AWS 西云笔试整体解决方案

URL 主页：http://52.80.213.27:8888/

1.upload photo to s3

1.1 install web server on ec2 instance host you code.

由于 EC2 默认是 python2.7 且不能连通互联网，所以选择的离线安装 Flask。具体步骤包括,通过远程复制命令将 flask.zip 上传到 ec2；然后使用 unzip 命令解压到当前目录；pip install flask 安装相关依赖包。由于 ec2 实例的默认 python 版本是 python2.7，所以下载 flask 离线安装文件应考虑到版本的问题。

flask 及其相关依赖包，如下图 1 所示

```
[ec2-user@ip-10-21-45-175 flask]$ ls
click-6.7-py2.py3-none-any.whl    itsdangerous-0.24.tar.gz          MarkupSafe-1.0.tar.gz
Flask-1.0.2-py2.py3-none-any.whl  Jinja2-2.10-py2.py3-none-any.whl  Werkzeug-0.14.1-py2.py3-none-any.whl
```

图 1 flask 相关依赖包

启动远程 web server 的方法很简单，只要在 python 代码中指定本地 IP(0.0.0.0),绑定开放的 8888 端口即可，如下图 2 所示。

```
* Serving Flask app "Cloud_2" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with stat
* Debugger is active!
* Debugger PIN: 123-927-792
* Running on http://0.0.0.0:8888/ (Press CTRL+C to quit)
```

图 2 在 EC2 上启动 web server 实例

1.2 write a photo upload webpage , user can upload jpeg photo to S3

首先在本地部属的 html 文件包括登录页，主页，如下图 3，图 4 所示。

图 3 登录页

由于是开放论坛，这里暂未对主页 post 提交的用户名进行处理，输入任意用户名，即可进入主页。

html 文件通过 post 请求转发到后台代码 login 函数，post 又通过 render_template("index.html")，跳转到主页。

图 4 主页

在主页中点击上传文件，输入文件标题（可以随便输入，后端未进行保存本字段），主页 comments 部分功能是：作者用来做图片说明。然后点击 Upload Imagine 即可上传文件。

本功能的实现方式：

1. 在 index.html 中通过表单提交请求
2. 后台 post()函数对提交的数据进行请求。

后台处理方式：

1. 通过 boto3 工具包建立 ec2 与 s3 的连接，具体实现如图 5 所示。

```
s3 = boto3.resource(
    's3',
    aws_access_key_id=ACCESS_KEY_ID,
    aws_secret_access_key=ACCESS_SECRET_KEY,
    config=Config(signature_version='s3v4',region_name='cn-north-1')
)


url = "http://s3.cn-north-1.amazonaws.com.cn/"
```

图 5 建立 ec2 与 s3 的连接

2. 判断上传文件是否是图片文件

```
if('.jpg'in filename or '.jpeg' in filename or '.png' in filename):
```

图 6 判断是否是图片文件

3. 通过

```
s3.Bucket(BUCKET_NAME).put_object(Key='hire2020-hire-luozhukun1031/'+filename, Body=f,ACL='public-read')
```

上传图片，ACL 设置图片的访问策略，这里设置为可读

4. 通过 put_object()上传图片到 s3 的某个对象

如果上传成功，会出现如图 7 所示结果。

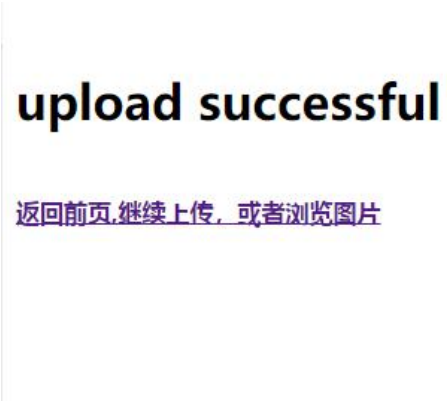


图 7 上传成功提示页

如果不成功，会跳转到上传错误页，提示重新上传。如图 8 所示

图 7 上传错误页

1.3 record user ip , date, time , photo name, photo url into DynamoDB when customer upload photo.

在上传到 s3 的同时，我们已经得到了主页 post 请求提交的照片名字以及当前上传时间（datetime.datetime.now()），将 s3 对象地址和照片名字进行拼接行成 Photourl.另外我们建立部分后面问题可能需要的字段。具体属性如下图 8 所示。

图 8 dynamodb 中包含的属性字段

通过如下代码建立与 Dynamodb 的连接

boto3.resource('dynamodb',region_name='cn-north-1',aws_access_key_id='AKIAVLSEBUAJRL52NPMZ', aws_secret_access_key='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')

然后通过 put_item（）将我们的字段插入数据库，包含（user ip , date, time , photo name, photo url），另外这里的 ID 字段，我们使用 UUid 函数生成一个唯一字段，作为一个 key 字段。同时建立一个二级索引 photoname。

2.Review Comment and Share Photo

2.1 create a photo review webpage, user can see all photo upload in the past time. when click a photo, user can see upload user ip, upload date time, photo name.

实现方式是，通过本地 post 请求提交到后台 upload（）

Upload 函数的实现逻辑：

扫描 dynamodb 中所建立的全表，建立 sorkey 在 date 上，按上传时间顺序扫描全表。将扫描结果，通过 render_template（）方法返回到照片概览页，获取全部图片的 user ip, upload date time, photo name 信息。如图 9，图 10 所示。

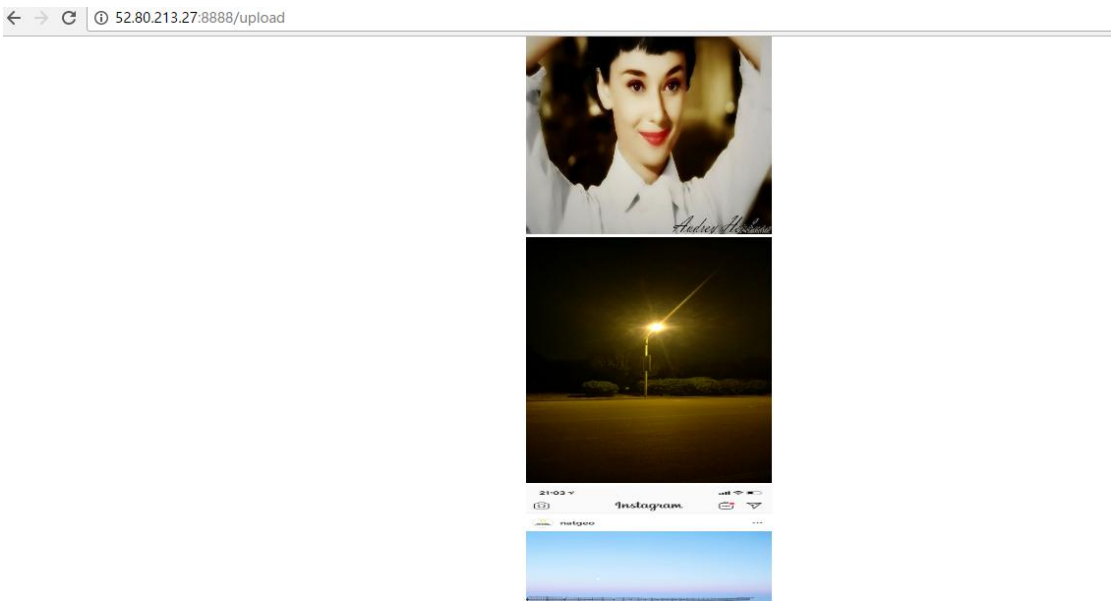


图 9 照片概览页面

图 10 照片详情页

2.2 create a search box , user can search photo by photo upload time or photo name.

在照片详情页创建一个添加一个关于照片名和时间框，并设置默认值，然后后台 picture_search（）函数对上传的时间字符串和照片名进行处理，具体方法是当检索的时间段或者照片名没有找到，返回全部照片。当只包含照片名，按照片名进行过滤，如果只包含上传时间字段，则按时间字段进行扫描。如果都包含，则按两个字段进行扫描。如图 12 所示。

图 12 照片搜索页

2.3 user can add comment in the photo view page, comment time and ip address will also store in DynamoDB, will other user can see the comment by timeline.

在照片详情页中添加了一个评论框，用户能够通过直接输入进行评论，评论之后通过 post 请求后端 comment 函数即时刷新，生成 timeine 评论返回照片详情页。如图 13，图 14 所示。

图 13 评论提交前

photoname: ww4sinaimgcnworiginal7095bce9gw1evfzbhu880j20c80a1ab9jpg
upload datetime: 2019-11-10 10:32:01
user ip: 10.21.14.117
user Comments: nihao
other Comments: b''
b"'userIP': b'127.0.0.1', 'commentContent': b'\\xe5\\x93\\x88\\xe5\\x93\\x88\\xe5\\x93\\x88\\xe5\\x93\\x88', 'commentTime': b'2019-11-10 19:19:48'"
b"'userIP': b'127.0.0.1', 'commentContent': b'\\xe5\\x93\\x88\\xe5\\x93\\x88\\xe5\\x93\\x88', 'commentTime': b'2019-11-10 19:19:57'"

Comments:
评论
分享图片
返回前页

图 14 评论提交后

2.4 create a button , user can generate a photo share link, with S3 PreSignedURL, and share s3 photo url to others, the link will valid for 1 minute, after 1 minute, the photo will can't be open.

方法 1：这里实现通过 hashlib 单向加密函数，通过 get 请求的方式设置参数 PID 和当前时间传递给服务端。服务端判断是否过期。如没有过期。转向照片的 url 链接，隐藏链接并访问。如若过期，重新生成单向加密 url 返回给客户端。具体代码实现：

```
current_time = time.time()
ramdom_str = "%s|%s" %(PID, current_time)
h = hashlib.md5()
h.update(bytes(ramdom_str, encoding='utf-8'))
UID = h.hexdigest()

q = "%s|%s|0" %(UID, current_time)
url = 'http://52.80.213.27:8888/picture_share?pid=%s' % q
```

如图 15 所示。

photoname: ww4sinaimgcnworiginal7095bce9gw1evfzbhu880j20c80a1ab9jpg
upload datetime: 2019-11-10 10:32:01
user ip: 10.21.14.117
user Comments: nihao
other Comments: b''
b"'userIP': b'127.0.0.1', 'commentContent': b'\\xe5\\x93\\x88\\xe5\\x93\\x88\\xe5\\x93\\x88\\xe5\\x93\\x88', 'commentTime': b'2019-11-10 19:19:48'"
b"'userIP': b'127.0.0.1', 'commentContent': b'\\xe5\\x93\\x88\\xe5\\x93\\x88\\xe5\\x93\\x88', 'commentTime': b'2019-11-10 19:19:57'"

Comments:
评论
分享图片

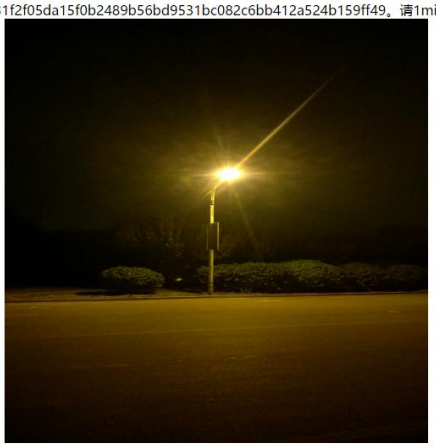
图 15 分享图片

方法二：通过 Boto 3 创建预签名 URL：

1. 配置 IAM 用户凭证以用于 Boto。

2. 编辑和运行以下代码段可创建用于 S3 对象的预签名 URL，并设置时限 1min

生成的照片链接: https://hire2020.s3.cn-north-1.amazonaws.com.cn/hire2020-hire-luozhukun1031/WP_20150827_003.jpg?X-Amz-Date=20191110T130006Z&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-SignedHeaders=host&X-Amz-Credential=AKIAVLSEBUAJRL52NPMZ%2F20191110%2Fcn-north-1%2Fs3%2Faws4_request&X-Amz-Expires=60&X-Amz-Signature=29805a6826a8031f2f05da15f0b2489b56bd9531bc082c6bb412a524b159ff49。请1mins中内使用，否则需要重新生成!



返回照片详情页
再次生成

图 16 使用 boto3 生成预签名

3. Work with a AWS ELB Load Banlancer.

3.1 create a AWS ELB Load Banlancer use your DynamoDB table name as Load Banlancer name, use subnet id subnet-01fb2a35ea0eb3b37,subnet-01b104c5dfeef6d6a and security group id sg-0b5c42f314f01b227.

使用 aws cli 进行创建，结果如图 16 所示

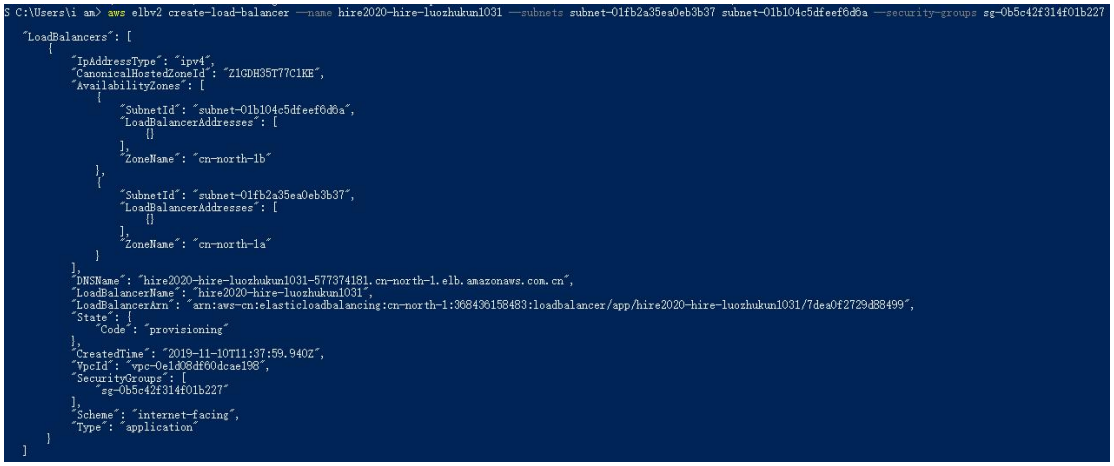


图 16 创建一个负载均衡器

3.2 使用 create-target-group 命令创建目标组，并指定用于 EC2 实例的相同 VPC

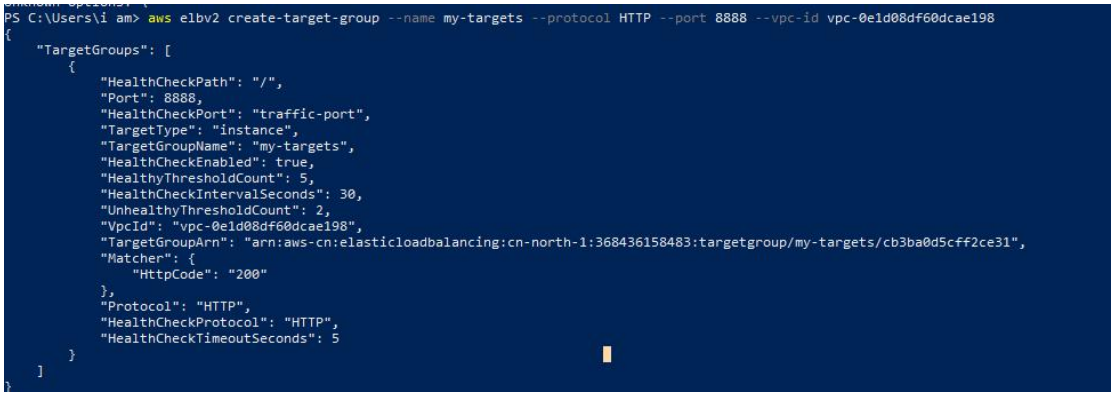


图 17 创建目标组

3.3 使用 register-targets 命令将您的实例注册到目标组

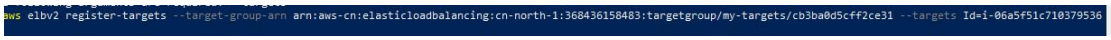


图 18 注册到目标组

3.4 使用此 describe-target-health 命令验证目标组的已注册目标的运行状况

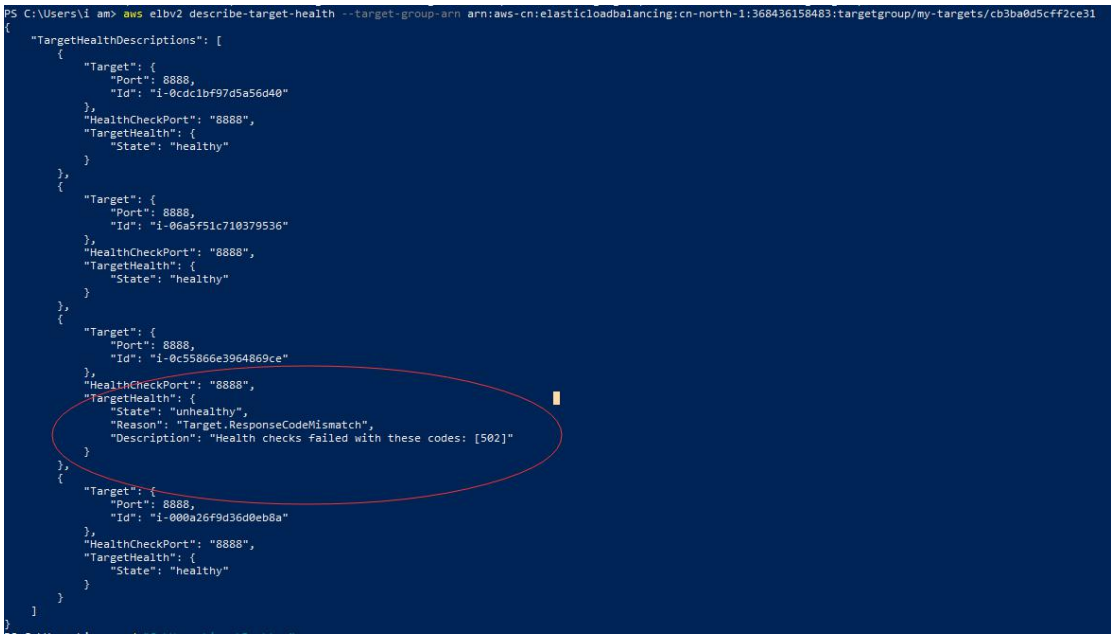
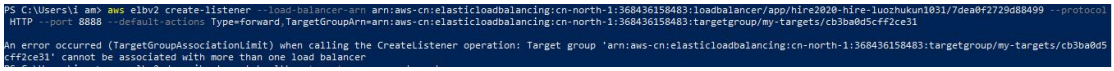


图 19 运行情况

3.5 报错情况



当我使用 create-listener 命令为您的负载均衡器创建侦听器，报错，我不知道是不是不让创建目标组。

3.6 创建代理

因为前面好像没有成功，所以不给出具体结果，得到真实 ip 步骤为:

1.1、为 elb（名称 YOU_ELB_NAME） 创建一个 Proxy Protocol，叫 EnableProxyProtocol

1.2、为 YOU_ELB_NAME 激活 EnableProxyProtocol

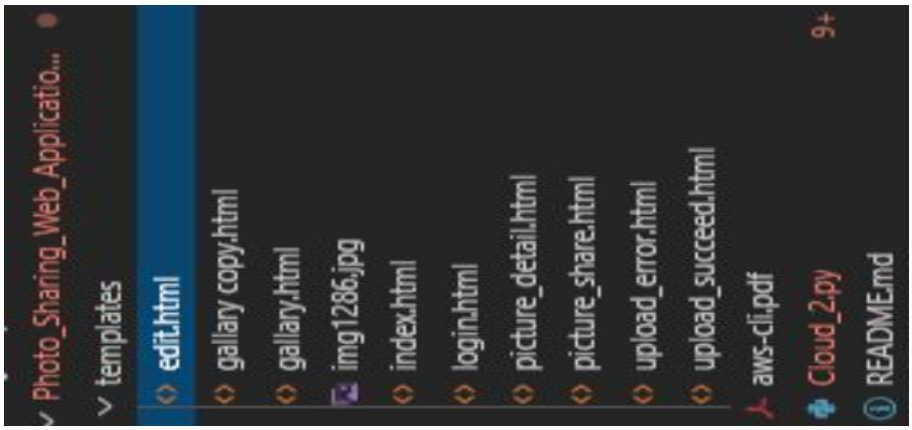
1.3、确认开启 aws elb describe-load-balancers --load-balancer-name YOU_ELB_NAME

1.4、查看是否启用 aws elb describe-load-balancers --load-balancer-name YOU_ELB_NAME

| jq '.LoadBalancerDescriptions[].BackendServerDescriptions'。

总结

本次笔试有较多瑕疵，网页存在很多编码的问题，暂时未能全部解决，建议使用英文。由于在笔试的同时，还得投入经历在我的论文之上。总体来说，我的时间非常不够，希望谅解。后面第三题的 ELB 负载，我后面会继续完成。感谢贵公司给我这次机会。附件包含我的工程文件和我的代码。由于时间关系，报告不能翻译成英文，请谅解！
工程目录结构：



Login.html->登录页
Index.html->主页
Gallery.html->概览页
Picture_detail.html-->照片详情页面
Picture_share.html-->共享链接生成页
Upload_error.html->上载出错页
Upload_succeed.html->上载成功页面
Cloud_2.py->后端代码