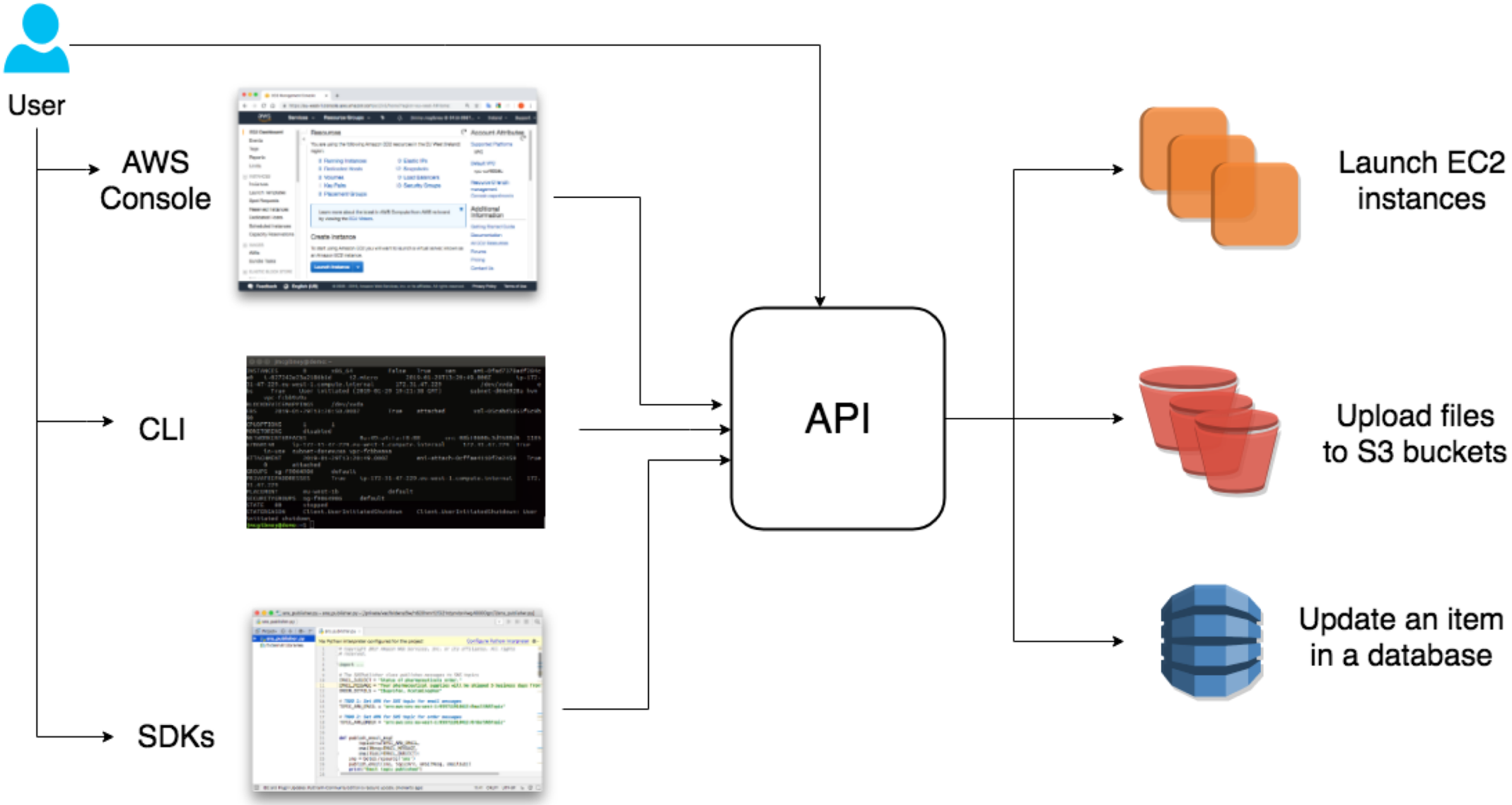
Developer Operations

Developing on AWS: Overview

Four ways to use AWS

- With AWS, all services are managed through an Application Programming Interface (API).
- A user can access this API in any of 4 ways:
  - Management Console
  - Command Line Interface (CLI)
  - Software Development Kit (SDK)
    - SDKs available for many languages (e.g. Python, Java, .NET)
    - Designed to simplify things for developers (compared with direct API calls)
  - Direct REST-like API calls
    - Using HTTP/HTTPS

Four ways to use AWS



Authentication

- Management Console based on Username and Password (+ optional MFA)
- All other methods based on API credentials
  - Access Key ID & Secret Access Key

IAM User

```
Access Key ID: AKIAJ4YAT7URNZPQTEST
Secret Access Key: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

AWS CLI

```
jmcgibney@demo: ~
jmcgibney@demo:~$ aws ec2 describe-instances
RESERVATIONS    118567060150    r-01d71c3f18124f689
INSTANCES       0       x86_64          False   True    xen     ami-0bce5ae5a9ce
6d623  i-00dc204260426a28e      t2.micro        jrh     2018-12-12T18:07:22.000Z
        ip-172-31-22-102.eu-west-1.compute.internal     172.31.22.102
/dev/xvda       ebs     True    User initiated (2018-12-12 19:04:17 GMT)
subnet-123d505a hvm     vpc-fcbb9a9a
BLOCKDEVICEMAPPINGS     /dev/xvda
```

AWS SDK & API

Java

Python

.NET

Interoperability

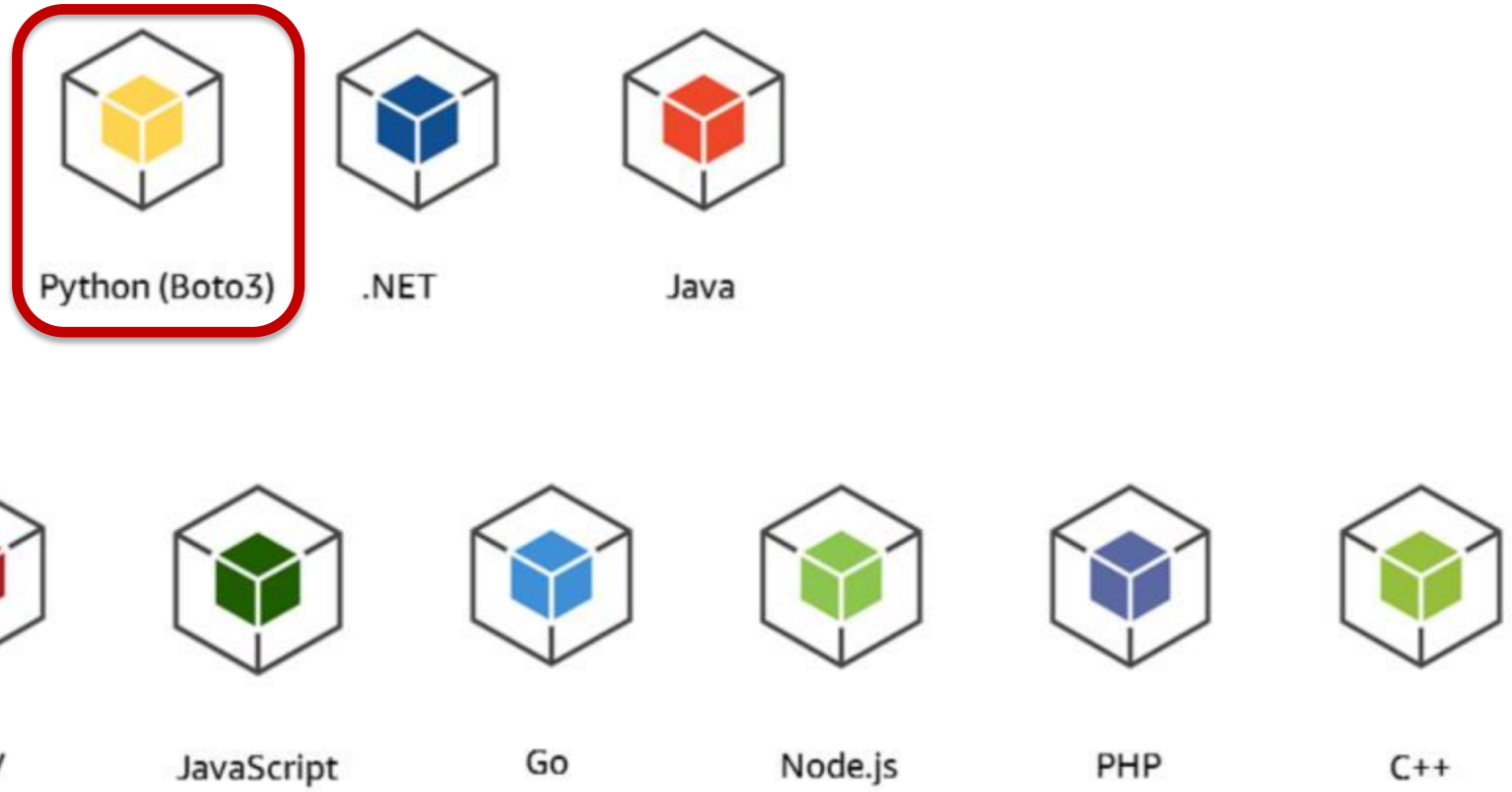
- The various access modes can be used interchangeably
  - e.g. you could create an instance using Python/boto3, start it using the CLI and stop it using the management console
- They all use the API underneath
- Note though that changes made using API may not appear immediately in the console (most pages have a 'refresh' button)

Command Line Interface (CLI) format


```
$ aws ec2 run-instances --cli-input-json file://./lcf/webserver.json
```

```
$ aws help
$ aws ec2 help
$ aws ec2 describe-instances help
```

Software Development Kit (SDK) libraries for several languages



- We are using the Boto3 SDK for Python in this module
- This is a library of classes and functions that we can use in our Python programs

Connecting to a Service

- Can use Service Client API vs Resource API
- Resource API newer and usually recommended approach

Client API vs Resource API

Service Client API

- Focused on **services**
- More low-level
- What operations can be carried out with the service (e.g. EC2 or S3)
- Has objects for request and result data

```
ec2client = boto3.client('ec2')
```

Client API vs Resource API

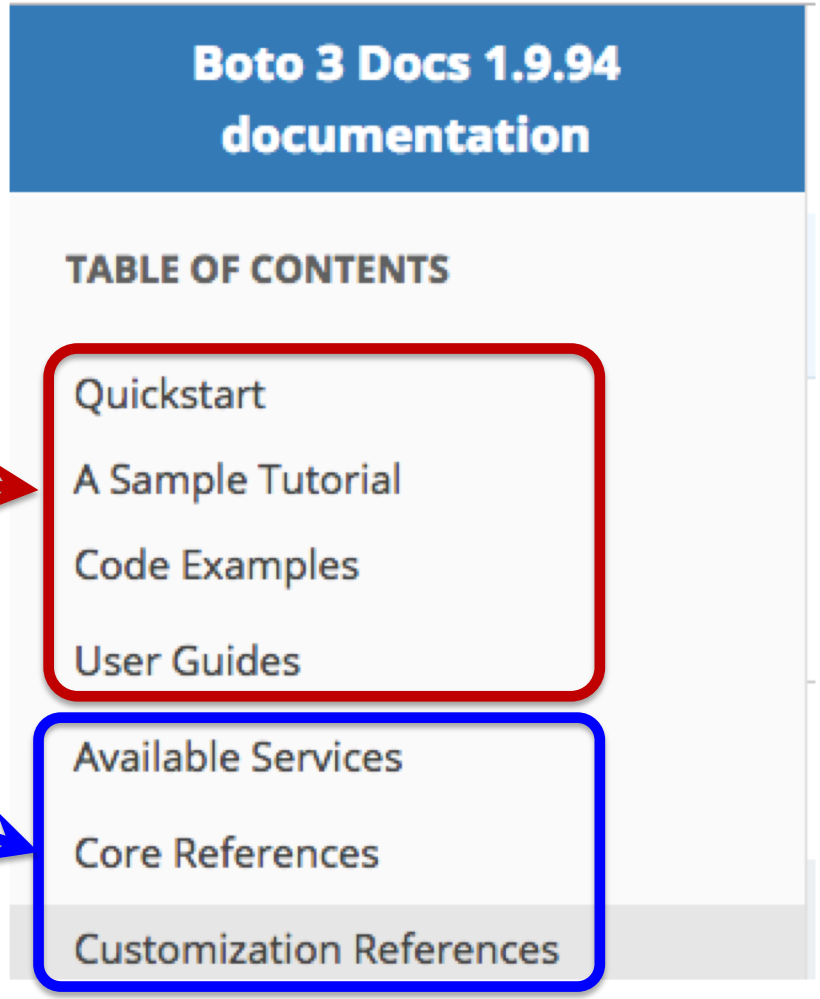
Resource API

- Focused on **resources**
- More high-level, more object oriented
- Has one class per conceptual resource
- Resource can be a service (e.g. EC2, S3) or an individual resource (e.g. an instance, a security group, an S3 bucket)

```
ec2resource = boto3.resource('ec2')
```

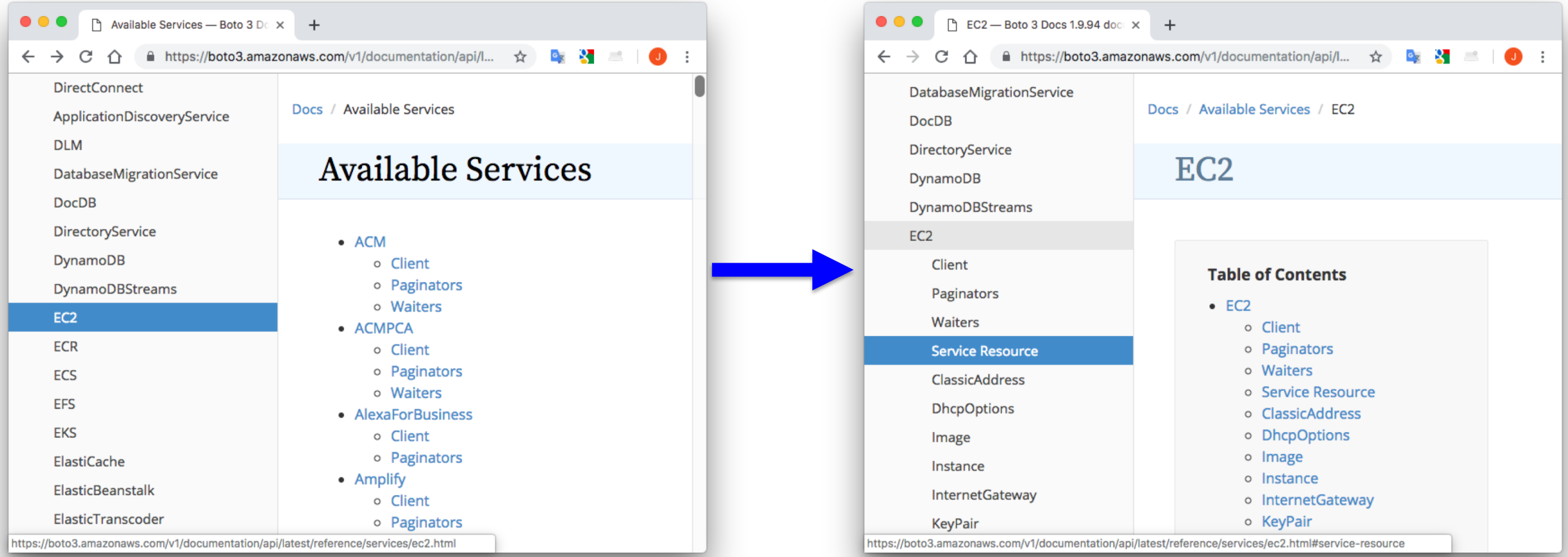
Navigating the Boto3 documentation

- Documentation home: https://boto3.readthedocs.io
- Mix of:
  - User guide / tutorial / examples
  - Full references



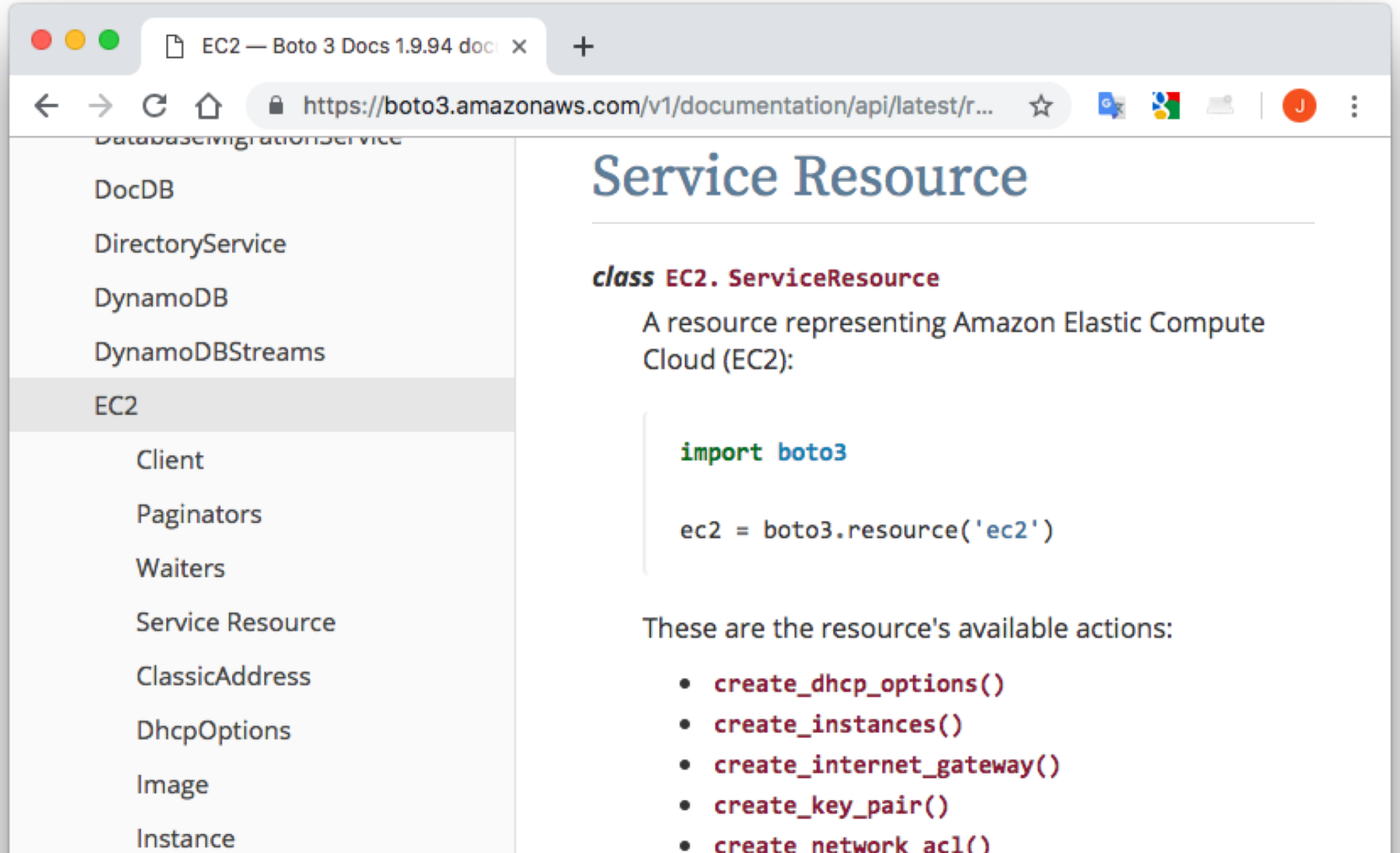
Navigating the Boto3 documentation

- **Available Services** is most useful reference section
  - Select the service you want; e.g. EC2, S3



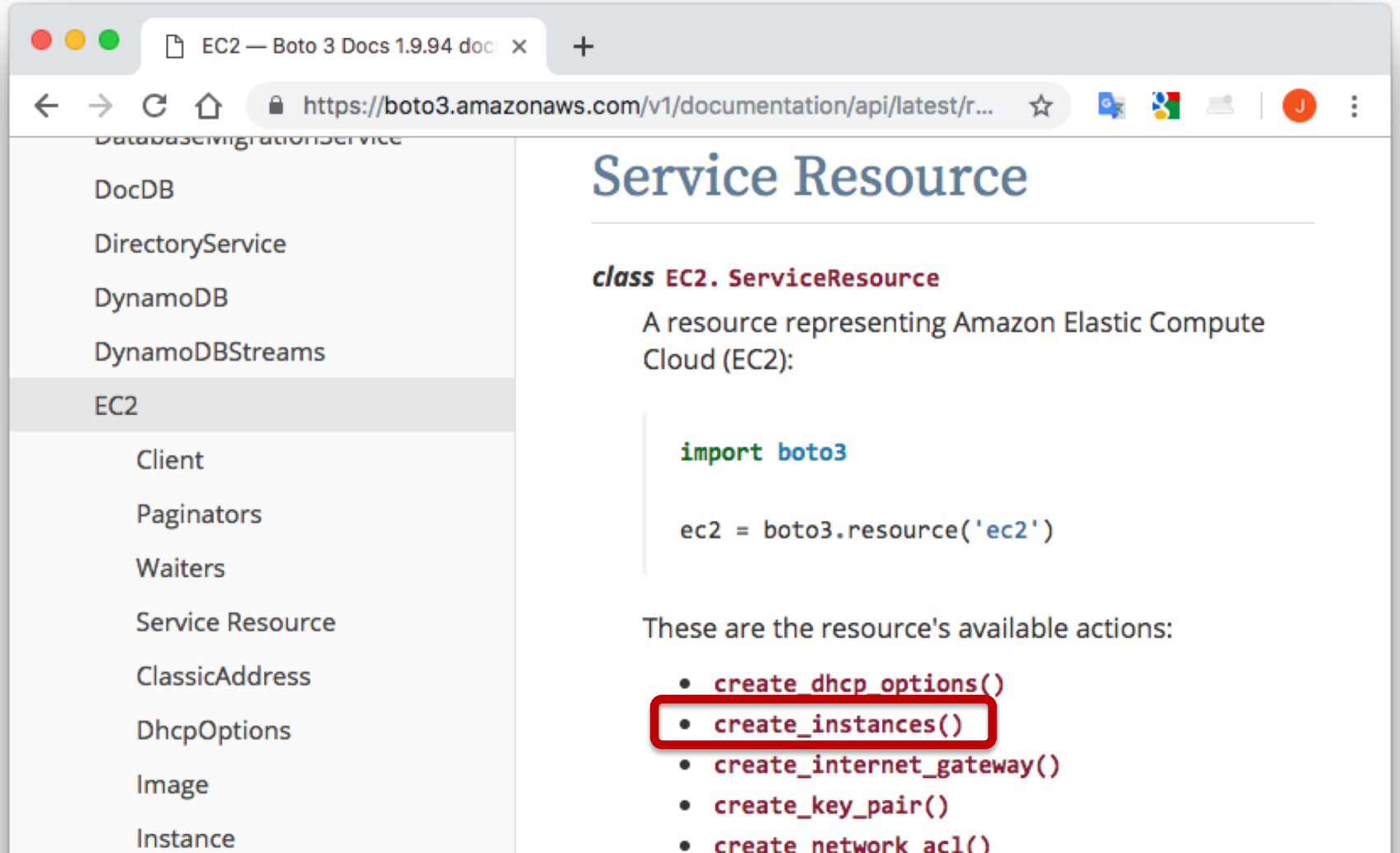
Navigating the Boto3 documentation

– And then the specific resource API (or client API)



Navigating the Boto3 documentation

– From which you can look at the detail of a specific method, for example



Navigating the Boto3 documentation

– From which you can look at the detail of a specific method, for example