

For this assignment, I have used an elastic beanstalk application that will load a docker image containing the python flask application. The beanstalk application is setup with load balancing and autoscaling, since I am using my personal AWS account, I have used t2.micro instances.

I am familiar with AWS by using the console, for this assignment I started to use the console to setup the beanstalk application and then used the knowledge obtained to code using AWS CDK for infrastructure as code. AWS CDK project I have placed in the infra folder will create a beanstalk application, the configurations are not completed. I was not able to complete the code within the 3 hours timeline.

In the three-hour timeline, I was able to create a docker image of the python application and upload to docker hub. I created a beanstalk application on console that will load create containers based on the docker image with autoscaling and load balancing enabled. Docker image is located on: [DockerHubLink](#). I originally planned to setup using ECR but considering the 3 hours timeline, I decided to use beanstalk instead.

Elastic Beanstalk > Environments > Pythonhello-env > Configuration

## Modify capacity

Configure the compute capacity of your environment and Auto Scaling settings to optimize the number of instances used.

### Auto Scaling Group

Environment type

Load balanced ▼

Instances

Min 1

Max 2

Fleet composition

Choose a mix of On-Demand and Spot Instances with multiple instance types. Spot Instances are automatically launched at the lowest available price. [Learn more](#)

☒ On-Demand instances

☐ Combine purchase options and instances

## Certificate

The self signed SSL certificate was created using the commands listed below. I have imported the certificate body and private key into AWS ACM.

The screenshot shows the AWS Certificate Manager console. At the top, there's a table with columns: Name, Domain name, Additional names, Status, Type, In use?, and Renewal eligibility. A single certificate is listed with the domain name 'pythonhello-env.eba-qby28efb.ca-central-1.elasticbeanstalk.com' and status 'Issued'.

Below the table, the 'Status' section shows 'Status: Issued' and 'Detailed status: The cert was imported at 2021-03-25T00:47:50UTC'. There is a 'Reimport certificate' button.

The 'Details' section provides more information:

- Type: Imported
- In use?: Yes
- Domain name: pythonhello-env.eba-qby28efb.ca-central-1.elasticbeanstalk.com
- Number of additional names: 0
- Identifier: f3f76bd1-cf8e-4e27-88f8-2af351d1e0e0
- Serial number: 06:b4:33:71:98:b4:6b:6f:34:ad:42:ce:57:69:ed:76:26:a8:a8:84
- Associated resources:
  - arn:aws:elasticloadbalancing:ca-central-1:172127287036:loadbalancer/app/awseb-AWSEB-FGJQWDYHZECEB/b6e6143ead1ac664
- Imported at: 2021-03-25T00:47:50UTC
- Not after: 2022-03-25T00:43:49UTC
- Expires in: 364 Days
- Public key info: RSA 2048-bit
- Signature algorithm: SHA256WITHRSA
- ARN: arn:aws:acm:ca-central-1:172127287036:certificate/f3f76bd1-cf8e-4e27-88f8-2af351d1e0e0
- Validation state: None

There is a 'Tags' section at the bottom.

```
openssl genrsa 2048 > privatekey1.pem
openssl req -new -key privatekey1.pem -out csr1.pem
openssl x509 -req -days 365 -in csr1.pem -signkey privatekey1.pem -out server.crt
cat server.crt # for certitacafate body
cat privatekey1.pem # for private key
```

I then updated the load balancer attached to the beanstalk application to use the certificate for port 443 and disabled port 80.

The screenshot shows the 'Listeners' section of the AWS Load Balancing console. It includes a description: 'You can specify listeners for your load balancer. Each listener routes incoming client traffic on a specified port using a specified protocol to your environment processes. By default, we've configured your load balancer with a standard web server on port 80.'

There are two buttons: 'Actions' and '+ Add listener'.

Below is a table of listeners:

<input type="checkbox"/>	Port	Protocol	SSL certificate	Default process	Enabled
<input type="checkbox"/>	443	HTTPS	pythonhello-env.eba-bxhfct5d.ca-central-1.elasticbeanstalk.com - 9f5aa83f-cd0d-4383-b749-4b913acdd672	default	<input checked="" type="checkbox"/>
<input type="checkbox"/>	80	HTTP	--	default	<input type="checkbox"/>

The link: <http://pythonflaskenv.eba-p8bqqr5.ca-central-1.elasticbeanstalk.com/> is the beanstalk application that is created using the code uploaded on github. This is not complete as it is missing load balancing and routing to only https.

The link: <https://pythonhello-env.eba-bxhfct5d.ca-central-1.elasticbeanstalk.com/> is the beanstalk application that was created using the console. This has routing to only HTTPS and load balancing with autoscaling.

HTTP Access:



## This site can't be reached

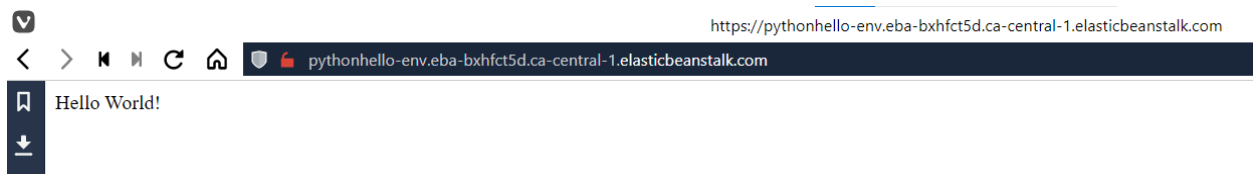
**pythonhello-env.eba-bxhfct5d.ca-central-1.elasticbeanstalk.com** took too long to respond.

Try:

- Checking the connection
- [Checking the proxy and the firewall](#)
- [Running Windows Network Diagnostics](#)

ERR\_CONNECTION\_TIMED\_OUT

HTTPs access:



Using curl command:

```
dshan@LAPTOP-GU1D7DPN:/mnt/c/Users/dshan$ curl -k "https://pythonhello-env.eba-bxhfct5d.ca-central-1.elasticbeanstalk.com"
Hello World!dshan@LAPTOP-GU1D7DPN:/mnt/c/Users/dshan$
dshan@LAPTOP-GU1D7DPN:/mnt/c/Users/dshan$ curl -k "http://pythonhello-env.eba-bxhfct5d.ca-central-1.elasticbeanstalk.com"
curl: (28) Failed to connect to pythonhello-env.eba-bxhfct5d.ca-central-1.elasticbeanstalk.com port 80: Connection timed out
dshan@LAPTOP-GU1D7DPN:/mnt/c/Users/dshan$ |
```

The GitHub repository is located: <https://github.com/shanmudi/pythonflask>. The app folder contains the code for the application such as the python flask script and the docker file to create the docker image containing the python script. The infra folder contains the CDK code. The GitHub pipeline if completed would run a nodejs container and install aws cdk package. The aws cli would be configured using secrets on the repo that contain the access and secret key. Once completed the command “cdk deploy” would synthesize the code into a CloudFormation template and build the infrastructure.