项目准备在vue最后一天的笔记里.

资料:

1.meiduo\_mall\_admin (前端文件)

2.美多商城后台项目统一环境

每次改完代码,后端需要重新运行,前端不用重新运行

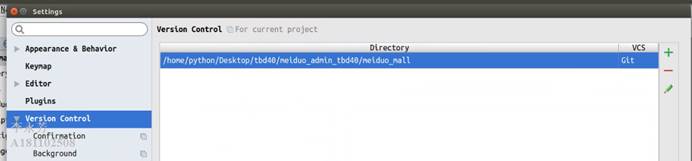
前端代码改了之后需要重新运行前端 npm run dev

# settings设置

#### 1 解释器

--建完虚拟环境,解释器重新选一下.直接里面会有,没有就add local

#### 2 版本控制



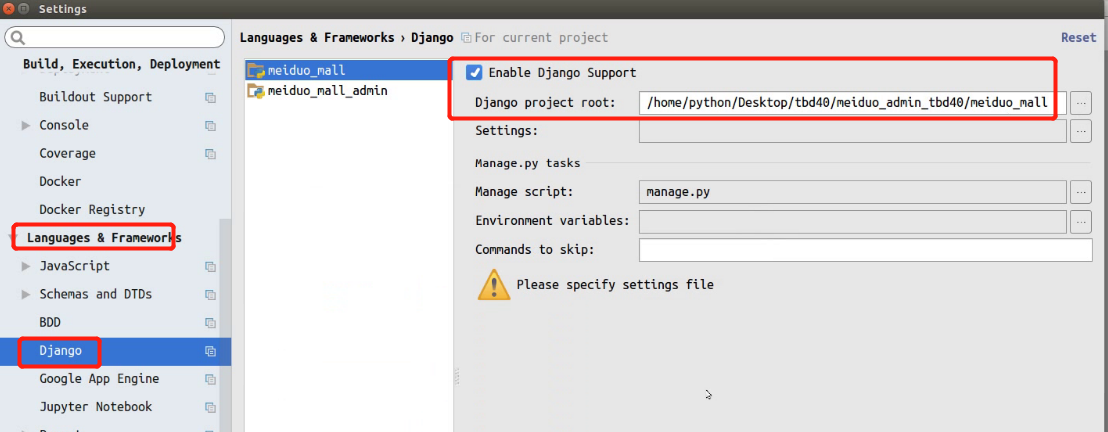
#### 3 ES6 （前端报红线设置一下）

#### 4 django路径问题

运行报错：No manage.py file specified in Settingf->Django Support

原因：不是自己新建的项目，是打开的一个目录。

解决：Django 项目路径,如下图 。

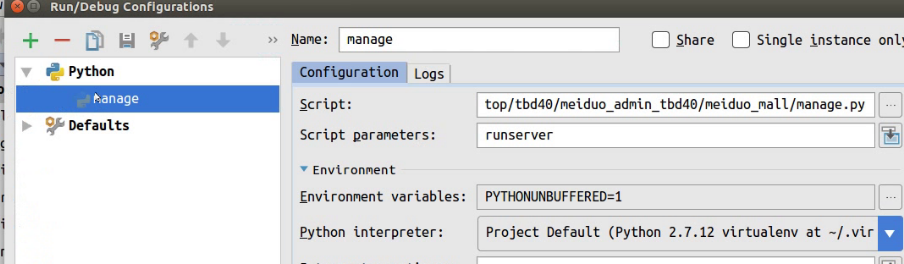


#### 5 运行配置

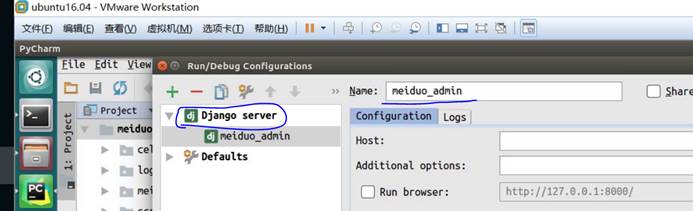
直接拷过来的代码运行可能有问题.在运行处点击 Edit configuration,(用pycharm 首次打开别人的文件Edit configuration处为灰色,点不了,这时需要close all 关闭所有打开 的py文件)

把原来的删除,自己新建一个 Django server ,只写名字就可以（如果原来的能运行可以不用改）（改成python也都可以）

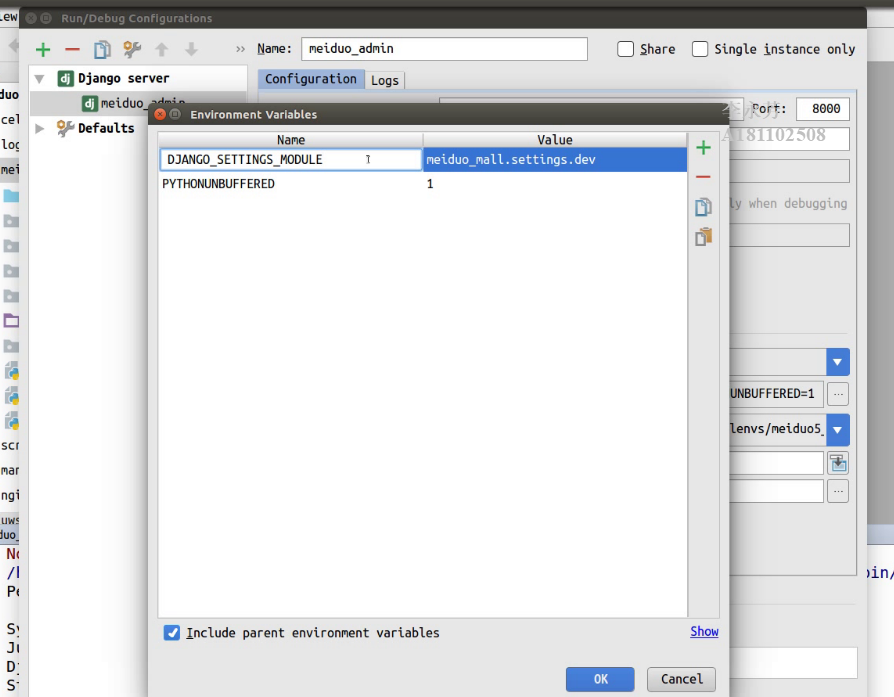
 原来:



改为:



#### 如果运行有问题就添加下环境变量:



#### pycharm运行按钮灰色,不能点击,

路径不对,需要重新配置下解释器或 关闭所有已经打开的py文件，再重新打开。

#### 端口占用

在终端输入

sudo fuser -k 8000/tcp

这样和端口8000相关的进程就都关了。

#### 导入数据库数据：【!导入数据前确保该迁移的文件已经迁移，确保表ok】

修改master\_db.sql 文件内容。（# USE'数据库名' 不能和Current Database 相同，就能导入成功。）

新建数据库 create database meiduo\_admin charset=utf8;

连接数据库，mysql -uroot -p

Use meiduo\_admin;

Source master\_db.sql;( 说明：如果是在 .sql文件的同级目录打开的数据库就直接不加路径，否则前面需要加上绝对路径)

注意：[导入数据库时, master\_db.sql中有要修改的内容,将使用数据库改成自己要用的数据库]



初始项目提交. 前端代码有问题(LF格式问题).忽略meiduo\_mall\_admin文件夹.

运行:分别运行后台和前台项目,前台项目运行： npm run dev

#### 问题：linux 运行npm run dev，报错：sh: 1: webpack-dev-server: Permission denied

解决：

1.将node和npm升级到最新版本。<https://www.jianshu.com/p/f2592d106aac>

2.再重新安装webpack,就可以了。

sudo npm install webpack -g

sudo npm install webpack-dev-server -g

**sudo apt-get update失败已经解决**

（1）网络原因----先检查网络好不好用

（2）ubuntu16 自带的源不好用，或者有问题

**【wi**ndows下】python3安装xadmin出现 UnicodeDecodeError: 'gbk' codec can't decode byte 0xa4 in position 3444: illegal multibyte sequence

解决:参考百度下载新的某文件的去替换掉原来site-package里面的.

安装依赖文件,先把所有的加载完,在安装,有没加载完的,影响后续所有其他的安装.

{

报错:error: Microsoft Visual C++ 14.0 is required. Get it with "Microsoft Visual C++ Build Tools": <http://landinghub.visualstudio.com/>

visual-cpp-build-tools

----------------------------------------

Command "D:\Code\pro\_houtai2020\houtai2020\meiduo\_mall\venv\Scripts\python.exe -u -c "import setuptools, tokenize;\_\_file\_\_='C:\\Users

\\lyf\\AppData\\Local\\Temp\\pip-install-qyet7hoa\\fdfs-client-py\\setup.py';f=getattr(tokenize, 'open', open)(\_\_file\_\_);code=f.read(

).replace('\r\n', '\n');f.close();exec(compile(code, \_\_file\_\_, 'exec'))" install --record C:\Users\lyf\AppData\Local\Temp\pip-record-

eq1zrqcu\install-record.txt --single-version-externally-managed --compile --install-headers D:\Code\pro\_houtai2020\houtai2020\meiduo\_

mall\venv\include\site\python3.6\fdfs-client-py" failed with error code 1 in C:\Users\lyf\AppData\Local\Temp\pip-install-qyet7hoa\fdf

s-client-py\

window安装pip,有的 会让安装"Microsoft Visual C++ ,有些python库需要使用Microsoft Visual C++编译.

这个是window一般自带的，一般是2015.

有些库是C,所以需要



}

#### 问题：windows上安装依赖有新问题

依赖文件里面有两个有问题,先注释掉,避免影响其他的安装:

#fdfs-client-py==1.2.6 (需要使用下载包,不直接在线安装)

#xadmin==0.6.1 (其实用不到，可以先不装)

#### 问题：Command "python setup.py egg\_info" failed with error code 1 (常见)

解决方法是更新 setuptools 和 pip：

pip install --upgrade setuptools python -m pip install --upgrade pip

#### 问题描述：启动django服务时出现“Error: [WinError 10013] 以一种访问权限不允许的方式做了一个访问套接字的尝试。”的错误 (常见)

问题原因：8000端口被占用了. [git.note](note://21E56BBC463140809E11A0103A3700EE)

解决方法1:默认启动的服务端口是8000，启动时修改该端口即可

解决方法2:

step1：通过命令行找出端口对应的PID进程

netstat -ano|findstr 8000

step2：命令行找出进程对应的详细信息

C:\Users\Administrator>tasklist |findstr 7336

step3:关闭该端口占用

C:\Users\Administrator>taskkill /pid 7336 /F

此时8000端口已经被释放，可以提供给其他服务使用

#### 问题：运行后台项目，导入不了Django。

**报 ImportError: Couldn't import Django. Are you sure it's installed and available on your PYTHONPATH en...**

解决：将 Edit Configuration 处的django server 改选成了python 模式，重新进行设置。

#### 问题：执行命令python manage.py XXX会报错,

#### ImportError: Couldn't import Django. Are you sure it's installed and available on your PYTHONPATH environment variable? Did you forget to activate a virtual environment?

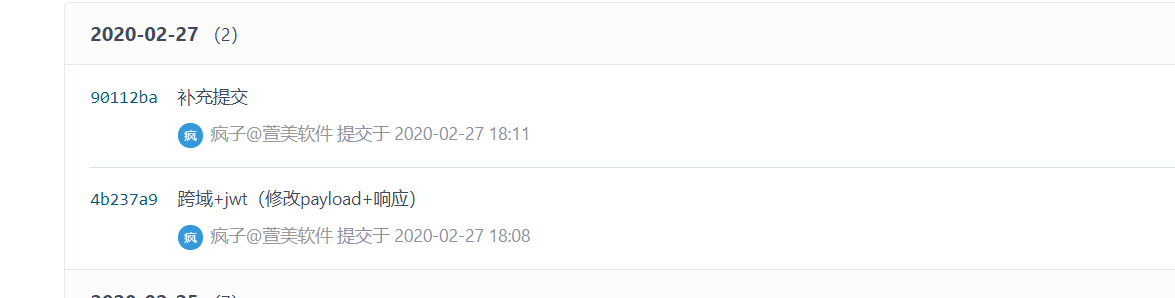
解决：

1.查看当前使用的Python3.5环境已经安装了django。（使用pip list 和pip3 list都能看到）

2.使用python3 manage.py runservser，成功。

#### Git

想使用pycharm提交，临时设置了启用git,邮箱写成了 lyf@qq.com，提交完之后成了下面这样。



# 2 管理员登录

**提交表单不受同源政策的限制**

**跨域只是浏览器中的安全策略,像android 或ios没有这个**

js ajax跨域时浏览器会屏蔽请求 会发起 option 试探, 服务器返回一个信息 , 浏览器认为可靠就可以访问

之前会话保持之前用session是基于cookie的,最后基于浏览器. 前后端分离直接用token

token是 技术

jwt是具体实现,最通用的一种token

jwt是一个包含身份信息的字符串,包含 header,payload,signature

正常路由是后端定,此处我们与vue一致

跨域文档 看http协议

### Jwt -- Json Web Token认证机制

#### token保存在哪?

保存在客户端。

你发送post请求，🡺f服务器验证成功生成token给你🡪每次交互的时候把token带上，一般放在header里。 HTML5的话一般有WebStorage(localStorage)做保存，如果是APP的话，这属于C/S系统，也是自己负责保存。 每次请求带过去，后端接收到你的请求后会从requestHeader中获取你的token

**cors跨域类CorsMiddleware中，3个重要函数：**

process\_request(）请求对象创建之后，路由规则匹配之前

process\_view() 路由规则匹配成功找到视图以后，视图执行之前 。

process\_response() 视图返回了response之后，数据真正返回给浏览器之前

**jwt 网站**

左右可以互查： 根据左边得到右边，也可以根据右边的到左边。

{

"alg": "HS256",

"typ": "JWT"

}

{

"username": "admin123",

"email": "admin123@qq.com",

"user\_id": 2,

"exp": 1582874896

}

**自定义payload：**

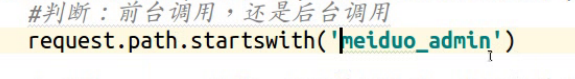
构造方法：在函数内调用源函数。

运行：去响应里面找到token，jwt反解码，来看payload的内容。

jwt验证方法是调用了DJNGO自带的，authenticate 方法。

jwt 验证的代码写在了序列化器里边,序列化器调用了django自带的用户认证authenticate 方法

**自定义验证条件时，按道理也可以这样判断：**



但是运行报错，没有request对象，

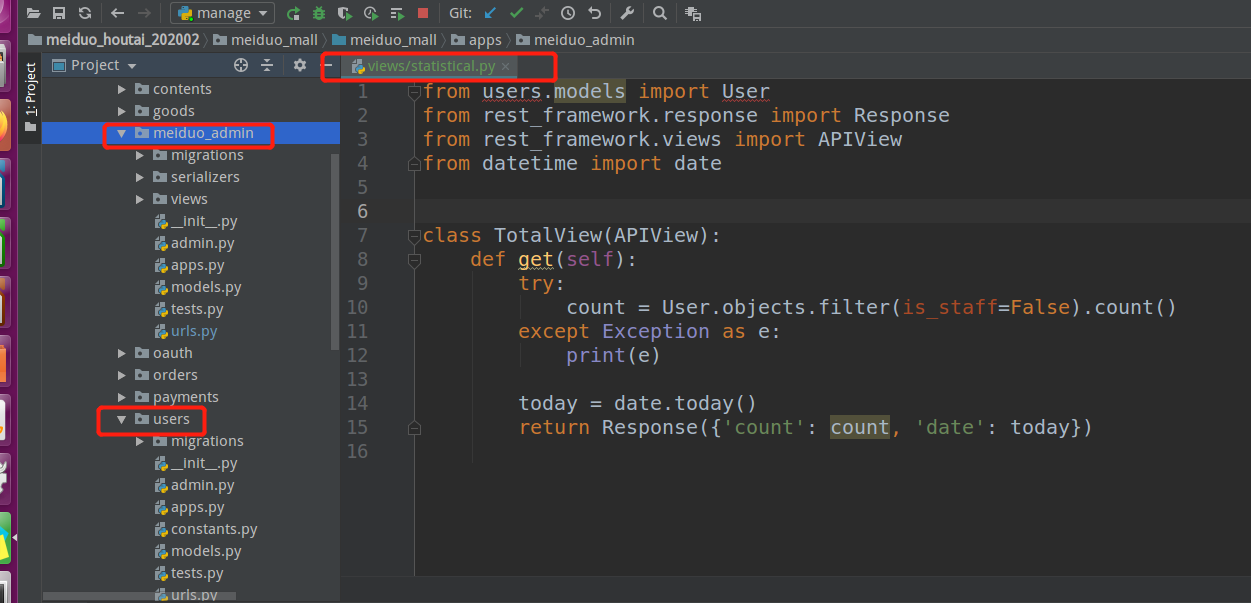
查看源码，JWT 在调用authenticate（）方法时没有传requests对象这个参数，所以走jwt时，request为空。故以此来进行前后端请求的区分。

#### 疑问： JWT 中导入的是django.contrib.auth的authenticate（），自己再项目中重写的，也能起作用

解答：JWT中 的导包from django.contrib.auth import authenticate，其实是调用了settings文件中指定的认证后端。

**疑问：导包。从一个子应用去调用另一个子应用里的东西？**

本项目将apps指定为了应用的导包路径，所以直接从子应用开始导入的。



#### 疑问：有的导包处有红线

解决：参照之前项目，把根目录设置为apps之后ok.

**drf 自带的过滤功能没有模糊查询**

**drf其实用的是django的插件，默认用的django自带的模板。**

**当想直接用后端得到运行结果时，需要直接在settings中改回成django自带的模板。**

**否则会报错如下：django.template.exceptions.TemplateDoesNotExist: rest\_framework/api.html**

------

查询集，每句代码返回的是一个叫查询集的东西，惰性执行，可以不断拼接。只有遍历或序列化时才会真正去查询。所以下面的代码可以不断拼接，在构造一条查询语句。

def get\_queryset(self):

# self.request,self.args,self.kwargs

# 普通会员

queryset = User.objects.filter(is\_staff=False)

keyword = self.request.query\_params.get('keyword')

if keyword:

queryset = queryset.filter(username\_\_contains=keyword)

# 排序：最新的放在最前面

queryset = queryset.order\_by('-id')

return queryset

===

查询所有用户时，把分页也做好后再去前端看运行结果。否则看不到效果。

分页会把查询结果放到 result里面去。

**分页：最终使用的django提供的分页类型。**

自己的分页直接继承这个就行，from rest\_framework.pagination import PageNumberPagination

用户名验证：1.位数，正则 2.不能重复

手机号验证: 1.正则 2.不能重复

# 4 用户管理

jinja2模板问题,先改成django.上线时再改回来

前端优化的几个位置:

**1 celery**

**2 缓存(省市区地址)**

**3 静态化（针对前后端不分离）**

静态化的原因：1. 查询数据耗时， 2. 根据数据生成html标签耗时. ==>所以静态化.

步骤：1.查到数据--生成HTML标签 ；2.将html的标签写到一个文件。

**静态化对前后端分离没有意义。**前端是浏览器静态化才有意义,前后端分离,前端不知道是不是浏览器,只能考虑缓存优化。

**4 mysql读写分离**

手机号 数字 不需要计算时 就不用定义成数字类型

#### 写视图

先写原始代码 再看是否有已有方法 能否抽取

get\_queryset 使用这个方法时不需要传参数request 只有get post 这些时才需要传.

在View 里有:

self.request

self.args 路由中提取的非命名参数

self.kwargs 路由中提取的命名参数

# 5商品管理

**问题：使用ModelViewSet，视图和路由写好后没有put和delete方法**

**商品-规格-新增**

**问题：点新增进去，下拉选项没有**

需要有一个spu的简单查询，注意路由和前端的一致才能正常显示。

**商品-规格-新增**

前天在新增功能时已经配置好spu的查询部分，今天再看又不能选了。 前端抓包，发现签名已经过期。修改后台jwt的签名有效期，解决。

#### read\_only=True

drf，写序列化器时，除了 read\_only=True,其他的在做输出时先不写条件。当输入时再添加需要的条件，一步一步来完善。

1.隐藏属性

2.关系属性

3.隐藏的关系属性

**class SkuModelSerializer**(serializers.ModelSerializer):  
 # 外键对应的隐藏属性需要明确定义  
 spu\_id = serializers.IntegerField()  
 category\_id = serializers.IntegerField()  
 # 关系属性默认输出主键，指定输出字符串  
 spu = serializers.StringRelatedField(read\_only=**True**)  
 category = serializers.StringRelatedField(read\_only=**True**)  
 # 隐藏的关系属性：specs，表示库存商品的规格选项信息  
 specs = SkuSpecSerializer(many=**True**) # 这里既输出又输入

# 明确的关系属性，会默认输出主键，自定义指定序列化器  
sku = SkuSerializer(read\_only=**True**)

# 隐藏的关系属性，在指定all时也不会输出，需要明确定义  
skus = OrderGoodsSerializer(many=**True**, read\_only=**True**)

**问题：完成规格-增加功能时，SPU 的简单查询视图，序列化器的id 一定需要写吗？**

如果去掉id,只有name.只从后端进行验证，能够提交成功。但从前端验证时，提交按钮点击没有反应。



读源码：按照执行顺序看

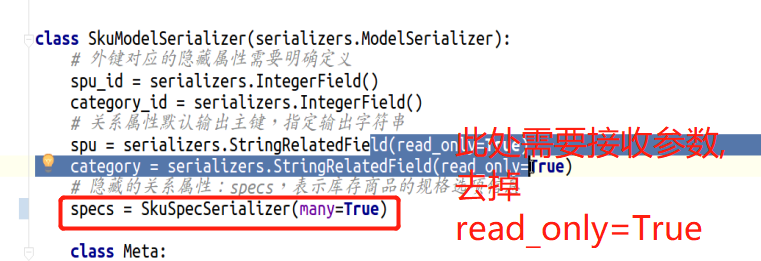
### sku

sku 表里。三级分类是冗余字段。便于查询

序列化器，一般除了用户，其他的直接继承ModelSerializer 比较方便，用户特殊因为密码要加密，需要自己重写保存方法。 create\_user()

#### 保存

对应多个规格 -保存多条-遍历



不能直接给关系属性赋值,所以此处先删掉 ,再去为剩下的赋值.[django数据库的要求]

#### sku-修改

修改对应的规格 有2个的有3个的.判断比较麻烦

**重写update:先删除 再遍历 创建**

**生成静态页面验证结果:**



#### sku-删除:

删除sku时,对应规格选项 自动被删除

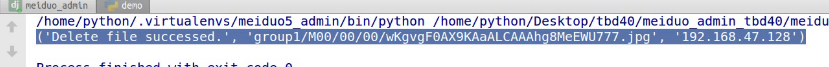
**逻辑删除怎么弄?**

**只要是删除, 请求方式不变, 就应该是delete ,处理方法就是destory . 重写destory**

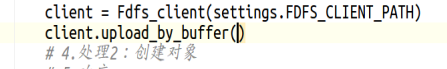
### 图片

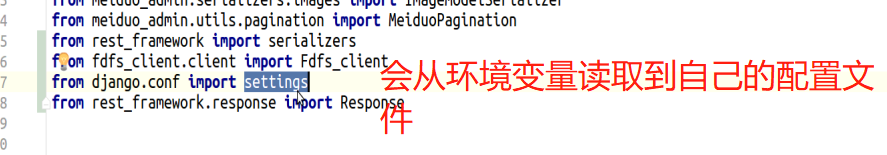
fastDFS 修改功能modify\_by\_filename()有问题,用替代方案

删除图片返回一个元组。

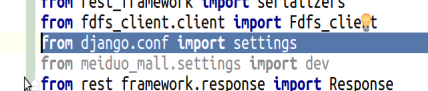


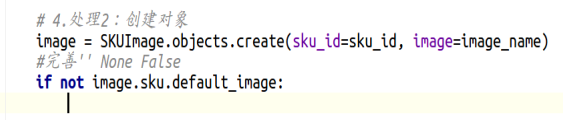
本地没有文件





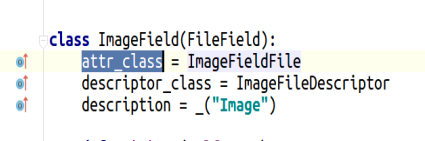
上面的灵活型号 (配置文件改名也无所谓)





**删除只能在视图中操作,因为没有用到序列化器**

在mysql还是字符串. 重写了属性.读的时候按照b这个类型. 赋值和调用有差异



#### 导包

python解释器的导包路径，默认以manage.py文件所在的路径为准。

当我们希望以某个路径导包，则将该路径加入sys.path 的列表中。

例如：

BASE\_DIR = os.path.dirname(os.path.dirname(os.path.abspath(\_\_file\_\_)))

# 指定应用的导包路径为meiduo\_mall/apps

sys.path.insert(0, os.path.join(BASE\_DIR, 'apps'))

将apps作为了导包路径。配置完之后，python解释器还是先从manage.py文件所在的路径找包，若没有，再去apps里找，若还没有，再从虚拟环境里找。

配置完后，sys.path的列表里也没有看到这个apps 的导包路径。

**商品-sku-保存**

**嵌套保存，没有重写时的异常：**

AssertionError: The `.create()` method does not support writable nested fields by default.

Write an explicit `.create()` method for serializer `meiduo\_admin.serializers.skus.SkuSerializer`, or set `read\_only=True` on nested serializer fields.

ERROR basehttp 124 "POST /meiduo\_admin/skus/ HTTP/1.1" 500 20594

前端当选中spu后，spu对应规格和选项才会显示。=》根据spu查规格，选项

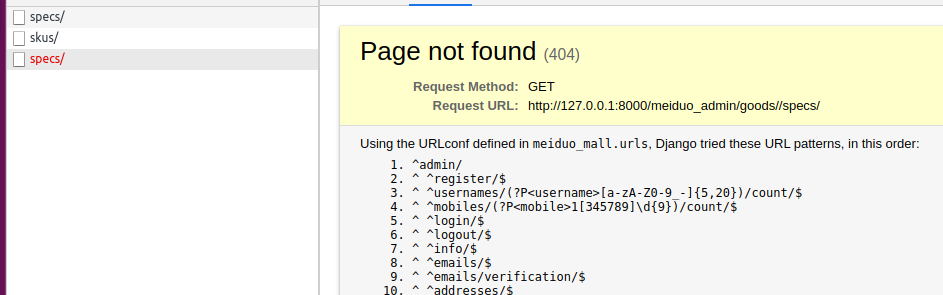


#### 重写保存和更新方法是怎么写的？【商品-sku-保存】

如果不重写，默认值保存sku,规格信息保存不上。

创建对象时,不能给关系属性赋值 创建对象，只能给关系属性的id赋值，当关系属性在序列化器中不是只读之后，创建sku时需要将其先pop 出去。

疑问：sku和规格都新增成功之后为什么spec还有个get请求，返回404 ？？【应该是前端的问题】



#### 静态化 【此时开前端，后端，celerys三个端口】

celery这部分不熟悉(l里面的代码时直接复制过来的 ，后续看)

**运行celery:**

[2020-03-04 20:55:08,781: WARNING/MainProcess] /home/lyf/.local/lib/python3.5/site-packages/celery/fixups/django.py:200: UserWarning: Using settings.DEBUG leads to a memory leak, never use this setting in production environments!

warnings.warn('Using settings.DEBUG leads to a memory leak, never '

[2020-03-04 20:55:08,781: INFO/MainProcess] celery@lyf ready.

解决：将settings中的debug改成False。

**报错：**

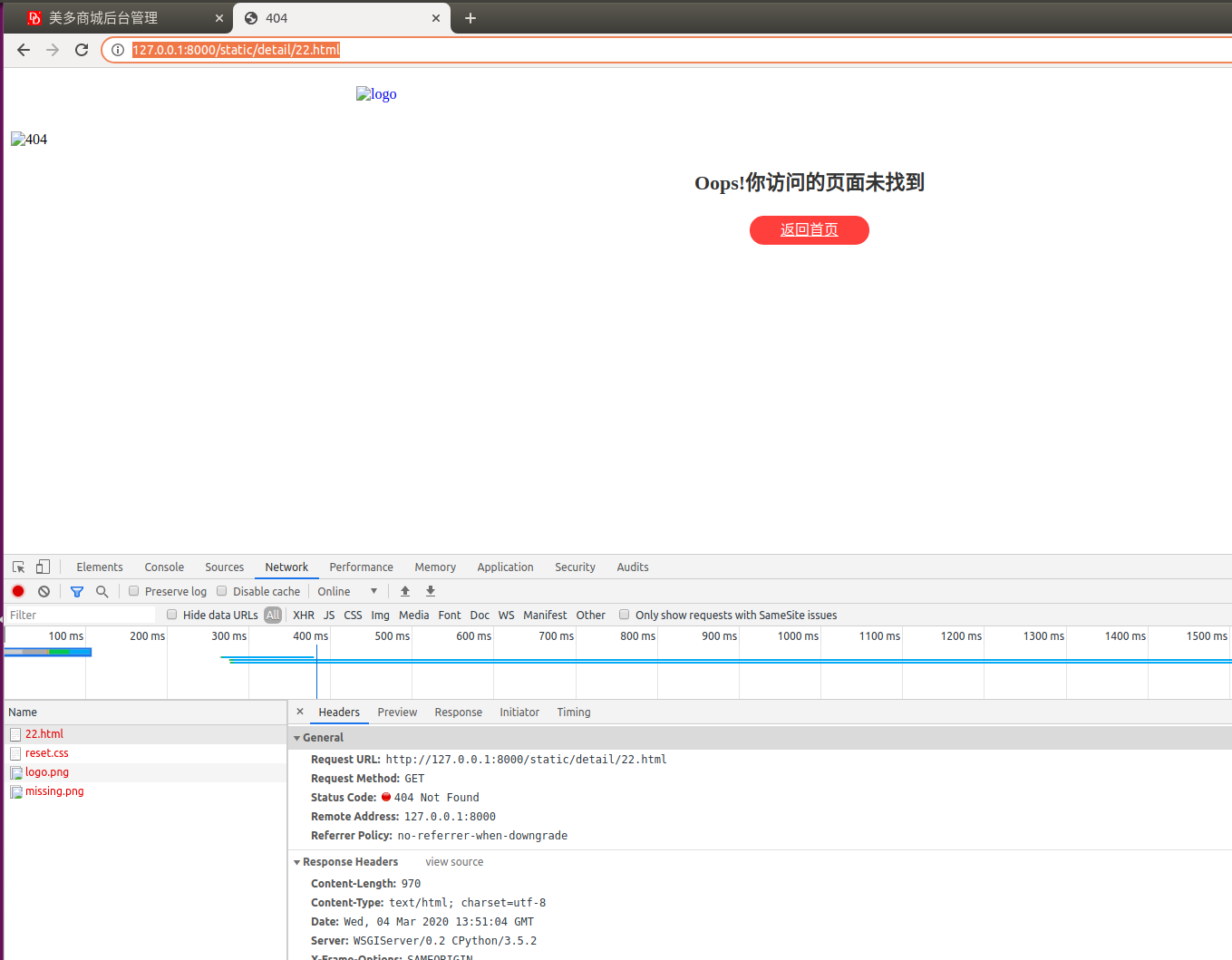
[2020-03-04 21:31:22,952: INFO/ForkPoolWorker-1] Task generate\_detail\_html[df7bf101-cbe1-4c1a-bbe6-7e1b888654b2] retry: Retry in 180s: TemplateSyntaxError("Could not parse the remainder: '()' from 'categories.values()'",)

原因：

在只运行后端时将模板改成了django自带的模板，因为静态化中间生成html所以模板改回成Jinja2.

**？？遗留问题： Python访问静态化后页面总报404**

<http://127.0.0.1:8000/static/detail/22.html>



删除的功能如果想改成逻辑删除，需要重写destory方法，只要是删除，就是使用destory方法，这是restful 规定的。

删除用不到序列化器。对于外键设置了级联删除，mysql自己就删完了sku 规格对象，不用再自己重写删除方法。

fastdfs存文件。 图片，音频，视频都可以存。

**商品-图片**

有的包比较小，没有专门的官方文档，说明这时候去pypi的网站上有简单的文档说明。

#### pypi

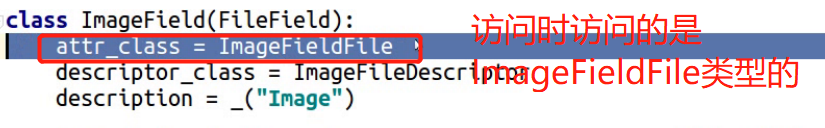
<https://pypi.org/>==》 browse projects==》[Python 3 compatible projects](https://pypi.org/search/?c=Programming+Language+%3A%3A+Python+%3A%3A+3) ,看所有包

crud都有，就用视图集。视图用什么类型只看需求，需要重写的，再去序列化里重写。（或视图中重写）

**fastdfs 代码运行报错,参见专题。**

【先 搁置图片，只完成了查询部分代码】

由于下下面这句话的作用，写和读其实是 两回事。访问时得到的对象和属性，其实是ImageFieldFile类型的。



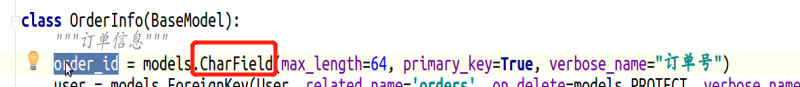
#### Viewset

视图集注册路由规则时，不能写^$,注意路由覆盖.

只要继承自试图集viewsets,就可以自动生成路由规则。

# 6.订单管理

项目中唯一自定义主键的例子



1.有修改 订单状态的需求，view为什么用ReadOnlyModelViewSet? 用到了默认的查询一个和查询多个。 修改只修改订单状态一个属性，不是默认的修改。

订单修改 【代码不熟】

因为只修改一个状态,所以重写update。

客户端不发过来所有值 只发过来 status

只返回id和status,不是对象

# 7 系统管理

权限表:迁移时会向权限表中添加数据

权限： 只需要做查询多个

用户组管理： crud都有，所以继承ModelViewSet. 保存时，序列化器自动为多对多关系属性进行保存。相当于操作了关系表。

做完group相关功能，提示如下，但功能正常。

UnorderedObjectListWarning: Pagination may yield inconsistent results with an unordered object\_list: <class 'django.contrib.auth.models.Group'> QuerySet.

paginator = self.django\_paginator\_class(queryset, page\_size)

管理员部分：【代码不熟】

管理员增加：

--group 的简单查询。 使用了动态获取序列化器的方法。 ？？get\_serialzer

--重写create方法。 密码需要加密，需要设置

问题：运行后 Django version 1.11.11, using settings 'mall.settings.dev'

怎么知道找的是dev呢？

解：在manage.py里面设置了。