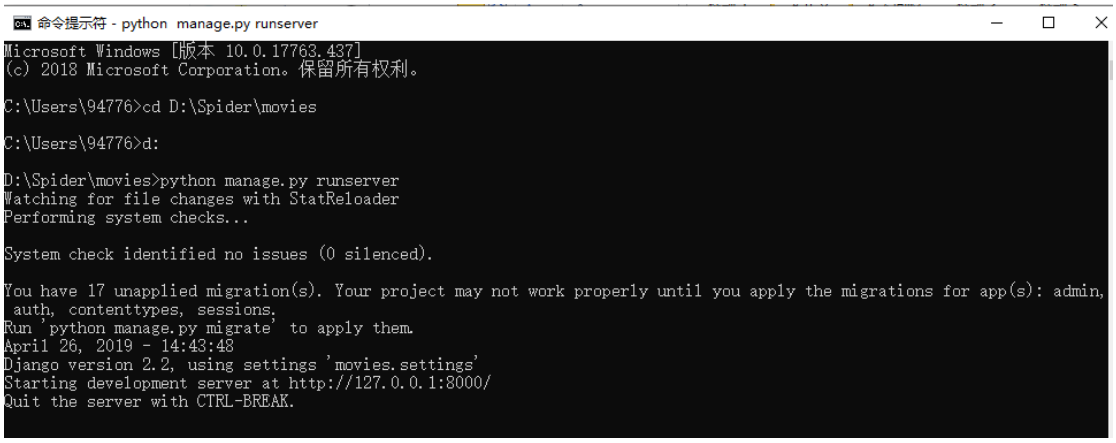
“conn=pymysql.connect(host='localhost',user='root',passwd='admin',db='douban',charset="utf8",cursorclass=pymysql.cursors.DictCursor)”，请根据自己创建的数据库，**修改相应的数据**；

(3)因为 chrome 的模拟浏览器插件在代码中是自定义的路径，所以请先**修改代码中的“chromedriver”**，代码位置如图：

```
class JDphone():
    def Open_web(self,url,key):      #找到查询板块，模拟浏览器输入“手机”搜索手机页面
        chromedriver="D:/平时文件/爬虫/京东手机/chromedriver.exe"
        os.environ["webdriver.chrome.driver"]=chromedriver
        self.driver=webdriver.Chrome(chromedriver)
        print("正在打开网页...")
        self.driver.get(url)
        keyput=self.driver.find_element_by_id("key")
        keyput.send_keys(key)
        keyput.send_keys(Keys.ENTER)
```

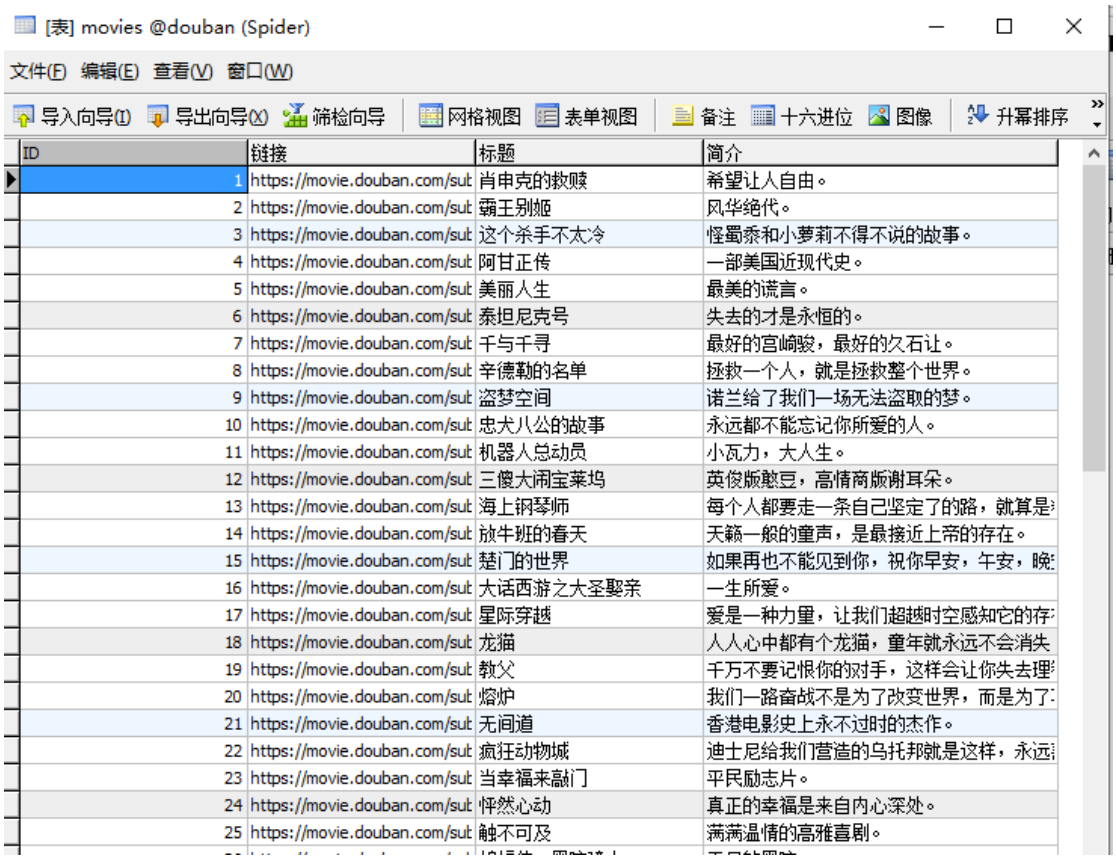
2、截图：





输入网址：http://127.0.0.1:8000/Show

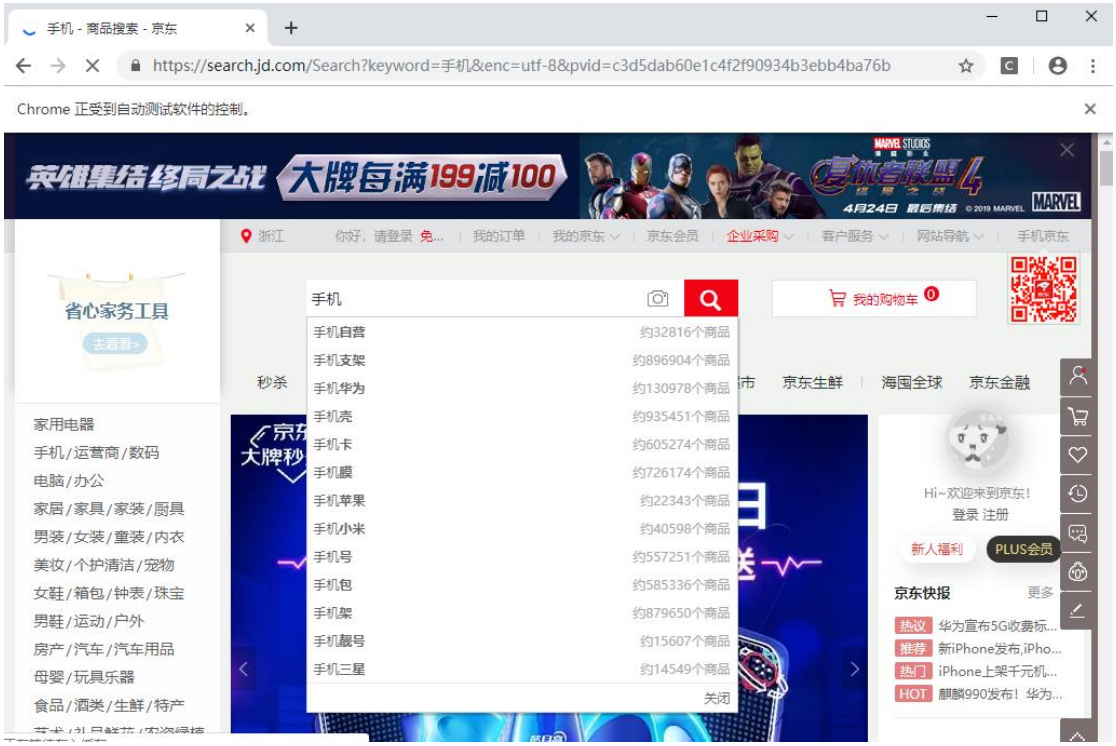
ID	链接	标题	简介
1	https://movie.douban.com/subject/1292052/	肖申克的救赎	希望让人自由。
2	https://movie.douban.com/subject/1291546/	霸王别姬	风华绝代。
3	https://movie.douban.com/subject/1295644/	这个杀手不太冷	怪蜀黍和小萝莉不得不说的故事。
4	https://movie.douban.com/subject/1292720/	阿甘正传	一部美国近现代史。
5	https://movie.douban.com/subject/1292063/	美丽人生	最美的谎言。
6	https://movie.douban.com/subject/1292722/	泰坦尼克号	失去的才是永恒的。
7	https://movie.douban.com/subject/1291561/	千与千寻	最好的宫崎骏，最好的久石让。
8	https://movie.douban.com/subject/1295124/	辛德勒的名单	拯救一个人，就是拯救整个世界。
9	https://movie.douban.com/subject/3541415/	盗梦空间	诺兰给了我们一场无法盗取的梦。
10	https://movie.douban.com/subject/3011091/	忠犬八公的故事	永远都不能忘记你所爱的人。
11	https://movie.douban.com/subject/2131459/	机器人总动员	小瓦力，大人生。
12	https://movie.douban.com/subject/3793023/	三傻大闹宝莱坞	英俊版憨豆，高情商版谢耳朵。

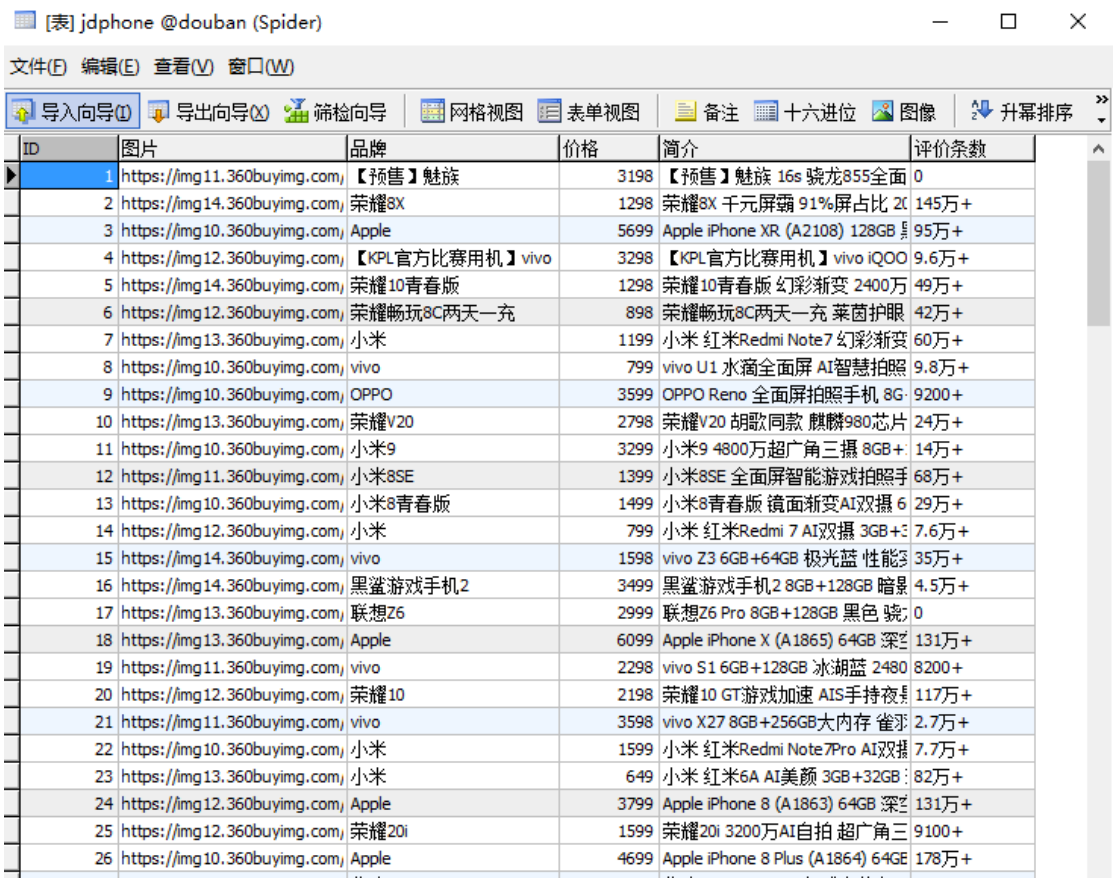


ID	链接	标题	简介
1	https://movie.douban.com/sub	肖申克的救赎	希望让人自由。
2	https://movie.douban.com/sub	霸王别姬	风华绝代。
3	https://movie.douban.com/sub	这个杀手不太冷	怪蜀黍和小萝莉不得不说的故事。
4	https://movie.douban.com/sub	阿甘正传	一部美国近现代史。
5	https://movie.douban.com/sub	美丽人生	最美的谎言。
6	https://movie.douban.com/sub	泰坦尼克号	失去的才是永恒的。
7	https://movie.douban.com/sub	千与千寻	最好的宫崎骏，最好的久石让。
8	https://movie.douban.com/sub	辛德勒的名单	拯救一个人，就是拯救整个世界。
9	https://movie.douban.com/sub	盗梦空间	诺兰给了我们一场无法盗取的梦。
10	https://movie.douban.com/sub	忠犬八公的故事	永远都不能忘记你所爱的人。
11	https://movie.douban.com/sub	机器人总动员	小瓦力，大人生。
12	https://movie.douban.com/sub	三傻大闹宝莱坞	英俊版憨豆，高情商版谢耳朵。
13	https://movie.douban.com/sub	海上钢琴师	每个人都要走一条自己坚定了的路，就算是
14	https://movie.douban.com/sub	放牛班的春天	天籁一般的童声，是最接近上帝的存在。
15	https://movie.douban.com/sub	楚门的世界	如果再也不能见到你，祝你早安，午安，晚
16	https://movie.douban.com/sub	大话西游之大圣娶亲	一生所爱。
17	https://movie.douban.com/sub	星际穿越	爱是一种力量，让我们超越时空感知它的存
18	https://movie.douban.com/sub	龙猫	人人心中都有个龙猫，童年就永远不会消失
19	https://movie.douban.com/sub	教父	千万不要记恨你的对手，这样会让你失去理
20	https://movie.douban.com/sub	熔炉	我们一路奋战不是为了改变世界，而是为了
21	https://movie.douban.com/sub	无间道	香港电影史上永不过时的杰作。
22	https://movie.douban.com/sub	疯狂动物城	迪士尼给我们营造的乌托邦就是这样，永远
23	https://movie.douban.com/sub	当幸福来敲门	平民励志片。
24	https://movie.douban.com/sub	怦然心动	真正的幸福是来自内心深处。
25	https://movie.douban.com/sub	触不可及	满满温情的高雅喜剧。

输入网址：http://127.0.0.1:8000/JD

会开始模拟浏览器操作，此次操作可能需要点时间





12306 验证码识别登录：

用户名和密码是手动输的…

四、未解决的问题

设计中，出现的一些至今还未解决的明显问题：

1、完整的执行一次代码后，再执行一次代码，同样的数据会再次存入数据库，数据会不断重复、叠加。比如第一次爬取了第一页数据，第二次爬取了第一、二页的数据，都存入数据库后，会导致第一页的数据出现了 2 次。（目前解决办法：每次执行代码前，将数据库的数据先全部清空）

2、数据展示页面的展示记录过多，未实现分页功能。

3、设计框架和美化能力有限。

4、12306 验证码识别登录用 Selenium 模拟登录时，用模拟器中的点击事件选择验证码图片时出现错误。