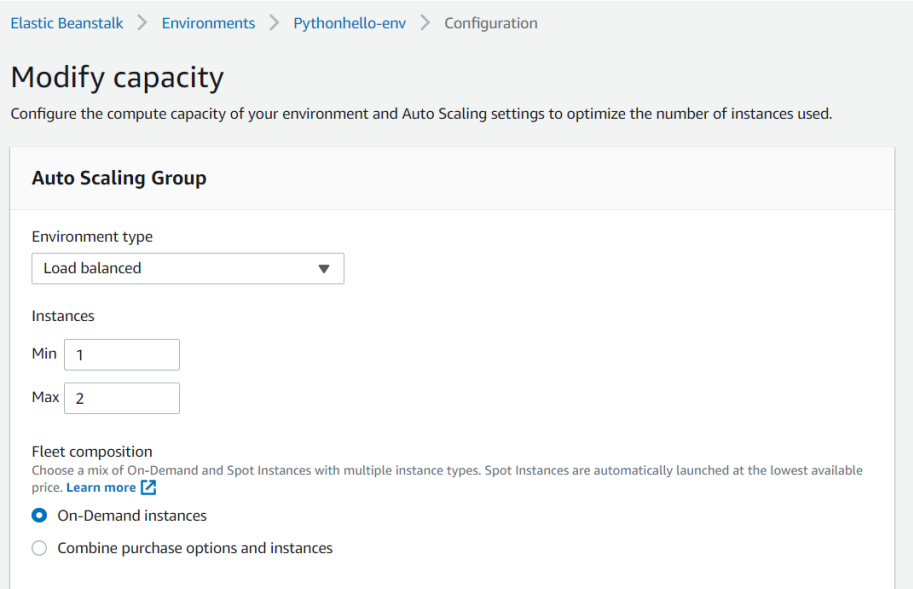
For this assignment, I have used an elastic beanstalk application that will load a docker image containing the python flask application. The beanstalk application is setup with load balancing and autoscaling, since I am using my personal AWS account, I have used t2.micro instances.

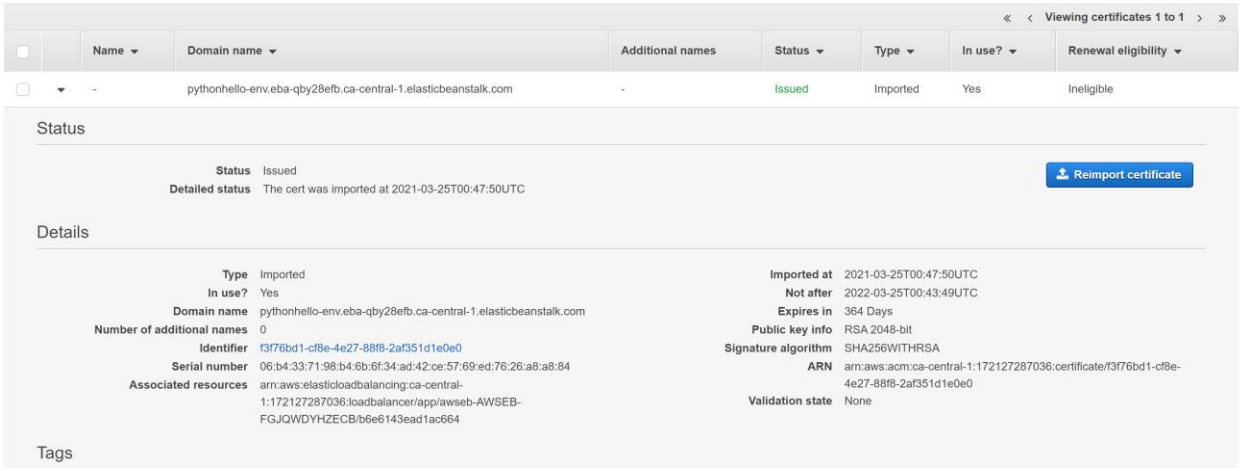
I am familiar with AWS by using the console, for this assignment I started to use the console to setup the beanstalk application and then used the knowledge obtained to code using AWS CDK for infrastructure as code. AWS CDK project I have placed in the infra folder will create a beanstalk application, 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.



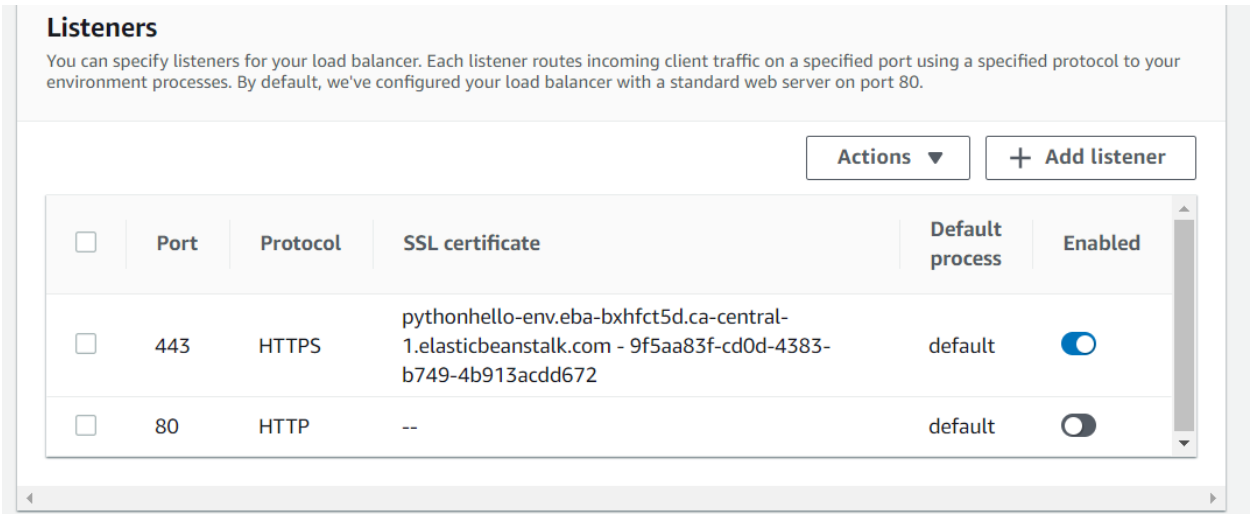
Certificate

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



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 certiticafate body
cat privatekey1.pem # for private key

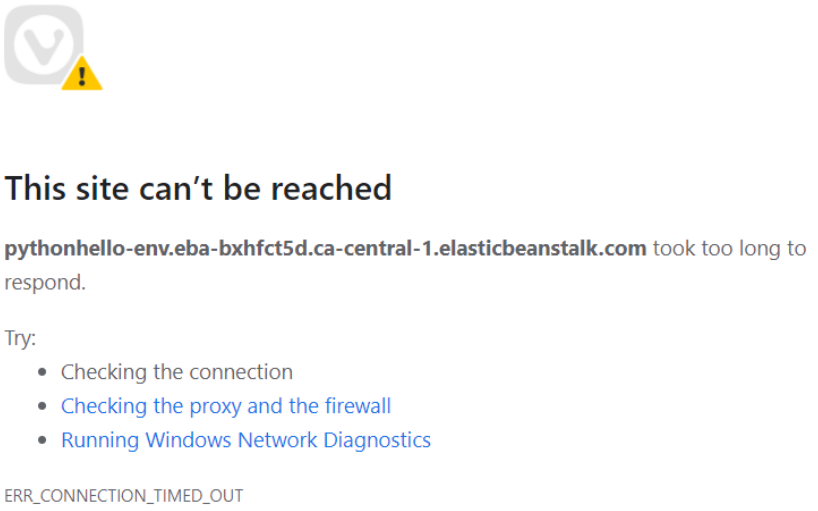
I then updated the load balancer attached to the beanstalk application to use the certificate for port 443 and disabled port 80.



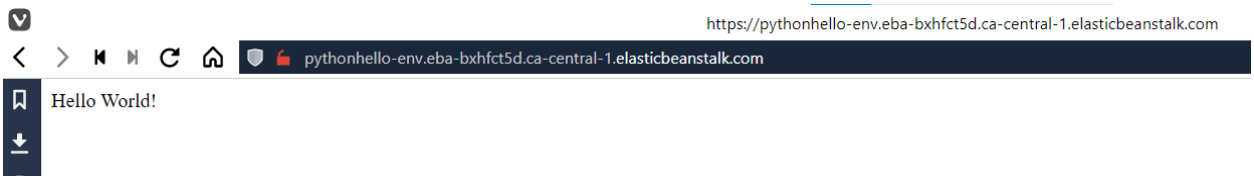
The link: http://pythonflaskenv.eba-p8bqqrt5.ca-central-1.elasticbeanstalk.com/ is the beanstalk application that is created using the code uploaded on github. This is not complete as it is missing load balancing and routing to only https.

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 autoscanling.

HTTP Access:



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.