主要代码及显示页面

显示页面:

Base.html:

```
{% load static %}


<html lang="en">

<head>

<title>{% block title %}{% endblock %} </title>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.5/css/bootstrap.min.css">

    <style>         ul li

{         display:inline;

list-style-type:none;

        }

    </style>

</head>


<body>

 <!-- Page content of course! -->

<main>

    <div class="container">

    {% block content %}

  {% if error_message %}<p><strong>{{ error_message}}</strong></p>{% endif %}

  {% endblock %}

        </div>

</main>



<footer class="footer">

  {% block footer %}{% endblock %}

</footer>

<!--End of Footer-->


<!-- Bootstrap core JavaScript

================================================== -->


<script src="https://code.jquery.com/jquery-3.3.1.min.js" integrity="sha256-FgpCb/KJQlLNfOu91ta32o/NMZxltwRo8QtmkMRdAu8="

crossorigin="anonymous"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

</body>


</html>
```

Index.html:

```
{% extends "mylink/base.html" %}


{% block content %}


<h3>爬取信息</h3>
<form method="POST" class="form-horizontal" role='form' action="{% url 'mylink:house_spider' %}">
  {% csrf_token %}
  {{ form.as_p }}
   <div class="form-group">
       <div class="col-md-12">
  <button type="submit" class="btn btn-primary form-control">开始爬取</button>
       </div>
   </div>
</form>


{% if page_obj %}
<h3>爬取结果</h3>
<table class="table table-striped">
    <thead>
        <tr>
            <th>名字</th>
            <th>小区</th>
            <th>房型</th>
            <th>面积</th>
            <th>年份</th>
            <th>区域</th>
            <th>总价(万)</th>
            <th>单价(元/平方米)</th>
        </tr>
    </thead>
    <tbody>
     {% for house in page_obj %}
        <tr>
            <td>
            {{ house.title }}
            </td>
            <td>
            {{ house.house }}
            </td>
            <td>
            {{ house.bedroom }}
            </td>
```

```
                <td>
                {{ house.area }}
                </td>
                <td>
                {{ house.year }}
                </td>
                <td>
                {{ house.location }}
                </td>
                <td>
                {{ house.total_price }}
                </td>
                <td>
                    {{ house.unit_price }}
                </td>
        {% endfor %}
            </tr>
        </tbody>
</table>

{% else %}
    <p>尚无二手房信息。</p>
{% endif %}

{% if is_paginated %}
     <ul class="pagination">
    {% if page_obj.has_previous %}
      <li class="page-item"><a class="page-link" href="?page={{ page_obj.previous_page_number }}">Previous</a></li>
{% else %}
      <li class="page-item disabled"><span class="page-link">Previous</span></li>
    {% endif %}

    {% for i in paginator.page_range %}
        {% if page_obj.number == i %}
      <li class="page-item active"><span class="page-link"> {{ i }} <span class="sr-only">(current)</span></span></li>
{% else %}
        <li class="page-item"><a class="page-link" href="?page={{ i }}">{{ i }}</a></li>
       {% endif %}
    {% endfor %}

         {% if page_obj.has_next %}
      <li class="page-item"><a class="page-link" href="?page={{ page_obj.next_page_number }}">Next</a></li>
    {% else %}
      <li class="page-item disabled"><span class="page-link">Next</span></li>
    {% endif %}
```

```
    </ul>

{% endif %}

{% endblock %}
```

Models.py:

```
from django.db import models #

Create your models here.


class HouseInfo(models.Model):
    title = models.CharField(max_length=256, verbose_name='名字')
house = models.CharField(max_length=20, verbose_name='小区')
bedroom = models.CharField(max_length=20, verbose_name='房型')
area = models.CharField(max_length=20, verbose_name='面积')
direction = models.CharField(max_length=20, verbose_name='朝向')
floor = models.CharField(max_length=60, verbose_name='朝向')     year
= models.CharField(max_length=10, verbose_name='年份')     location =
models.CharField(max_length=10, verbose_name='区域')     total_price
= models.IntegerField(verbose_name='总结(万元)')     unit_price =
models.IntegerField(verbose_name='单价(元/平方米)')
     add_date = models.DateTimeField(auto_now_add=True, verbose_name="创建日期
")     mod_date = models.DateTimeField(auto_now=True, verbose_name="修改日期")
     def
__str__(self):
        return "{}-{}-{}".format(self.house,self.bedroom, self.total_price)
class Meta:
        verbose_name = "二手房"
```

forms.py:

```
from django import forms


DISTRICT_CHOICES = (('baoshan', '宝山'), ('putuo', '普陀'), ('songjiang', '松江'))
```

```
PRICE_CHOICES = (('p2', '200-300 万'), ('p3', '300-400 万'), ('p4', '400-500 万'), ('p5', '500-800 万'))
BEDROOM_CHOICES = (('12','二室'), ('13', '三室'), ('14', '四室'))
 class
HouseChoiceForm(forms.Form):
    district = forms.CharField(label="区域", widget=forms.RadioSelect(choices=DISTRICT_CHOICES))
price = forms.CharField(label="价格", widget=forms.RadioSelect(choices=PRICE_CHOICES))     bedroom
= forms.CharField(label="房型", widget=forms.RadioSelect(choices=BEDROOM_CHOICES))
```

views.py:

```
from django.shortcuts import render
from .models import HouseInfo from .forms
import HouseChoiceForm from
django.core.paginator import Paginator from
django.http import HttpResponseRedirect
  from fake_useragent import
UserAgent import requests from bs4
import BeautifulSoup import re

# Create your views here.

def house_index(request):     form = HouseChoiceForm()
house_list = HouseInfo.objects.all().order_by('-add_date')
if house_list:
        paginator = Paginator(house_list, 20)
page = request.GET.get('page')          page_obj
= paginator.get_page(page)
         return render(request,
'mylink/index.html',
                      {'page_obj': page_obj, 'paginator': paginator,
'is_paginated': True, 'form': form,})     else:
        return render(request,'mylink/index.html', {'form': form,})
 def house_spider(request):
if request.method == 'POST':
        form = HouseChoiceForm(request.POST)
        if form.is_valid():
            district = form.cleaned_data.get('district')              price =
form.cleaned_data.get('price')              bedroom = form.cleaned_data.get('bedroom')
url = 'https://sh.lianjia.com/ershoufang/{}/{}{}'.format(district, price, bedroom)
             home_spider = HomeLinkSpider(url)
home_spider.get_max_page()
home_spider.parse_page()
home_spider.save_data_to_model()              return
HttpResponseRedirect('/mylink/')     else:
        return HttpResponseRedirect('/mylink/')
```

```
  class
HomeLinkSpider(object):
def __init__(self, url):
        self.ua = UserAgent(verify_ssl=False)
self.headers = {"User-Agent": self.ua.random}
self.data = list()          self.url = url
     def
get_max_page(self):
        response = requests.get(self.url, headers=self.headers)
if response.status_code == 200:
            soup = BeautifulSoup(response.text, 'html.parser')
a = soup.select('div[class="page-box house-lst-page-box"]')
max_page = eval(a[0].attrs["page-data"])["totalPage"]
return max_page         else:
            print("请求失败 status:{}".format(response.status_code))
return None
     def
parse_page(self):
        max_page = self.get_max_page()
for i in range(1, max_page + 1):
            url = "{}pg{}/".format(self.url, i)             response =
requests.get(url, headers=self.headers)             soup =
BeautifulSoup(response.text, 'html.parser')             ul =
soup.find_all("ul", class_="sellListContent")             li_list =
ul[0].select("li")             for li in li_list:                 detail =
dict()                 detail['title'] =
li.select('div[class="title"]')[0].get_text()             house_info =
li.select('div[class="houseInfo"]')[0].get_text()             house_info_list =
house_info.split(" | ")
             detail['house'] = house_info_list[0]
detail['bedroom'] = house_info_list[1]
detail['area'] = house_info_list[2]
detail['direction'] = house_info_list[3]
             position_info = li.select('div[class="positionInfo"]')[0].get_text().split(' -
')
             floor_pattern = re.compile(r'.+\)')          match1
= re.search(floor_pattern, position_info[0])                 if
match1:
                detail['floor'] = match1.group()
else:
                detail['floor'] = "未知"
             year_pattern = re.compile(r'\d{4}')          match2
= re.search(year_pattern, position_info[0])          if match2:
                detail['year'] = match2.group()
else:
```

```
                detail['year'] = "未知"
detail['location'] = position_info[1]
                price_pattern = re.compile(r'\d+')                total_price =
li.select('div[class="totalPrice"]')[0].get_text()
detail['total_price'] = re.search(price_pattern, total_price).group()
                unit_price = li.select('div[class="unitPrice"]')[0].get_text()
detail['unit_price'] = re.search(price_pattern, unit_price).group()
self.data.append(detail)
    def save_data_to_model(self):          for
item in self.data:              new_item =
HouseInfo()             new_item.title =
item['title']             new_item.house =
item['house']             new_item.bedroom =
item['bedroom']             new_item.area =
item['area']             new_item.direction =
item['direction']             new_item.floor =
item['floor']             new_item.year =
item['year']             new_item.location =
item['location']             new_item.total_price
= item['total_price']
new_item.unit_price = item['unit_price']
new_item.save()
```

爬取结果

名字	小区	房型	面积	年份	区域	总价(万)	单价(元/平方米)
新城云间锦苑 4房电梯叠加 一家老小住方便	新城云间锦院	4室2厅	129平米	未知	松江老城	630	48838
上坤公园天地 5房间2卫 260万	上坤公园天地	3室2厅	70平米	未知	顾村	260	37143
妙境公寓 2室2厅 315万新上	妙境公寓	2室2厅	83.42平米	未知	川沙	315	37761
产证已到手 新装目前已出租 看房方便 诚意出售	碧桂园浦东星作	2室2厅	92.11平米	未知	泥城镇	340	36913
长达佳苑宜居尚城 2室2厅 310万	长达佳苑宜居尚城	2室2厅	84平米	未知	航头	310	36905

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
```

```
    'mylink',
]


DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'myfile',
        'USER': 'root',
        'PASSWORD': '123',
        'HOST': 'localhost',
        'PORT': '3306',
    }
}


STATIC_URL = '/static/'

STATICFILES_DIRS = [os.path.join(BASE_DIR, "static"), ]
```

		厅		知			
妙境公寓 2室2厅 315万新上	妙境公寓	2室2厅	83.42平米	未知	川沙	315	37761
产证已到手 新装目前已出租 看房方便 诚意出售	碧桂园浦东星作	2室2厅	92.11平米	未知	泥城镇	340	36913
长达佳苑宜居尚城 2室2厅 310万	长达佳苑宜居尚城	2室2厅	84平米	未知	航头	310	36905
1楼带天井，满五年唯一，看房方便，诚意出售	白杨小区	2室1厅	71.3平米	未知	北蔡	368	51613
妙境公寓 2室2厅 315万新上	妙境公寓	2室2厅	83.42平米	未知	川沙	315	37761

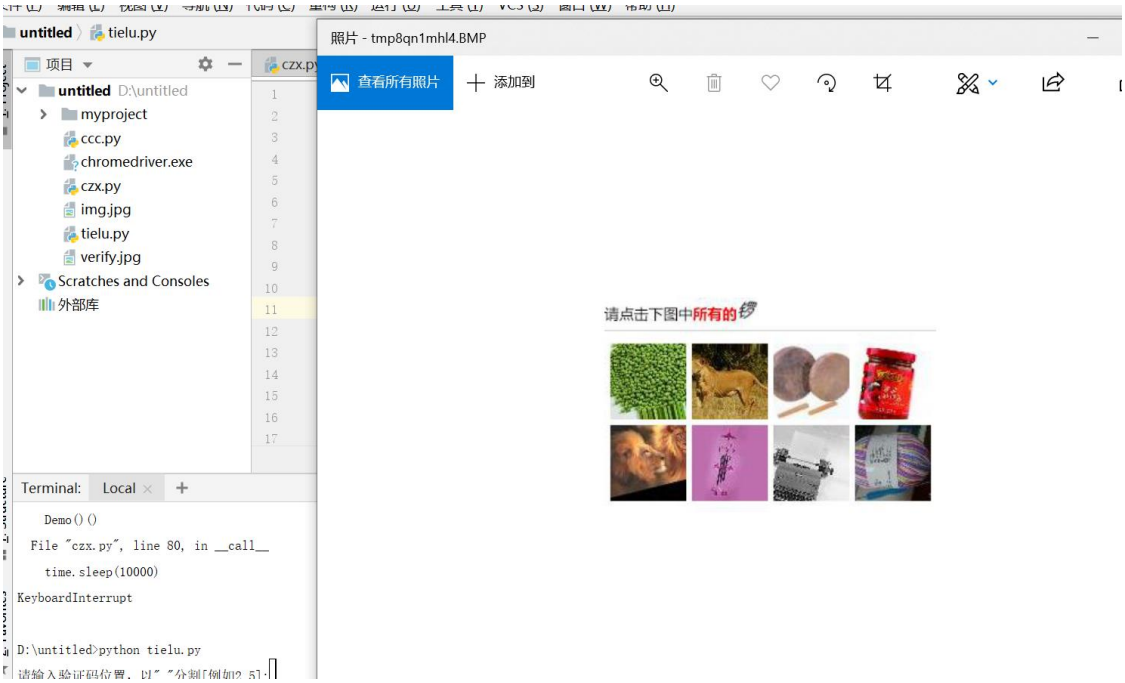
```python
# coding:u8

from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.action_chains import ActionChains
import requests
import base64
import re
import time

class Demo():
    def __init__(self):
        self.coordinate = [[-105, -20], [-35, -20], [40, -20], [110, -20], [-105, 50], [-35, 50], [40, 50], [110, 50]]

    def login(self):
        login_url = "https://kyfw.12306.cn/otn/resources/login.html"
        webdriverUrl = r'D:\untitled\chromedriver.exe'
        driver = webdriver.Chrome(webdriverUrl)
        driver.set_window_size(1200, 900)
        driver.get(login_url)
        account = driver.find_element_by_class_name("login-hd-account")
        account.click()
        userName = driver.find_element_by_id("J-userName")
        userName.send_keys("531218020@qq.com")
        password = driver.find_element_by_id("J-password")
        password.send_keys("**********")
        self.driver = driver
```

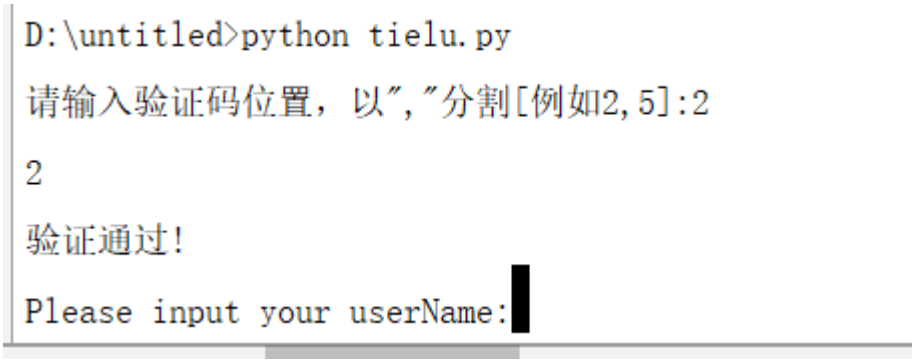
```
    def getVerifyImage(self):
        try:

            img_element = WebDriverWait(self.driver, 100).until(
                EC.presence_of_element_located((By.ID, "J-loginImg"))
            )

        except Exception as e:
            print(u"网络开小差,请稍后尝试")
        base64_str = img_element.get_attribute("src").split(",")[-1]
        imgdata = base64.b64decode(base64_str)
        with open('verify.jpg', 'wb') as file:
            file.write(imgdata)
        self.img_element = img_element

    def getVerifyResult(self):
        url = "http://littlebigluo.qicp.net:47720/"
        response = requests.request("POST", url, data={"type": "1"}, files={'pic_xxfile': open('verify.jpg', 'rb')})
        result = []
        print(response.text)
        for i in re.findall("<B>(.*)</B>", response.text)[0].split(" "):
            result.append(int(i) - 1)
        self.result = result
        print(result)

    def moveAndClick(self):
        try:
            Action = ActionChains(self.driver)
            for i in self.result:
                Action.move_to_element(self.img_element).move_by_offset(self.coordinate[i][0],
                                                                         self.coordinate[i][1]).click()
            Action.perform()
        except Exception as e:
            print(e.message())

    def submit(self):
        self.driver.find_element_by_id("J-login").click()

    def __call__(self):
        self.login()
        time.sleep(3)
        self.getVerifyImage()
        time.sleep(1)
        self.getVerifyResult()
        time.sleep(1)
        self.moveAndClick()
        time.sleep(1)
        self.submit()
        time.sleep(10000)

if __name__ == '__main__':
    Demo()()
```

见 gif 动画

```
import requests
from PIL import Image
from json import loads
import getpass
from requests.packages.urllib3.exceptions import InsecureRequestWarning

# 禁用安全请求警告
requests.packages.urllib3.disable_warnings(InsecureRequestWarning)

class LoginTic(object):
    def __init__(self):
        self.headers = {
            "User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_3) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/59.0.3071.115
Safari/537.36"
        }
        # 创建一个网络请求 session 实现登录验证
        self.session = requests.session()

    # 获取验证码图片
    def getImg(self):
        url = "https://kyfw.12306.cn/passport/captcha/captcha-image?login_site=E&module=login&rand=sjrand";
        response = self.session.get(url=url, headers=self.headers, verify=False)
        # 把验证码图片保存到本地
        with open('img.jpg', 'wb') as f:
            f.write(response.content)
```

```
        # 用 pillow 模块打开并解析验证码,这里是假的，自动解析以后学会了再实现
        try:
            im = Image.open('img.jpg')
            # 展示验证码图片，会调用系统自带的图片浏览器打开图片，线程阻塞
            im.show()
            # 关闭，只是代码关闭，实际上图片浏览器没有关闭，但是终端已经可以进行交互了(结束阻塞)
            im.close()
        except:
            print(u'请输入验证码')
        captcha_solution = input('请输入验证码位置，以","分割[例如 2,5]:')
        return captcha_solution

    # 验证结果
    def checkYanZheng(self, solution):
        # 分割用户输入的验证码位置
        soList = solution.split(',')
        # 由于 12306 官方验证码是验证正确验证码的坐标范围,我们取每个验证码中点的坐标(大约值)
        yanSol = ['35,35', '105,35', '175,35', '245,35', '35,105', '105,105', '175,105', '245,105']
        yanList = []
        for item in soList:
            print(item)
            yanList.append(yanSol[int(item)])
        # 正确验证码的坐标拼接成字符串，作为网络请求时的参数
        yanStr = ','.join(yanList)
        checkUrl = "https://kyfw.12306.cn/passport/captcha/captcha-check"
        data = {
            'login_site': 'E',  # 固定的
            'rand': 'sjrand',  # 固定的
            'answer': yanStr  # 验证码对应的坐标，两个为一组，跟选择顺序有关,有几个正确的，输入几个
        }
        # 发送验证
        cont = self.session.post(url=checkUrl, data=data, headers=self.headers, verify=False)
        # 返回 json 格式的字符串，用 json 模块解析
        dic = loads(cont.content)
        code = dic['result_code']
        # 取出验证结果，4：成功  5：验证失败  7：过期
        if str(code) == '4':
            return True
        else:
            return False

    # 发送登录请求的方法
    def loginTo(self):
        # 用户输入用户名，这里可以直接给定字符串
        userName = input('Please input your userName:')
        # 用户输入密码，这里也可以直接给定
        # pwd = raw_input('Please input your password:')
        # 输入的内容不显示，但是会接收，一般用于密码隐藏
        pwd = getpass.getpass('Please input your password:')
        loginUrl = "https://kyfw.12306.cn/passport/web/login"
        data = {
            'username': userName,
            'password': pwd,
            'appid': 'otn'
        }
        result = self.session.post(url=loginUrl, data=data, headers=self.headers, verify=False)
        dic = loads(result.content)
        print(result.content)
        mes = dic['result_message']
        # 结果的编码方式是 Unicode 编码，所以对比的时候字符串前面加 u,或者 mes.encode('utf-8') == '登录成功'进行判断，否则报错
        if mes == u'登录成功':
            print('恭喜你，登录成功，可以购票!')
        else:
            print('对不起，登录失败，请检查登录信息!')

if __name__ == '__main__':
    # checkYanZheng('0,3')
    login = LoginTic()
    yan = login.getImg()
    chek = False
    # 只有验证成功后才能执行登录操作
    while not chek:
        chek = login.checkYanZheng(yan)
        if chek:
            print('验证通过!')
        else:
            print('验证失败，请重新验证!')
    login.loginTo()
```





遇见的问题

数据库问题:

Q：#1251 - Client does not support authentication protocol requested by server

A：数据库连接出错，查资料后发现可能是没有设置密码问题，就重置了 MYSQL 密码，之后就可以运行了

Q：PyCharm:Error running xxx:Cannot run program "D:\Python27\python.exe"

A：这个错误是之前误删了下载好的 python 文件，后来在设置里，Project Interpreter 里添加找到了另外一个文件夹中下载好的 Python 文件，重新添加进去就不会报错了

Q：'gbk' codec can't decode byte 0xa6 in position 9737: illegal multibyte sequence A：打开 django/views 下的 debug.py 文件，转到 line331 行:

with Path(CURRENT_DIR, 'templates', 'technical_500.html').open() as fh

将其改成:

with Path(CURRENT_DIR, ‘templates‘, ‘technical_500.html‘).open(encoding="utf-8") as fh

就成功了。

Q：AttributeError: 'str' object has no attribute 'decode'

A：把出错代码中的 decode 改为 encode 即可

Q：1146, "Table 'myfile.mylink_houseinfo' doesn't exist")

A：python manage.py makemigrations python
manage.py migrate

运行后成功

Q：django.core.exceptions.ImproperlyConfigured: mysqlclient 1.3.13 or newer is required; you have 0.9.3

A：找到 Python 安装路径下的 base.py 文件，将文件中的如下代码注释

if version < (1, 3, 3):

raise ImproperlyConfigured("mysqlclient 1.3.3 or newer is required; you have %s" % Database.__version__)

重新在项目 manage.py 路径下执行如下命令即可

python manage.py makemigrations python

manage.py migrate

Q：还有少部分属于语法错误