后端反馈

关于错误信息

- 某个对象不存在有个优先级问题，如果先发现一个东西不存在，就不会报另一个不存在的错误了
- parameter error 我整成更细致的错误信息了，比如输入太长了太短了，或者是提示某个东西需要输入然后"msg"可能会有多个错误信息，每条信息对应一个参数的错误信息，例如 "msg": { "requestId": ["Enter a whole number."] } requestId是前端发过来的json中的参数名，如果没有问题那么"msg"中就没有

url

下面是每个请求对应的url，前端需要调用某个api的时候只需要发送 localhost:8000/commentarea/对应API的英文 get_comment_area 获取讨论区内容 post_long_comment 发送长评 post_short_comment 发送短评 post_short_comment_for_long_comment 为长评写短评 rose_comment 点赞评论 egg_comment 点踩评论 star_comment 收藏长评 star_comment_area 收藏讨论区 request_create_comment_area 提交申请 approve_create_comment_area_request 批准创建讨论区 get_create_comment_area_request 获取创建讨论区请求 （这个还没完善，因为这个需要用cookie判断权限，现在不管是谁要都会给他返回全部请求信息） 顺便说一下，现在那些前端需要发送userId的尤其是get请求，把userId放到url里面（即明文传输）的都是很不安全的，但是我大概还是按照要求写了，之后登录系统完成之后，应该需要使用cookie来判断发出请求的用户是谁 get_star_comment_area_list 获取讨论区列表（用户收藏的） get_short_comment 获取短评 虽然前端思路不用rose_user_list（点赞用户列表，但是我还是返回给你了，bool变量在后面也加上了） get_long_comment 获取长评

关于Response的格式

例如response中包含一个CommentArea，意思就是发送了一个dict其中有CommentArea的每个属性 对于ManyToManyField（即对象之间的多对多关系，比如用户和讨论区的收藏与被收藏关系），对应的条目是一个包含所有关联对象id的列表 下面是一个后端返回的CommentArea的例子 { "code": 200, "data": { "msg": "success", "comment_area": { "id": 2, "name": "Prime in P", "master": 1, "paper": 2, "long_comment_list": [1, 2], "short_comment_list": [], "star_user_list": [1, 3, 4] } } } 可以看到star_user_list中有1,3,4就意味着id为1,3,4的用户收藏了这个讨论区

前端原有信息

这部分我做了一些修改

相关数据结构

这部分我直接搬运python代码了，后面的参数对前端来说不重要，你只要知道不同的属性的名字就行了 上面那个例子中的commentarea的属性的名字就是用的我在下面列出的Python类的属性名

```
class Paper(models.Model):
    title = models.CharField(max_length=255)
    path = models.CharField(max_length=255)
```

```
class ShortComment(models.Model):
    poster = models.ForeignKey('user.User', on_delete=models.CASCADE,
related_name = 'post_short_comment')
    post_time = models.DateTimeField(auto_now_add=True)
    content = models.CharField(max_length=255)
    rose_number = models.IntegerField(default=0)
    egg_number = models.IntegerField(default=0)
    rose_user_list = models.ManyToManyField('user.User',
related_name='rose_short_comment')
    egg_user_list = models.ManyToManyField('user.User',
related_name='egg_short_comment')

class LongComment(models.Model):
    poster = models.ForeignKey('user.User', on_delete=models.CASCADE,
related_name='post_long_comment')
    post_time = models.DateTimeField(auto_now_add=True)
    title = models.CharField(max_length = 255)
    star_number = models.IntegerField(default=0)
    star_user_list = models.ManyToManyField('user.User',
related_name='star_long_comment')
    content = models.TextField()
    short_comment_list = models.ManyToManyField(ShortComment)

class CreateRequest(models.Model):
    requestor = models.ForeignKey('user.User', on_delete=models.CASCADE)
    paper = models.ForeignKey(Paper, on_delete=models.CASCADE)

class CommentArea(models.Model):
    name = models.CharField(max_length=255)
    master = models.ForeignKey('user.User', on_delete=models.CASCADE)
    paper = models.ForeignKey(Paper, on_delete=models.CASCADE)
    long_comment_list = models.ManyToManyField(LongComment)
    short_comment_list = models.ManyToManyField(ShortComment)
    star_user_list = models.ManyToManyField('user.User',
related_name='star_comment_area')
```

Codes

```
CODE = {
    "success": 200,
    "database_error": 300,
    "user_error": 400,
    "method_error": 600,
    "parameter_error": 700,
}
```

Response Body Fundamental

CommentAreaPage

获取讨论区内容

```
// 获取讨论区内容
// userId代表的用户获取commentAreaId代表的讨论区内容
request.body = {
    "commentAreaId": number,
}

// 成功
response.body = {
    "code": 200,
    "data": {
        "msg": "success",
            "comment_area": CommentArea,
    }
}

// 失败
// id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "comment area id does not exist"
    }
}
// 请求格式错误
response.body = {
    "code": 300,
    "data": {
        "msg": "parameter error"
    }
}
```

点赞评论

```
// 点赞评论
// userId代表的用户赞shortCommentId代表的评论
request.body = {
    "shortCommentId": number,
    "userId": number,
}

// 成功
response.body = {
    "code": 200,
    "data": {
        "msg": "success"
```

```
    }
}

// 失败
// short comment id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "short comment id does not exist"
    }
}
// user id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "user id does not exist"
    }
}
// 请求格式错误
response.body = {
    "code": 300,
    "data": {
        "msg": "parameter error"
    }
}
```

点踩评论

```
// 点踩评论
// userId代表的用户踩shortCommentId代表的评论
request.body = {
    "shortCommentId": number,
    "userId": number,
}

// 成功
response.body = {
    "code": 200,
    "data": {
        "msg": "success"
    }
}

// 失败
// short comment id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "short comment id does not exist"
    }
}
```

```
// user id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "user id does not exist"
    }
}
// 请求格式错误
response.body = {
    "code": 300,
    "data": {
        "msg": "parameter error"
    }
}
```

收藏长评

```
// 收藏长评
// userId代表的用户收藏longCommentId代表的长评
request.body = {
    "longCommentId": number,
    "userId": number,
}

// 成功
response.body = {
    "code": 200,
    "data": {
        "msg": "success"
    }
}

// 失败
// long comment id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "long comment id does not exist"
    }
}
// user id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "user id does not exist"
    }
}
// 请求格式错误
response.body = {
    "code": 300,
    "data": {
```

```
        "msg": "parameter error"
    }
}
```

收藏论文(收藏讨论区)

```
// 收藏论文
// userId代表的用户收藏paperId代表的论文
response.body = {
    "commentAreaId": number,
    "userId": number,
}

// 成功
response.body = {
    "code": 200,
    "data": {
        "msg": "success"
    }
}

// 失败
// paper id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "paper id does not exist"
    }
}
// user id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "user id does not exist"
    }
}
// 请求格式错误
response.body = {
    "code": 300,
    "data": {
        "msg": "parameter error"
    }
}
```

ApproveCreateCommentAreaPage

获取创建讨论区请求

```
// 获取创建讨论区请求
request.body = {
}
** 这里应该用cookie来实现，不能直接发用户id，目前没有cookie,你只需要向对应url发送请求即可**

// 成功
response.body = {
    "code": 200,
    "data": {
        "msg": "success",
        "createRequestList": List<CreateRequest>,
    }
}

// 失败
// user id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "user id does not exist"
    }
}
// 当前用户不是管理员
response.body = {
    "code": 300,
    "data": {
        "msg": "not administrator"
    }
}
// 请求格式错误
response.body = {
    "code": 300,
    "data": {
        "msg": "parameter error"
    }
}
```

批准创建讨论区

```
// 批准创建讨论区
request.body = {
    ** 这里感觉不需要发用户id，因为如果用这个发过来的id检测权限，那太容易伪装成管理员了,我觉得应该用当前用户的cookie来检测 **
    "requestId": number,
}

// 成功
response.body = {
    "code": 200,
```

```
    "data": {
        "msg": "success",
    }
}

// 失败
// user id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "user id does not exist"
    }
}
// request id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "request id does not exist"
    }
}
// 当前用户不是管理员
response.body = {
    "code": 300,
    "data": {
        "msg": "not administrator"
    }
}
// 请求格式错误
response.body = {
    "code": 300,
    "data": {
        "msg": "parameter error"
    }
}
```

LongComment

获取短评

```
// 获取短评
// 获取shortCommentId代表的长评
// 需要UserID因为后端要判断有没有点过赞
request.body = {
      "shortCommentId": number,
      "userId", number
}

// 成功
response.body = {
    "code": 200,
    "data": {
```

```
        "msg": "success",
        "comment": ShortComment,
        "rose": boolean,
        "egg": boolean
    }
}

// 失败
// short comment id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "short comment id does not exist"
    }
}
// 请求格式错误
response.body = {
    "code": 300,
    "data": {
        "msg": "parameter error"
    }
}
```

获取长评

```
// 获取长评
// 获取longCommentId代表的长评
// 需要用户id因为需要判断是否收藏过
request.body = {
    "userId", number,
    "longCommentId": number,
}


// 成功
response.body = {
    "code": 200,
    "data": {
        "msg": "success",
        "comment": LongComment,
        "star": boolean
    }
}

// 失败
// long comment id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "long comment id does not exist"
    }
```

```
}
// 请求格式错误
response.body = {
    "code": 300,
    "data": {
        "msg": "parameter error"
    }
}
```

发送短评

```
// 发送短评
// userId代表的用户在paperId代表的论文发送内容为content的短评
response.body = {
    "paperId": number,
    "userId": number,
    "content": string,
}

// 成功
response.body = {
    "code": 200,
    "data": {
        "msg": "success"
    }
}

// 失败
// paper id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "paper id does not exist"
    }
}
// user id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "user id does not exist"
    }
}
// 请求格式错误
response.body = {
    "code": 300,
    "data": {
        "msg": "parameter error"
    }
}
```

发送长评

暂时和发送长评一样?

```
// 发送长评
// userId代表的用户在paperId代表的论文发送内容为content的长评
response.body = {
    "commentAreaId": number,
    "userId": number,
    "content": string,
    "title": string,
}

// 成功
response.body = {
    "code": 200,
    "data": {
        "msg": "success"
    }
}

// 失败
// paper id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "paper id does not exist"
    }
}
// user id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "user id does not exist"
    }
}
// 请求格式错误
response.body = {
    "code": 300,
    "data": {
        "msg": "parameter error"
    }
}
```

收藏长评

```
// 收藏长评
// userId代表的用户收藏longCommentId代表的长评
request.body = {
    "longCommentId": number,
```

```
    "userId": number,
}

// 成功
response.body = {
    "code": 200,
    "data": {
        "msg": "success"
    }
}

// 失败
// long comment id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "long comment id does not exist"
    }
}
// user id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "user id does not exist"
    }
}
// 请求格式错误
response.body = {
    "code": 300,
    "data": {
        "msg": "parameter error"
    }
}
```

为长评写短评

```
// 为长评写短评
// userId代表的用户将shortComment代表的短评发送到longCommentId代表的长评下

request.body = {
    "longCommentId": number,
    "shortComment": ShortComment,
    "userId": number,
}

// 成功
response.body = {
    "code": 200,
    "data": {
        "msg": "success"
    }
```

```
}

// 失败
// long comment id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "long comment id does not exist"
    }
}
// user id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "user id does not exist"
    }
}
// 请求格式错误
response.body = {
    "code": 300,
    "data": {
        "msg": "parameter error"
    }
}
```

RequestCreateCommentAreaPage

提交申请

```
// 提交申请
// userId代表的用户提交paperPdfInStr代表的论文
request.body = {
    "userId": number,
      "paperPdfInStr": string,
      "paperTitle": string,
}
// 成功
response.body = {
    "code": 200,
    "data": {
        "msg": "success"
    }
}

// 失败
// user id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "user id does not exist"
    }
```

```
}
// 请求格式错误
response.body = {
    "code": 300,
    "data": {
        "msg": "parameter error"
    }
}
```

MyCommentAreaListPage

获取讨论区列表

```
// 获取评论区列表
// 获取userId的评论区列表
request.body = {
    "userId": number,
}

// 成功
response.body = {
    "code": 200,
    "data": {
        "msg": "success",
        "commentAreaList": List<CommentArea>
    }
}

// 失败
// user id不存在
response.body = {
    "code": 300,
    "data": {
        "msg": "user id does not exist"
    }
}
// 请求格式错误
response.body = {
    "code": 300,
    "data": {
        "msg": "parameter error"
    }
}
```