个人中心

个人信息获取

1）给User添加`email_active`字段，用于记录邮箱`email`是否被激活。

2）**API接口**：获取用户个人信息。

```
请求方式和URL地址：GET /user/
前端传递参数：
    在请求头中携带jwt token。
返回值：
    {
        "id": "<用户id>",
        "username": "<用户名>",
        "mobile": "<手机号>",
        "email": "<邮箱>",
        "email_active": "<激活标记>"
    }
注：DRF中`JSONWebTokenAuthentication`认证机制会根据jwt token对用户身份进行认证，如果认证失败返回401错误，如果是权限禁止返回403错误。
```

3）使用RetrieveAPIView时，其获取单个对象时是根据pk获取，我们这里所有获取的是当前登录的用户，所以需要把`get_object`方法进行重写。

4）request对象的user属性。

对于request对象，有一个user属性。这个user属性：

a）如果用户认证成功，`request.user`是User模型类的实例对象，存放的是当前登录用户的信息。

b）如果用户未认证，`request.user`是`AnonymousUser`类的实例对象。

用户邮箱设置

1）**API接口**：设置用户个人邮箱。

```
请求方式和URL地址：PUT /email/
前端传递参数：
    1）在请求头中携带jwt token。
    2）邮箱email
返回值：
    {
        "id": "<用户id>",
        "email": "<邮箱>",
    }
```

2）处理流程:

a）接收参数并进行校验(email是否传递，格式是否正确)

b）保存用户的邮箱信息并发送激活邮件

c）返回应答，设置邮箱成功

3）发送激活邮件

a）生成激活链接：在激活链接中需要保存待激活用户的id和email，但是为了安全，需要对信息进行加密。

b）发送邮件：配置文件中先进行邮件发送配置，在使用django的send_mail方法发送邮件，为了不影响邮件设置过程，邮件采用celery发送。

```
from django.core.mail import send_mail
send_mail(subject='邮件主题', message='正文', from_email='发件人',
recipient_list='收件人列表', html_message='html邮件正文')
```

个人邮箱激活

1）**API接口**：激活用户个人邮箱。

```
请求方式和URL地址：GET /emails/verification/?token=xxx
前端传递参数：
    1）token
返回值：
    {
        "message": "处理结果"
    }
```

2）处理流程:

a）接收参数token并进行校验(token是否传递，token是否有效)

b）将用户邮箱激活标记设置为已激活。

c）返回应答，邮箱激活成功。

省市县三级信息

1）信息存储(自关联) 地区的自关联其实是一个特殊一对多的关系。

一个省包含很多个市，一个市包含很多县。

id(地区id)	name(地区名)	parent_id(父级地区ID)
320000	江苏省	NULL
320200	无锡市	320000
320282	宜兴市	320200

2）模型类自关联(地区的自关联其实是一个特殊的一对多)

```
class Area(models.Model):
    """
    行政区域
    """
    name = models.CharField(max_length=20, verbose_name='名称')
    parent = models.ForeignKey('self', on_delete=models.SET_NULL,
related_name='subs', null=True, blank=True, verbose_name='上级行政区域')
```

注：模型类自关联，ForeignKey第一个参数传`self`。

另外之前的关联查询中，有了一个area对象之后，查询关联的下级地区信息和父级地区，例如:

```
area = Area.objects.get(id='320200')
area.parent # 父级地区，由多查一，对象名.关联属性
area.area_set.all() # 子级地区，由一查多，对象名.多类名_set.all()
```

当ForeignKey创建关联属性时，指定了`related_name='subs'`之后，再查询和area对象关联的子级地区，不在使用`area.area_set.all()`，而是使用`area.subs.all()`。

3）定义导入地区信息shell脚本

```
#! /bin/bash
mysql -u'<用户名>' -p'<密码>' -h'<数据库主机IP>' '<数据库名>' < '<sql文件>'
```

4）地区视图集。

补充(客户端发送请求时携带JWT token数据)

```
axios.get(this.host + '/user/', {
    headers: {
        // 通过请求头向后端传递JWT token的方法
        'Authorization': 'JWT ' + <JWT token数据>
    },
    responseType: 'json',
})
.then(response => {
    ...
})
.catch(error => {
        ...
});
```