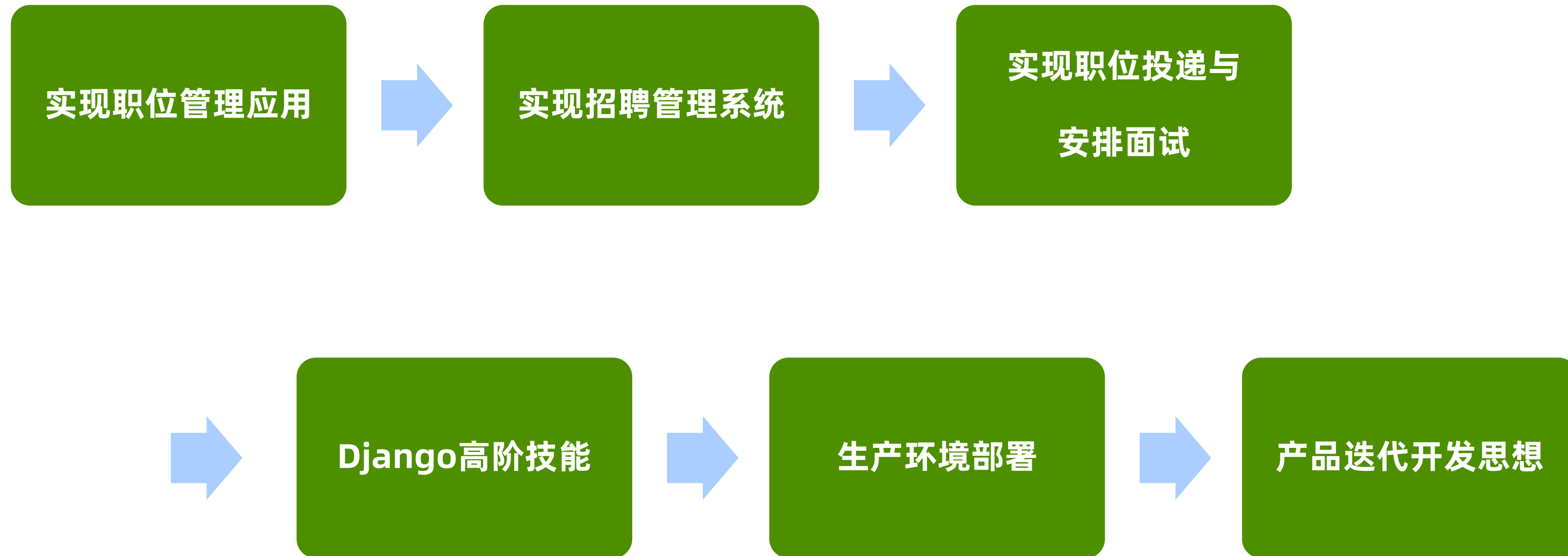


课程整体脉络





扫码试看/订阅

《 Django 快速开发实战 》视频课程

Agenda

- 一、初识 Django & 开发环境准备
- 二、使用 Django 创建一个应用 - 职位管理系统

一、初识 Django & 开发环境准备

初识 Django

- 课程需要的基础
- 学完课程后你能掌握的内容
- Django 适用于哪些场景
- Django 的优点和缺点
- 哪些著名产品使用了 Django
- Django 的 MTV 架构
- Django 的设计思想

课程需要的基础

基础条件

- 有一定 Python 基础，能够使用 Python 编写代码
- 有一定的 HTML/CSS/JavaScript 基础（非必须）
- 理解 Web 应用的前后端交互

学完课程后你能掌握的内容

- 能够使用 Django Admin 搭建管理后台
- 掌握 Django 管理后台的深度定制方法，能够添加定制的功能
- Django 中间件的工作原理，能够自己设计和实现一个中间件
- 能够使用 Django 快速为企业现有系统搭建管理后台
- 精益创业的产品思维，结合 Django 1-2 天快速迭代开发出有用的企业应用

Django 做了什么

- 参考 PHP MyAdmin
- 参考 MySQL 客户端

Current Server: phpMyAdmin demo - MySQL

(Recent tables) ...

filter databases by name > >>

- ...
- 000000
- Chad
- Erik
- Eton
- JanetM
- asd
- atestdb1986
- avinash
- bancaythongnoel
- bancaythongnoel.com
- binus
- ciago92
- cosmopolita
- entreprise
- fred
 - New
 - students
 - zip
- graphistedacunha
- grosappareils
- hi
- information_schema
- joins
- kayla
- menagerie
- music
- mysql

phpMyAdmin demo - MySQL

Databases SQL Status Users Export Import Settings Replication More

Users overview

User	Host	Password	Global privileges	Grant	Action
<input type="checkbox"/> debian-sys-maint	localhost	Yes	ALL PRIVILEGES	No	Edit Privileges Export
<input type="checkbox"/> lucasbrasilconta	localhost	Yes	USAGE	No	Edit Privileges Export
<input type="checkbox"/> pma	localhost	Yes	USAGE	No	Edit Privileges Export
<input type="checkbox"/> root	127.0.0.1	No	ALL PRIVILEGES	Yes	Edit Privileges Export
<input type="checkbox"/> root	localhost	No	ALL PRIVILEGES	Yes	Edit Privileges Export
<input type="checkbox"/> root	pmademo	No	ALL PRIVILEGES	Yes	Edit Privileges Export
<input type="checkbox"/> test_user	localhost	--	USAGE	No	Edit Privileges Export
<input type="checkbox"/> toor	%	Yes	ALL PRIVILEGES	Yes	Edit Privileges Export

☐ Check All
 With selected: Export

Add user

Remove selected users

(Revoke all active privileges from the users and delete them afterwards.)

☐ Drop the databases that have the same names as the users.

Go

Note: phpMyAdmin gets the users' privileges directly from MySQL's privilege tables. The content of these tables may differ from the privileges the server uses, if they have been changed manually. In this case, you should [reload the privileges](#) before you continue.

Server Status

Database stats

Collection Stats

Query

Activity Monitor

Export(File)

Import(File)

admin

config

leanote

32

albums

attachs

blog_comments

blog_likes

blog_singles

configs

email_logs

files

find_pwds

group_users

groups

has_share_notes

leanote.ShareNotes

leanote.has_share_notes

note_content_histories

note_contents

note_images

note_tags

notebooks

notes

reports

sessions

share_notebooks

share_notes

suggestions

tag_count

tags

themes

localhost

leanote.notes Stats

leanote.blog_singles

leanote.notes

Find

Insert

Update

Remove

Index

Aggregation

db.notes.find().sort({"_id": 1}).skip(0).limit(30)

Query or id

Sort {"_id": 1}

Fields {}

Skip 0

Limit 30

Run

▼ _id	ObjectId("540817e099c37b583c000005")	Object id
_id	ObjectId("540817e099c37b583c000005")	Object id
CreatedTime	2014-09-04 15:42:24+0800	Date
Desc	1. IntroductionLeanote, not just a notepad! Some Features Knowledge: Manage your kno...	String
ImgSrc	http://7xj51o.com1.z0.glb.clouddn.com/default_markdown.png	String
IsBlog	true	Boolean
IsMarkdown	false	Boolean
IsTrash	false	Boolean
NotebookId	ObjectId("540817e099c37b583c000002")	Object id
PublicTime	2014-09-04 15:42:24+0800	Date
RecommendTime	2014-09-04 15:42:24+0800	Date
Title	About Leanote	String
UpdatedTime	2015-06-15 18:42:09+0800	Date
UpdatedUserId	ObjectId("540817e099c37b583c000001")	Object id
UriTitle	%E6%AC%A2%E8%BF%8E%E6%9D%A5%E5%88%B0leanote	String
UserId	ObjectId("540817e099c37b583c000001")	Object id
IsDeleted	false	Boolean
Usn	200006	Integer
ReadNum	1	Integer
► _id	ObjectId("5447a20a19807a7b6e000000")	Object id
► _id	ObjectId("5481481bf4e87273d2000003")	Object id
► _id	ObjectId("5481489cf4e8721ee3000000")	Object id
► _id	ObjectId("54814eb4f4e872189d000003")	Object id
► _id	ObjectId("54817b33f4e8725881000000")	Object id
► _id	ObjectId("54817b49f4e8725881000001")	Object id
► _id	ObjectId("5482b541f4e87253cb000000")	Object id

Total Results: 17 (0.00s)

Remove

Expand 0

◀

▶

Django 适用于哪些场景

- 内容管理系统
 - 博客
 - CMS
 - Wiki
- 企业内部系统
 - 会议室预定
 - 招聘管理
 - ERP & CRM
 - 报表系统
- 运维管理系统
 - CMDB
 - 发布管理
 - 作业管理
 - 脚本管理
 - 变更管理
 - 故障管理

Django 的优点和缺点

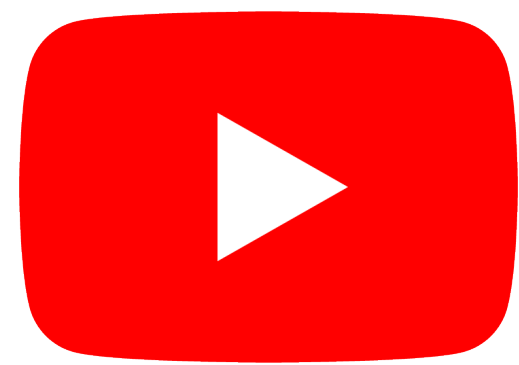
- 优点

- Python 实现，代码干净、整洁
- 提供管理后台，能够快速开发
- 复用度高，设计、使用上遵循 DRY 原则
- 易于扩展复用的中间件
- 内置的安全框架
- 丰富的第三方类库

- 缺点

- 单体应用-不易并行开发，单点扩展
- 不适合非常小的几行代码的项目
- 不适合于高并发的 to C 互联网项目

哪些著名产品使用了 Django



YouTube



Instagram



Dropbox

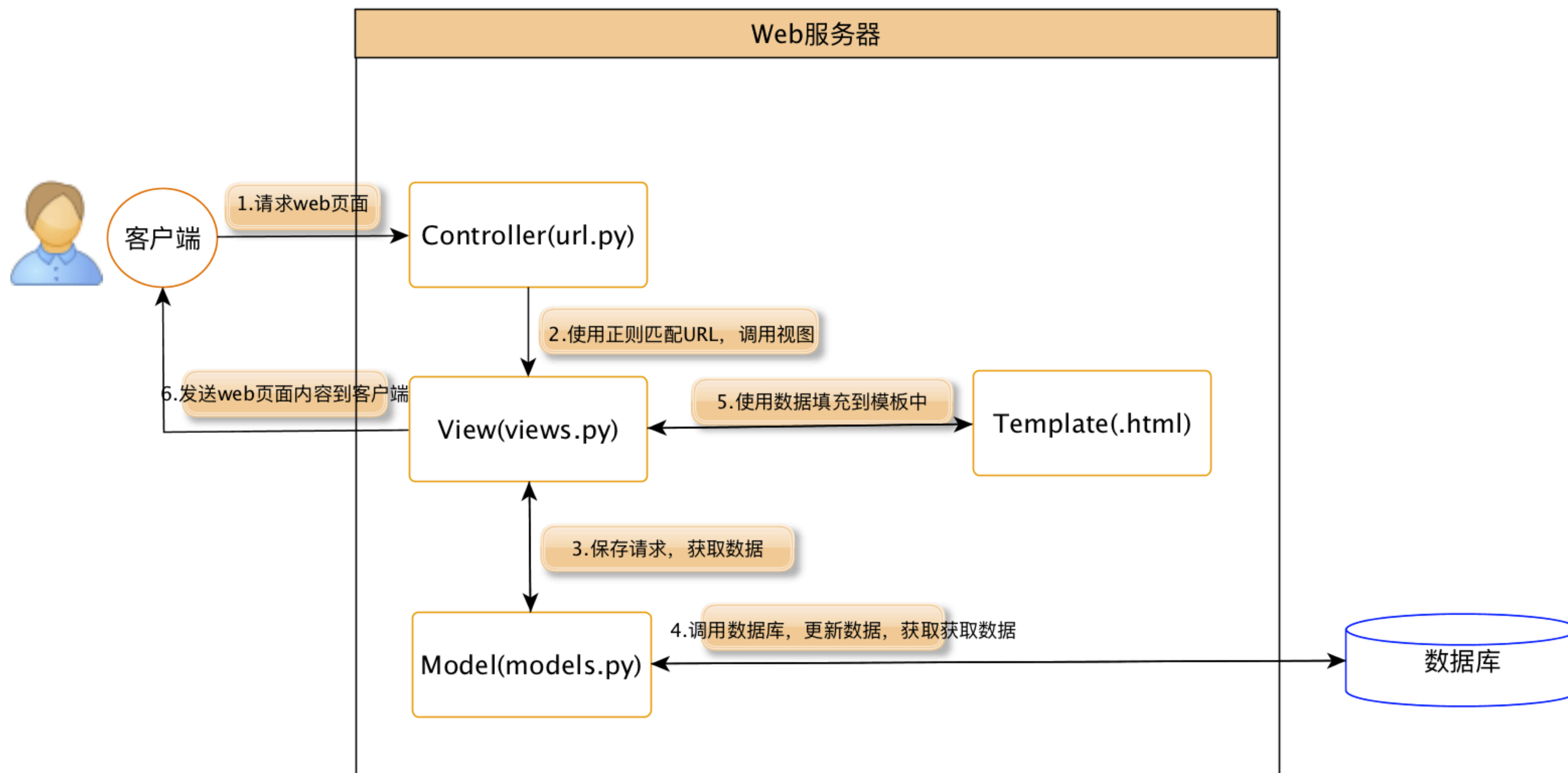


Spotify



Disqus

Django 的 MTV 架构



Django 的设计思想

- DRY (Don't repeat yourself) : 不重复造轮子
- MVT
- 快速开发
- 灵活易于扩展
- 松耦合
- 显式优于隐式

Django 开发环境准备

1. Python 开发环境准备 - Anaconda 介绍
2. Python 开发环境准备 - PyCharm 介绍
3. Python 开发环境准备 - PyCharm 中的 Django 支持
4. Django 安装与创建第一个应用

Anaconda 介绍

- Python 科学计算工具包：数据科学家的工具箱
- 包含了 Python 二进制发行包
- 包含 Numpy, Pandas, Matplotlib, SciPy, Bokeh, Jupyter, PyTorch, Tensorflow 等科学处理工具
- 包含了一个开源的 Python IDE: Spyder
- 包含了 Conda 包管理软件: `conda install xxx`

PyCharm 介绍

- PyCharm IDE, 出自 JetBrains 公司, IDEA 系列产品为 JetBrains 产出的产品
- 最好用的免费 Python IDE
- PyCharm 有 Community 版本, 有 Enterprise 版本
- PyCharm 社区版不支持 Django 开发, 但可以安装 Django 类库, 能够实现 Django 代码的自动提示

环境准备

- 推荐使用 Anaconda 版本，下载安装 Anaconda

<https://www.anaconda.com/products/individual>

```
bash ~/Downloads/Anaconda3-2020.02-MacOSX-x86_64.sh
```

```
conda install django
```

- 安装 PyCharm

<https://www.anaconda.com/pycharm>

使用 Django 创建第一个项目

创建会议室管理项目，项目名为 meetingroom

```
django-admin startproject meetingroom
```

```
cd meetingroom
```

启动项目

```
python manage.py runserver 0.0.0.0:8080
```

访问项目

<http://127.0.0.1:8080>

二、使用 Django 创建一个应用：职位管理系统

产品需求

1. 管理员能够发布职位
2. 匿名用户能够浏览职位
3. 匿名用户能够投递职位

职位管理系统-建模

职位名称，类别，工作地点，职位职责，职位要求，发布人，发布日期，修改日期

所有职位

共764个职位在招

职位名称	职位类别	工作地点
高级Java开发工程师（账号）	技术类	上海市
测试开发工程师	技术类	上海市
推荐系统开发工程师（漫画）	技术类	上海市
unity3D引擎工具开发工程师	技术类	上海市
资深流量运营（漫画）	产品运营类	上海市
IP周边品类运营（电商）	产品运营类	上海市
对韩版权高级经理	内容类	上海市
市场品牌经理（电商）	市场营销类	上海市
音视频工程师	技术类	上海市
类目运营（演唱会）	产品运营类	
资深内容运营（电商）	产品运营类	上海市
后端开发工程师	技术类	上海市

音视频工程师

工作地点

- 上海市

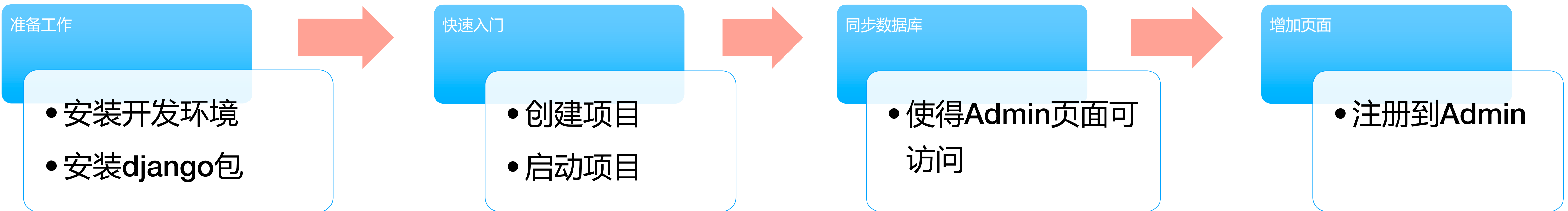
工作职责

- 负责视频录制及编辑功能开发；
- 负责图像处理算法的移动端实现以及优化；
- 视频剪辑/滤镜/转场/特效的实现及优化；
- 为公司视频拍摄和视频编辑提供技术支持和解决方案

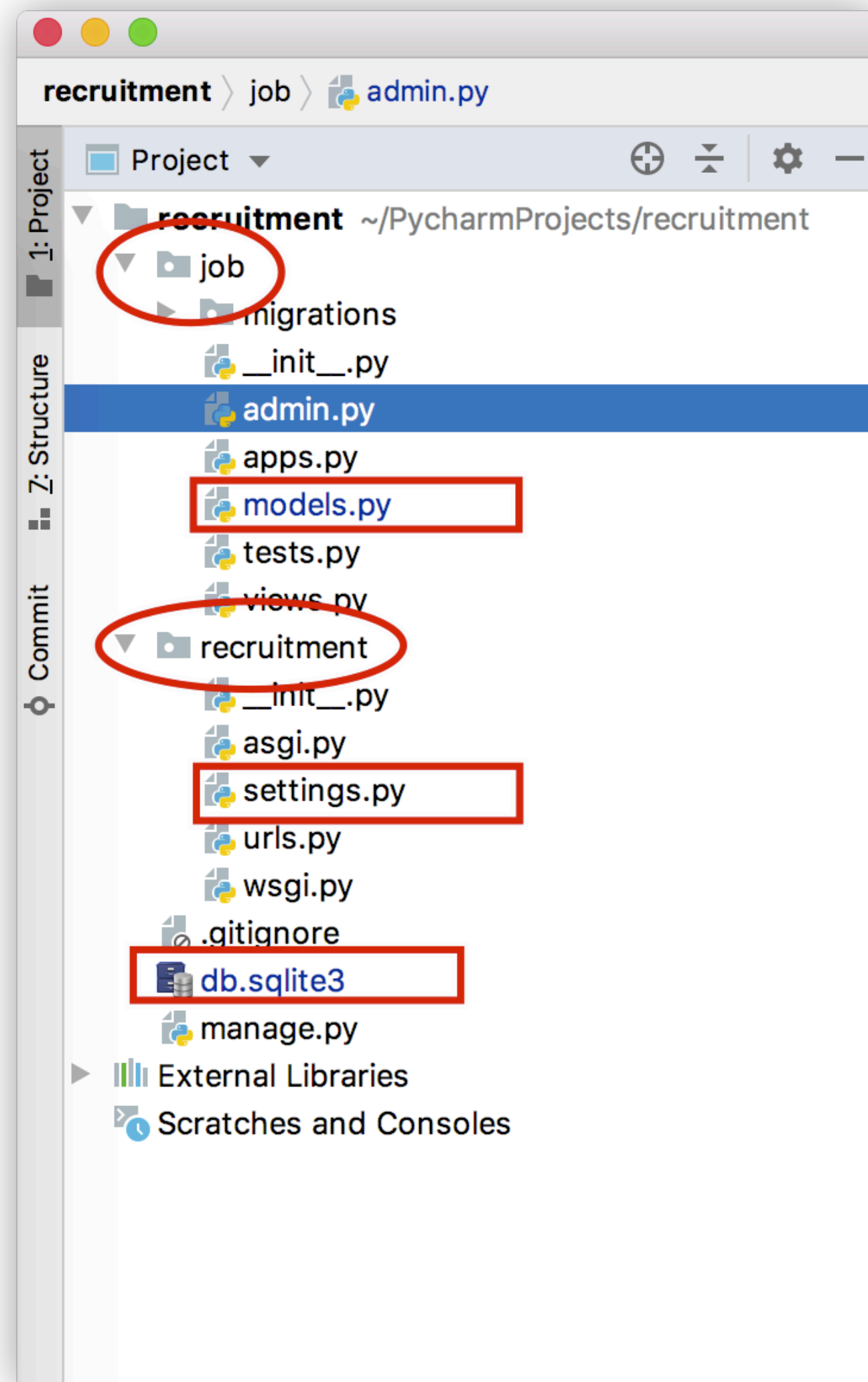
职位要求

- 本科及以上学历；
- 具有扎实的编程功底，良好的设计能力和编程习惯；
- 熟悉Android/iOS开发，熟悉C++，AndroidNDK开发；
- 有视频录制、编辑相关开发经验、熟悉视频编解码优先；
- 有FFmpeg，Mediacodec，VideoToolBox，OpenCV，开发经验者优先；
- 有移动端视频拍摄和视频编辑处理框架设计经验者优先；
- 熟悉并掌握TCP/UDP;RTP/RTCP等协议，熟悉voip相关技术，熟悉webrtc开源框架者优先；
- 积极乐观，责任心强，工作认真细致，具备良好的服务意识，具有良好的团队沟通与协作能力。

同步数据库 & 创建Admin后台账号



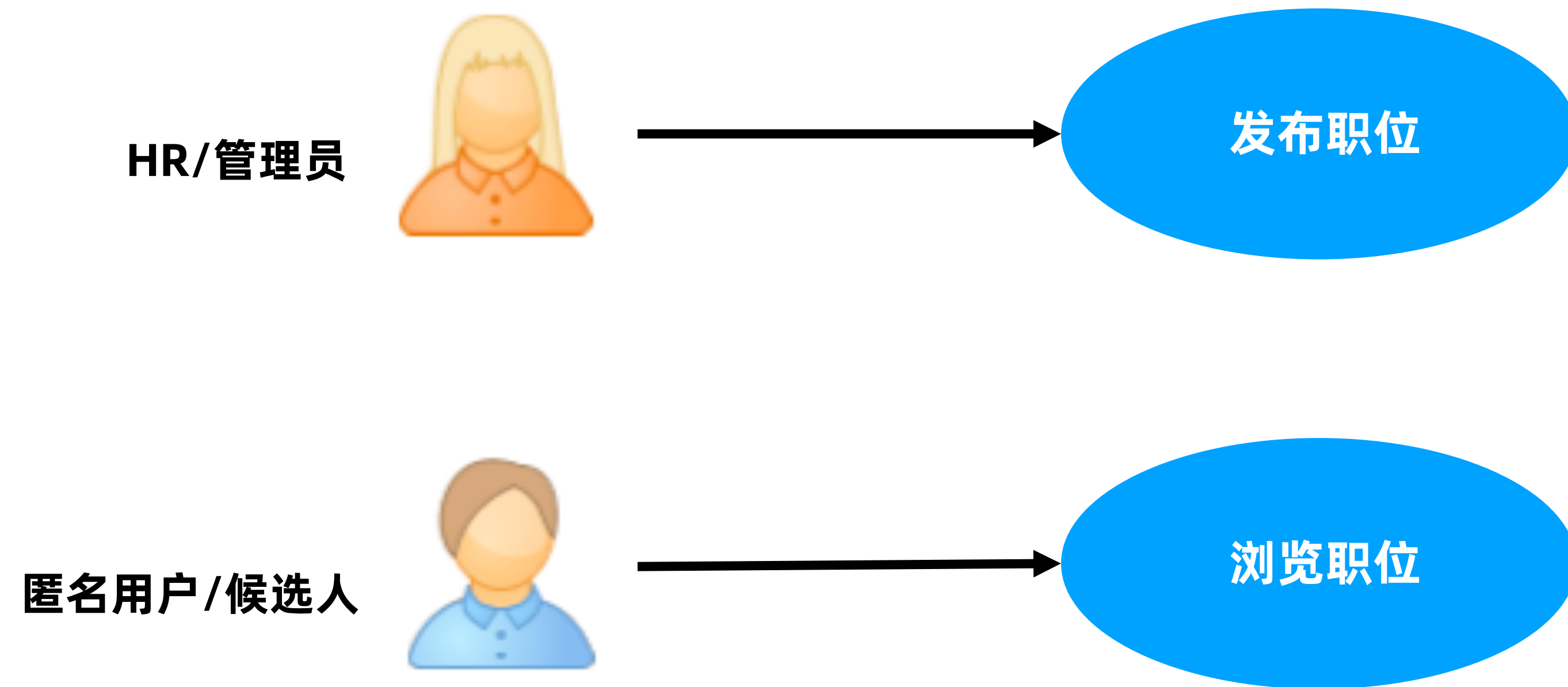
Django 项目代码结构



`./manage.py startproject recruitment`

`./manage.py startapp job`

职位列表展示



职位列表展示

- 列表页是独立页面，使用自定义的页面
- 添加如下页面
 - 职位列表页
 - 职位详情页
- 匿名用户可以访问

Django 的自定义模板

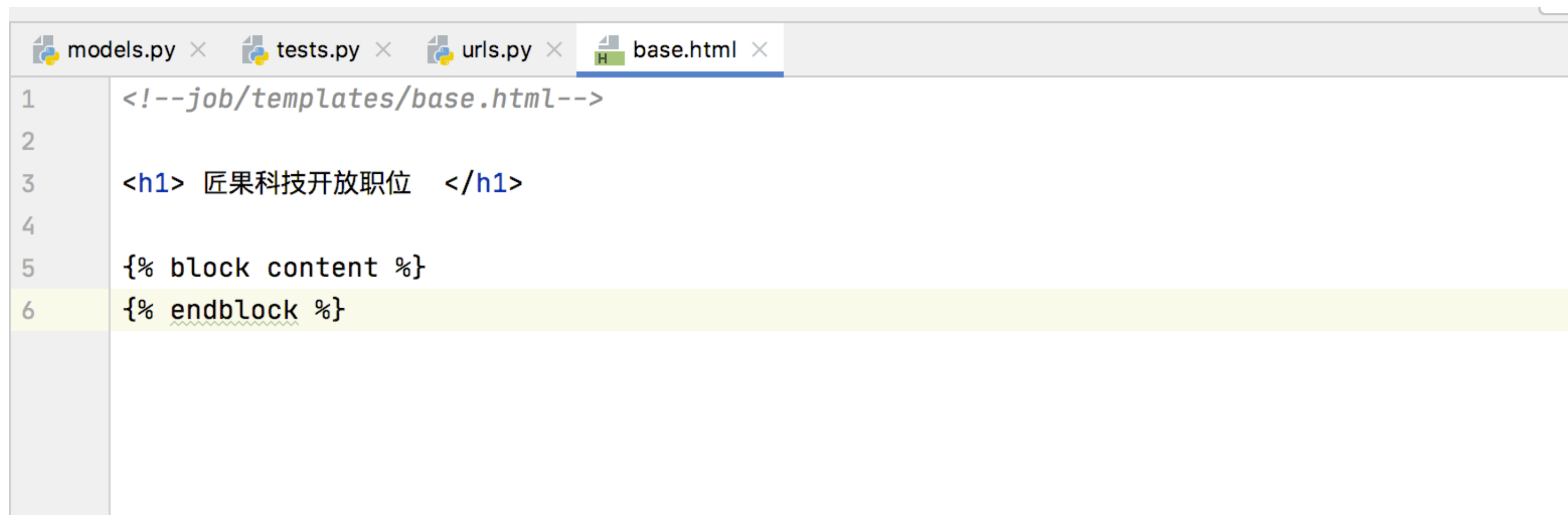
- Django 模板包含了输出的 HTML 页面的静态部分的内容
- 模板里面的动态内容在运行时被替换
- 在 views 里面指定每个 URL 使用哪个模板来渲染页面

Django 的自定义模板

- 模版继承与块 (Template Inheritance & Block)
 - 模板继承允许定义一个骨架模板，骨架包含站点上的公共元素（如头部导航，尾部链接）
 - 骨架模板里面可以定义 Block 块，每一个 Block 块都可以在继承的页面上重新定义/覆盖
 - 一个页面可以继承自另一个页面
- 定义一个匿名访问页面的基础页面，基础页面中定义页头
- 添加页面 `job/templates/base.html`

Base 模板

- 如下 job/templates/base.html 定义了站点的标题
- 使用 block 指令定义了页面内容块，块的名称为 content，这个块可以在继承的页面中重新定义



```
1 <!--job/templates/base.html-->
2
3 <h1> 匠果科技开放职位 </h1>
4
5 {% block content %}
6 {% endblock %}
```

添加职位列表页模板 - 继承自 base.html

- 这里使用 extends 指令来表示，这个模板继承自 base.html 模板
- Block content 里面重新定义了 content 这个块
- 变量：运行时会被替换，变量用 {{variable_name}} 表示，变量是 views 层取到内容后填充到模板中的参数
- Tag：控制模板的逻辑，包括 if, for, block 都是 tag

添加职位列表页模板 - 继承自 base.html



The screenshot shows the PyCharm IDE with the 'recruitment' project open. The 'job' directory under 'templates' is expanded, showing 'base.html', 'job.html', and 'joblist.html'. The 'joblist.html' file is selected and its content is displayed in the editor. The code is a Django template that extends 'base.html' and renders a list of jobs. The left sidebar shows the project structure, and the top bar shows the file tabs and a search bar.

```
1 {% extends 'base.html' %}
2
3
4 {% block content %}
5     终于等到你，期待加入我们，用技术去探索一个新世界
6
7 {% if job_list %}
8     <ul>
9         {% for job in job_list %}
10            <li>{{job.type_name}} <a href="/job/{{ job.id }}" style="color:blue">{{ job.job_name }}</a>  {{job.city_name}} </li>
11        {% endfor %}
12    </ul>
13 {% else %}
14     <p>No jobs are available.</p>
15 {% endif %}
16
17 {% endblock %}
```

模板内容自动转义

- 不做内容转义的问题，对于以下的模板内容
 - 你好 {{ name }}
- 当用户的名字输入如下内容时：
 - **<script>alert('hello')</script>**
- 结果
 - 模板展现的时候，用户打开的页面会出来一个弹窗（即页面展现时可以动态的代码）
 - 严重的安全漏洞（XSS 跨站脚本攻击）
- 模板内容自动转义：用户输入一段脚本作为名字时，页面展现时标签都被转义

职位列表的视图

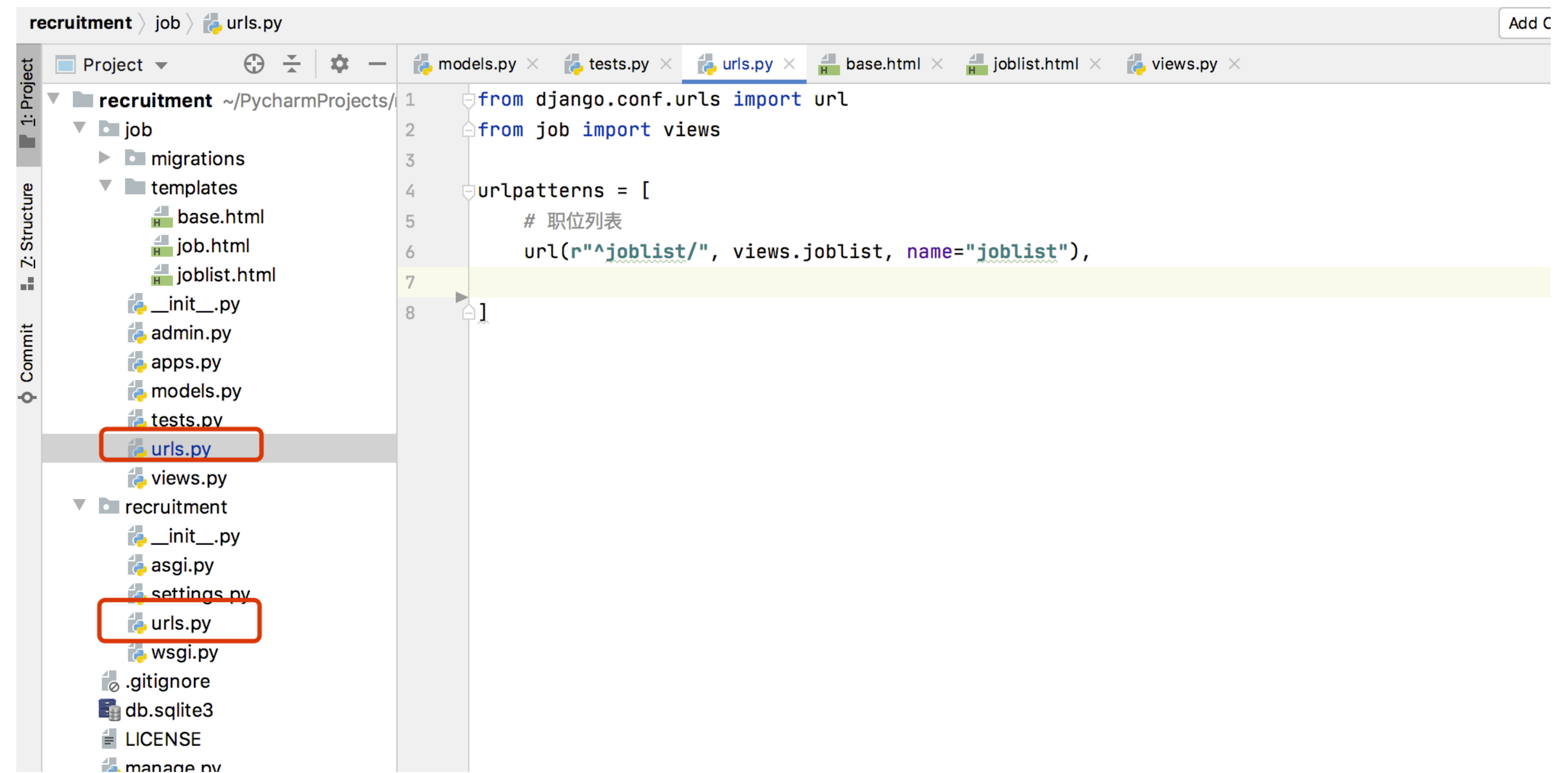
- 视图里面获取数据，把数据传入到模板中
- 示例中，使用 Django 的 model 来获取数据，数据按照职位类型排序
- 模板渲染指定了使用前面定义的 joblist.html，把一个含有 job_list 这个 key 的 map 传入到模板

职位列表的视图

```
models.py × tests.py × urls.py × base.html × joblist.html × views.py ×
3  # Create your views here.
4  from django.shortcuts import render
5  from django.http import HttpResponse
6  from django.http import Http404
7  from django.template import RequestContext, loader
8
9  from job.models import Job
10 from job.models import Cities, JobTypes
11
12
13 def joblist(request):
14     job_list = Job.objects.order_by('job_type')
15     template = loader.get_template('joblist.html')
16     context = {'job_list': job_list}
17     for job in job_list:
18         job.city_name = Cities[job.job_city][1]
19         job.type_name = JobTypes[job.job_type][1]
20     return HttpResponse(template.render(context))
21
```

添加 URL 路径映射

- 让添加的页面，能够通过 URL 访问到
- /joblist/ 的路径访问到 views 里面定义的 joblist 视图
- 这个视图是一个 Method View，方法表示一个视图



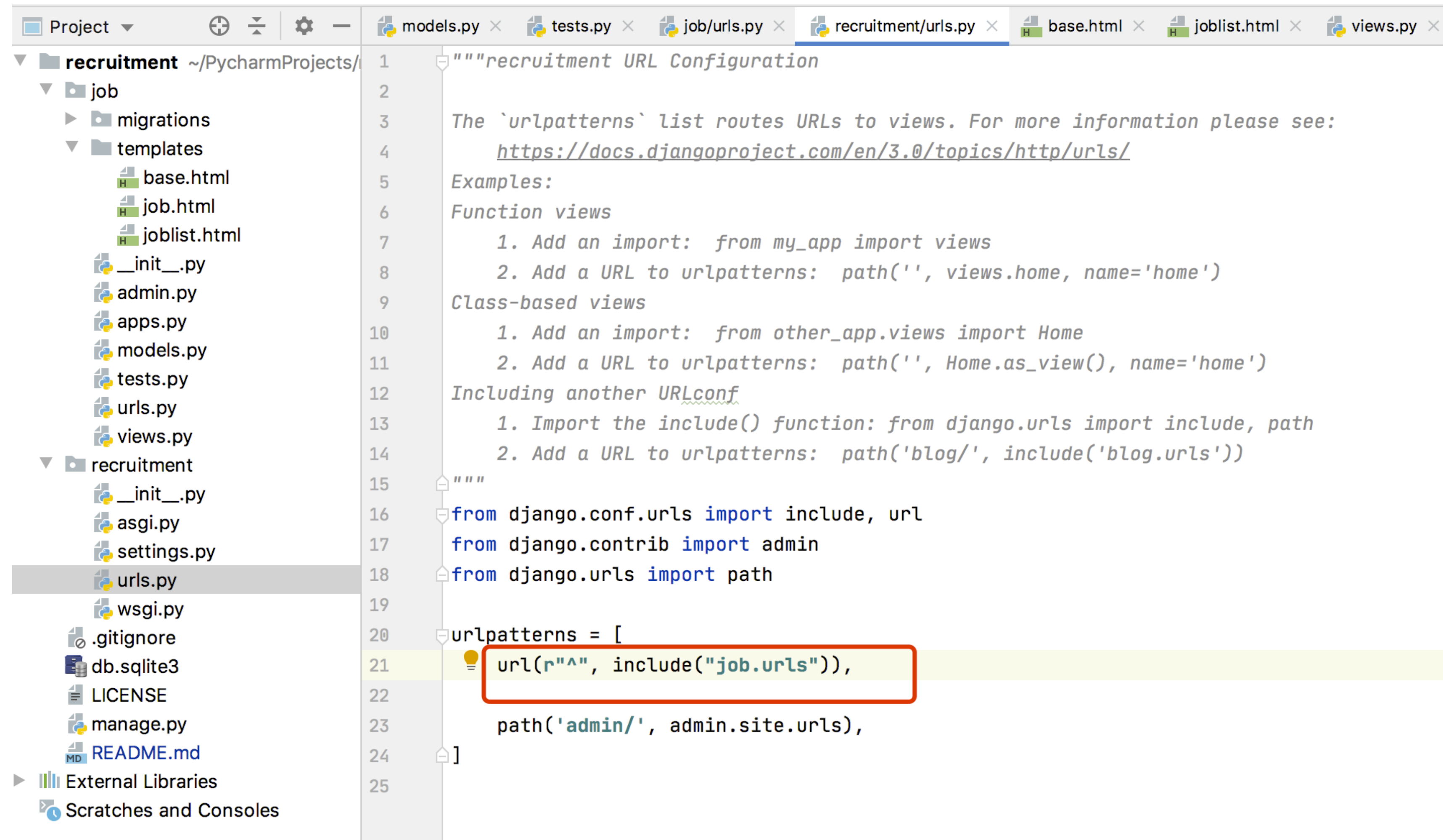
```
recruitment > job > urls.py
Project
1: Project
Z: Structure
Commit
recruitment
├── job
│   ├── migrations
│   └── templates
│       ├── base.html
│       ├── job.html
│       └── joblist.html
│   ├── __init__.py
│   ├── admin.py
│   ├── apps.py
│   ├── models.py
│   ├── tests.py
│   └── urls.py
├── recruitment
│   ├── __init__.py
│   ├── asgi.py
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
├── .gitignore
├── db.sqlite3
├── LICENSE
└── manage.py
```

```
1 from django.conf.urls import url
2 from job import views
3
4 urlpatterns = [
5     # 职位列表
6     url(r'^joblist/', views.joblist, name="joblist"),
7 ]
8
```

应用（app）的所有 URL 定义加入到项目（recruitment）中

如下图，把 job 应用下面的 URL 都加到路由中；

收到请求时，先走 jobs 应用下面的 URL 路由找页面，然后再按照 admin/ 路径匹配请求 URL



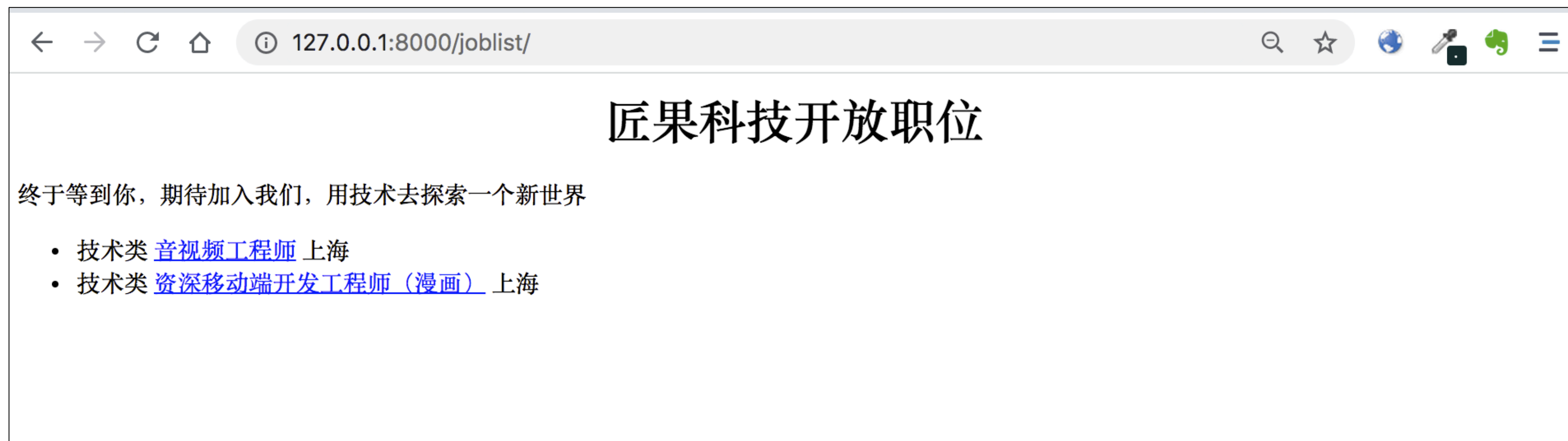
The screenshot shows the PyCharm IDE interface. On the left, the Project Explorer displays the 'recruitment' project structure, including sub-projects 'job' and 'recruitment'. The 'job' project contains 'migrations', 'templates', and 'urls.py'. The 'recruitment' project contains 'urls.py'. The 'urls.py' file for 'recruitment' is open in the editor, showing the following code:

```
1 """recruitment URL Configuration
2
3 The `urlpatterns` list routes URLs to views. For more information please see:
4     https://docs.djangoproject.com/en/3.0/topics/http/urls/
5 Examples:
6 Function views
7     1. Add an import: from my_app import views
8     2. Add a URL to urlpatterns: path('', views.home, name='home')
9 Class-based views
10    1. Add an import: from other_app.views import Home
11    2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
12 Including another URLconf
13    1. Import the include() function: from django.urls import include, path
14    2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
15 """
16 from django.conf.urls import include, url
17 from django.contrib import admin
18 from django.urls import path
19
20 urlpatterns = [
21     url(r'^$', include("job.urls")),
22     path('admin/', admin.site.urls),
23 ]
```

The line `url(r'^$', include("job.urls")),` is highlighted with a red box, indicating the URL definition for the 'job' application being added to the 'recruitment' project's routing.

职位列表页

- 模板添加定义，View 页面添加完，URL 中也定义路由之后，再访问页面：
- <http://127.0.0.1:8000/joblist/>



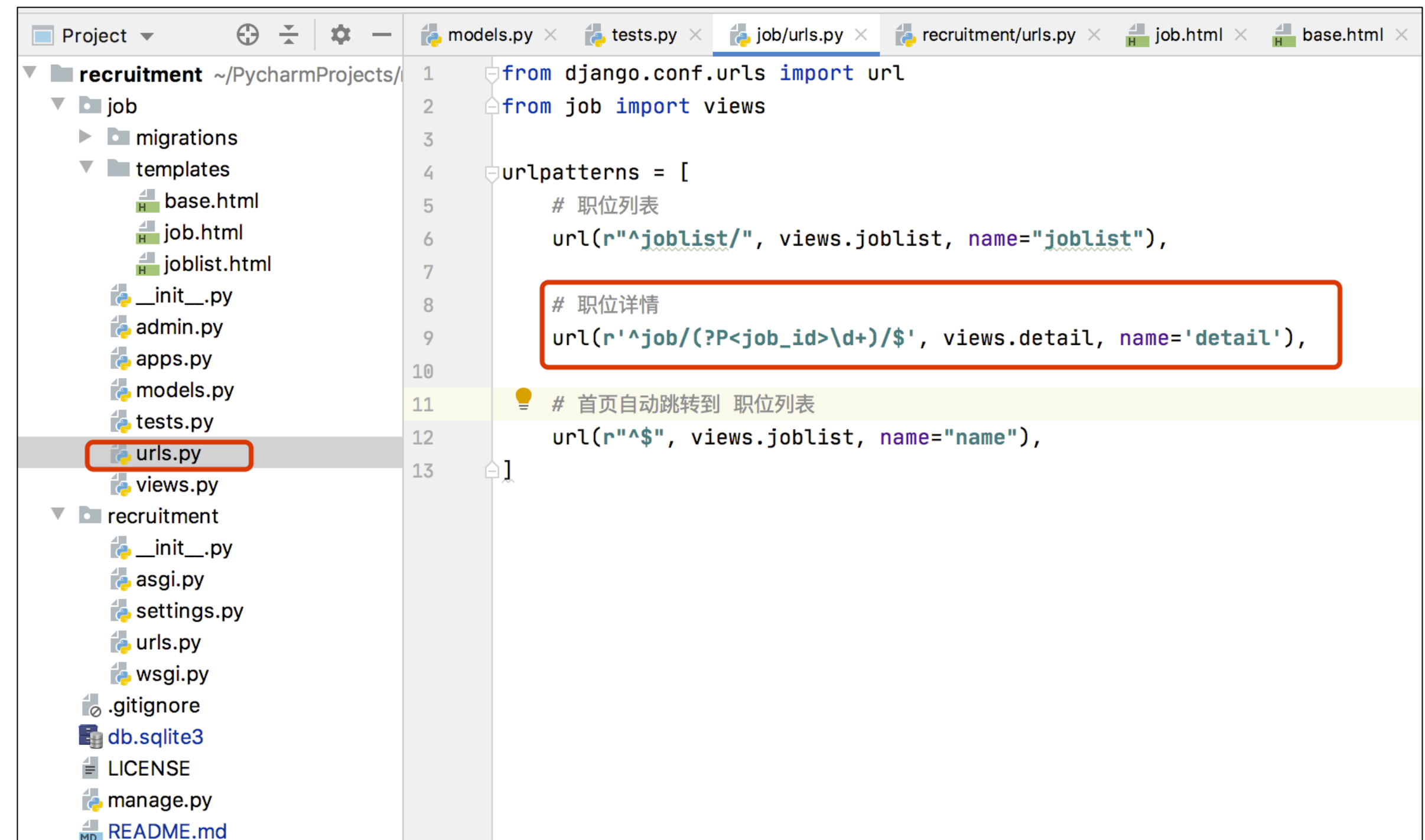
职位详情页

- 前面列表页，每个职位上有一个链接，指向职位详情页
- 同样添加如下 3 块内容：
 - 详情页模板 - 定义内容呈现 (Template)
 - 详情页视图 - 获取数据逻辑 (View)
 - 定义 URL 路由

职位详情页

如下在 views.py, urls.py 中分别定义了 View 视图，以及 URL 的路由规则 /job/job_id 来访问详情

```
22 def detail(request, job_id):
23     try:
24         job = Job.objects.get(pk=job_id)
25         job.city_name = Cities[job.job_city][1]
26     except Job.DoesNotExist:
27         raise Http404("Job does not exist")
28     return render(request, 'job.html', {'job': job})
29     return HttpResponse("You're looking at job %s." % job_id)
30
31
```



The screenshot shows the PyCharm IDE interface. On the left, the 'Project' view displays the file structure of a Django project named 'recruitment'. The 'job' directory is expanded, showing files like 'migrations', 'templates', 'base.html', 'job.html', 'joblist.html', '__init__.py', 'admin.py', 'apps.py', 'models.py', 'tests.py', 'urls.py' (highlighted with a red box), and 'views.py'. The 'recruitment' directory also shows files like '__init__.py', 'asgi.py', 'settings.py', 'urls.py', 'wsgi.py', '.gitignore', 'db.sqlite3', 'LICENSE', 'manage.py', and 'README.md'. On the right, the 'job/urls.py' file is open, showing the following code:

```
1 from django.conf.urls import url
2 from job import views
3
4 urlpatterns = [
5     # 职位列表
6     url(r'^joblist/', views.joblist, name='joblist'),
7
8     # 职位详情
9     url(r'^job/(?P<job_id>\d+)/$', views.detail, name='detail'),
10
11     # 首页自动跳转到 职位列表
12     url(r'^$', views.joblist, name='name'),
13 ]
```

职位详情页

匠果科技开放职位

[返回职位列表](#)

岗位名称：音视频工程师

城市：上海

岗位职责：

负责视频录制及编辑功能开发；
负责图像处理算法的移动端实现以及优化；
视频剪辑/滤镜/转场/特效的实现及优化；
为公司视频拍摄和视频编辑提供技术支持和解决方案

任职要求：

本科及以上学历；
具有扎实的编程功底，良好的设计能力和编程习惯；
熟悉Android/iOS开发，熟悉C++，AndroidNDK开发；
有视频录制、编辑相关开发经验、熟悉视频编解码优先；
有FFmpeg，Mediacodec，VideoToolBox，OpenCV，开发经验者优先；
有移动端视频拍摄和视频编辑处理框架设计经验者优先；
熟悉并掌握TCP/UDP；RTP/RTCP等协议，熟悉voip相关技术，熟悉webrtc开源框架者优先；
积极乐观，责任心强，工作认真细致，具备良好的服务意识，具有良好的团队沟通与协作能力。

申请



扫码试看/订阅

《 Django 快速开发实战 》视频课程