day07 功能

1.发布

1.1 小程序

- 选图片
- 填内容
- 提交

```
{
    cover:"https://mini-1251317460.cos.ap-chengdu.myqcloud.com/08a9daei1578736867828.png",
    content:"小程序开发太简单了",
    address:"北京市",
    topic:1,
    images:[
        "https://mini-1251317460.cos.ap-chengdu.myqcloud.com/08a9daei1578736867828.png",
        "https://mini-1251317460.cos.ap-chengdu.myqcloud.com/08a9daei1578736867828.png"
    ]
}
```

```
{
    cover:"https://mini-1251317460.cos.ap-chengdu.myqcloud.com/08a9daei1578736867828.png",
    content:"小程序开发太简单了",
    address:"北京市",
    topic:1,



    images:[
        {
            path:"https://mini-1251317460.cos.ap-chengdu.myqcloud.com/08a9daei1578736867828.png",
            cos_key:"08a9daei1578736867828.pn"
        },
        {
            path:"https://mini-1251317460.cos.ap-chengdu.myqcloud.com/08a9daei1578736867828.png",
            cos_key:"08a9daei1578736867828.pn"
        },
    ]
}
```

1.2 API

```
from rest_framework.views import APIView
from rest_framework.generics import CreateAPIView
from rest_framework import serializers
from apps.api import models
class NewsDetailModelSerializer(serializers.Serializer):
    key = serializers.CharField()
    cos_path = serializers.CharField()

class NewsModelSerializer(serializers.ModelSerializer):
    images = NewsDetailModelSerializer(many=True)
    class Meta:
        model = models.News
        fields = "__all__"

class NewsView(CreateAPIView):
    """ 创建动态的API """
    serializer_class = NewsModelSerializer
```

```
class News(models.Model):
    """
    动态
    """
    cover = models.CharField(verbose_name='封面', max_length=128)
    content = models.CharField(verbose_name='内容', max_length=255)
    topic = models.ForeignKey(verbose_name='话题', to='Topic', null=True,
blank=True)
    address = models.CharField(verbose_name='位置', max_length=128, null=True,
blank=True)

    user = models.ForeignKey(verbose_name='发布者', to='UserInfo',
related_name='news')

    favor_count = models.PositiveIntegerField(verbose_name='赞数', default=0)

    viewer_count = models.PositiveIntegerField(verbose_name='浏览数', default=0)

    comment_count = models.PositiveIntegerField(verbose_name='评论数', default=0)

    create_date = models.DateTimeField(verbose_name='创建时间',
auto_now_add=True)
```

```
class NewsDetail(models.Model):
    """
    动态详细
    """
    key = models.CharField(verbose_name='腾讯对象存储中的文件名', max_length=128,
help_text="用于以后在腾讯对象存储中删除")
    cos_path = models.CharField(verbose_name='腾讯对象存储中图片路径',
max_length=128)
    news = models.ForeignKey(verbose_name='动态', to='News')
```

1.3 规则

```
{
    k1:v1,
    k2:v2,
    k3:{...},
    k4:[
        {....}
    ]
}
```

2.restful api回顾

2.1 APIView （可以）

```python
from rest_framework.response import Response
class UserModelSerializer(serializers.ModelSerializer):
    class Meta:
        model = models.UserInfo
        fields = "__all__"

class UserView(APIView):

    def get(self,request,*args,**kwargs):
        user_list = models.UserInfo.objects.all()
        ser = UserModelSerializer(instance=user_list,many=True)
        return Response(ser.data)

    def post(self,request,*args,**kwargs):
        ser = UserModelSerializer(data=request.data)
        if ser.is_valid():
            # models.UserInfo.objects.create(**ser.validated_data)
            ser.save(user_id=1)
            return Response(ser.data)
        return Response(ser.errors)
```

2.2 ListAPIView

```
ListAPIView,CreateAPIView,RetrieveAPIView,UpdateAPIView,DestroyAPIView
```

```python
class NewTestModelSerializer(serializers.ModelSerializer):
    class Meta:
        model = models.News
        fields = "__all__"

class NewTestView(CreateAPIView,ListAPIView):
    serializer_class = NewTestModelSerializer
    queryset = models.News.objects.filter(id__gt=4)

```

2.2.1 用户传递某些值

创建用户时，自己在后台生成一个UID。

```
class NewTestModelSerializer(serializers.ModelSerializer):
    class Meta:
        model = models.News
        fields = "__all__"

class NewTestView(CreateAPIView,ListAPIView):
    serializer_class = NewTestModelSerializer
    queryset = models.News.objects.filter(id__gt=4)

    def perform_create(self, serializer):
        serializer.save(uid=str(uuid.uuid4()))
```

2.2.2 fields和exclude的区别?

通过fields和exclude定制页面展示数据。

需求：只显示用户表的id,name,age的数据，其他不显示。

```
class NewTestModelSerializer(serializers.ModelSerializer):
    class Meta:
        model = models.News
        # fields = ["id","name",'age']
        # fields = "__all__"
        exclude = ['gender']

class NewTestView(ListAPIView):
    serializer_class = NewTestModelSerializer
    queryset = models.User.objects.all()

[
    {id:1,name:'xxx',age:18},
    {id:1,name:'xxx',age:11},
    {id:1,name:'xxx',age:99},
]
```

需求：数据库有5个字段，显示7个字段。

```
class NewTestModelSerializer(serializers.ModelSerializer):
    xx = serializers.CharField(source='id')
    x1 = serializers.SerializerMethodField()
    class Meta:
        model = models.News
        # fields = "__all__"
        # fields = ['id','name','age','gender','phone','xx','x1']
        exclude = ['id','name']

    def get_x1(self,obj):
        return obj.id

class NewTestView(ListAPIView):
    serializer_class = NewTestModelSerializer
    queryset = models.User.objects.all()

[
    {id:1,name:'xxx',age:18...   xx:1,x1:1},
    {id:2,name:'xxx',age:11...   xx:2,x1:2},
    {id:3,name:'xxx',age:99,     xx:3,x1:3},
]
```

2.2.3 read_only

添加时不要，查看时候需要。

需求：编写两个接口 添加（3字段）、获取列表（5个字段）

```
class NewTestModelSerializer(serializers.ModelSerializer):
    # phone = serializers.CharField(source='phone',read_only=True)
    # email = serializers.CharField(source='email',read_only=True)
    class Meta:
        model = models.News
        fields = "__all__"
        read_only_fields = ['phone','email',]

class NewTestView(CreateAPIView, ListAPIView):
    serializer_class = NewTestModelSerializer
    queryset = models.User.objects.all()

添加：
    {
        name:'xx',
        age:'19',
        gender:1
    }
获取：
    [
        {name:'xx',age:'xx',gender:'',phone:'xx',email:xxx}
    ]
```

2.3.4 复杂需求

添加时用一个serializers、列表时用一个serializers

```
class NewTestModelSerializer1(serializers.ModelSerializer):
    class Meta:
        model = models.News
        fields = "__all__"

class NewTestModelSerializer2(serializers.ModelSerializer):
    class Meta:
        model = models.News
        fields = "__all__"

class NewTestView(CreateAPIView, ListAPIView):
    queryset = models.User.objects.all()

    def get_serializer_class(self):
        if self.request.method == 'POST':
            return NewTestModelSerializer1
        if self.request.method == 'GET':
            return NewTestModelSerializer2
```

2.3.5 serializers嵌套

```python
class CreateNewsTopicModelSerializer(serializers.Serializer):
    key = serializers.CharField()
    cos_path = serializers.CharField()


class CreateNewsModelSerializer(serializers.ModelSerializer):
    imageList = CreateNewsTopicModelSerializer(many=True)

    class Meta:
        model = models.News
        exclude = ['user', 'viewer_count', 'comment_count',"favor_count"]

    def create(self, validated_data):
        # 把imageList切走
        image_list = validated_data.pop('imageList')

        # 创建New表中的数据
        news_object = models.News.objects.create(**validated_data)

        data_list = models.NewsDetail.objects.bulk_create(
            [models.NewsDetail(**info, news=news_object) for info in image_list]
        )
        news_object.imageList = data_list

        if news_object.topic:
            news_object.topic.count += 1
            news_object.save()

        return news_object

class NewsView(CreateAPIView):
    """
    发布动态
    """
    serializer_class = CreateNewsModelSerializer
    def perform_create(self, serializer):
        # 只能保存：News表中的数据（）
        # 调用serializer对象的save（先调用create）
        new_object = serializer.save(user_id=1)
        return new_object
```

3. 首页展示

- 小程序
    - 初始化
    - 下拉刷新
    - 上翻页
    - 瀑布流
- 后端API
    - APIView
    - ListAPIView

        - filter
        - pagination

扩展：分页的优化

> 记录最大值和最小值，防止切片全部数据扫描的问题。

4.详细页面（3点）

- 写脚本构造数据
- 最近的访客
- 一级评论

作业

1. 赞文章
2. 赞评论
3. 关注
4. 访问记录

    > 进入详细页面时，先判断用户是否已经登录。
    > 未登录，不操作。
    > 登录：添加到访问记录中。