达内-博客项目-用户模块API说明文档

[Python-教研部]

目录

[一， 概述： 1](#_Toc17517_WPSOffice_Level1)

[二， 事件定义： 1](#_Toc13432_WPSOffice_Level1)

[三， 开发规范: 1](#_Toc13973_WPSOffice_Level1)

[四， 数据库结构： 2](#_Toc10395_WPSOffice_Level1)

[五， 接口说明： 3](#_Toc28262_WPSOffice_Level1)

[六， 常见问题： 9](#_Toc31397_WPSOffice_Level1)

# 概述：

随着近期公司内部个人兴趣社区活动的如火如荼，为满足广大社区爱好者追求自我的梦想，特此开发内部博客；方便大家整理及自己社区活动中的一些优质内容，增加社区的互动性。此文档为用户模块的API详情。

# 事件定义：

1. 注册 - 新用户可通过此功能注册为内部论坛的注册用户。每个注册用户可申请一个自己独立的博客空间，只有注册用户方可在自己的博客中发表自己的博客及在博客内容上留言及回复他人的留言。
2. 登陆 - 用户可通过此功能进行用户登陆操作，只有登陆的用户方可执行会员的相关操作-如发布博客及留言功能。
3. 修改个人信息 - 每个注册用户都有 个人描述，个人签名及头像和昵称 可供自定义编写；用户需要进行登陆后，方可使用该功能。

# 开发规范:

1. 后端环境

Python 3.7.3 + django 1.11.8 + mysql 5.5 + Ubuntu19.04 + vim

1. 通信协议

http

1. 通信格式

json

1. API规范

一定程度上符合RESTful 定义

# 数据库结构：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 作用 | 备注1 | 备注2 |
| username | varchar(11) | 用户名 | 注册时填写的用户名，不可修改 | 主键 |
| nickname | varchar(30) | 昵称 | 在博客中显示的名字，可修改 | 无 |
| email | varchar(50) | 邮箱 | 预留 | 无 |
| phone | varchar(11) | 手机号 | 无 | 无 |
| password | varchar(32) | 密码 | 用户密码，已散列存储 | 无 |
| sign | varchar(50) | 个人签名 | 无 | 无 |
| info | varchar(150) | 个人描述 | 无 | 无 |
| avatar | varchar(100) | 头像 | 无 | 无 |
| created\_time | datetime | 创建时间 | 无 | 无 |
| updated\_time | datetime | 更新时间 | 无 | 无 |

# 接口说明：

## 1 注册接口

URL : [http://127.0.0.1:8000/v1/users](http://127.0.0.1/users)

### 请求方式

POST

### 请求格式

json 具体参数如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| username | 用户名 | char | 必填 |
| email | 用户邮箱 | char | 必填 |
| password1 | 第一次输入的密码 | char | 必填 |
| password2 | 第二次输入的密码 | char | 必填 |

请求示例：

{‘username’: jack, ‘email’: [‘abc@qq.com’,](mailto:‘abc@qq.com’,) ‘password1’: ‘abcdef’, ‘password2’: ‘abcdef’}

### 1.3响应格式

json 具体参数如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| code | 状态 | int | 默认正常为200，异常请见1.4 |
| username | 用户名 | char | 无 |
| data | 返回具体的数据都包含在data中 | {} | {  token: xxxx 此为会话保持用的令牌-char  } |

响应示例：

{‘code’: 200 , ‘username’: ‘abc’, ’data’: {‘token’: ‘asdadasd.cvreijvd.dasdadad’} }

### 1.4异常码

|  |  |  |
| --- | --- | --- |
| 异常码 | 含义 | 备注 |
| 202 | 请求中无内容 |  |
| 203 | 请求中未提交用户名 |  |
| 204 | 请求中未提交邮箱 |  |
| 205 | 请求中未提交密码 |  |
| 206 | 两次提交的密码不一致 | 即password1 != password2 |
| 207 | 用户名已存在 |  |
| 500 | 服务器异常 |  |

异常响应示例

{'code':203, 'error':'请输入用户名'}

## 2 获取用户数据接口

URL: [http://127.0.0.1:8000/v1/users/<username>](http://127.0.0.1/users/%3cusername%3e)

### 2.1 请求方式

GET

### 2.2 请求格式

2.2.1 直接GET请求，可获取全量数据

2.2.2 GET请求后添加查询字符串，可根据具体查询字符串查询：[http://127.0.0.1:8000/v1/users/<username>](http://127.0.0.1/users/%3cusername%3e)?nickname=1

### 2.3 响应格式

json 具体参数如下

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| code | 状态 | int | 请求成功，code为200，异常码请见2.4 |
| username | 用户名 | char | 此次欲获取的用户的用户名 |
| data | 返回的具体数据均在data里 | {} | {‘info’:个人描述-char，’sign’:个人签名-char, ‘nickname’:昵称-char, ‘avatar’: 头像地址-char} |
|  |  |  |  |

响应示例：

全量响应：

{‘code’:200,‘username’:’xiaoming’,‘data’:{‘nickname’:’abc’, ’sign’:’hellow’, ‘avatar’: ’abc.jpg’, ‘info’: ‘hahahahah’}}

局部响应（GET请求添加查询字符串- 以下为查询nickname的返回）:

{‘code’:200, ‘username’:’123’, ‘data’:{‘nickname’:’abcde’} }

### 2.4 异常码

|  |  |  |
| --- | --- | --- |
| 异常码 | 含义 | 备注 |
| 208 | 用户名不存在 |  |
|  |  |  |

## 3 修改用户个人信息接口

URL: [http://127.0.0.1:8000/v1/users](http://127.0.0.1/users/%3cusername%3e)/<username>

### 3.1 请求方式

PUT

### 3.2 请求格式

json 具体参数如下

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| sign | 个人签名 | char | 非必填 |
| info | 个人描述 | char | 非必填 |
| nickname | 昵称 | char | 非必填 |

该请求需客户端在HTTP header 里添加 token, 格式如下：

Authorization ： token

请求示例：{‘sign’:xxx, ‘info’:xxxx, ‘nickname’:xxxx}

### 3.3 响应格式

json , 具体参数如下

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| code | 状态 | Int | 正常为200，异常码见3.4 |
| username | 用户名 | char | 此次请求的用户名 |

响应示例：{‘code’:200, ‘username’:’char’}

### 3.4 异常码

|  |  |  |
| --- | --- | --- |
| 异常码 | 含义 | 备注 |
| 202 | 空提交 | 请求中无任何信息 |

异常响应示例:

{‘code’:202, ‘error’: ‘please put data’}

## 4 上传头像接口

URL:[http://127.0.0.1:8000/v1/users](http://127.0.0.1/users/%3cusername%3e)/<username>/avatar

### 4.1 请求方式

POST multipart/form-data

### 4.2 请求格式

表单 具体表单参数如下

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| avatar | 表单中的图片 | char | 必填 |

该请求需客户端在HTTP header 里添加 token, 格式如下：

Authorization ： token

### 4.3 响应格式

json, 具体参数如下

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| code | 状态 | Int | 正常为200 |
| username | 用户名 | char | 此次修改的用户名 |

## 5 获取token

【即登陆请求】 http://127.0.0.1:8000/v1/tokens

### 5.1 请求方式

POST

### 5.2 请求格式

json , 具体参数如下

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| username | 用户名 | char | 必填 |
| password | 密码 | char | 必填 |

请求示例： {‘username’: ‘xxx’, ‘password’: ‘yyy’}

### 5.3 响应格式

json, 具体参数如下

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| code | 200 | Int | 正常为200，异常码请见5.4 |
| username | 用户名 | char |  |
| data | 返回具体数据都在data中 | {} | {‘token’: ‘xxxx’-char} |

响应示例：

{‘code’: 200, ‘username’: ‘asc’, ‘data’: {‘token’: ‘zdsadasd’}}

### 5.4 异常码

|  |  |  |
| --- | --- | --- |
| 异常码 | 含义 | 备注 |
| 201 | 请求方式并非POST |  |
| 202 | 请求为空 |  |
| 203 | 请求中未提交用户名 |  |
| 205 | 请求中未提交密码 |  |
| 208 | 用户名不存在 |  |
| 209 | 提交的密码不正确 |  |

异常码响应示例

{‘code’: 205, ‘error’: ‘no password’}

# 常见问题：

1，注意个别客户端请求需要添加 token 传回服务器端，否则异常