

# PROJECT REPORT

Group number: 9

Sagar Premchand Yatam: 000306T058

## PROJECT DESCRIPTION:

The main aim of this project is to deploy a simple calculator scalable web application using AWS Cloud infrastructure and implement the auto scaling policy group for the instance. The auto scaling shall be performed on the instance if there is high CPU utilization than average threshold utilization with increase in data usage and more of users.

The features that are considered are scalability with respect to computation and security of data at rest.

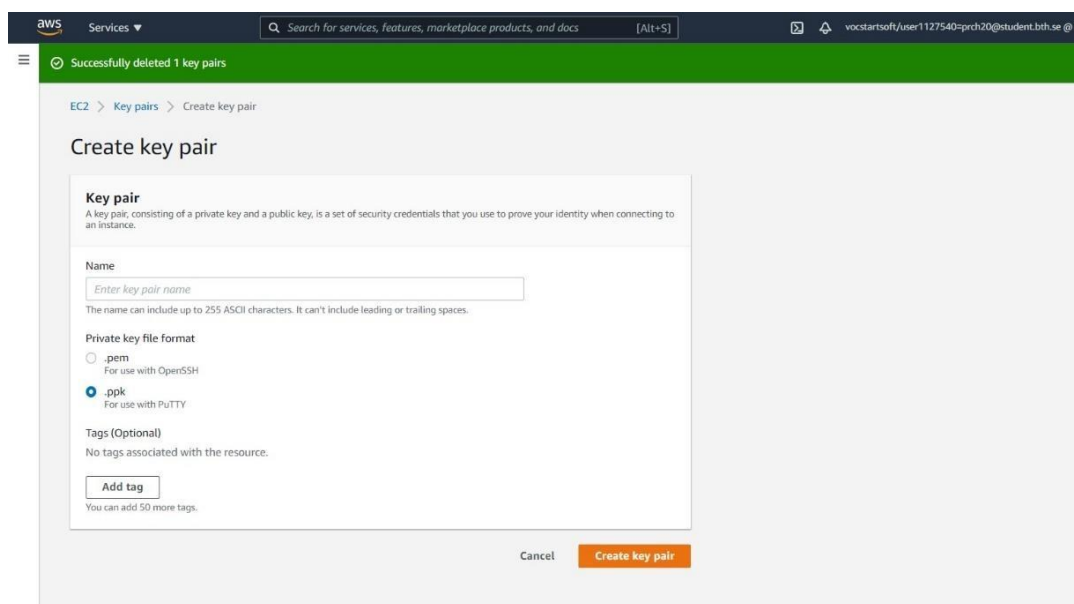
- The large amount of input data given while using the application can be handled and fulfill the feature scalability with respect to CPU utilization.
- If user does not perform any operations i.e., at the rest time the data present previously is secured while using the application. Hence, the feature security of data at rest is achieved.

AWS cloud technology is chosen to accomplish the given task. The main intention is to use web application framework i.e., Flask with python programming in AWS cloud infrastructure and test/validate our application with large amount of data from many users which in turn increases CPU utilization.

## IMPLEMENTATION:

### STEP-1: LAUNCHING THE EC2 INSTANCE

1. Firstly, key pair should be created to launch the EC2 instance.



2. On the AWS console choose EC2 instance option and click on launch instance button.
3. Select the ubuntu server image from the list of images (AMI).



## 6. Adding storage

The screenshot shows the 'Add Storage' step in the AWS Management Console. It displays a table with storage configuration details for the root volume.

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Throughput (MB/s)	Delete on Termination	Encryption
Root	/dev/sda1	snap-0a52a0f51496c3762	8	General Purpose SSD (gp2)	100 / 3000	N/A	<input checked="" type="checkbox"/>	Not Encrypted

Below the table is an 'Add New Volume' button and a note: 'Free tier eligible customers can get up to 30 GiB of EBS General Purpose (SSD) or Magnetic storage. Learn more about free usage tier eligibility and usage restrictions.'

## 7. Configuring the security group.

- Click on Launch instance button and to access the created instance we need to use putty on the system and launch the instance using IP address and previously created key file. the instance will be launched as shown in the below figure.

The screenshot shows the 'Connect to instance' page in the AWS Management Console for instance `i-055d47f336546048c`. It provides options to connect via EC2 Instance Connect, Session Manager, SSH client, or EC2 Serial Console. The 'Public IP address' is `172.31.92.225`. A note states: 'Note: In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.'

Next to it is a terminal window showing the output of the `lsb_release -a` command:

```
ubuntu@ip-172-31-92-225:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description: Ubuntu 18.04.2 LTS
Release: 18.04
Codename: bionic

Management: https://landscape.canonical.com
Support: https://ubuntu.com/advantage

System information disabled due to load higher than 1.0

1 update can be applied immediately.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-92-225:~$
```

## STEP-2: DEPLOYING THE CALCULATOR APP

To deploy the flask application in ubuntu it is important to implement the following commands in putty:

- Updating existing packages – as the ubuntu server is a new operating system where it contains outdated packages, so it is important to update the existing packages in the operating system.

*cmd: sudo apt-get update*

```

ubuntu@ip-172-31-92-225: ~
Using username "ubuntu".
Authenticating with public key "Flask-AWS"
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-1045-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information disabled due to load higher than 1.0

1 update can be applied immediately.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-92-225:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease [101 kB]
Get:4 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 Packages [8628 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe Translation-en [5124 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 c-n-f Metadata [265 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/multiverse amd64 Packages [144 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/multiverse Translation-en [104 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/multiverse amd64 c-n-f Metadata [9136 B]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [1127 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates/main Transla

```

- ii. Installing python – to run the flask applications it is important to have python in our ubuntu server.

*cmd: sudo apt-get install python3*

```

Fetched 19.6 MB in 3s (5775 kB/s)
Reading package lists... Done
ubuntu@ip-172-31-92-225:~$ sudo apt-get install python3
Reading package lists... Done
Building dependency tree
Reading state information... Done
python3 is already the newest version (3.8.2-0ubuntu2).
python3 set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 88 not upgraded.
ubuntu@ip-172-31-92-225:~$ █

```



- iii. Installing pip i.e python package manager – after installing python pip is installed into the ubuntu server which allows us to install and manage additional libraries and dependencies that are not distributed as part of python library.

*cmd: sudo apt-get install python3-pip*

```
ubuntu@ip-172-31-92-225:~$ sudo apt-get install python3-pip
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu build-essential cpp cpp-9 dpkg-dev
  fakeroot g++ g++-9 gcc gcc-10-base gcc-9 gcc-9-base libalgorithm-diff-perl
  libalgorithm-diff-xs-perl libalgorithm-merge-perl libasan5 libatomic1 libbinutils
  libc-dev-bin libc6-dev libcc1-0 libcrypt-dev libctf-nobfd0 libctf0 libdpkg-perl
  libexpat1-dev libfakeroot libfile-fcntllock-perl libgcc-9-dev libgcc-s1 libgomp1 libisl22
  libitm1 liblsan0 libmpc3 libpython3-dev libpython3.8 libpython3.8-dev
  libpython3.8-minimal libpython3.8-stdlib libquadmath0 libstdc++-9-dev libstdc++6 libtsan0
  libubsan1 linux-libc-dev make manpages-dev python-pip-whl python3-dev python3-wheel
  python3.8 python3.8-dev python3.8-minimal zlib1g-dev
Suggested packages:
  binutils-doc cpp-doc gcc-9-locales debian-keyring g++-multilib g++-9-multilib gcc-9-doc
  gcc-multilib autoconf automake libtool flex bison gdb gcc-doc gcc-9-multilib glibc-doc
  bzip2 libstdc++-9-doc make-doc python3.8-venv python3.8-doc binfmt-support
The following NEW packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu build-essential cpp cpp-9 dpkg-dev
  fakeroot g++ g++-9 gcc gcc-9 gcc-9-base libalgorithm-diff-perl libalgorithm-diff-xs-perl
  libalgorithm-merge-perl libasan5 libatomic1 libbinutils libc-dev-bin libc6-dev libcc1-0
  libcrypt-dev libctf-nobfd0 libctf0 libdpkg-perl libexpat1-dev libfakeroot
  libfile-fcntllock-perl libgcc-9-dev libgomp1 libisl22 libitm1 liblsan0 libmpc3
  libpython3-dev libpython3.8-dev libpython3.8-minimal libpython3.8-stdlib libquadmath0
  libstdc++-9-dev libstdc++6 libtsan0 libubsan1
  linux-libc-dev make manpages-dev python-pip-whl python3-dev python3-pip python3-wheel
  python3.8-dev zlib1g-dev
The following packages will be upgraded:
  gcc-10-base libgcc-s1 libpython3.8 libpython3.8-minimal libpython3.8-stdlib libstdc++6
  python3.8 python3.8-minimal
8 upgraded, 50 newly installed, 0 to remove and 80 not upgraded.
Need to get 56.7 MB of archives.
After this operation, 214 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates/main amd64 libpython3.8 amd64 3.8.10-0ubuntu1~20.04 [1625 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates/main amd64 python3.8 amd64 3.8.10-0ubuntu1~20.04 [387 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates/main amd64 libpython3.8-stdlib amd64 3.8.10-0ubuntu1~20.04 [1675 kB]
```

- iv. Downloading flask into our server – as we are deploying a flask application our server needs to be installed with supported package called flask.

*cmd: sudo pip3 install flask*

```
ubuntu@ip-172-31-92-225:~$ sudo pip3 install flask
Collecting flask
  Downloading Flask-2.0.1-py3-none-any.whl (94 kB)
    | 94 kB 4.3 MB/s
Collecting itsdangerous>=2.0
  Downloading itsdangerous-2.0.1-py3-none-any.whl (18 kB)
Collecting Jinja2>=3.0
  Downloading Jinja2-3.0.1-py3-none-any.whl (133 kB)
    | 133 kB 29.5 MB/s
Collecting click>=7.1.2
  Downloading click-8.0.1-py3-none-any.whl (97 kB)
    | 97 kB 8.4 MB/s
Collecting Werkzeug>=2.0
  Downloading Werkzeug-2.0.1-py3-none-any.whl (288 kB)
    | 288 kB 33.8 MB/s
Collecting MarkupSafe>=2.0
  Downloading MarkupSafe-2.0.1-cp38-cp38-manylinux2010_x86_64.whl (30 kB)
Installing collected packages: itsdangerous, MarkupSafe, Jinja2, click, Werkzeug, flask
  Attempting uninstall: MarkupSafe
    Found existing installation: MarkupSafe 1.1.0
    Not uninstalling markupsafe at /usr/lib/python3/dist-packages, outside environment /usr
    Can't uninstall 'MarkupSafe'. No files were found to uninstall.
  Attempting uninstall: Jinja2
    Found existing installation: Jinja2 2.10.1
    Not uninstalling jinja2 at /usr/lib/python3/dist-packages, outside environment /usr
    Can't uninstall 'Jinja2'. No files were found to uninstall.
  Attempting uninstall: click
    Found existing installation: Click 7.0
    Not uninstalling click at /usr/lib/python3/dist-packages, outside environment /usr
    Can't uninstall 'Click'. No files were found to uninstall.
Successfully installed Jinja2-3.0.1 MarkupSafe-2.0.1 Werkzeug-2.0.1 click-8.0.1 flask-2.0.1 itsdangerous-2.0.1
```

- v. Nginx is used as reverse proxy and load balancer. We will be using it as web server for deploying our web application along with that we are using guicorn for the purpose of python web application.

*cmd: sudo apt-get install nginx*

*cmd: sudo apt-get install guicorn3*

The below figure shows the installation of nginx

```
ubuntu@ip-172-31-92-225:~$ sudo apt-get install nginx
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  fontconfig-config fonts-dejavu-core libfontconfig1 libgd3 libjpeg-turbo8
  libjpeg8 libnginx-mod-http-image-filter libnginx-mod-http-xslt-filter libnginx-mod-mail
  libnginx-mod-stream libtiff5 libwebp6 libxpm4 nginx-common nginx-core
Suggested packages:
  libgd-tools fcgiwrap nginx-doc ssl-cert
The following NEW packages will be installed:
  fontconfig-config fonts-dejavu-core libfontconfig1 libgd3 libjpeg-turbo8
  libjpeg8 libnginx-mod-http-image-filter libnginx-mod-http-xslt-filter libnginx-mod-mail
  libnginx-mod-stream libtiff5 libwebp6 libxpm4 nginx nginx-common nginx-core
0 upgraded, 17 newly installed, 0 to remove and 80 not upgraded.
Need to get 2431 kB of archives.
After this operation, 7891 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/main amd64 fonts-dejavu-core all 2.37-1 [1041 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/main amd64 fontconfig-config all 2.13.1-2ubuntu3 [28.8 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/main amd64 libfontconfig1 amd64 2.13.1-2ubuntu3 [114 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates/main amd64 libjpeg-turbo8 amd64 2.0.3-0ubuntu1.20.04.1 [117 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/main amd64 libjpeg8 amd64 8c-2ubuntu8 [2194 B]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/main amd64 libjpeg-turbo8 amd64 2.0.3-0ubuntu1.20.04.1 [117 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates/main amd64 libwebp6 amd64 0.6.1-2ubuntu0.20.04.1 [185 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates/main amd64 libtiff5 amd64 4.1.0+git191117-2ubuntu0.20.04.1 [162 kB]
```

The below figure shows the installation of guicorn

```
ubuntu@ip-172-31-92-225:~$ sudo apt-get install guicorn
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  python3-gunicorn
Suggested packages:
  python3-pastedeploy python3-setproctitle python3-tornado
The following NEW packages will be installed:
  guicorn python3-gunicorn
0 upgraded, 2 newly installed, 0 to remove and 80 not upgraded.
Need to get 68.5 kB of archives.
After this operation, 323 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 python3-gunicorn all 20.0.4-3 [56.8 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 guicorn all 20.0.4-3 [11.8 kB]
Fetched 68.5 kB in 0s (2017 kB/s)
Selecting previously unselected package python3-gunicorn.
(Reading database ... 66489 files and directories currently installed.)
Preparing to unpack .../python3-gunicorn_20.0.4-3_all.deb ...
Unpacking python3-gunicorn (20.0.4-3) ...
Selecting previously unselected package guicorn.
Preparing to unpack .../guicorn_20.0.4-3_all.deb ...
Unpacking guicorn (20.0.4-3) ...
Setting up python3-gunicorn (20.0.4-3) ...
Setting up guicorn (20.0.4-3) ...
Processing triggers for man-db (2.9.1-1) ...
```

- vi. A virtual environment is created for installing and running the flask application in an isolated python environment.

*cmd: sudo apt install python3-virtualenv*

```
ubuntu@ip-172-31-92-225:~$ sudo apt install python3-virtualenv
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  python3-appdirs python3-distlib python3-filelock
The following NEW packages will be installed:
  python3-appdirs python3-distlib python3-filelock python3-virtualenv
0 upgraded, 4 newly installed, 0 to remove and 80 not upgraded.
Need to get 197 kB of archives.
After this operation, 1032 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/main amd64 python3-appdirs all 1.4.3-2.1 [10.8 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 python3-distlib all 0.3.0-1 [116 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 python3-filelock all 3.0.12-2 [7948 B]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates/universe amd64 python3-virtualenv all 20.0.17-1ubuntu0.4 [62.7 kB]
Fetched 197 kB in 0s (7249 kB/s)
Selecting previously unselected package python3-appdirs.
(Reading database ... 66547 files and directories currently installed.)
Preparing to unpack .../python3-appdirs_1.4.3-2.1_all.deb ...
Unpacking python3-appdirs (1.4.3-2.1) ...
Selecting previously unselected package python3-distlib.
Preparing to unpack .../python3-distlib_0.3.0-1_all.deb ...
Unpacking python3-distlib (0.3.0-1) ...
Selecting previously unselected package python3-filelock.
Preparing to unpack .../python3-filelock_3.0.12-2_all.deb ...
Unpacking python3-filelock (3.0.12-2) ...
Selecting previously unselected package python3-virtualenv.
Preparing to unpack .../python3-virtualenv_20.0.17-1ubuntu0.4_all.deb ...
Unpacking python3-virtualenv (20.0.17-1ubuntu0.4) ...
Setting up python3-filelock (3.0.12-2) ...
Setting up python3-distlib (0.3.0-1) ...
Setting up python3-appdirs (1.4.3-2.1) ...
Setting up python3-virtualenv (20.0.17-1ubuntu0.4) ...
Processing triggers for man-db (2.9.1-1) ...
ubuntu@ip-172-31-92-225:~$
```

- vii. To install the flask application, we are going to create a new directory

*cmd: mkdir flaskapp*

- viii. Go to created directory.

*cmd: cd flaskapp*

- ix. Our flask application is present in the git repository to clone the repository into our web server the following command is used.

*cmd: git clone <git link>*

```
ubuntu@ip-172-31-92-225:~$ mkdir flaskapp
ubuntu@ip-172-31-92-225:~$ cd flaskapp
ubuntu@ip-172-31-92-225:~/flaskapp$ git clone https://github.com/helloflask/calculator.git
Cloning into 'calculator'...
remote: Enumerating objects: 105, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 105 (delta 0), reused 0 (delta 0), pack-reused 102
Receiving objects: 100% (105/105), 35.36 KiB | 7.07 MiB/s, done.
Resolving deltas: 100% (48/48), done.
ubuntu@ip-172-31-92-225:~/flaskapp$
```

- x. To run the application we need to initiate virtual environment the following commands are used to initiate the virtual environment.

*cmd: virtualenv venv*

*cmd: source venv/bin/activate*



```

ubuntu@ip-172-31-92-225:~/flaskapp$ virtualenv venv
created virtual environment CPython3.8.10.final.0-64 in 185ms
  creator CPython3Posix(dest=/home/ubuntu/flaskapp/venv, clear=False, global=False)
  seeder FromAppData(download=False, pip=latest, setuptools=latest, wheel=latest, pkg_resources=latest, via=copy, app_data_dir=/home/ubuntu/.local/share/virtualenv/seed-app-data/v1.0.1.debian.1)
  activators BashActivator,CShellActivator,FishActivator,PowerShellActivator,PythonActivator,XonshActivator
ubuntu@ip-172-31-92-225:~/flaskapp$ ls
calculator  venv
ubuntu@ip-172-31-92-225:~/flaskapp$

```

- xi. The following command is used to download all the required files into the webserver to run the flask application in the virtual environment.

*cmd: pip install -r calculator/requirements.txt*

```

ubuntu@ip-172-31-92-225:~/flaskapp$ source venv/bin/activate
(venv) ubuntu@ip-172-31-92-225:~/flaskapp$ pip install -r calculator/requirements.txt
Collecting Flask==0.12.2
  Downloading Flask-0.12.2-py2.py3-none-any.whl (83 kB)
    |#####| 83 kB 1.9 MB/s
Collecting Jinja2==2.8
  Downloading Jinja2-2.8-py2.py3-none-any.whl (263 kB)
    |#####| 263 kB 33.2 MB/s
Collecting Werkzeug==0.13
  Downloading Werkzeug-0.13-py2.py3-none-any.whl (311 kB)
    |#####| 311 kB 44.6 MB/s
Collecting gunicorn==19.5.0
  Downloading gunicorn-19.5.0-py2.py3-none-any.whl (113 kB)
    |#####| 113 kB 50.7 MB/s
Collecting gevent
  Downloading gevent-21.1.2-cp38-cp38-manylinux2010_x86_64.whl (6.3 MB)
    |#####| 6.3 MB 37.9 MB/s
Collecting itsdangerous==2.0.1
  Downloading itsdangerous-2.0.1-py3-none-any.whl (18 kB)
Collecting click==2.0
  Downloading click-8.0.1-py3-none-any.whl (97 kB)
    |#####| 97 kB 9.3 MB/s
Collecting MarkupSafe
  Downloading MarkupSafe-2.0.1-cp38-cp38-manylinux2010_x86_64.whl (30 kB)
Collecting zope.event
  Downloading zope.event-4.5.0-py2.py3-none-any.whl (6.8 kB)
Collecting greenlet<2.0,>=0.4.17; platform_python_implementation == "CPython"
  Downloading greenlet-1.1.0-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (164 kB)
    |#####| 164 kB 40.8 MB/s
Collecting zope.interface
  Downloading zope.interface-5.4.0-cp38-cp38-manylinux2010_x86_64.whl (259 kB)
    |#####| 259 kB 39.2 MB/s
Requirement already satisfied: setuptools in ./venv/lib/python3.8/site-packages (from gevent->-r calculator/requirements.txt (line 6)) (44.0.0)
Installing collected packages: MarkupSafe, Jinja2, Werkzeug, itsdangerous, click, Flask, gunicorn, zope.event, greenlet, zope.interface, gevent
Successfully installed Flask-0.12.2 Jinja2-2.8 MarkupSafe-2.0.1 Werkzeug-0.13 click-8.0.1 gevent-21.1.2 greenlet-1.1.0 gunicorn-19.5.0 itsdangerous-2.0.1 zope.event-4.5.0 zope.interface-5.4.0
(venv) ubuntu@ip-172-31-92-225:~/flaskapp$

```

- xii. The following commands are used to change the directory where the flask application is located and run the flask application.

*cmd: cd calculator*

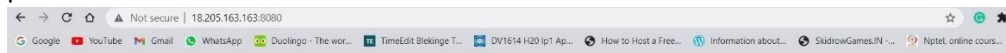
*cmd: python3 app.py*

```

(venv) ubuntu@ip-172-31-92-225:~/flaskapp/calculator$ python3 app.py
* Running on http://0.0.0.0:8080/ (Press CTRL+C to quit)
193.11.185.72 - - [27/Jul/2021 10:16:35] "GET / HTTP/1.1" 200 -
193.11.185.72 - - [27/Jul/2021 10:16:35] "GET /static/css/style.css HTTP/1.1" 200 -
193.11.185.72 - - [27/Jul/2021 10:16:35] "GET /static/js/main.js HTTP/1.1" 200 -
193.11.185.72 - - [27/Jul/2021 10:16:35] "GET /static/banner.png HTTP/1.1" 200 -
193.11.185.72 - - [27/Jul/2021 10:16:37] "GET /static/favicon.ico HTTP/1.1" 200 -
193.11.185.72 - - [27/Jul/2021 10:16:45] "GET /_calculate?number1=7&operator=%2B&number2=3 HTTP/1.1" 200 -

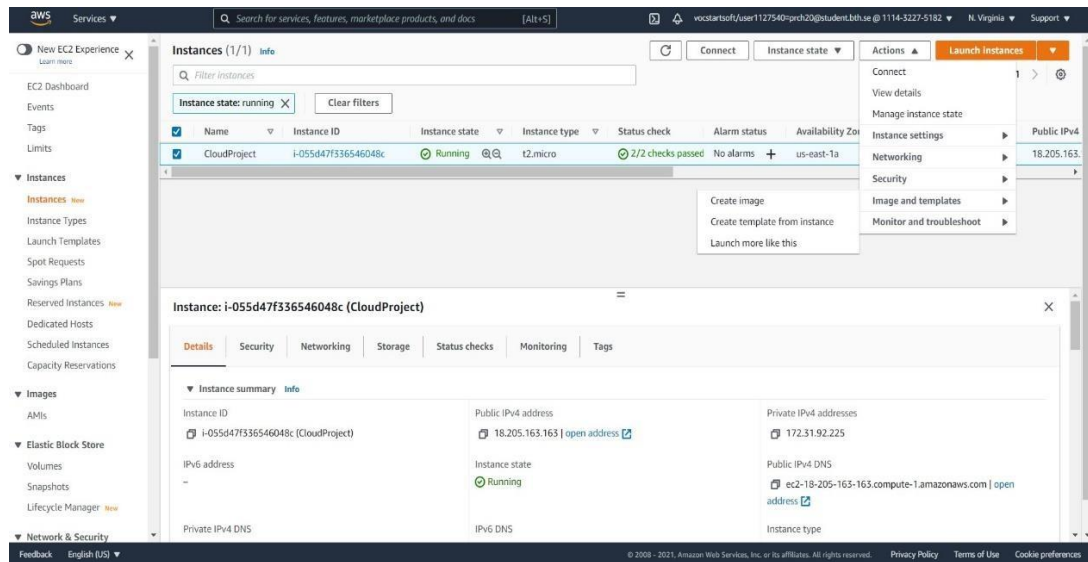
```

- xiii. Launching the flask web application in the web browser using the public IP address and port number: 8080 of our ubuntu webserver.

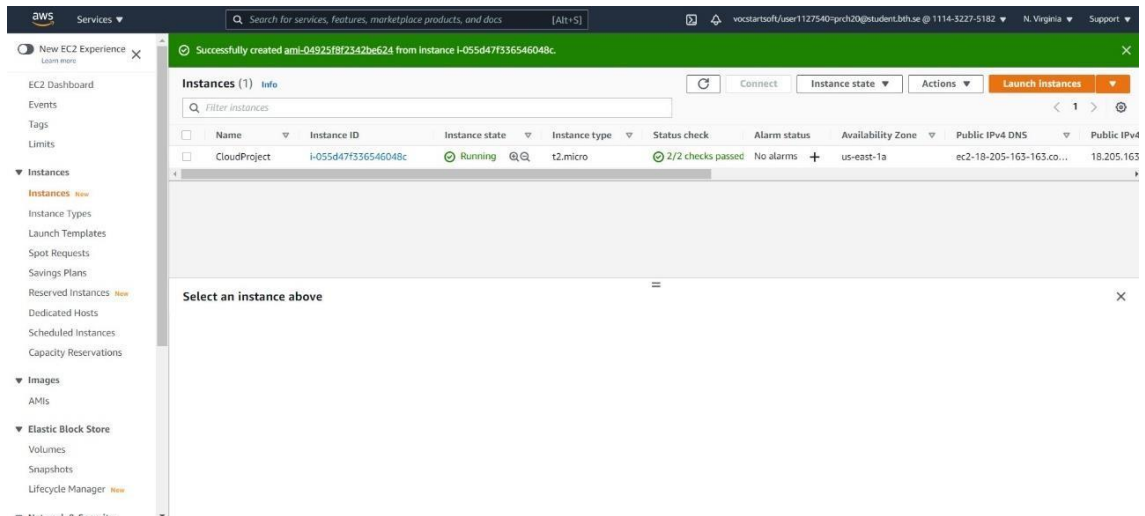


### STEP-3: IMAGE FOR THE CREATED INSTANCE

In this step we are going to create image for our instance where this image is used to launch the additional EC2 instances when there is stress generated on our flask web application so that a greater number of people can use this application without any interruption. The below image shows how to create the image for our instance in AWS.



After successful creation of image for our instance we get the results as shown in the figure below.

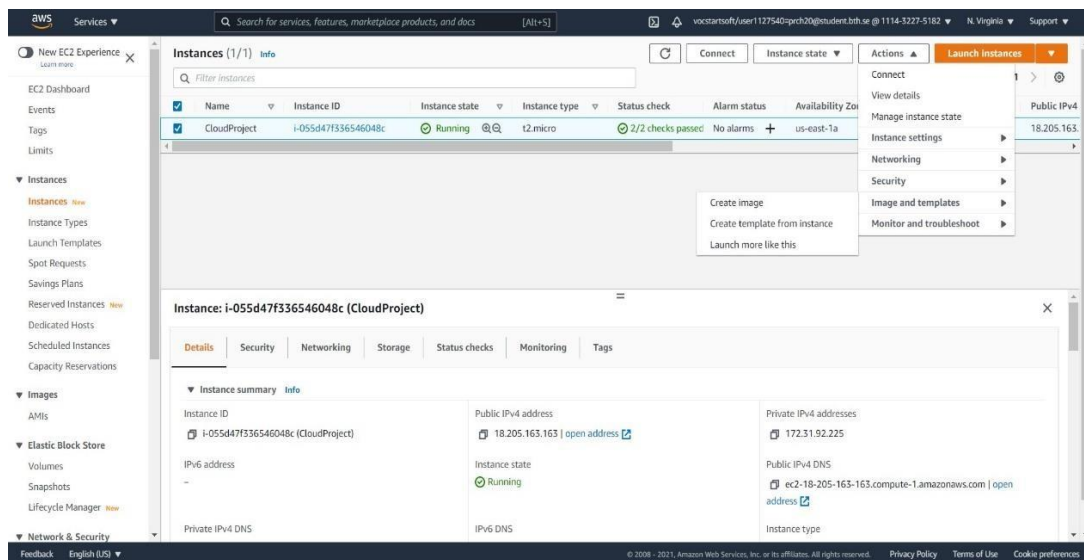


#### STEP 4: TEMPLATE CREATION FOR INSTANCE.

In this step we are going to create template for our instance where this template contains all the necessary instance configuration information that includes id of our instance, instance type, a key pair, security groups and other parameters that we used to launch EC2 instance.

Launch templates are very streamline and simplify the launch process for autoscaling and on demand instances. It reduces the number of steps required to create an instance by capturing all launch parameters within one resource and this makes the process simple and to reproduce.

The below figure shows how to initiate the creation of template for our instance.



The below figure shows the steps involved in creating instance templates.

aws

Services

Search for services, features, marketplace products, and docs

[Alt+5]

🔍

🔔

vocartsoft/user1127540-prch20@student.bth.se @ 1114-3227-5182

N.V.

EC2 > Launch templates > Create template from instance

## Create launch template

Creating a launch template allows you to create a saved instance configuration that can be reused, shared and launched at a later time. Templates can have multiple versions.

### Launch template name and description

Source instance  
i-055d47f336546048c

Launch template name - *required*

CloudProject

Must be unique to this account. Max 128 chars. No spaces or special characters like '&', '"', '@'.

Template version description

template of flask application instance

Max 255 chars

[Auto Scaling guidance](#) [Info](#)  
Select this if you intend to use this template with EC2 Auto Scaling.

☒ Provide guidance to help me set up a template that I can use with EC2 Auto Scaling

► Template tags

### Launch template contents

Specify the details of your launch template below. Leaving a field blank will result in the field not being included in the launch template.

After successful creation of templates to our image we get the results as shown in the figure below.

aws

Services ▼

Search for services, features, marketplace products, and docs

[Alt+S]

vocstartsoft/user1127540-prch20@student.bfh.se @ 1114-3227-5182 ▼

N. Virginia ▼

Support

EC2 > Launch templates > Create template from instance

Success

Successfully created CloudProject [t-09808482f2727cc5c]

► Actions log

Next steps

Launch an instance

With On-Demand instances, you pay for compute capacity by the second (for Linux, with a minimum of 60 seconds) or by the hour (for all other operating systems) with no long-term commitments or upfront payments. Launch an On-Demand Instance from your launch template.

[Launch instance from this template](#)

Create an Auto Scaling group from your template

Amazon EC2 Auto Scaling helps you maintain application availability and allows you to scale your Amazon EC2 capacity up or down automatically according to conditions you define. You can use Auto Scaling to help ensure that you are running your desired number of Amazon EC2 instances during demand spikes to maintain performance and decrease capacity during lulls to reduce costs.

[Create Auto Scaling group](#)

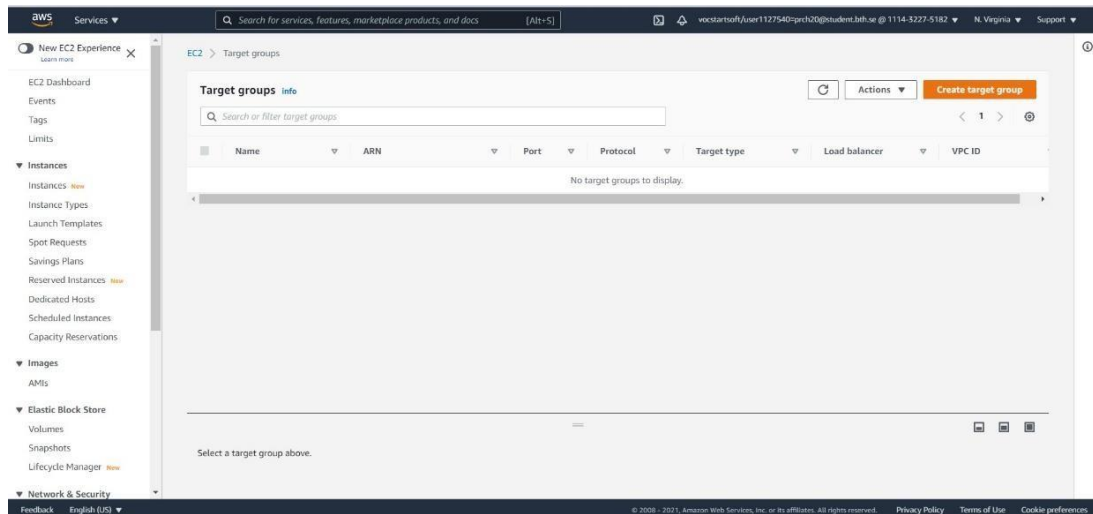
Create Spot Fleet

A Spot Instance is an unused EC2 instance that is available for less than the On-Demand price. Because Spot Instances enable you to request unused EC2 instances at steep discounts, you can lower your Amazon EC2 costs significantly. The hourly price for a Spot Instance (of each instance type in each Availability Zone) is set by Amazon EC2, and adjusted weekly based on the current supply and demand for Spot instances. Spot instances are well-suited for flexible, noncritical workloads that can tolerate interruption.

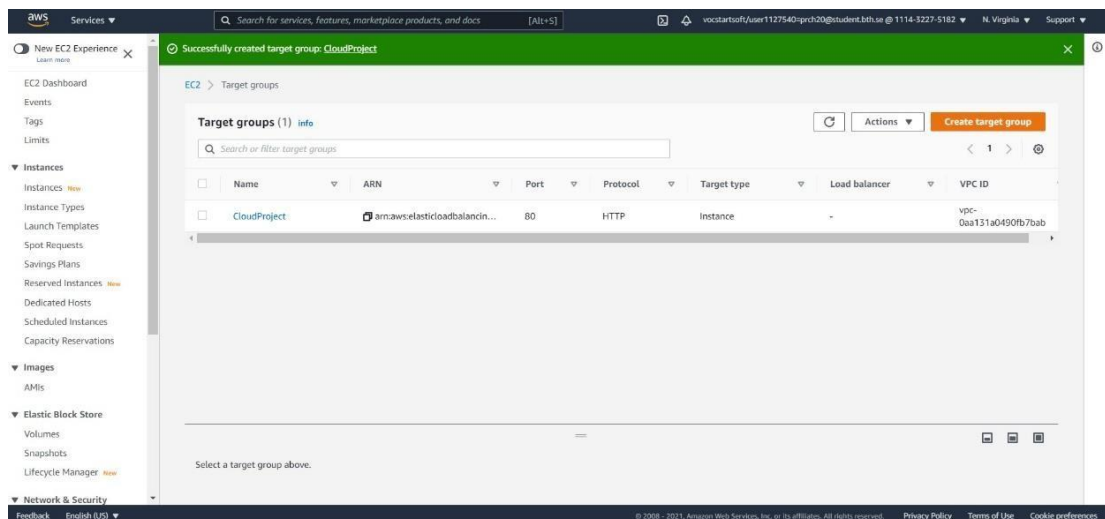
## STEP 5: CREATING TARGET GROUPS FOR AUTOSCALING

In this step we are going to create a target group that is used to inform the load balancer or initiate the load balancer where to direct the traffic to: EC2 instances, fixed IP addresses. In this target groups we configure the instance that we are target and its related image.

The below figure explains about the creation of target groups for our instance.



The below figure shows the successful creation of target groups.



## STEP-6: AUTO SCALING OF OUR INSTANCE

In this project we are going to use auto scaling of our instance when there is excess data generated while using the flask application by greater number of users.

The main purpose of autoscaling is that it allows us to automatically add or remove EC2 instances according to the conditions that we define while setting up autoscaling policy.

The benefits of autoscaling are

- It can detect an unhealthy instance, terminate it, and replace it with the new one.
- It ensures that our application always has the right amount of compute and proactively provisions capacity with predictive scaling.



- It only adds instances when needed and it can scale across different purchase options to optimize the price and performance.

In this project the following steps are performed in creating and configuring autoscaling to our instance.

1. Choose launch template or configuration.

Step 1: Choose launch template or configuration
Edit

**Group details**

Auto Scaling group name  
CloudProject

Launch template

Launch template  
CloudProject [↗](#)  
lt-09808482f7277cc5c

Version  
Default

Description  
template of flask application instance

2. Configure settings.

Step 2: Configure settings
Edit

**Instance purchase options**

This Auto Scaling group will adhere to the launch template

**Network**

Network

VPC  
vpc-0aa131a0490fb7bab [↗](#)

Availability Zone	Subnet	
us-east-1a	<a href="#">subnet-0006cd05d4c707137</a> <a href="#">↗</a>	172.31.80.0/20
us-east-1b	<a href="#">subnet-0092a6959d352e3cc</a> <a href="#">↗</a>	172.31.16.0/20
us-east-1c	<a href="#">subnet-04c454267121e9ff6</a> <a href="#">↗</a>	172.31.32.0/20
us-east-1d	<a href="#">subnet-0daec8d55d031c9d5</a> <a href="#">↗</a>	172.31.0.0/20
us-east-1e	<a href="#">subnet-04e98488902d2aae5</a> <a href="#">↗</a>	172.31.48.0/20
us-east-1f	<a href="#">subnet-01b3d67d022bd5eb5</a> <a href="#">↗</a>	172.31.64.0/20

### 3. Configure advanced options.

Step 3: Configure advanced options

Edit

Load balancing

Load balancer 1

Name

-

Type

--/HTTP

Target group

CloudProject [↗](#)

Health checks

Health check type

EC2

Health check grace period

300 seconds

Additional settings

Monitoring

Enabled

### 4. Configure group size and scaling policies

Step 4: Configure group size and scaling policies

Edit

Group size

Desired capacity

2

Minimum capacity

1

Maximum capacity

2

Scaling policy

Target tracking scaling

Policy type

Target tracking scaling

Scaling policy name

Target Tracking Policy

Execute policy when

As required to maintain Average CPU utilization at 5

Take the action

Add or remove capacity units as required

Instances need

300 seconds to warm up before including in metric

Scale in

Enabled

Instance scale-in protection

Instance scale-in protection

☐ Enable instance protection from scale in

## 5. Adding notifications and tags

Step 5: Add notifications

Edit

Notifications

No notifications

Step 6: Add tags

Edit

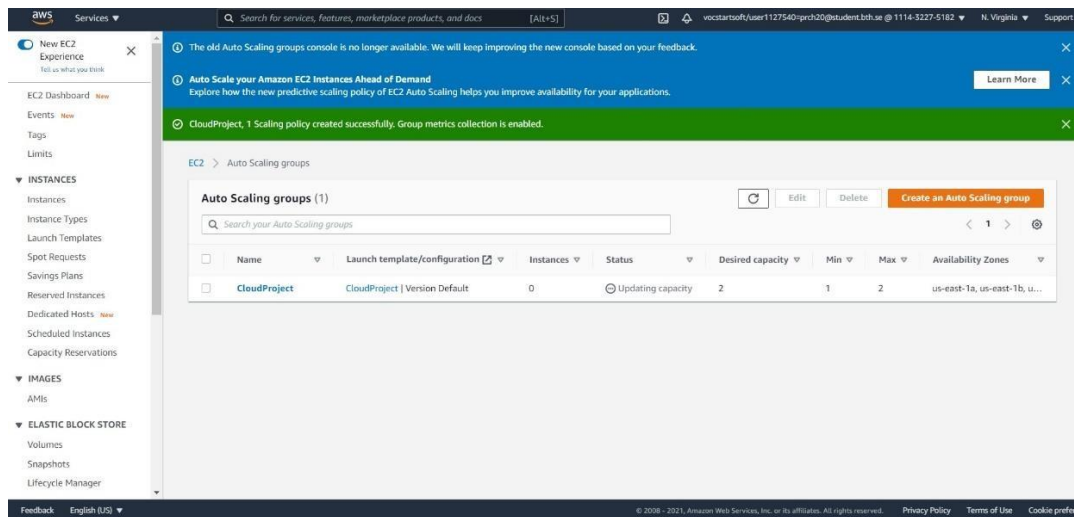
Tags (0)

Key	Value	Tag new instances
No tags		

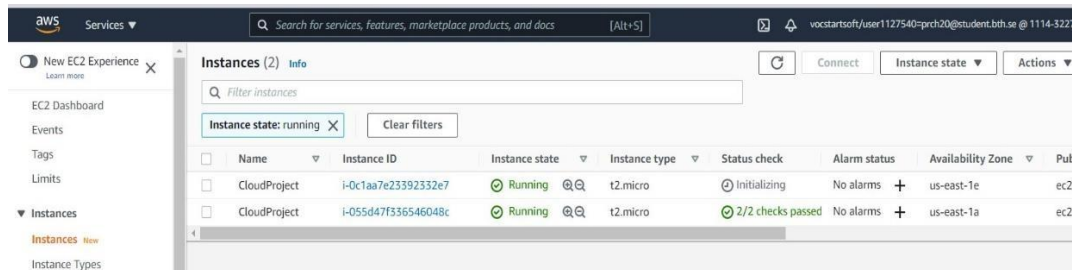
Cancel

Create Auto Scaling group

## 6. After performing above steps auto scaling group is successfully created as shown in the figure below.



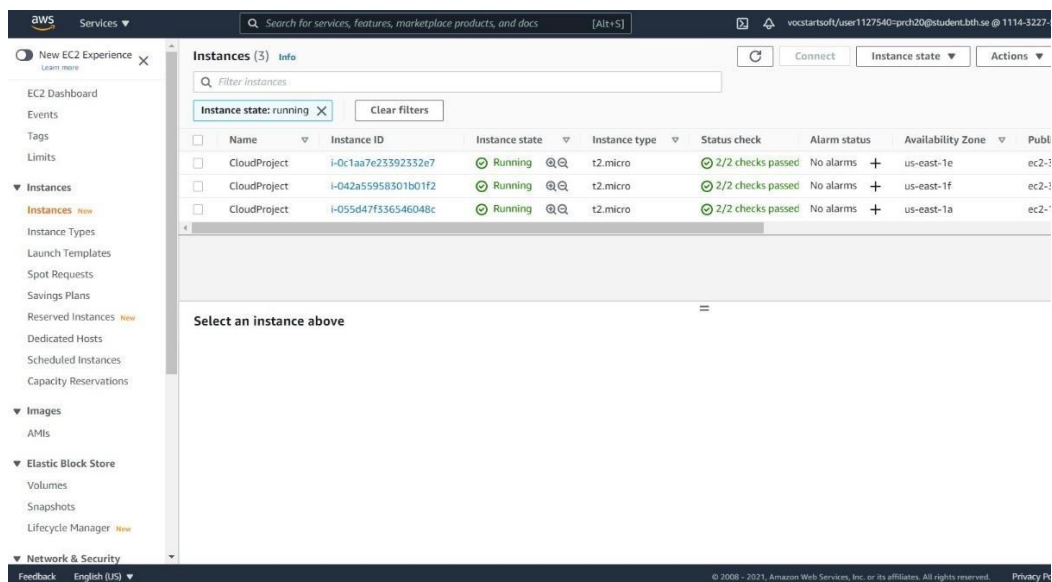
After setting up of auto scaling policy group for the instance “CloudProject” we have given the desired minimum capacity as 1 with similar configuration then an additional instance is activated as shown in the below figure.



As the number of requests and data increases as n number of users access the application there is an increase in the load on CPU. Therefore, another instance is launched when the threshold value is greater than 20.

The max capacity of our auto scaling group is 2, therefore another resource is added to the instance CloudProject after satisfying the auto scaling policies.

The below figure shows the added resources to our instance.

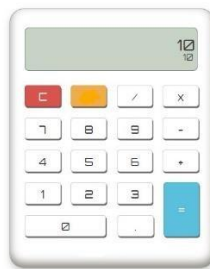
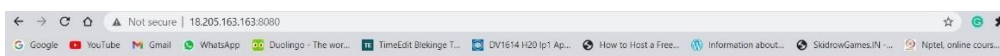


The below image shows the monitoring of activity of auto scaling policy group.

Status	Description	Cause	Start time
Successful	Launching a new EC2 instance: i-0c1aa7e23392332e7	At 2021-07-27T10:49:29Z a user request created an AutoScalingGroup changing the desired capacity from 0 to 2. At 2021-07-27T10:49:52 an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 0 to 2.	2021 July 27, 12:49:57 PM +02:00
Successful	Launching a new EC2 instance: i-042a55958301b01f2	At 2021-07-27T10:49:29Z a user request created an AutoScalingGroup changing the desired capacity from 0 to 2. At 2021-07-27T10:49:52 an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 0 to 2.	2021 July 27, 12:49:57 PM +02:00

## Validation:

- Deploying the flask web application in the instance and accessing the application using public DNS IP address 18.205.163.163 and port number 8080.

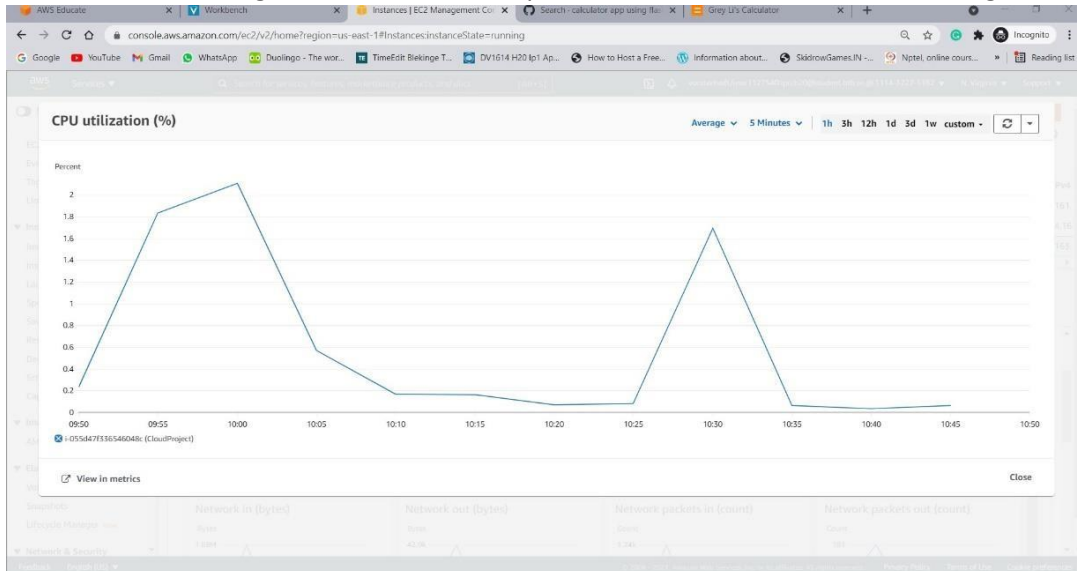


- After reaching max CPU threshold value by 20, a new resource is added in the instance list.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
CloudProject	i-0c1aa7e23392332e7	Running	t2.micro	2/2 checks passed	No alarms	us-east-1e
CloudProject	i-042a55958301b01f2	Running	t2.micro	2/2 checks passed	No alarms	us-east-1f
CloudProject	i-055d47f336546048c	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a



- CPU utilization through cloud watch in the span of 5 minutes is shown the below figure.



## Results:

- Scalability with respect to data is achieved in this project by adding the required number of resources to the instance according to the conditions present in the auto scaling policy group.
- Here the scalability is obtained as there is an exceeded in the number of users of the application the load on the CPU increased that the average threshold.
- The security of data at rest is achieved by utilizing the AWS KMS feature as the access to encrypt or decrypt the data within the service is independently controlled by AWS KMS policies under the customer's control, customers can isolate control over access to the data, from access to the keys.
- We tried or utilize this feature to secure the data of our application using client-side encryption in AWS KMS, but the results obtained were not appropriate to the proposed solution. Though we have learned a lot during this project.