api 文档

错误码

- 200:成功
- 400:bad arguments，参数格式不对
- 401:Invalid credentials，jwt校验失败，未登录
- 404:not found
- 405:Method not allowed
- 500:Internal Server Error

User

1. 登录

发送code，后端处理后发送给微信API，返回openid

后端处理后返回给前端jwt，之后前端发请求需要在header里加上Authorization：jwt

```
Method: PATCH
URL: /api/user/login
Request:
{
    'code': '' # wx.login()获取的code
}
Response:
{
    'jwt': '',
    'new':True, # 是否是新用户
    'nickname': '',
    'age':'',
    'addr':'',
    'gender':'',
    'phone':'',
    'intro':'',
    'avatarUrl':'',    # 头像 url
    'updated':'',      # 资料上次更新时间
    'message':'ok'
}

```

2. 获取个人信息

```
Method: GET
URL: /api/user/user
# header需要加Authorization: jwt
Response:
{
    'openid': '',
    'nickname': '',
    'jwt': '',
    'age':'',
    'addr':'',
    'gender':'',
    'phone':'',
    'intro':'',
    'avatarUrl':'',#url
    'updated':''
}

```

3. 编辑个人资料

```
Method: PATCH
URL: /api/user/edit_info
# header需要加Authorization: jwt
Request:
{
    'avatarUrl':'',    # 头像url
    'nickname':'',
    'gender':'', # 男 or 女
    'phone':'',
    'intro':'',
    'addr':'',
    'age':'',
}
Response:
{
    'avatarUrl':'',
    'updated':'',
    'message':'ok'/'Invalid arguments:xxx'
}
```

4. 上传头像

用wx.uploadfile上传

```
Method: POST
URL: /api/user/avatar
# header需要加Authorization：jwt
Request:
{
    filePath: avatarUrl, # 要上传的文件路径
    name: 'file', # 后端接收文件的字段名
}
Response:
{
    'avatarUrl':'',
    'updated':'',
    'message':'ok'/'Invalid arguments:xxx'
}
```

5. 获取天气

```
Method: GET
URL: /api/user/weather
# header需要加Authorization：jwt
Response:
{
    'icon':'',
    'text':'',
    'temperature':''
}
```

6. 通过wx.uploadfile上传头像

```
Method: GET
URL: /api/user/weather
Request:
{
    filename设置为‘file'
}
Response:
{
    "avatarUrl":user.avatarUrl,
    "updated": user.updated
}
```

Closet

1. 添加衣服

```
Method: POST
URL: /api/closet/add_clothes
Request:
{
    'clothes_name':''
    'clothes_main_type':''
    'clothes_detail_type':''
}
Response:
{
    'clothes_ID':''
    'message':'Clothes added to the closet successfully'
}

```

2. 编辑衣服（）

```
Method: POST
URL: /api/closet/edit_clothes
Request:
{
    'clothes_ID':''
    'clothes_name':''
    'clothes_main_type':''
    'clothes_detail_type':''
}
Response:
{
    'message':'Clothes updated successfully'
}

```

3. 删除衣服（）

```
Method: POST
URL: /api/closet/delete_clothes
Request:
```

```
{
    'clothes_ID':''
}
Response:
{
    'message':'Clothes deleted successfully'
}

```

4. 显示衣服列表（）

```
Method: GET
URL: /api/closet/get_clothes
Response:
{
    'clothes':[
        {'id':''
        'name':''
        'Mtype':''
        'Dtyoe':''
        'pictureUrl':''},
        ...],
     'message':'ok'
}

```

5. 添加穿搭衣服（）

```
Method: POST
URL: /api/closet/add_outfit
Request:
{
    'id':
}
Response:
{
    'clothes':[
        {'id':''
        'name':''
        'Mtype':''
        'Dtyoe':''
        'pictureUrl':''},
        ...],
    'message':'ok'/'clothes not found'
}
```

6. 删除穿搭衣服（）

```
Method: POST
URL: /api/closet/remove_outfit
Request:
{
    'id':
}
Response:
{
    'clothes':[
        {'id':''
        'name':''
        'Mtype':''
        'Dtyoe':''
        'pictureUrl':''},
        ...],
    'message':'ok'/'clothes not found'
}
```

7. 对当前穿搭评分（）

```
Method: POST
URL: /api/closet/score
Response:
{
    'rate':'',
    'have_better:'1'/'0',
     # 以下have_better=1则有，否则为空
    'best_score:'',
    'replace:[
        {'id':''
        'name':''
        'Mtype':''
        'Dtyoe':''
        'pictureUrl':''},
        ...],
    'message':'ok'/'no outfit'
}
```

8. 获取最新穿搭（）

```
Method: GET
URL: /api/closet/get_outfit
Response:
{
    'clothes':[
        {'id':''
        'name':''
        'Mtype':''
        'Dtyoe':''
        'pictureUrl':''},
        ...],
    'message':'ok'/'not found'
}
```

9. 生成推荐穿搭（）

```
Method: POST
URL: /api/closet/generate
Response:
{
    'rate':'',
    'clothes':[
        {'id':''
        'name':''
        'Mtype':''
        'Dtyoe':''
        'pictureUrl':''},
        ...],
    'message':'ok'
}
```

10. 替换成推荐搭配（）

```
Method: POST
URL: /api/closet/replace
Response:
{
    'rate':'',
    'clothes':[
        {'id':''
        'name':''
        'Mtype':''
        'Dtyoe':''
        'pictureUrl':''},
        ...],
    'message':'ok'/'outfit not found'
}
```

11. 获取穿搭评分（）

```
Method: GET
URL: /api/closet/get_score
Request:
{
    'index':
}
Response:
{
    'dates':'',
    'ratings':'',
    'message':'ok'/'outfit not found'/'Method not allowed'
}
```

12. 获取最喜欢上衣（）

```
Method: GET
URL: /api/closet/get_favorite_clothes
Request:
{
    'index':
}
Response:
{
    'clothes_list':'',
    'count_list':'',
    'message':'ok'/'outfit not found'/'Method not allowed'
}
```

13. 获取用户衣服总数（）

404:Clothes not found

```
Method: GET
URL: /api/closet/get_clothes_count
Request:
{
    'index':
}
Response:
{
    "clothes_count": clothes_count,
    "message": "ok"
}
```

Post

以下 api 都需要在用户登录状态下调用，即需要在 header 中填写 Authorazation 字段

1. 创建贴文

```
Method: POST
URL: /api/posting/create_post
Request:
{
    "title":"",
    "content":"",
    "show_rate":"",
    "show_weather":""
}
Response:
{
    "post":{
        title
        content
        images
        rate
        weather
        create_time
        user
        likes
        favorites
        show_rate
        show_weather
    }
  }
```

2. 上传帖子图片

```
Method: POST
URL: /api/posting/upload_image/<int:post_id>/
Request:
{
    filename设置为‘file'
}
Response:
{
    "message": "Image uploaded successfully"
}
```

3. 贴文详情

```
Method: GET
URL: /api/posting/post_detail/id=<post_id>
Response:
{
    "post": post_serializer.data,
    "comments": comments_serializer.data,
    "is_liked":is_liked,
    "is_fav":is_fav
}
```

4. 评论

```
Method: POST
URL: /api/posting/comment_post/<int:post_id>/
Request:
{
    "content"
}
Response:
{
   "comment":[]
}
```

5. 点赞

```
Method: POST
URL: /api/posting/like_post/<int:post_id>/
Response:
{
   "message": "Success"
}
```

6. 收藏

```
Method: POST
URL: /api/posting/favorite_post/<int:post_id>/
Response:
{
   "message": "Success"
}
```

7. 获取贴文列表

```
hod: GET
URL: /api/posting/user_posts/<post_type>
Request: {}    # 'post_type' 可为 'create'/'favorite'/'all'
Response:
{
    'posts': PostForm[]    # 贴文模型列表的 Json
    'post_type':
}
```

8. 获取用户统计信息

```
hod: GET
URL: /api/posting/user_post_info
Response:
{
   "favorites":
   "posts":
   "likes":
}
```