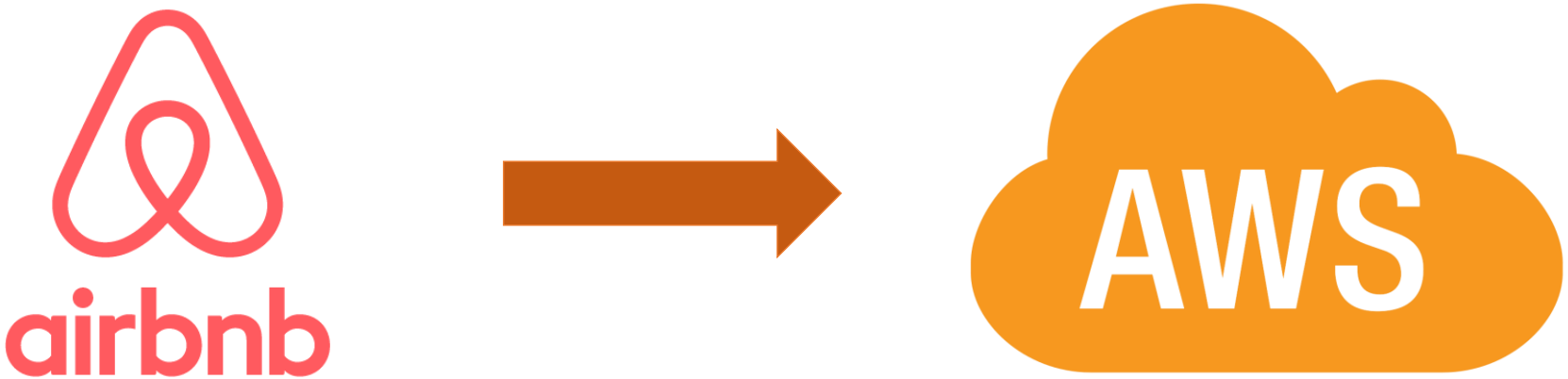
AirBnB API
to
AWS



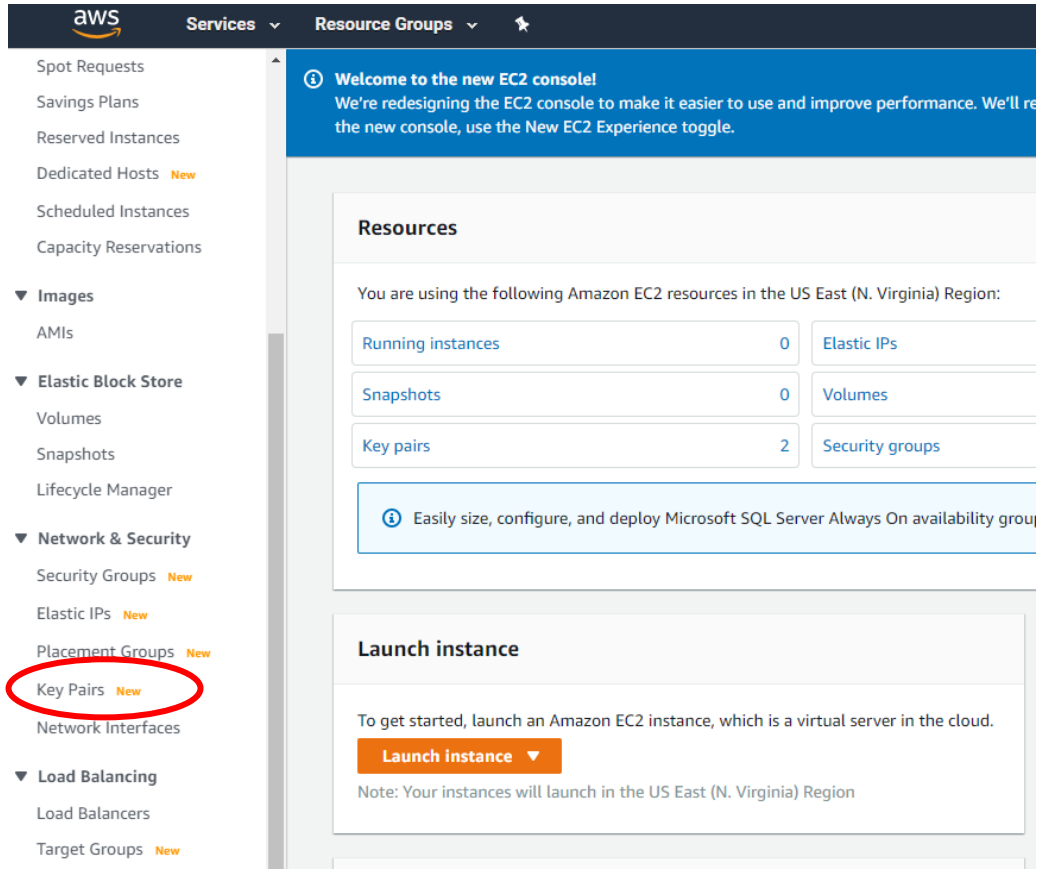
By: Jhoan Stiven Zamora Caicedo

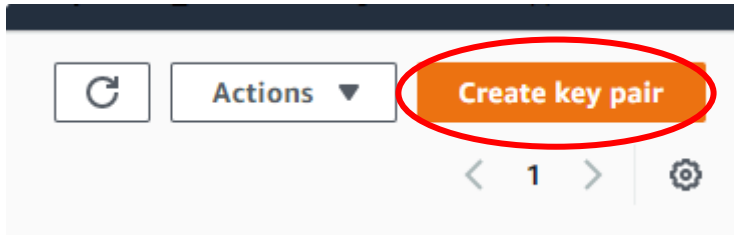
APP DEPLOYMENT

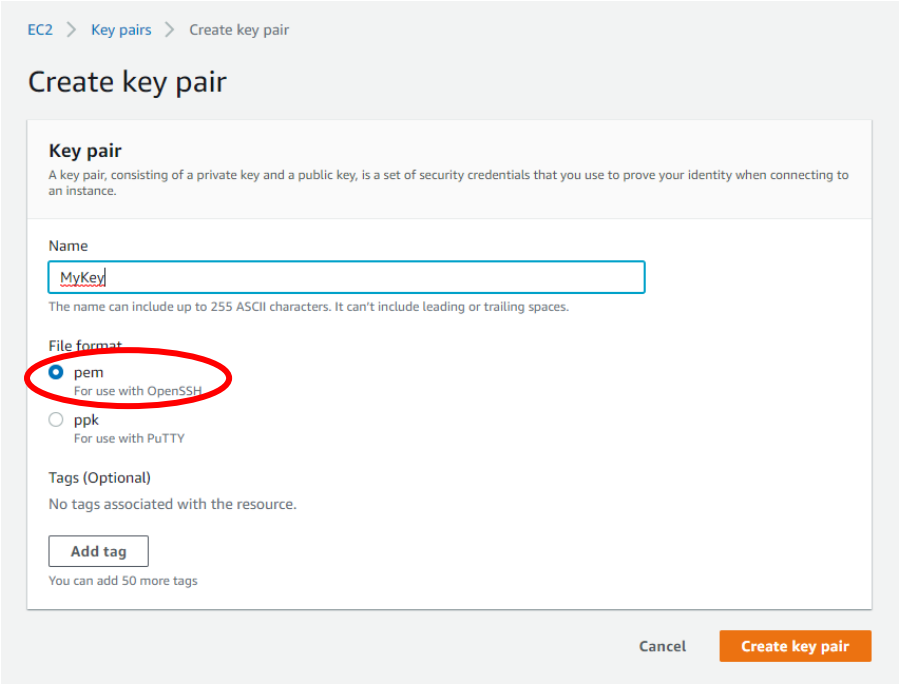
IT IS RECOMMENDED TO CREATE A REGULAR USER IN AWS.

ACADEMIC USERS CAN RESULT IN A LOT OF PERMISSION ISSUES OR OTHER PROBLEMS.

(IN EC2 Dash Board) GENERATE KEYS IN CASE DEBUGGING IS NECESSARY







SAVE THE KEY AND DON'T LOSE IT, ALTHOUGH IF YOU DO, IT AIN'T THE END OF THE WORLD, YOU CAN CREATE AS MUCH AS YOU NEED

(IN Elastic Beanstalk Dash Board)

Elastic Beanstalk > Environments

All environments

Actions | Create a new environment

Filter results matching the display values

1

Environment name	Health	Application name	Date created	Last modified	URL	Running versions	Platform	Platform state	Tier name

Empty

No environments to display.

Elastic Beanstalk > Create environment

Select environment tier

AWS Elastic Beanstalk has two types of environment tiers to support different types of web applications. Web servers are standard applications that listen for and then process HTTP requests, typically over port 80. Workers are specialized applications that have a background processing task that listens for messages on an Amazon SQS queue. Worker applications post those messages to your application by using HTTP.

- (●) Web server environment
  Run a website, web application, or web API that serves HTTP requests.
  Learn more
- ( ) Worker environment
  Run a worker application that processes long-running workloads on demand or performs tasks on a schedule.
  Learn more

Cancel | Select

Create a web server environment

Launch an environment with a sample application or your own code. By creating an environment, you allow AWS Elastic Beanstalk to manage AWS resources and permissions on your behalf. Learn more

Application information

Application name

AirBnB_API

Up to 100 Unicode characters, not including forward slash (/).

▸ **Application tags (optional)**

Environment information

Choose the name, subdomain, and description for your environment. These cannot be changed later.

Environment name

AirbnbApi-env

Domain

Leave blank for autogenerated value .us-east-1.elasticbeanstalk.

Check availability

Description

Platform

Managed platform
Platforms published and maintained by AWS Elastic Beanstalk. Learn more

Custom platform
Platforms created and owned by you.

Platform

Python

Platform branch

Python 3.7 running on 64bit Amazon Linux 2

Platform version

3.1.0 (Recommended)

Application code

Sample application
Get started right away with sample code.

Existing version
Application versions that you have uploaded for AirBnB_API.

-- Choose a version --

Upload your code
Upload a source bundle from your computer or copy one from Amazon S3.

Don't click this yet

Cancel | Configure more options | Create environment

Software

AWS X-Ray: disabled
Rotate logs: disabled (default)
Log streaming: disabled (default)
Environment properties: 1
PYTHONPATH

Edit

Instances

IMDSv1: enabled
Root volume type: container default
Root volume size (GB): container default
Root volume IOPS: container default
Security groups: *none*

Edit

Capacity

Environment type: single instance
Fleet composition: On-Demand instance
EC2 instance type: t2.micro
EC2 image ID:ami-094d9cd7536bcef21

Edit

Load balancer

This configuration does not contain a load balancer.

Rolling updates and deployments

Deployment policy: All at once
Rolling updates: disabled

Edit

Security

Service role: arn:aws:iam::633519187499:role/aws-elasticbeanstalk-service-role
Virtual machine key pair: --
Virtual machine instance profile: aws-elasticbeanstalk-ec2-role

Edit

Monitoring

Health reporting system: Enhanced
Health event log streaming: disabled

Edit

Managed Updates

Managed updates: enabled
Weekly update window: Thu:22:00 UTC

Edit

Notifications

Email address: --

Edit

Network

This environment is not part of a VPC.

Edit

Database

Engine: --
Instance class: --
Storage (GB): --
Multi-AZ: --

Edit

Tags

Tags: *none*

Edit

Cancel Previous Create environment

Elastic Beanstalk > Create environment

Modify security

Service role

Service role

aws-elasticbeanstalk-service-role

Virtual machine permissions

EC2 key pair

Select the key you created before

IAM instance profile

aws-elasticbeanstalk-ec2-role

Cancel Save

Software

AWS X-Ray: disabled
Rotate logs: disabled (default)
Log streaming: disabled (default)
Environment properties: 1
PYTHONPATH

Edit

Instances

IMDSv1: enabled
Root volume type: container default
Root volume size (GB): container default
Root volume IOPS: container default
Security groups: *none*

Edit

Capacity

Environment type: single instance
Fleet composition: On-Demand instance
EC2 instance type: t2.micro
EC2 image ID:ami-094d9cd7536bcef21

Edit

Load balancer

This configuration does not contain a load balancer.

Rolling updates and deployments

Deployment policy: All at once
Rolling updates: disabled

Edit

Security

Service role: arn:aws:iam::633519187499:role/aws-elasticbeanstalk-service-role
Virtual machine key pair: **Your key**
Virtual machine instance profile: aws-elasticbeanstalk-ec2-role

Edit

Monitoring

Health reporting system: Enhanced
Health event log streaming: disabled

Edit

Managed Updates

Managed updates: enabled
Weekly update window: Thu:22:00 UTC

Edit

Notifications

Email address: --

Edit

Network

This environment is not part of a VPC.

Edit

Database

Engine: --
Instance class: --
Storage (GB): --
Multi-AZ: --

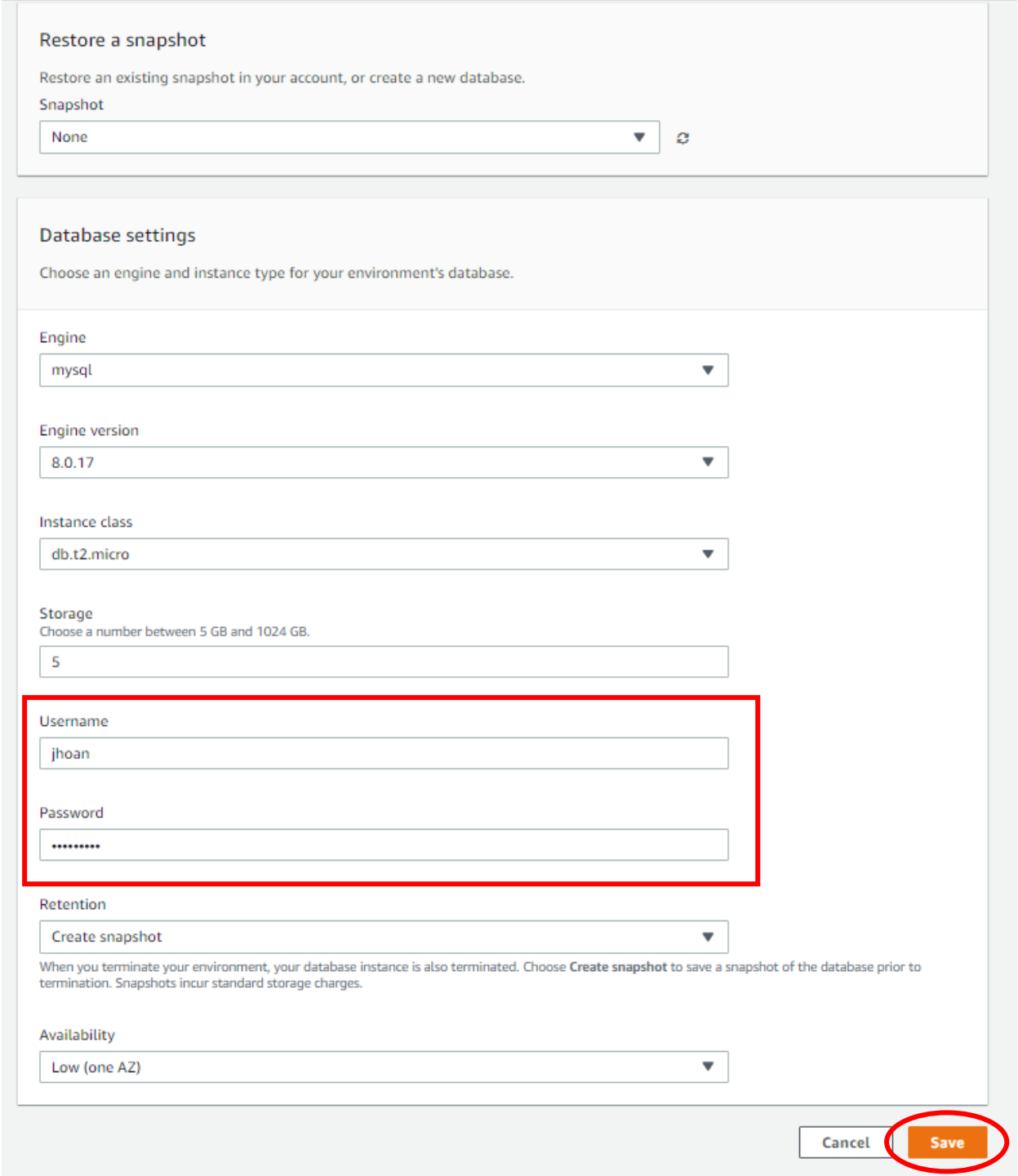
Edit

Tags

Tags: *none*

Edit

Cancel | Previous | Create environment



You can write down the user and password, although you won't be using them directly in your app unless you want to interact with the mysql console.

(Which is not necessary because the ORM is doing everything)

Software

AWS X-Ray: disabled
Rotate logs: disabled (default)
Log streaming: disabled (default)
Environment properties: 1
PYTHONPATH

Edit

Instances

IMDSv1: enabled
Root volume type: container default
Root volume size (GB): container default
Root volume IOPS: container default
Security groups: *none*

Edit

Capacity

Environment type: single instance
Fleet composition: On-Demand instance
EC2 instance type: t2.micro
EC2 image ID:ami-094d9cd7536bcef21

Edit

Load balancer

This configuration does not contain a load balancer.

Rolling updates and deployments

Deployment policy: All at once
Rolling updates: disabled

Edit

Security

Service role: arn:aws:iam::633519187499:role/aws-elasticbeanstalk-service-role
Virtual machine key pair: **Your key**
Virtual machine instance profile: aws-elasticbeanstalk-ec2-role

Edit

Monitoring

Health reporting system: Enhanced
Health event log streaming: disabled

Edit

Managed Updates

Managed updates: enabled
Weekly update window: Thu:22:00 UTC

Edit

Notifications

Email address: --

Edit

Network

This environment is not part of a VPC.

Edit

Database

Engine: --
Instance class: --
Storage (GB): --
Multi-AZ: --

Edit

Tags

Tags: *none*

Edit

Cancel | Previous | **Create environment**

The process can take between 10 and 20 minutes the first time. You just have to wait.

PROJECT CONFIGURATION

INSTALL PIP AND PYTHON VENV

```
AirBnB_clone_v4$ sudo apt-get install python3-pip
AirBnB_clone_v4$ sudo apt-get install python3-venv
```

CREATE VIRTUAL ENVIRONMENT

```
AirBnB_clone_v4$ python3 -m venv ./venv
```

ACTIVATE VIRTUAL ENVIRONMENT

```
AirBnB_clone_v4$ source ./venv/bin/activate
(venv) AirBnB_clone_v4$
```

DEACTIVATE

```
(venv) AirBnB_clone_v4$ deactivate
AirBnB_clone_v4$
```

INSTALL DEPENDENCIES (Don’t use sudo)

```
(venv) AirBnB_clone_v4$ pip3 install mysqlclient
(venv) AirBnB_clone_v4$ pip3 install SQLAlchemy==1.2.5
(venv) AirBnB_clone_v4$ pip3 install flask
(venv) AirBnB_clone_v4$ pip3 install flasgger
(venv) AirBnB_clone_v4$ pip3 install flask_cors
```

LIST DEPENDENCIES

```
(venv) AirBnB_clone_v4$ pip3 freeze
```

CHANGE ENV VARIABLES AND ADD PORT VARIABLE

```
HBNB_MYSQL_USER = getenv('RDS_USERNAME')
HBNB_MYSQL_PWD = getenv('RDS_PASSWORD')
HBNB_MYSQL_HOST = getenv('RDS_HOSTNAME')
HBNB_MYSQL_PORT = getenv('RDS_PORT')
HBNB_MYSQL_DB = getenv('RDS_DB_NAME')
```

SET DEFAULT DB MODE

```
HBNB_ENV = "db"
```

*(In db_storage and __init__ modules)

ADD PORT TO CREATE ENGINE

```
create_engine('mysql+mysqldb://{}:{}@{}:{}/{}'.
format(HBNB_MYSQL_USER, HBNB_MYSQL_PWD, HBNB_MYSQL_HOST,
HBNB_MYSQL_PORT, HBNB_MYSQL_DB))
```

TEST LOCALLY USING VENV

```
(venv) AirBnB_clone_v4$ export FLASK_APP=api.v1.app
(venv) AirBnB_clone_v4$RDS_USERNAME=test_user
RDS_PASSWORD=test_pwd RDS_HOSTNAME=localhost RDS_PORT=3306
RDS_DB_NAME=test_db flask run
```

(This does not run as __main__ nor it will do in AWS)

CHECK IF SERVER IS RUNNING LOCALLY

```
AirBnB_clone_v4$ curl localhost:5000/api/v1/status
{
  "status": "OK"
}
```

YOU CAN ALSO CHECK MYSQL TO VERIFY IF TABLES ARE CREATED

ELASTIC BEANSTALK CONFIG FILES

COMMANDS AND CONFIGURATION ON DEPLOYMENT
(venv does not matter, you can have it on or off)

```
AirBnB_clone_v4$ mkdir .ebextensions
```

These files run in order of name (can be any)

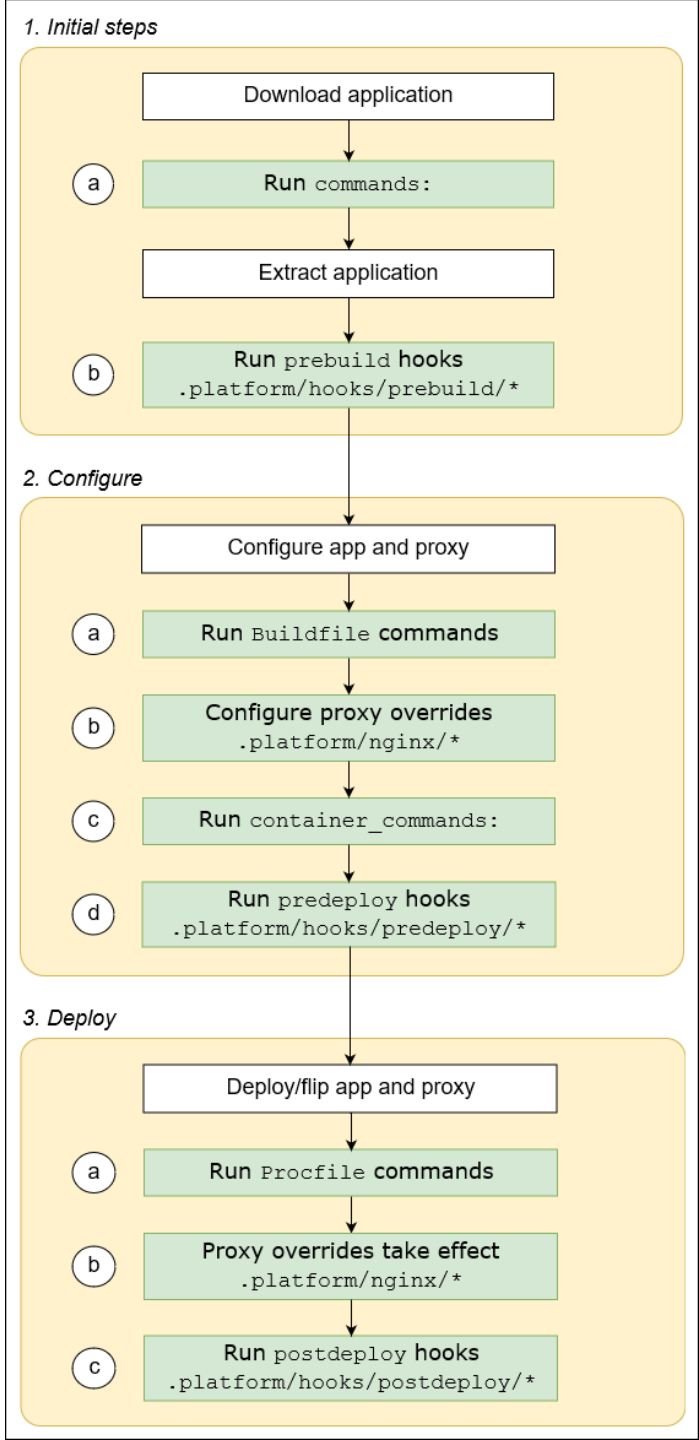
```
.ebextension$ touch 01_mysql.config
.ebextension$ touch 02_wsgi_path.config
```

On 01_mysql.config (Spaces are important)

```
commands:
  01_mysql_devel:
    command: "sudo yum install -y mysql-devel"
```

On 02_wsgi_path.config (Spaces are important)

```
option_settings:
  aws:elasticbeanstalk:container:python:
    WSGIPath: api.v1.app
```



CREATE REQUIREMENTS FILE

```
(venv) AirBnB_clone_v4$ pip3 freeze > requirements.txt
```

(DON’T DO THIS, IT IS SIMPLY BONUS INFORMATION)

IF YOU WANT TO INSTALL DEPENDENCIES FROM FILE

```
(venv) AirBnB_clone_v4$ pip3 install -r requirements.txt
```

APP DEPLOYMENT

IT IS RECOMMENDED TO CREATE A REGULAR USER IN AWS.

ACADEMIC USERS CAN RESULT IN A LOT OF PERMISSION ISSUES OR OTHER PROBLEMS.

Elastic Beanstalk > Environments > AirbnbApi-env

AirbnbApi-env

AirbnbApi-env.eba-4tzphxcn.us-east-1.elasticbeanstalk.com (e-inhmpfidih)
Application name: AirBnB_API

Refresh | Actions

Health

Ok

Causes

Running version

Sample Application

Upload and deploy

Platform

Python 3.7 running on 64bit
Amazon Linux 2/3.1.0

Change

Recent events

Show all

1

Time	Type	Details
2020-08-20 11:33:06 UTC-0500	INFO	Successfully launched environment: AirbnbApi-env
2020-08-20 11:33:04 UTC-0500	INFO	Application available at AirbnbApi-env.eba-4tzphxcn.us-east-1.elasticbeanstalk.com.
2020-08-20 11:32:50 UTC-0500	INFO	Added instance [i-057f95836fb5f5ea2] to your environment.
2020-08-20 11:32:50 UTC-0500	INFO	Environment health has transitioned from Pending to Ok. Initialization completed 12 seconds ago and took 8 minutes.
2020-08-20 11:32:29 UTC-0500	INFO	Instance deployment completed successfully.

Once ready, you can click the link given to you in order to verify the sample app was deployed.

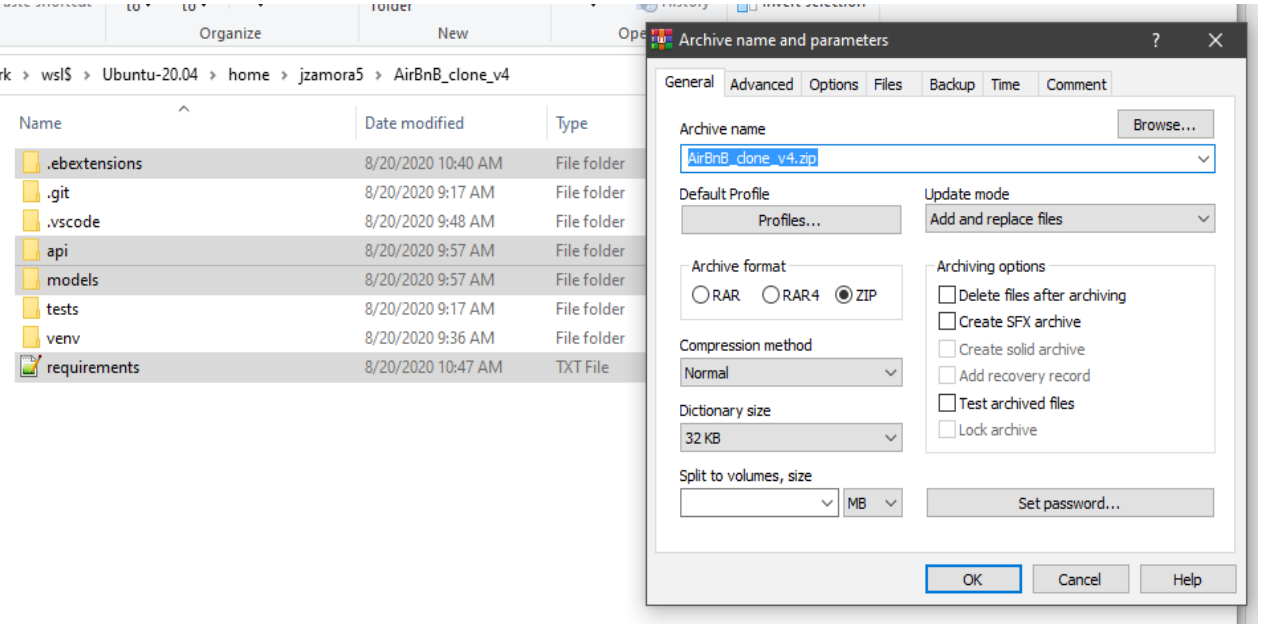
Congratulations

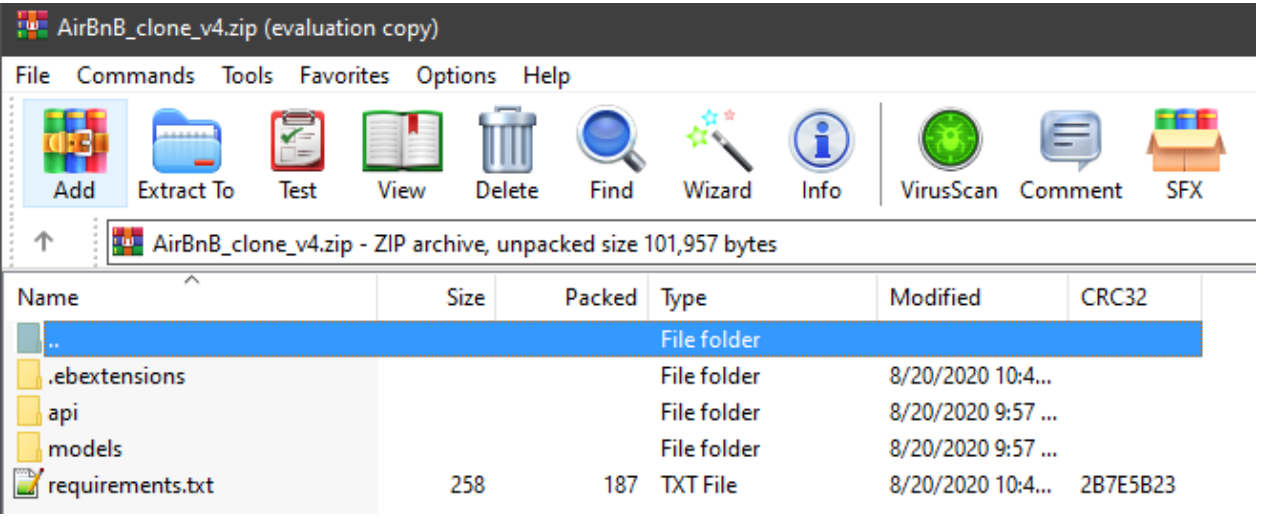
Your first AWS Elastic Beanstalk Python Application is now running on your own dedicated environment in the AWS Cloud

This environment is launched with Elastic Beanstalk Python Platform

What's Next?

- AWS Elastic Beanstalk overview
- AWS Elastic Beanstalk concepts
- Deploy a Django Application to AWS Elastic Beanstalk
- Deploy a Flask Application to AWS Elastic Beanstalk
- Customizing and Configuring a Python Container
- Working with Logs





Now you need to compress your app into a ZIP file.

(MUST BE THIS FORMAT)

ONLY INCLUDE THE HIGHLIGHTED FOLDERS AND THE REQUIREMENTS FILE

(IGNORE THE VENV FOLDER AND ANYTHING ELSE THAT IS NOT PART OF YOUR MAIN APP)

YOU MUST ZIP ALL OF THESE FOLDERS AND FILE DIRECTLY

(DO NOT ZIP A FOLDER CONTAINING THEM)

WRONG

ZIP AirBnB_v4 Folder

GOOD

ZIP (.ebextensions, api, models, requirements)

Elastic Beanstalk > Environments > AirbnbApi-env

AirbnbApi-env

AirbnbApi-env.eba-4tzphxcn.us-east-1.elasticbeanstalk.com (e-inhmpfidih)
Application name: AirBnB_API

Refresh | Actions

Health

Ok

Causes

Running version

Sample Application

Upload and deploy

Platform

Python 3.7 running on 64bit
Amazon Linux 2/3.1.0

Change

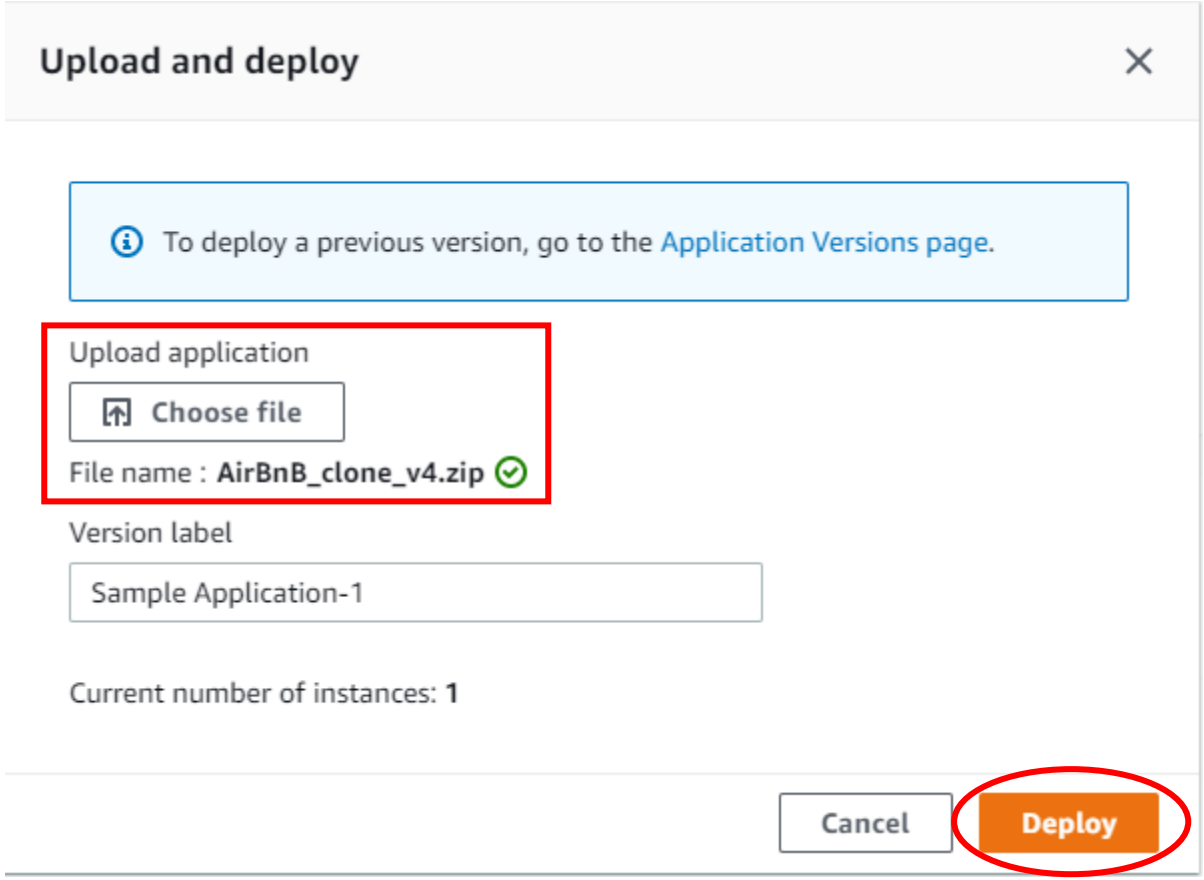
Recent events

Show all

1

Time	Type	Details
2020-08-20 11:33:06 UTC-0500	INFO	Successfully launched environment: AirbnbApi-env
2020-08-20 11:33:04 UTC-0500	INFO	Application available at AirbnbApi-env.eba-4tzphxcn.us-east-1.elasticbeanstalk.com.
2020-08-20 11:32:50 UTC-0500	INFO	Added instance [i-057f95836fb5f5ea2] to your environment.
2020-08-20 11:32:50 UTC-0500	INFO	Environment health has transitioned from Pending to Ok. Initialization completed 12 seconds ago and took 8 minutes.
2020-08-20 11:32:29 UTC-0500	INFO	Instance deployment completed successfully.

Time to upload your app!

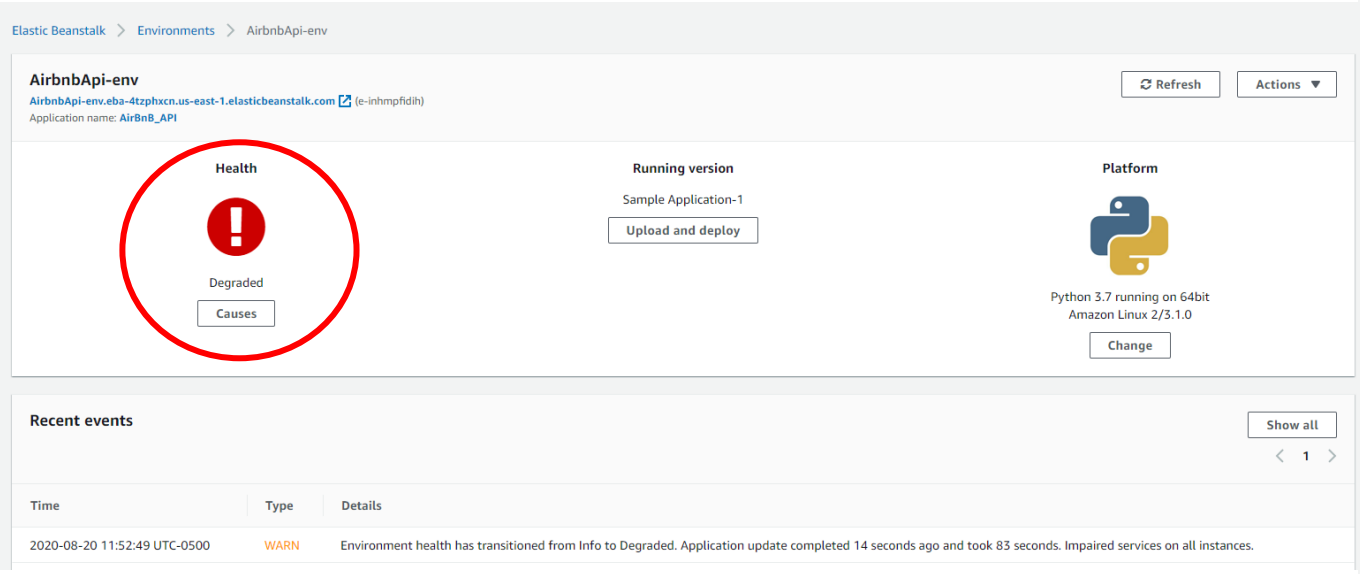


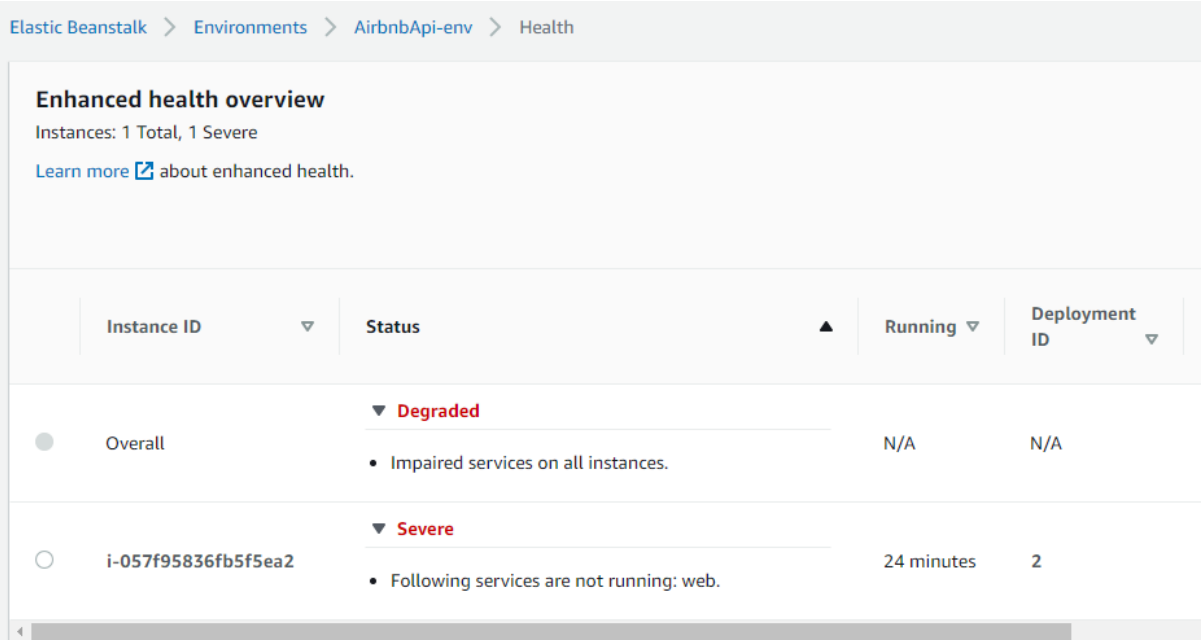
You can let the version label be set automatically or specify one, but make sure to keep the same format:

name-number (And the number should increase on each app deployment)

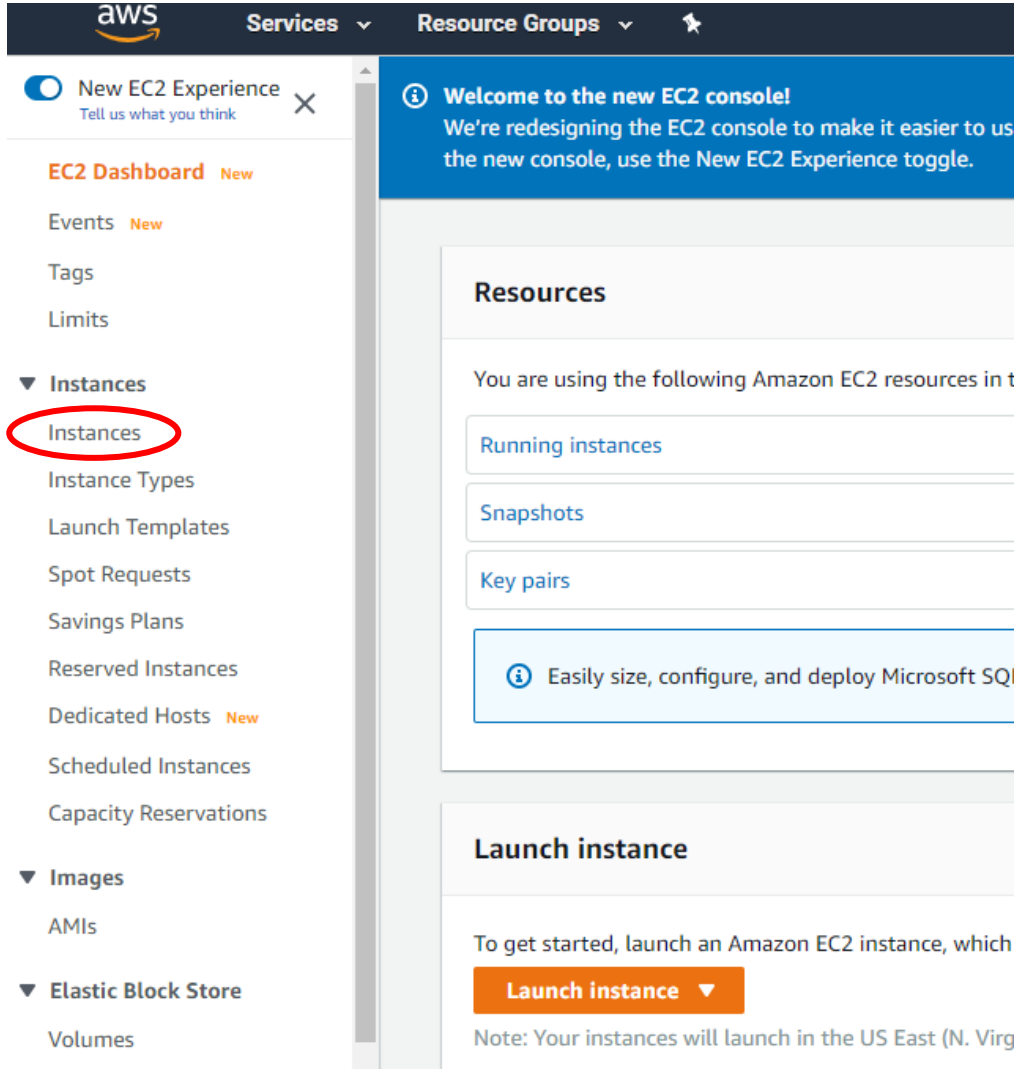
Otherwise you will run into versioning issues.

If the deployment fails you will have to SSH into the EC2 instance and debug the system because the causes shown in Elastic Beanstalk are ambiguous.

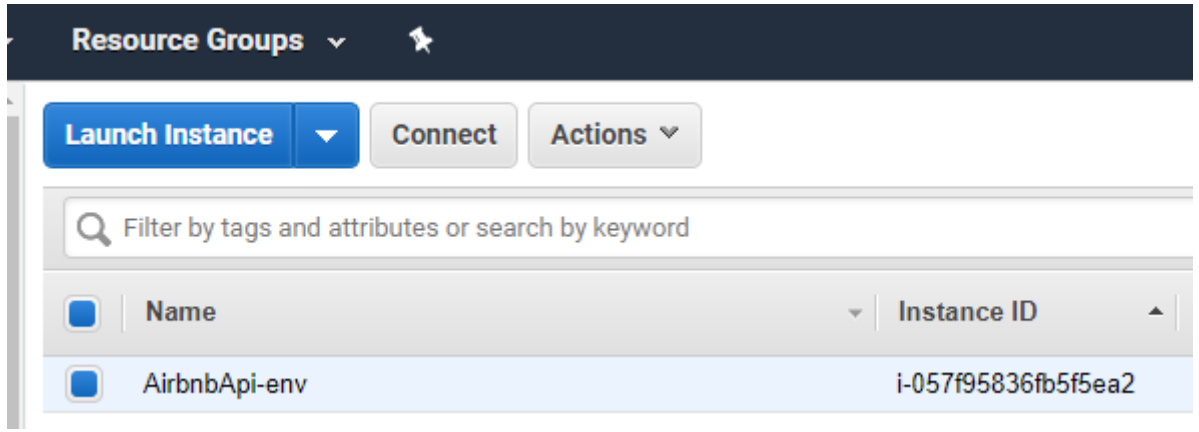




(IN EC2 Dash Board) Find your instance



It should have the name you defined in Elastic Beanstalk



Scroll to the right and find its Public IP, it shouds also show the key you defined before.

Now you can SSH into the server with the IP you just found.

The default use for ec2 instances is ec2-user. So simply run the next command in your terminal.

```
~$ ssh -i path_to_key ec2-user@instance_ip
```

```
➜  ~ ssh -i [Your key] ec2-user@54.237.163.186
The authenticity of host '54.237.163.186 (54.237.163.186)' can't be established.
ECDSA key fingerprint is SHA256:gozq8AcT5YjH0aKLDnC4u2vdlmlQVmUpvzL1jaYrRNE.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '54.237.163.186' (ECDSA) to the list of known hosts.
  _____ _           _   _      ____                     _        _ _
 | ____| | __   ___| |_(_) ___| __ )  ___  __ _ _ __  ___| |_ __ _| | | __
 |  _| | |/ _ \/ __| __| |/ __|  _ \ / _ \/ _\ | '_ \/ __| __/ _\ | | |/ /
 | |___| | (_| \__ \ |_| | (__| |_) |  __/ (_| | | | \__ \ || (_| | |   <
 |_____|_|\__,_|___/\__|_|\___|____/ \___|\__,_|_| |_|___/\__\__,_|_|_|\_\

 Amazon Linux 2 AMI

 This EC2 instance is managed by AWS Elastic Beanstalk. Changes made via SSH
 WILL BE LOST if the instance is replaced by auto-scaling. For more information
 on customizing your Elastic Beanstalk environment, see our documentation here:
 http://docs.aws.amazon.com/elasticbeanstalk/latest/dg/customize-containers-ec2.html

[ec2-user@ip-172-31-94-205 ~]$
```

If your app was uploaded to the server but failed when being run, it will be located in the next path for python 3.7:

/var/app/current

If you used other version like python 3.6, the app might be located in:

/opt/python/bundle

```
[ec2-user@ip-172-31-94-205 ~]$ cd /var/app/current
[ec2-user@ip-172-31-94-205 current]$ ls
Procfile  api  models  requirements.txt
[ec2-user@ip-172-31-94-205 current]$ cd ..
[ec2-user@ip-172-31-94-205 app]$ ls
current  venv
[ec2-user@ip-172-31-94-205 app]$ source venv/staging-LQM1lest/bin/activate
(staging) [ec2-user@ip-172-31-94-205 app]$
```

The app also runs on a virtual environment, so if you want to test it locally on the server, you must go back one directory, and activate the venv.

The path changes a little (adding a staging folder) but the process for activation is the same.

Instead of (venv), we get (staging), but it indicates we are already in the virtual environment.

If you simply want to check the logs of the deployment in order to find possible issues, you can go to the directory.

/var/log

Here you will find all the logs for different things such as elastic beanstalk boot, gunicorn and nginx initialization, etc.

```
[ec2-user@ip-172-31-94-205 ~]$ cd /var/log/
[ec2-user@ip-172-31-94-205 log]$ ls
amazon        cfn-init-cmd.log  cloud-init-output.log  eb-cfn-init-call.log  healthd   nginx    web.stdout.log
boot.log      cfn-init.log      cloud-init.log         eb-cfn-init.log       httpd     rotated  wtmp
btmp          cfn-wire.log      cron                   eb-engine.log         maillog   sa       xray
cfn-hup.log   chrony            dmesg                  eb-publish.log        messages  secure   yum.log
[ec2-user@ip-172-31-94-205 log]$
```

You can simply cat the files; some of them require sudo.

```
[ec2-user@ip-172-31-94-205 log]$ sudo cat web.stdout.log
```

This one is for checking gunicorn initialization.

One of the most common issues is this one:

```
Aug 20 16:52:01 ip-172-31-94-205 web: [2020-08-20 16:52:01 +0000] [5546] [INFO] Using worker: threads
Aug 20 16:52:01 ip-172-31-94-205 web: [2020-08-20 16:52:01 +0000] [5553] [INFO] Booting worker with pid: 5553
Aug 20 16:52:02 ip-172-31-94-205 web: Failed to find attribute 'application' in 'api.v1.app'.
Aug 20 16:52:02 ip-172-31-94-205 web: [2020-08-20 16:52:02 +0000] [5553] [INFO] Worker exiting (pid: 5553)
Aug 20 16:52:02 ip-172-31-94-205 web: [2020-08-20 16:52:02 +0000] [5546] [INFO] Shutting down: Master
Aug 20 16:52:02 ip-172-31-94-205 web: [2020-08-20 16:52:02 +0000] [5546] [INFO] Reason: App failed to load.
[ec2-user@ip-172-31-94-205 log]$
```

Which means the name of the app in flask is incorrect.

(By default AWS searches for application instead of app inside the python file)

01_mysql.config | 02_wsgi_path.config | app.py | requirements.txt

api > v1 > app.py > not_found

```python
from models import storage
from api.v1.views import app_views
from os import environ
from flask import Flask, render_template, make_response, jsonify
from flask_cors import CORS
from flasgger import Swagger
from flasgger.utils import swag_from

application = app = Flask(__name__)
app.config['JSONIFY_PRETTYPRINT_REGULAR'] = True
app.register_blueprint(app_views)
cors = CORS(app, resources={r"/api/v1/*": {"origins": "*"}})

@app.teardown_appcontext
def close_db(error):
    """ Close Storage """
    storage.close()


@app.errorhandler(404)
def not_found(error):
    """ 404 Error
    ---
    responses:
      404:
        description: a resource was not found
    """
    return make_response(jsonify({'error': "Not found"}), 404)

app.config['SWAGGER'] = {
    'title': 'AirBnB clone Restful API',
    'uiversion': 3
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 2: Python

```
(venv) ➜ AirBnB_clone_v4 git:(master) ✗
```

Because we do not want to go through the hassle of changing every place where we have the word app, we can take advantage of python referencing, and do this:

```python
app = Flask(__name__)
```

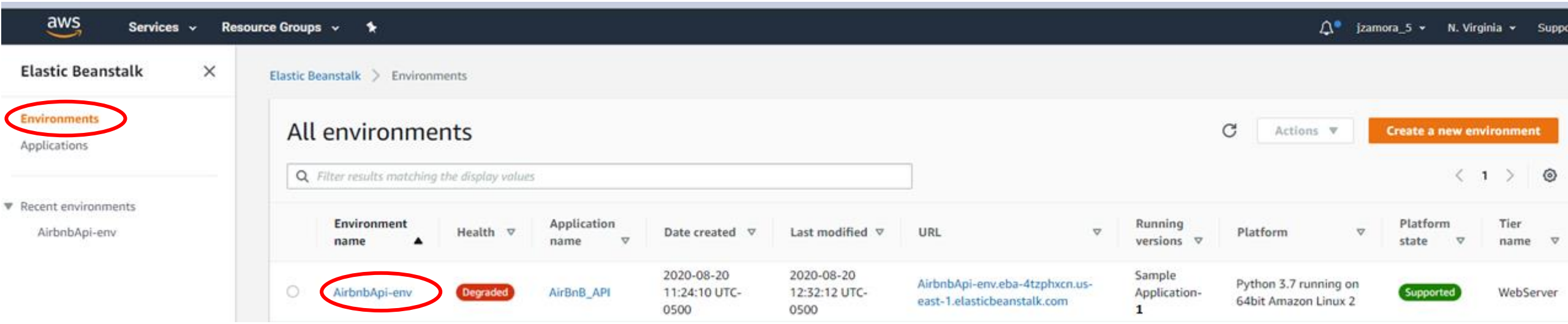
↓

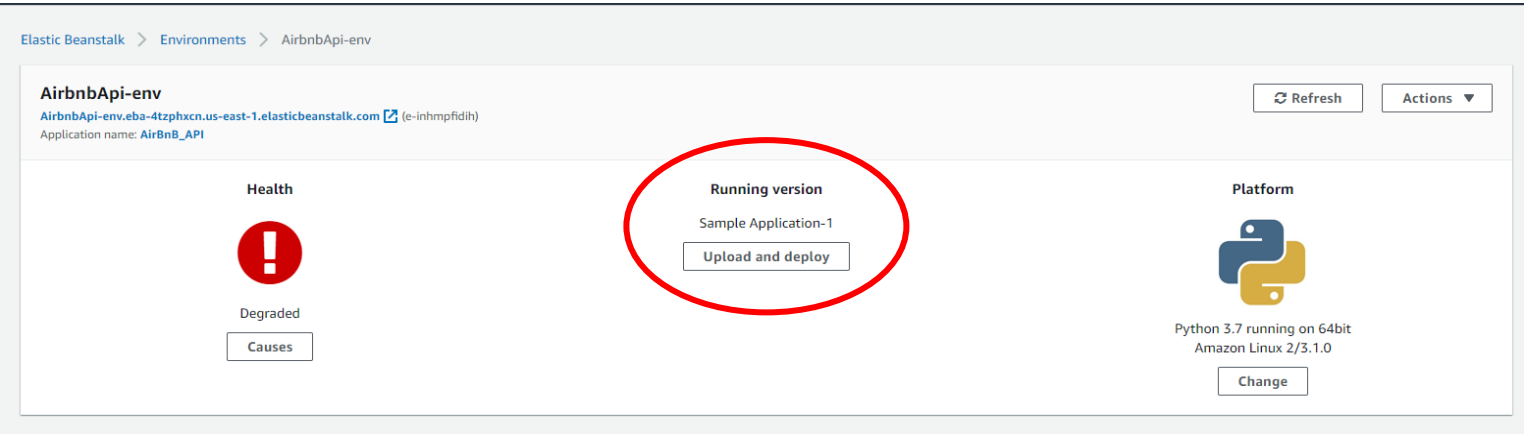
```python
application = app = Flask(__name__)
```

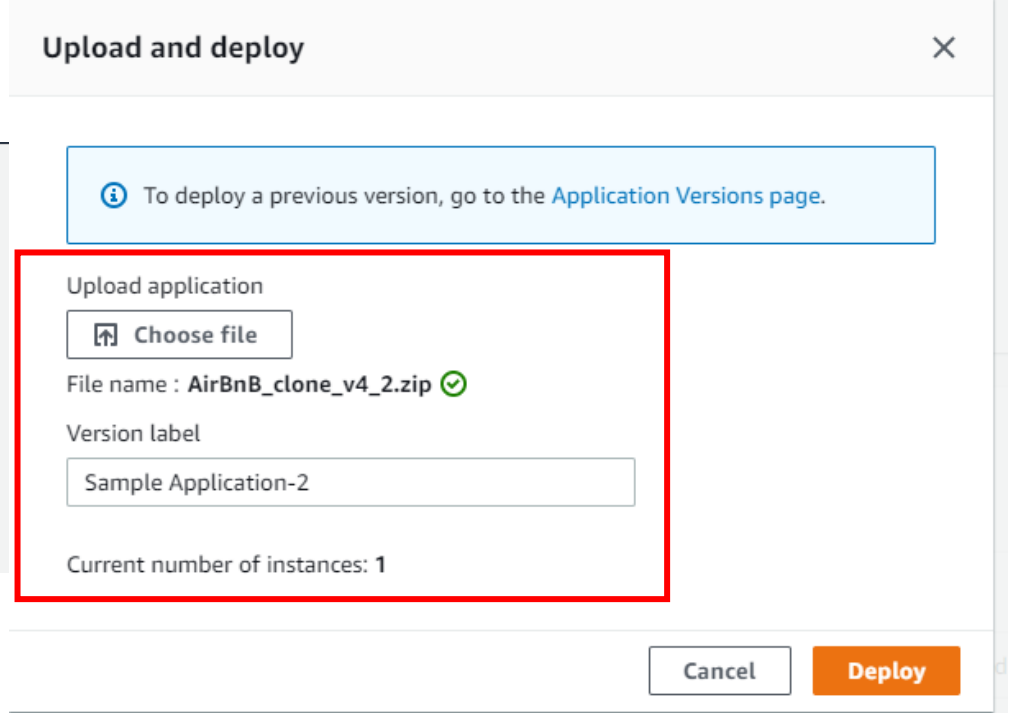
Now that the issue is fixed in the code, you simply create a new ZIP file, just as before.

Name	Date modified	Type	Size
.ebextensions	8/20/2020 10:40 AM	File folder	
.git	8/20/2020 9:17 AM	File folder	
.vscode	8/20/2020 9:48 AM	File folder	
api	8/20/2020 9:57 AM	File folder	
models	8/20/2020 9:57 AM	File folder	
tests	8/20/2020 9:17 AM	File folder	
venv	8/20/2020 9:36 AM	File folder	
AirBnB_clone_v4	8/20/2020 11:41 AM	WinRAR ZIP archive	61 KB
AirBnB_clone_v4_2	8/20/2020 12:19 PM	WinRAR ZIP archive	61 KB
requirements	8/20/2020 10:47 AM	TXT File	1 KB

In Elastic Beanstalk go to your environments, and deploy the new ZIP file.







If you did everything correctly, your APP should have green Health.

Elastic Beanstalk > Environments > AirbnbApi-env

AirbnbApi-env

AirbnbApi-env.eba-4tzphxcn.us-east-1.elasticbeanstalk.com (e-inhmpfidih)

Application name: AirBnB_API

Refresh | Actions

Health

Ok

Causes

Running version

Sample Application-2

Upload and deploy

Platform

Python 3.7 running on 64bit Amazon Linux 2/3.1.0

Change

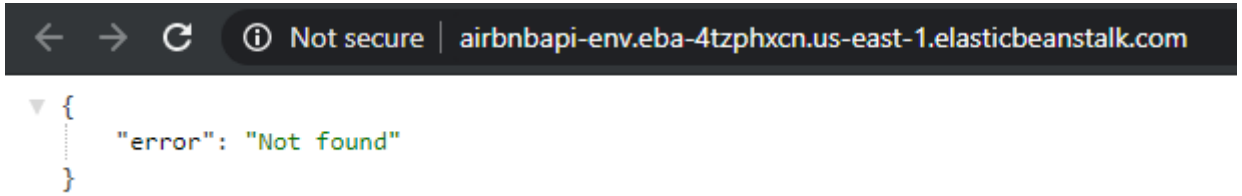
Recent events

Show all

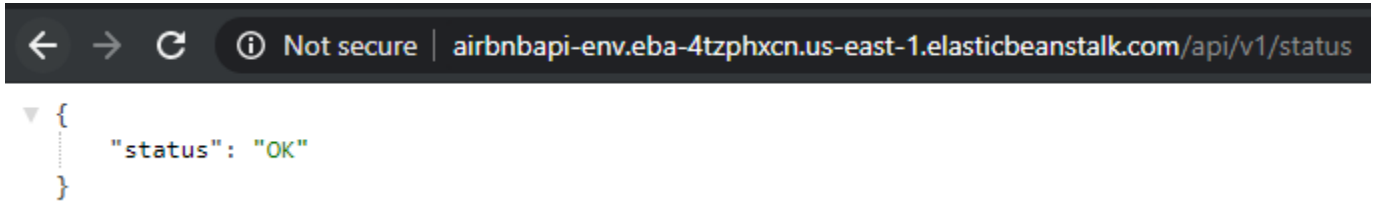
1

Time	Type	Details
2020-08-20 12:37:46 UTC-0500	INFO	Environment health has transitioned from Degraded to Ok. Application update completed 41 seconds ago and took 15 seconds.
2020-08-20 12:37:00 UTC-0500	INFO	Environment update completed successfully.
2020-08-20 12:37:00 UTC-0500	INFO	New application version was deployed to running EC2 instances.
2020-08-20 12:36:54 UTC-0500	INFO	Instance deployment completed successfully.
2020-08-20 12:36:51 UTC-0500	INFO	Instance deployment successfully generated a 'Procfile'.

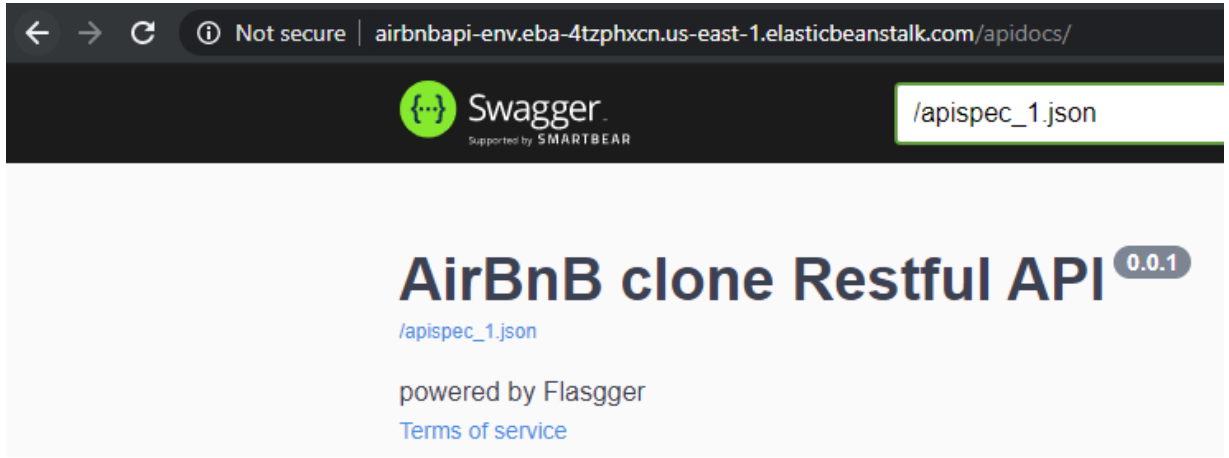
You can now try the link of your API. (The root has no endpoint in flask)



You can access the defined endpoints such as status



If you created documentation with Swagger, you can also access it



Parameters

Cancel

Name	Description
request * required (body)	Edit Value |

```
{
  "name": "Louisiana"
}
```

Cancel

Parameter content type

application/json

Execute | Clear

Responses

Response content type application/json

Curl

```
curl -X POST "http://airbnbapi-env.eba-4tzphxcn.us-east-1.elasticbeanstalk.com/api/v1/states" -H "accept: application/json" -H "Content-Type: application/json" -d "{ \"name\": \"Louisiana\"}"
```

Request URL

```
http://airbnbapi-env.eba-4tzphxcn.us-east-1.elasticbeanstalk.com/api/v1/states
```

Server response

Code	Details
201	Response body

```
{
  "__class__": "State",
  "created_at": "2020-08-20T17:41:16.343982",
  "id": "447c1fff-2016-4520-a06f-2352376d0b47",
  "name": "Louisiana",
  "updated_at": "2020-08-20T17:41:16.344036"
}
```

Download

Response headers

```
access-control-allow-origin: http://airbnbapi-env.eba-4tzphxcn.us-east-1.elasticbeanstalk.com
connection: keep-alive
content-length: 194
content-type: application/json
date: Thu, 20 Aug 2020 17:41:16 GMT
server: nginx/1.18.0
vary: Origin
```

Responses

Code	Description
201	Request completed successfully
400	Missing name or Not Valid JSON

You can use postman, curl, or even swagger to try out the different endpoints in the API.

Here swagger is used to POST a state with name Louisiana.

The state was created successfully.

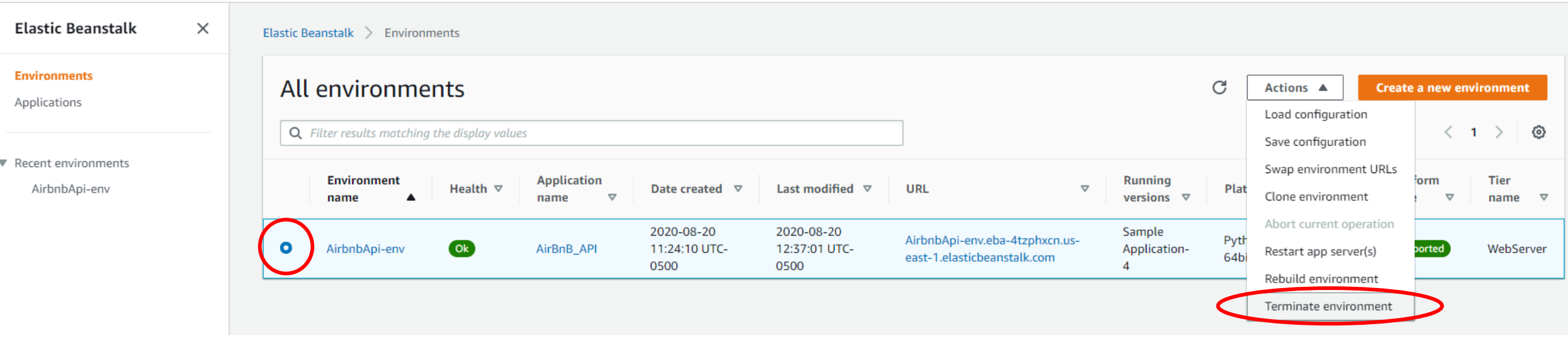
Not secure | airbnbapi-env.eba-4tzphxcn.us-east-1.elasticbeanstalk.com/api/v1/states


```
[
  {
    "__class__": "State",
    "created_at": "2020-08-20T17:41:16.000000",
    "id": "447c1fff-2016-4520-a06f-2352376d0b47",
    "name": "Louisiana",
    "updated_at": "2020-08-20T17:41:16.000000"
  }
]
```

Now that you have a working ZIP of your file, you can easily deploy it with elastic beanstalk in the future!

Remember servers can cost money if you surpass the AWS free tier, so make sure you terminate your APP when you don’t need it anymore.

As of this moment it is not possible to pause it, although you can schedule it to work in certain time periods during the day.



Confirm Environment Termination

Permanently terminate **AirbnbApi-env**. This action can't be undone.

- Tier: Web Server
- Platform: Python 3.7 running on 64bit Amazon Linux 2/3.1.0
- Version: Sample Application-4
- Last modified: 2020-08-20 12:37:01 UTC-0500

If you proceed with this action, the following will happen:

- *AirbnbApi-env.eba-4tzphxcn.us-east-1.elasticbeanstalk.com* will be released.
- The attached database **will be terminated**.
  Terminate *aa1f50uvcrahbip.cyhrfnmonamn.us-east-1.rds.amazonaws.com:3306* with snapshot.
- Any additional resources associated with your Elastic Beanstalk environment **will be destroyed**.

Enter the name of the environment to confirm:

AirbnbApi-env

Cancel **Terminate**

This can take some time, but after it is done, your app should appear like this:

Elastic Beanstalk > Environments

All environments

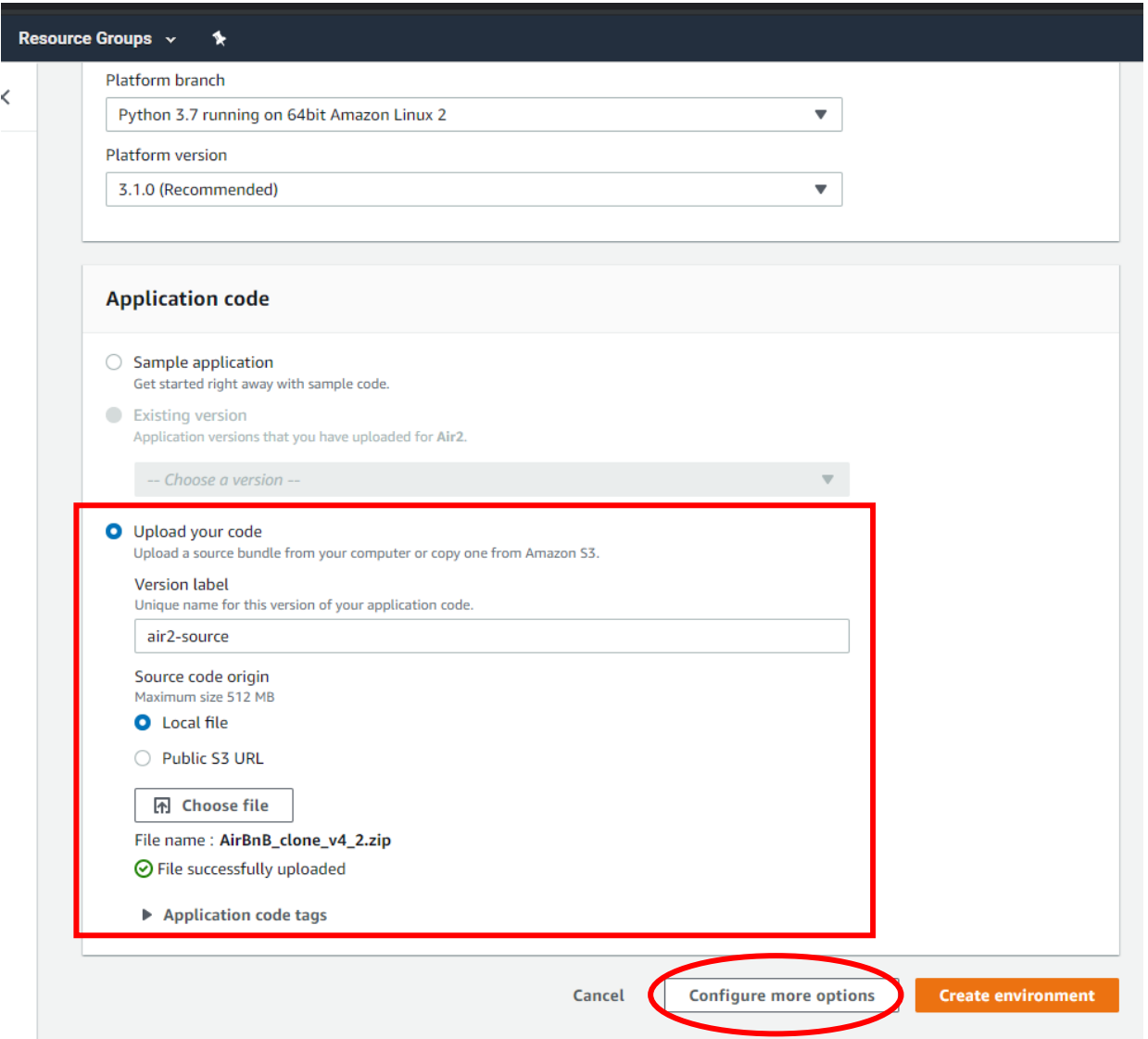
Actions | Create a new environment

Filter results matching the display values

1

Environment name	Health	Application name	Date created	Last modified	URL	Running versions	Platform	Platform state	Tier name
AirbnbApi-env (terminated)	-	AirBnB_API	2020-08-20 11:24:10 UTC-0500	2020-08-20 12:59:40 UTC-0500	AirbnbApi-env.eba-4tzphxcn.us-east-1.elasticbeanstalk.com	Sample Application-4	Python 3.7 running on 64bit Amazon Linux 2	Supported	WebServer

Next time you want to create the app, you will have to repeat the steps of creating the environment in elastic beanstalk with data base and key (although the key is not obligatory, and you can add it later if you want)



The main difference will be that instead of using a sample application you can upload your code so that it is deployed knowing that it is in a working state.

Don't forget to go to
Configure more options
to set up the DB and key before
Create Environment.

Works right away! no server configuration needed

Elastic Beanstalk > Environments > Air2-env

Air2-env

Air2-env.eba-9g2e2kvq.us-east-1.elasticbeanstalk.com (e-brsa9trvbm)

Application name: Air2

Refresh | Actions

Health

Ok

Causes

Running version

air2-source

Upload and deploy

Platform

Python 3.7 running on 64bit
Amazon Linux 2/3.1.0

Change

Recent events

Show all

1

Time	Type	Details
2020-08-20 13:42:37 UTC-0500	INFO	Successfully launched environment: Air2-env
2020-08-20 13:42:36 UTC-0500	INFO	Application available at Air2-env.eba-9g2e2kvq.us-east-1.elasticbeanstalk.com.
2020-08-20 13:42:18 UTC-0500	INFO	Added instance [i-0e491b42dd5d811dc] to your environment.
2020-08-20 13:42:04 UTC-0500	INFO	Instance deployment completed successfully.
2020-08-20 13:42:02 UTC-0500	INFO	Instance deployment successfully generated a 'Procfile'.