1. 注册登录接口

1.1 jwt 模块的使用

```python

import jwt
from jwt import PyJWTError

# 加密:
# 1. 准备要加密的数据字典
#     需要包含 exp 字段, 这个字段的值是 datetime 对象, 是指过期的时间
payload = {
    'user_id': 100,
    'exp': datetime.now() + timedelta(days=7)
}
# 2. 准备加密的秘钥
key = 'N1UzXOFKJRZk5cslhMDcbqHZ0lKvCAyL85fufewVmUF9bGPAlAXw9w=='
# 3. 选择加密的算法 一般是 HS256
algorithm = 'HS256'
# 4. 生层加密的 token: jwt.encode(要加密的数据字典, key=加密的秘钥, algorithm=加密的算法)
#    注意: 这里返回的 token 是字节码类型
token = jwt.encode(payload, key=key, algorithm=algorithm)
print(token)

# 解密:
# 通过 jwt.decode(要解密的 token,key=加密的秘钥, algorithm=加密的算法)
# 1. 如果 token 合法那么返回数据字典
# 2. 如果 token 不合法抛出 jwt.PyJWTError 异常
token='xxxx'
try:
    res = jwt.decode(token, key=key, algorithm=algorithm)
    print(res)
except PyJWTError:
```

1.2 集成到项目中

```
# 1. 将加密解密的 秘钥，算法，token 有效期配置到项目中
# 2. 封装加密过程为单独函数
# 3. 封装解密过程为单独函数
```

1.3 自定义手机号码校验函数

```python
# 自定义手机号码校验函数
# 1. 定义函数，接受一个参数
# 2. 函数中校验该参数
#      如果校验失败抛出 ValueError(错误消息)
#      如果校验成功，返回数据
def parse_mobile(mobile_str):
    if re.match(r'1[3-9]\d{9}', mobile_str):
        return mobile_str
    else:
        raise ValueError('手机号码格式不正确')
```

1.4 接口代码实现

2. 获取用户信息接口

2.1 请求钩子

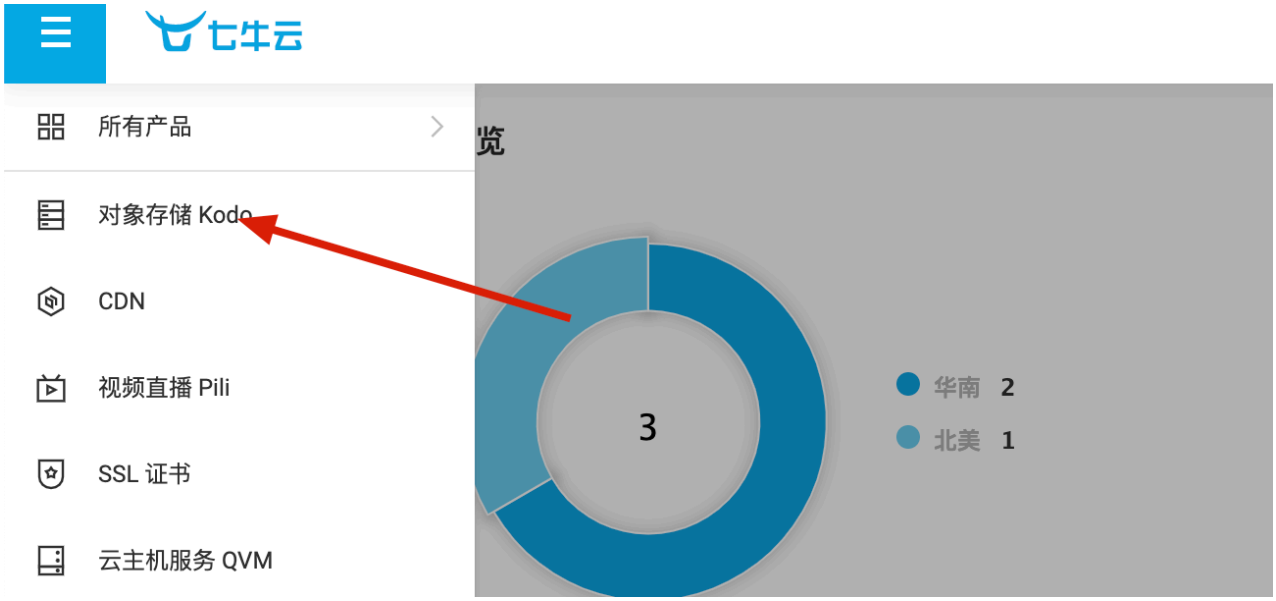
2.2 装饰器的实现

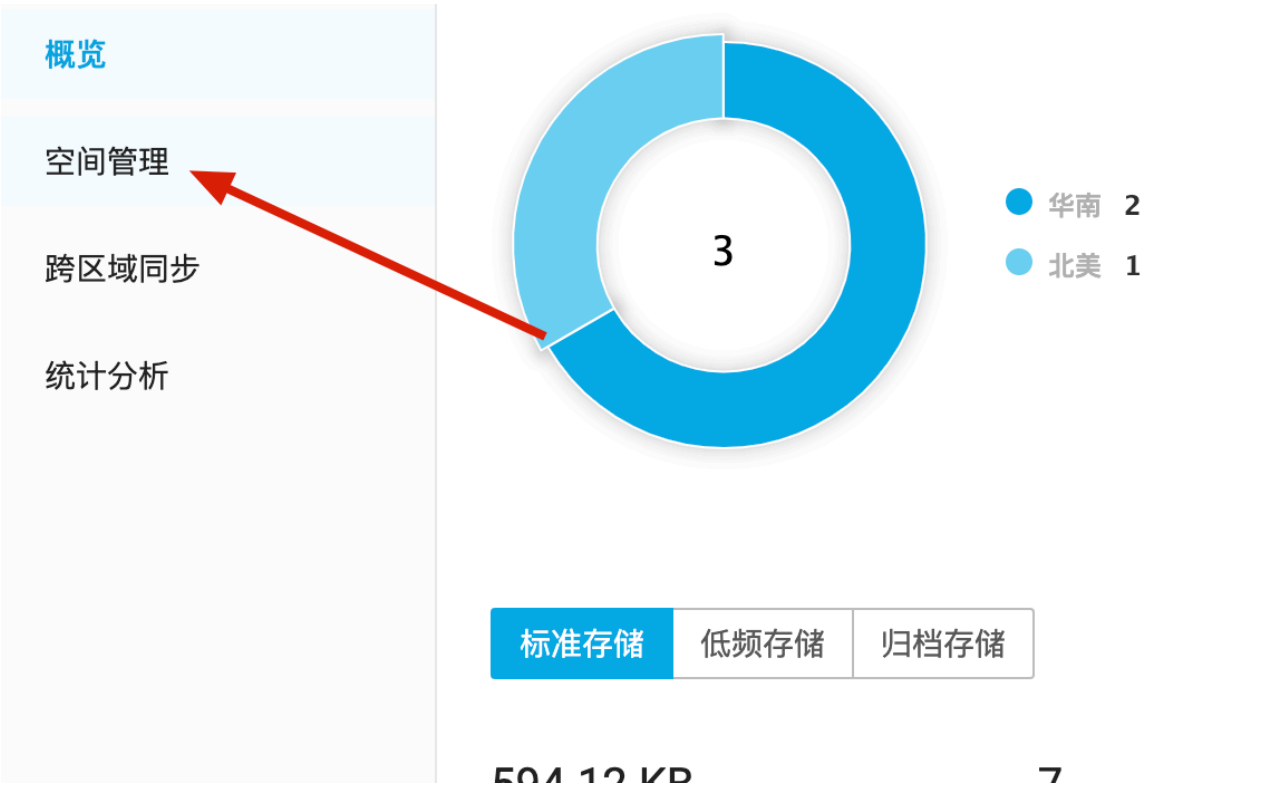
```python
from functools import wraps

from flask import g


def login_required(func):
    # 1. 判断 g.user_id 是否存在
    #    1. 不存在 返回 401
    #    2. 存在调用视图函数
    @wraps(func)
    def wrapper(*args, **kwargs):
        if hasattr(g, 'user_id'):
            return func(*args, **kwargs)
        else:
            return {'message': "用户未登录"}, 401

    return wrapper
```

2.3 接口的实现

3. 七牛云介绍

3.1 七牛云使用





空间管理

+ 新建空间　刷新列表

全部　华东　华北　华南　北美　东南亚

空间名称	空间类型	存储区域	访问控制	创
py38	自有空间	华南	公开	202
hmnewus	自有空间	北美	公开	202

新建存储空间

存储空间名称：shpy39

存储空间名称不允许重复，遇到冲突请更换名称。
名称格式为 3 ~ 63 个字符，可以包含小写字母、数字、短划线，且必须以小写字母或者数字开头和结尾。

存储区域：华东 华北 华南 北美 东南亚

存储空间单价最低。

访问控制：公开 私有

公开和私有仅对 Bucket 的读文件生效，修改、删除、写入等对 Bucket 的操作均需要拥有者的授权才能进行操作。

空间概览 | 文件管理 | 域名管理 | 图片样式 | 转码样式 | 空间设置

上传文件至 shpy39

返回 | 选择文件

No file chosen

选择上传文件存储类型

标准存储 | 低频存储

设置路径前缀：

示例：image/**

转码样式：

请选择转码样式

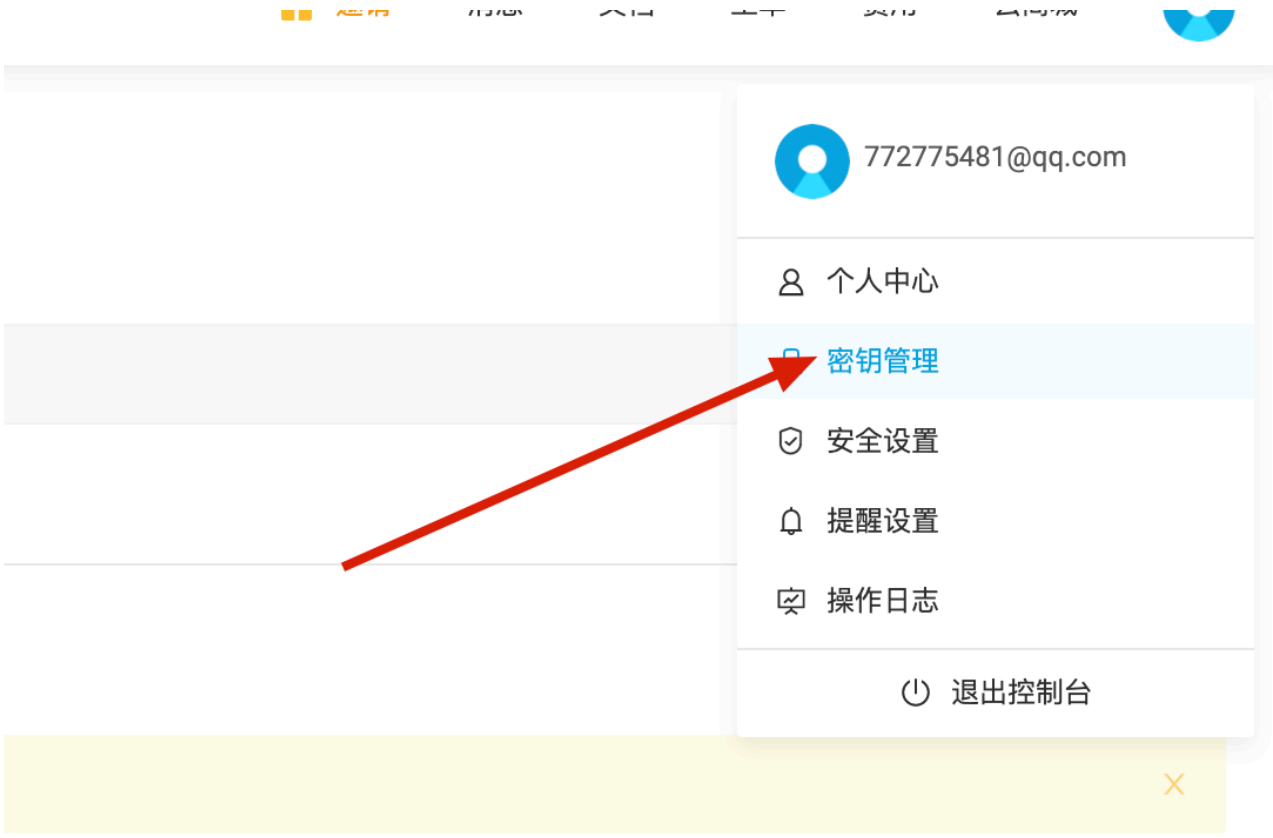
新增转码样式

上传覆盖： 关闭

⚠上传须知

七牛云存储严禁上传包括反动、及侵权内容的文件。七牛有义务违规文件的用户信息保存，并保账号的权利。


文件详情
Xnip2020-12-15_16-42-43.png
基础信息
该文件无法预览
文件名 Xnip2020-12-15_16-42-43.png
文件类型 image/png
复制文件访问的域名
ETag FkLQ-sx0aUVPw_g9NLnobufQgV8r
文件大小 203.68 KB
存储类型 标准存储
文件链接 http://qldhzpaxz.hn-bkt.clouddn.com/Xnip2020-12-15_16-42-43.png
元数据定义
自定义元数据，在 Header 中返回时均会冠以 x-qn-meta 的前缀，例如，name: value 的元数据，会以 x-qn-meta-name: value 在 Header 中返回。
注意：参数名不区分大小写



创建时间 AccessKey/SecretKey

2020-08-24

AK: vtT_LwBjGgt5JGL-bP6im1tldGcZwOGHRNdyWUFp

SK: ····· 显示

```python
# 定义函数(二进制图片数据)
def upload_image(data):
    # 1. 准备配置信息
    #    access_key
    #    secret_key
    # 需要填写你的 Access Key 和 Secret Key
    access_key = 'vtT_LwBjGgt5JGL-bP6im1tldGcZwOGHRNdyWUFp'
    secret_key = 'b5DQrYLidafU-u0oI_NygoTunyZgDHuD7ptFKimf'
    #    空间名
    bucket_name = 'shpy39'
    # 2. 实例化 auth = Auth 对象(access_key, secret_key)
    auth = Auth(access_key, secret_key)
    # 3. 生成上传时需要使用的 token = auth.upload_token(空间名)
    token = auth.upload_token(bucket_name)
    # 4. 上传图片 put_data(token, key=None, data=图片的二进制数据)
    ret, resp = put_data(token, key=None, data=data)
    #     key 是指上传之后，七牛云存储时用的名字，
    #     如果是 None 表示七牛云自动计算出文件的唯一名字
    #     通常情况下，保持为 None，让七牛云自动处理
    # 5. 读取七牛云返回的文件 id
    #      put_data 返回 一个元组(ret, resp)
    #      ret['key'] 就是文件的 id
    # 6. 拼接 域名和文件 id  得到访问图片的完整 url
    if resp.status_code == 200:
        key = ret['key']
        url = 'http://qldhzpaxz.hn-bkt.clouddn.com/' + key
        return url
    else:
        print('上传失败')
        print(ret)
```

"Datalore" plugin update available
Update Plugin Settings... Ignore this update

if __name__ == "__main__" › with open('666.png', 'rb') as f

3.2 集成到项目

```python
    # 七牛云
    QINIU_ACCESS_KEY = 'vtT_LwBjGgt5JGL-bP6im1tldGcZwOGHRNdyWUFp'
    QINIU_SECRET_KEY = 'b5DQrYLidafU-u0oI_NygoTunyZgDHuD7ptFKimf'
    QINIU_BUCKET_NAME = 'shpy39'
    QINIU_DOMAIN = 'http://qldhzpaxz.hn-bkt.clouddn.com/'
```

```python
    #    access_key
    #    secret_key
    # 需要填写你的 Access Key 和 Secret Key
    access_key = current_app.config['QINIU_ACCESS_KEY']
    secret_key = current_app.config['QINIU_SECRET_KEY']
    #    空间名
    bucket_name = current_app.config['QINIU_BUCKET_NAME']
    # 2. 实例化 auth = Auth 对象(access_key, secret_key)
    auth = Auth(access_key, secret_key)
    # 3. 生成上传时需要使用的 token = auth.upload_token(空间名)
    token = auth.upload_token(bucket_name)
    # 4. 上传图片 put_data(token, key=None, data=图片的二进制数据)
    ret, resp = put_data(token, key=None, data=data)
    #     key 是指上传之后，七牛云存储时用的名字，
    #     如果是 None 表示七牛云自动计算出文件的唯一名字
    #     通常情况下，保持为 None，让七牛云自动处理
    # 5. 读取七牛云返回的文件 id
    #      put_data 返回 一个元组(ret, resp)
    #      ret['key'] 就是文件的 id
    # 6. 拼接 域名和文件 id  得到访问图片的完整 url
    if resp.status_code == 200:
        key = ret['key']
        url = current_app.config['QINIU_DOMAIN'] + key
        return url
    else:
        print('上传失败')
        print(ret)
        return None
```

"Datalore" plugin update available
Update Plugin Settings... Ignore this update

```python
if __name__ == '__main__':
    with open('666.png', 'rb') as f:
        # 判断文件是否为图片类型

        # 方式1: 传递 文件路径/文件对象
        # type = imghdr.what('666.png')
        # type = imghdr.what(f)
        # print(type)

        # 方式2: 传递 二进制数据
        data_bytes = f.read()
        type = imghdr.what(None, data_bytes)
        print(type)
```

```python

# 自定义图片校验函数(参数)
def parse_image(image_file):
    try:
        file_type = imghdr.what(image_file)
    except Exception:
        raise ValueError("文件不是图片")

    if not file_type:
        raise ValueError("文件不是图片")

    return image_file

```