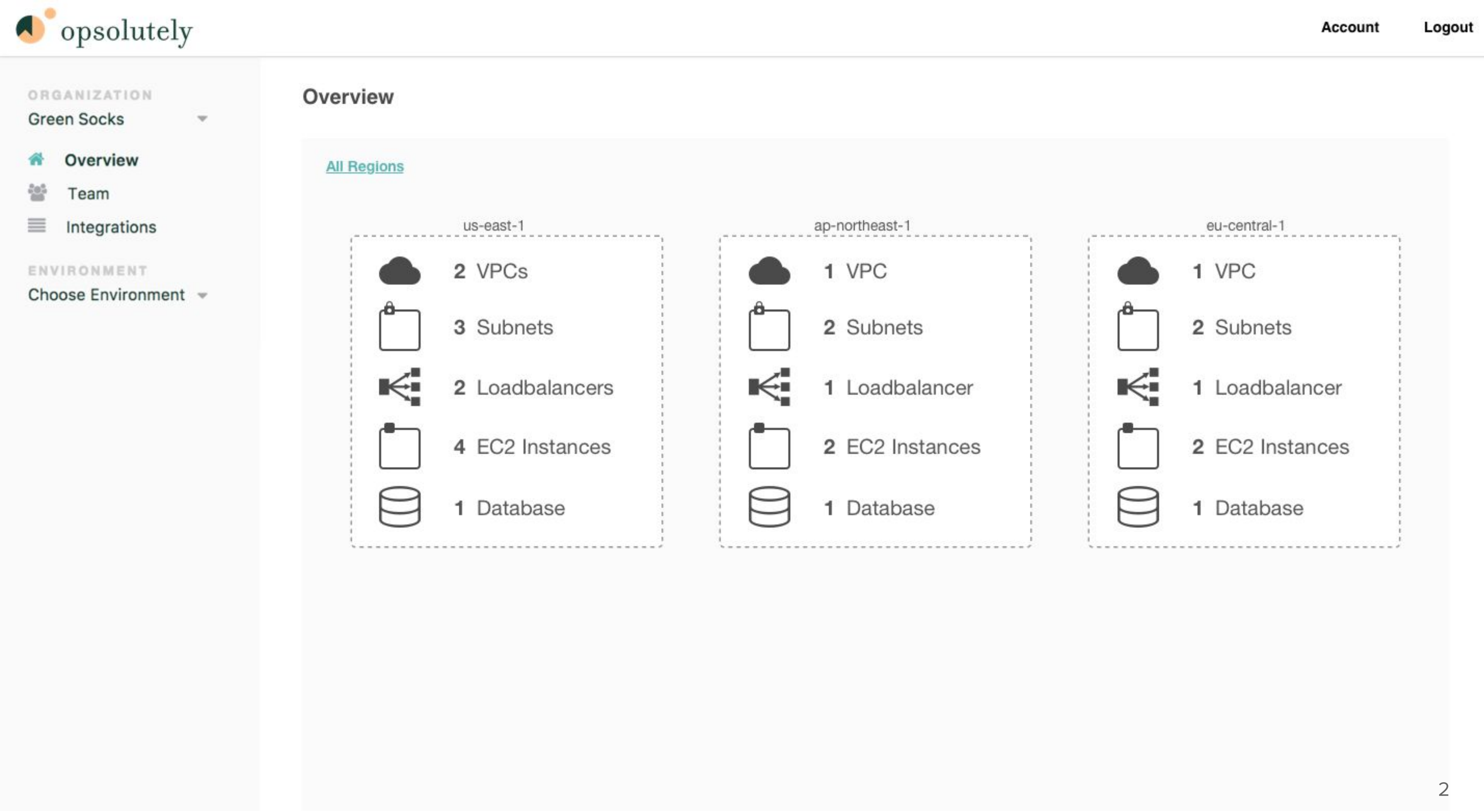
How we replaced AWS with a Python script

Jack Danger
@jackdanger
opsolutely.com



ORGANIZATION

Green Socks ▾

- Overview
- Team
- Integrations

ENVIRONMENT

Choose Environment ▾

Overview

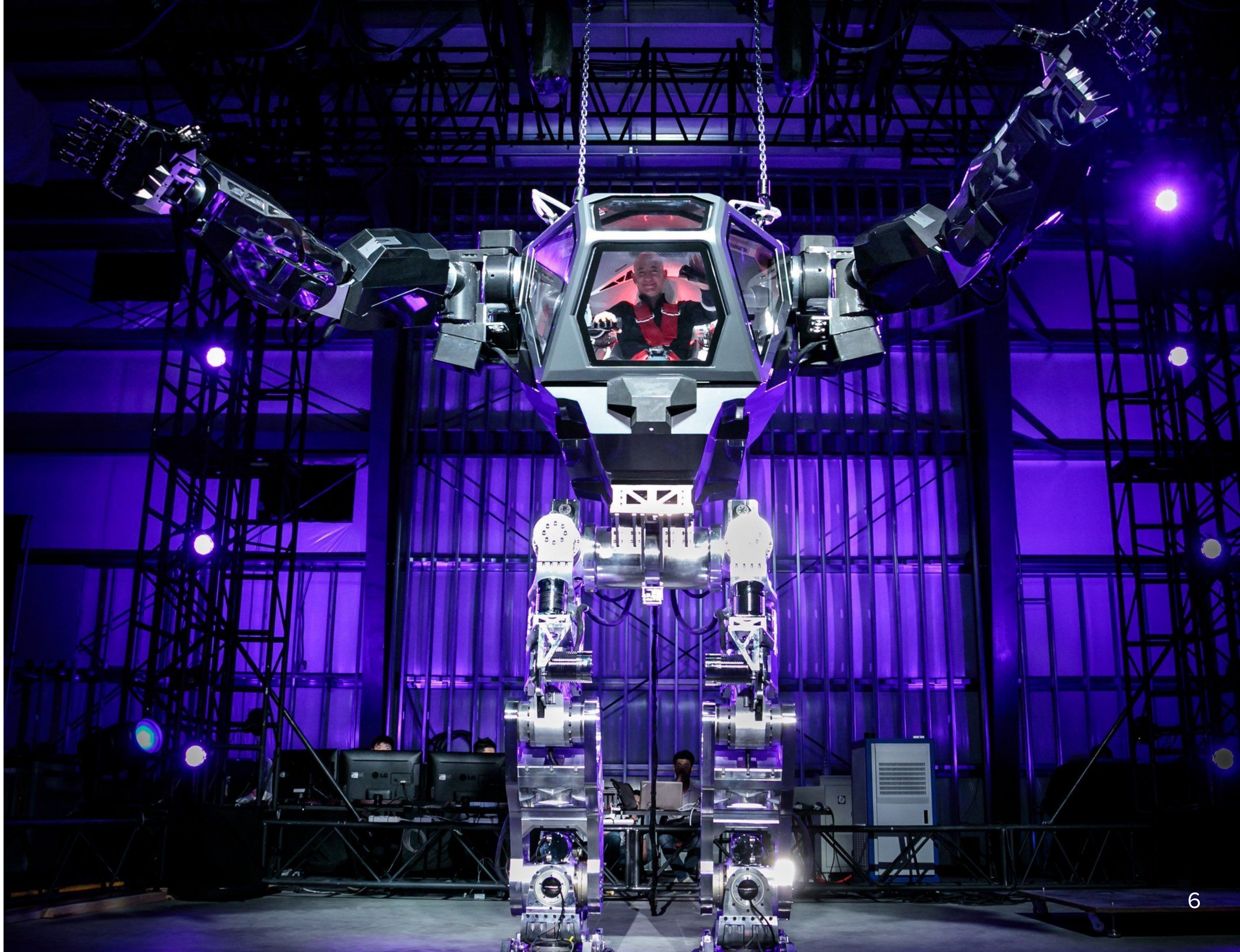
All Regions

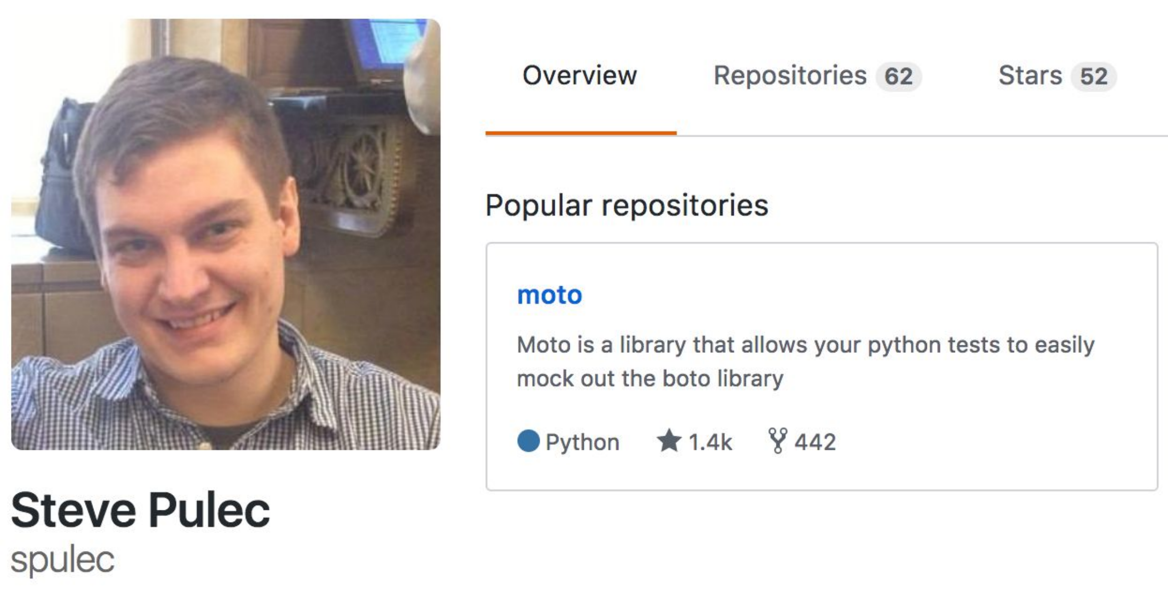
us-east-1	ap-northeast-1	eu-central-1
2 VPCs	1 VPC	1 VPC
3 Subnets	2 Subnets	2 Subnets
2 Loadbalancers	1 Loadbalancer	1 Loadbalancer
4 EC2 Instances	2 EC2 Instances	2 EC2 Instances
1 Database	1 Database	1 Database

"Round Robin"

Tests Pass = Code Works

"I have more money than I can spend"





Steve Pulec
spulec

Popular repositories

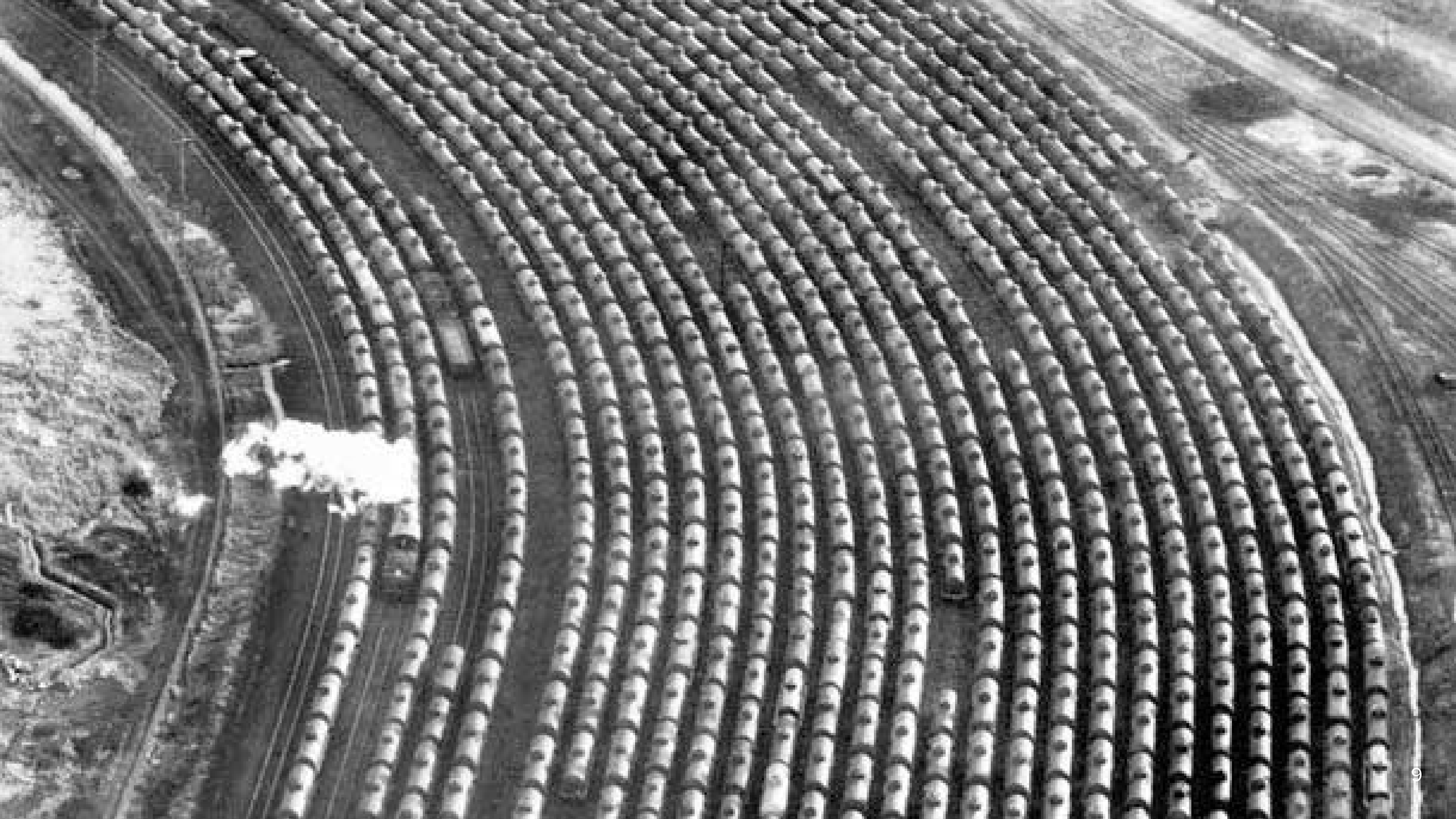
moto

Moto is a library that allows your python tests to easily mock out the boto library

● Python ★ 1.4k 442

Where it all started

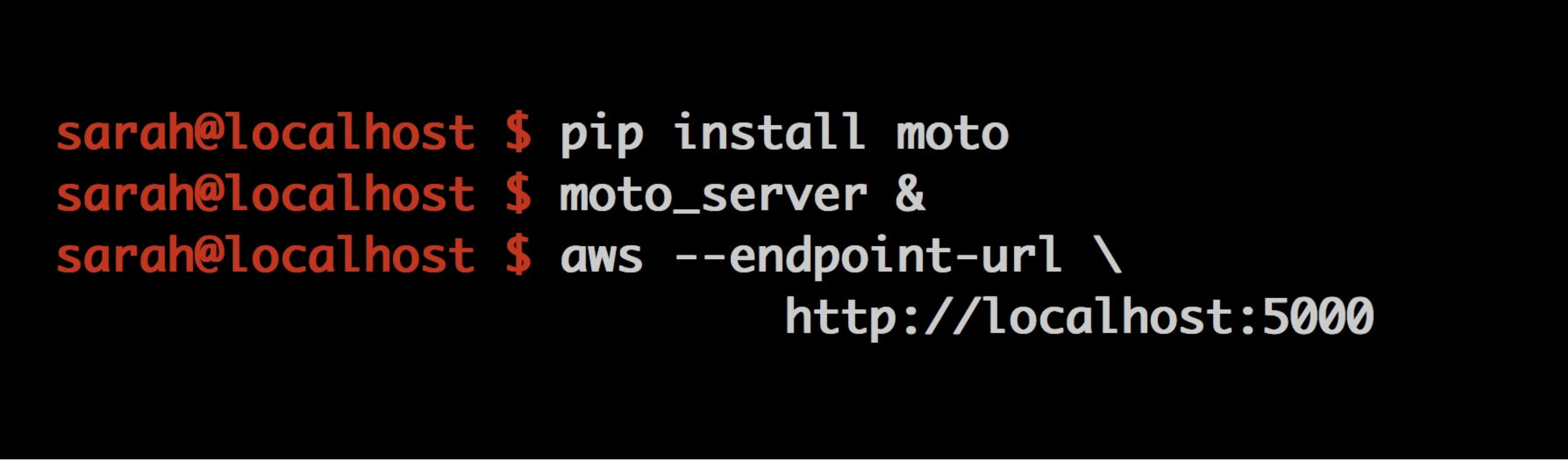
BNSF



Perfect copy.

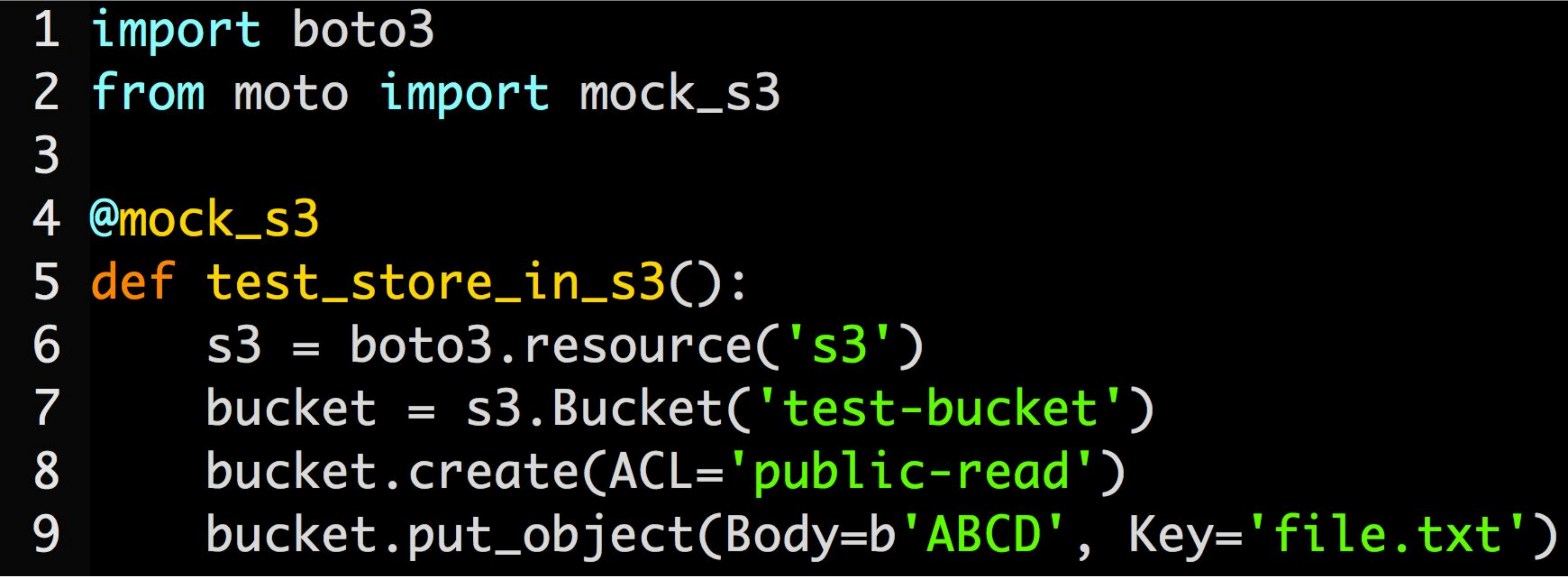
Nobody will ever notice.

"I promise I won't mock anymore!"



```
sarah@localhost $ pip install moto
sarah@localhost $ moto_server &
sarah@localhost $ aws --endpoint-url \
                          http://localhost:5000
```

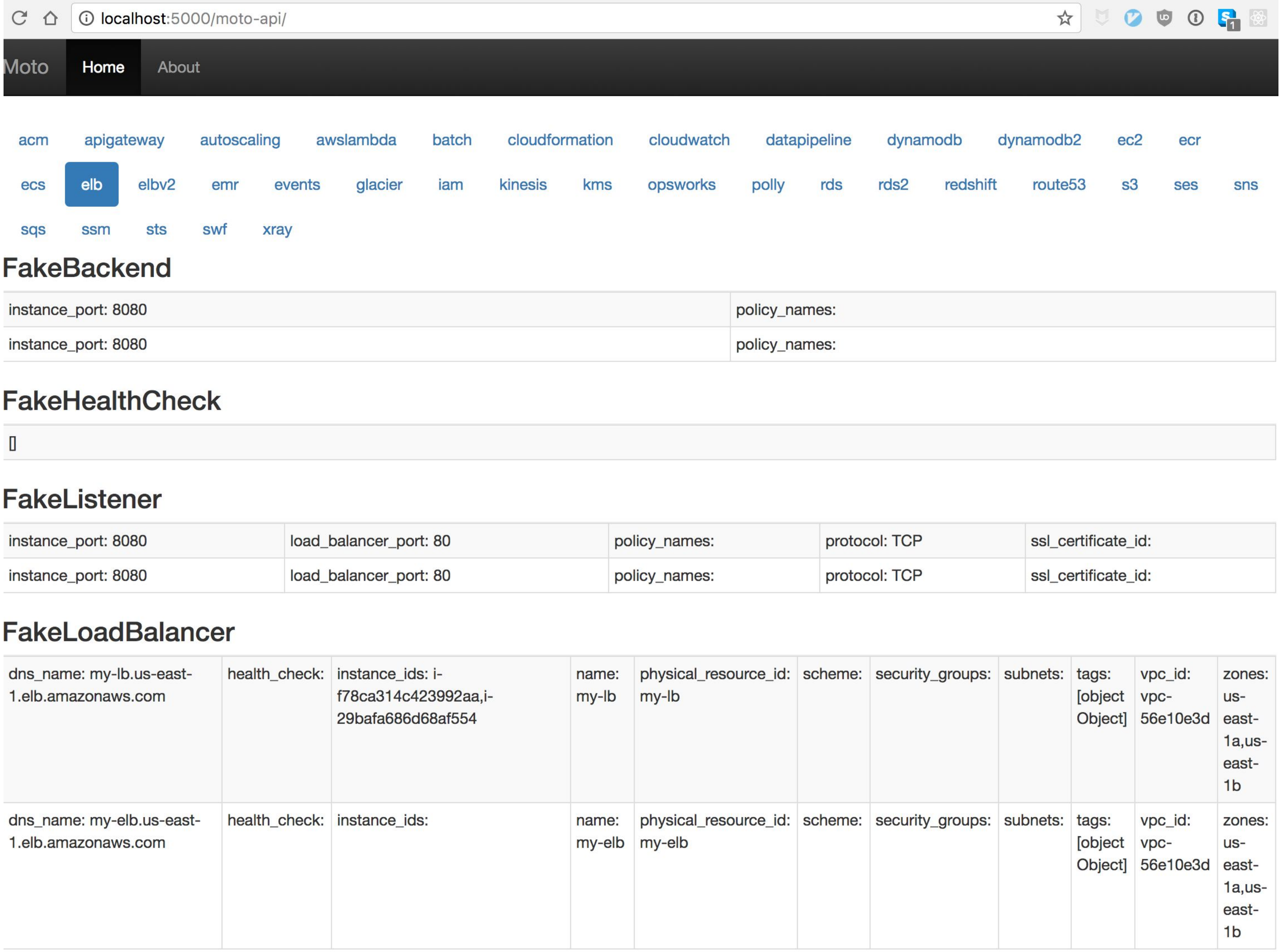
Server Mode



```python
import boto3
from moto import mock_s3

@mock_s3
def test_store_in_s3():
    s3 = boto3.resource('s3')
    bucket = s3.Bucket('test-bucket')
    bucket.create(ACL='public-read')
    bucket.put_object(Body=b'ABCD', Key='file.txt')
```

Embedded Python



FakeBackend

instance_port: 8080	policy_names:
instance_port: 8080	policy_names:

FakeHealthCheck

[]

FakeListener

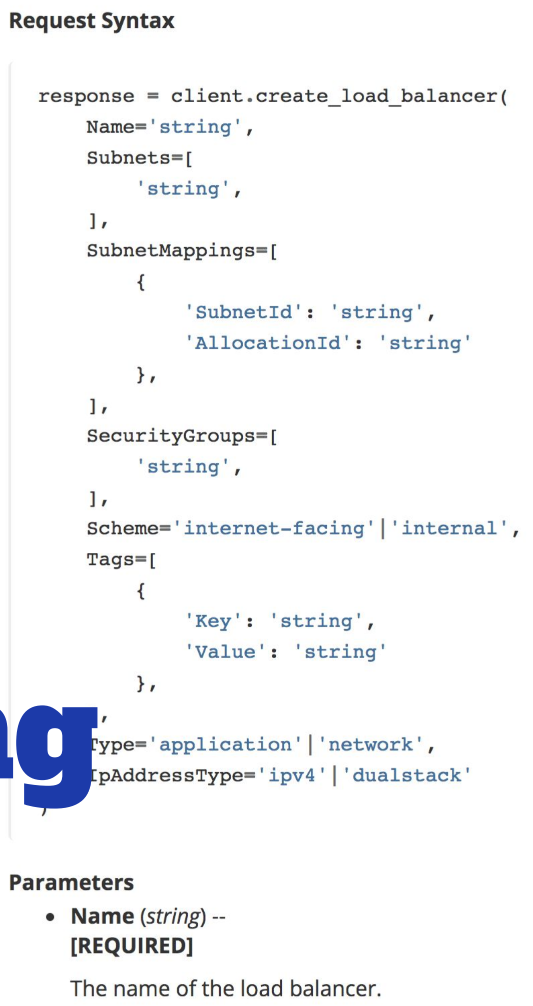
instance_port: 8080	load_balancer_port: 80	policy_names:	protocol: TCP	ssl_certificate_id:
instance_port: 8080	load_balancer_port: 80	policy_names:	protocol: TCP	ssl_certificate_id:

FakeLoadBalancer

dns_name: my-lb.us-east-1.elb.amazonaws.com	health_check:	instance_ids: i-f78ca314c423992aa,i-29bafa686d68af554	name: my-lb	physical_resource_id: my-lb	scheme:	security_groups:	subnets:	tags: [object Object]	vpc_id: vpc-56e10e3d	zones: us-east-1a,us-east-1b
dns_name: my-elb.us-east-1.elb.amazonaws.com	health_check:	instance_ids:	name: my-elb	physical_resource_id: my-elb	scheme:	security_groups:	subnets:	tags: [object Object]	vpc_id: vpc-56e10e3d	zones: us-east-1a,us-east-1b

see inside AWS

The docs Are amazing



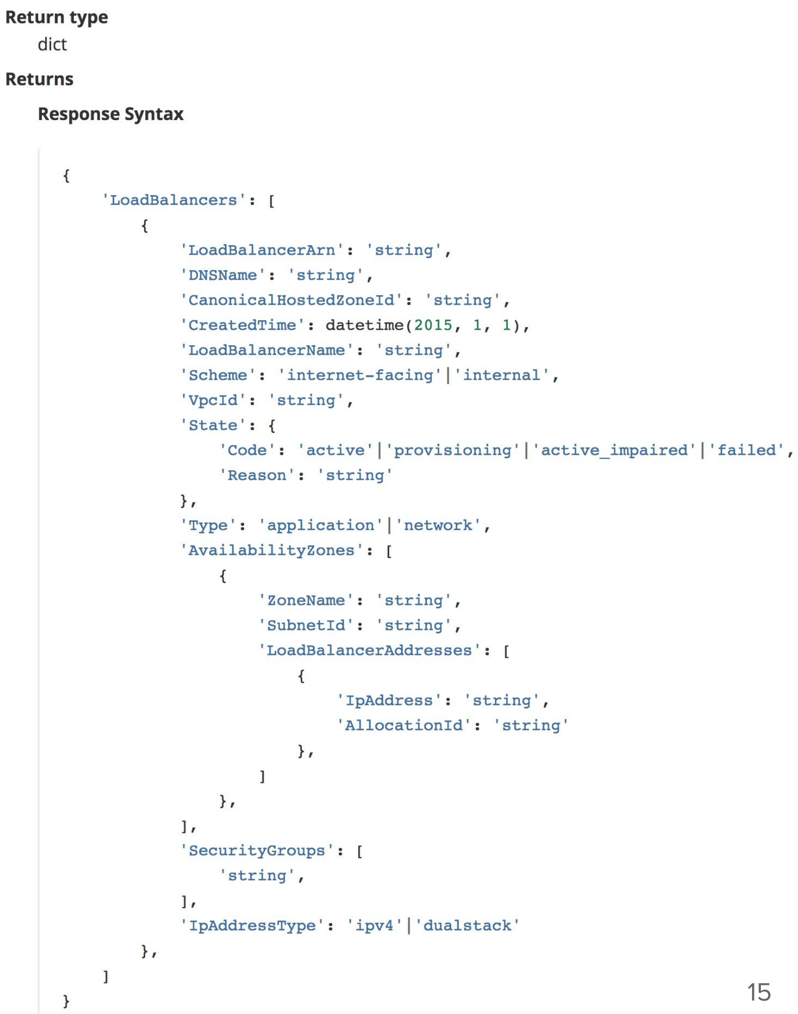
Request Syntax

```
response = client.create_load_balancer(
    Name='string',
    Subnets=[
        'string',
    ],
    SubnetMappings=[
        {
            'SubnetId': 'string',
            'AllocationId': 'string'
        },
    ],
    SecurityGroups=[
        'string',
    ],
    Scheme='internet-facing'|'internal',
    Tags=[
        {
            'Key': 'string',
            'Value': 'string'
        },
    ],
    Type='application'|'network',
    IpAddressType='ipv4'|'dualstack'
)
```

Parameters

- **Name** (*string*) --
  [REQUIRED]

  The name of the load balancer.



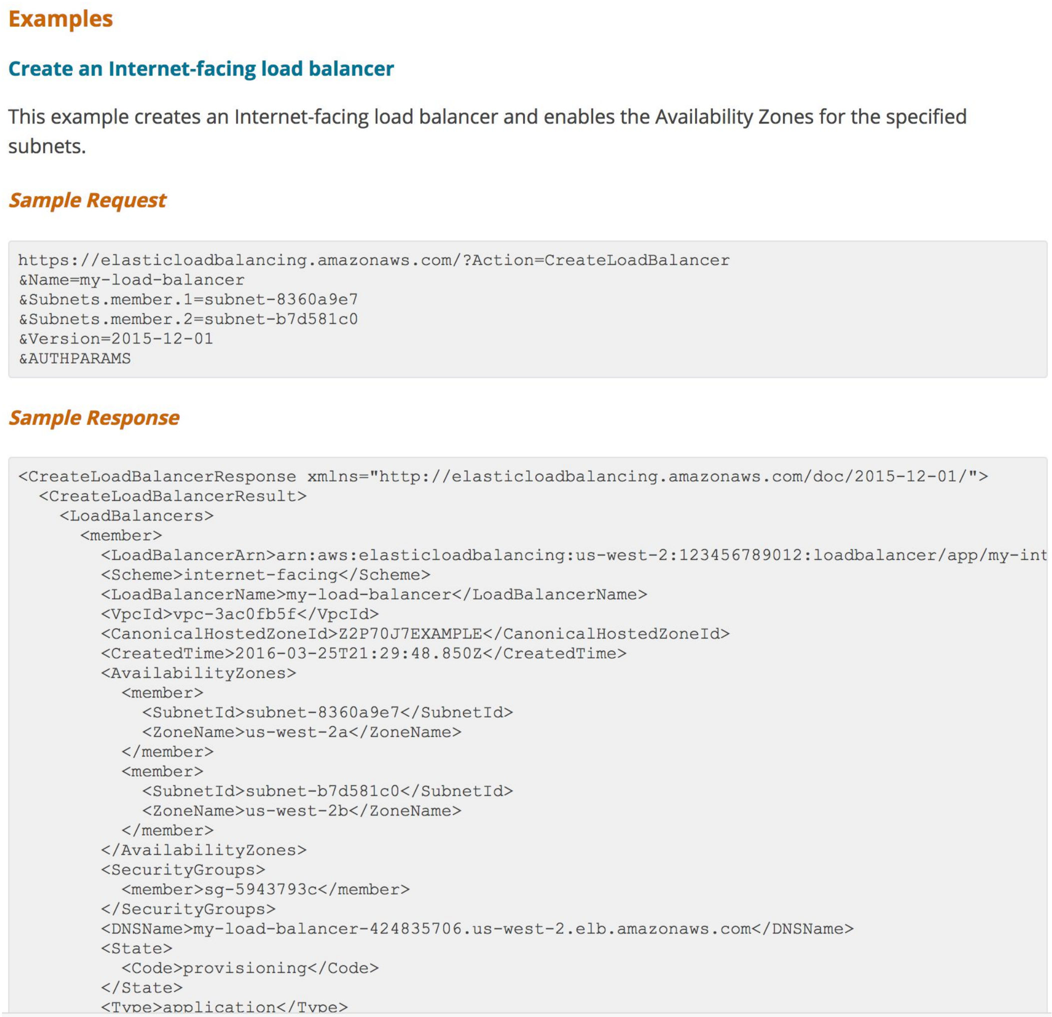
Return type

dict

Returns

Response Syntax

```
{
    'LoadBalancers': [
        {
            'LoadBalancerArn': 'string',
            'DNSName': 'string',
            'CanonicalHostedZoneId': 'string',
            'CreatedTime': datetime(2015, 1, 1),
            'LoadBalancerName': 'string',
            'Scheme': 'internet-facing'|'internal',
            'VpcId': 'string',
            'State': {
                'Code': 'active'|'provisioning'|'active_impaired'|'failed',
                'Reason': 'string'
            },
            'Type': 'application'|'network',
            'AvailabilityZones': [
                {
                    'ZoneName': 'string',
                    'SubnetId': 'string',
                    'LoadBalancerAddresses': [
                        {
                            'IpAddress': 'string',
                            'AllocationId': 'string'
                        },
                    ]
                },
            ],
            'SecurityGroups': [
                'string',
            ],
            'IpAddressType': 'ipv4'|'dualstack'
        },
    ]
}
```



Examples

Create an Internet-facing load balancer

This example creates an Internet-facing load balancer and enables the Availability Zones for the specified subnets.

Sample Request

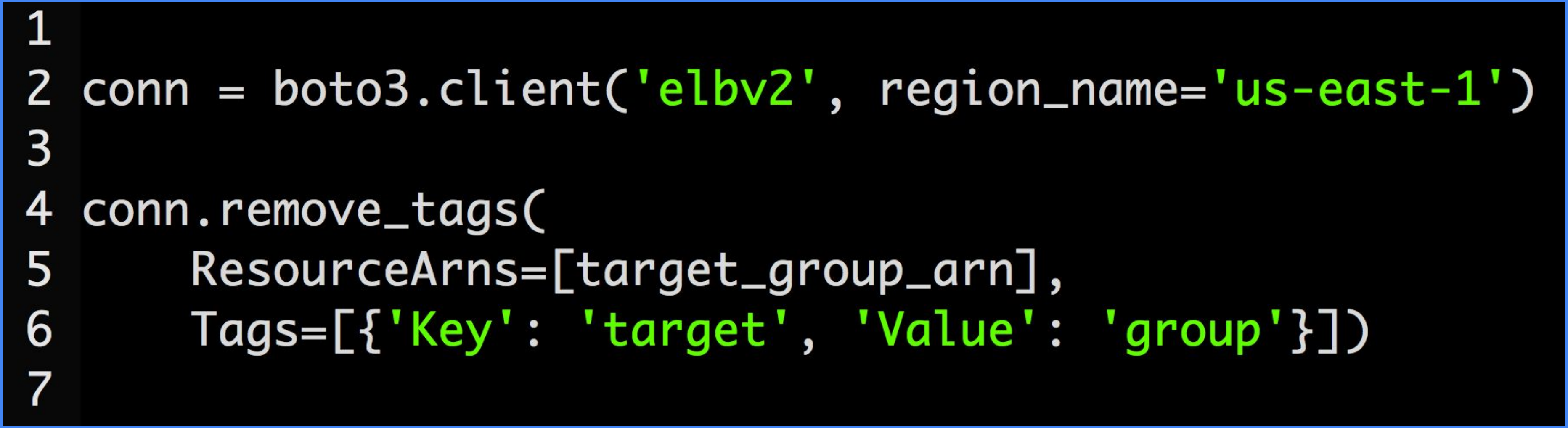
```
https://elasticloadbalancing.amazonaws.com/?Action=CreateLoadBalancer
&Name=my-load-balancer
&Subnets.member.1=subnet-8360a9e7
&Subnets.member.2=subnet-b7d581c0
&Version=2015-12-01
&AUTHPARAMS
```

Sample Response

```
<CreateLoadBalancerResponse xmlns="http://elasticloadbalancing.amazonaws.com/doc/2015-12-01/">
  <CreateLoadBalancerResult>
    <LoadBalancers>
      <member>
        <LoadBalancerArn>arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-int
        <Scheme>internet-facing</Scheme>
        <LoadBalancerName>my-load-balancer</LoadBalancerName>
        <VpcId>vpc-3ac0fb5f</VpcId>
        <CanonicalHostedZoneId>Z2P70J7EXAMPLE</CanonicalHostedZoneId>
        <CreatedTime>2016-03-25T21:29:48.850Z</CreatedTime>
        <AvailabilityZones>
          <member>
            <SubnetId>subnet-8360a9e7</SubnetId>
            <ZoneName>us-west-2a</ZoneName>
          </member>
          <member>
            <SubnetId>subnet-b7d581c0</SubnetId>
            <ZoneName>us-west-2b</ZoneName>
          </member>
        </AvailabilityZones>
        <SecurityGroups>
          <member>sg-5943793c</member>
        </SecurityGroups>
        <DNSName>my-load-balancer-424835706.us-west-2.elb.amazonaws.com</DNSName>
        <State>
          <Code>provisioning</Code>
        </State>
        <Type>application</Type>
```

The best anywhere

Come on in!



```python

conn = boto3.client('elbv2', region_name='us-east-1')

conn.remove_tags(
        ResourceArns=[target_group_arn],
        Tags=[{'Key': 'target', 'Value': 'group'}])

```

Which service?	ELBv2
Which endpoint?	remove_tags
Where to start?	moto/elbv2/responses.py

EASY

MEDIUM

HARD

Jack Danger

@jackdanger

opsolutely.com