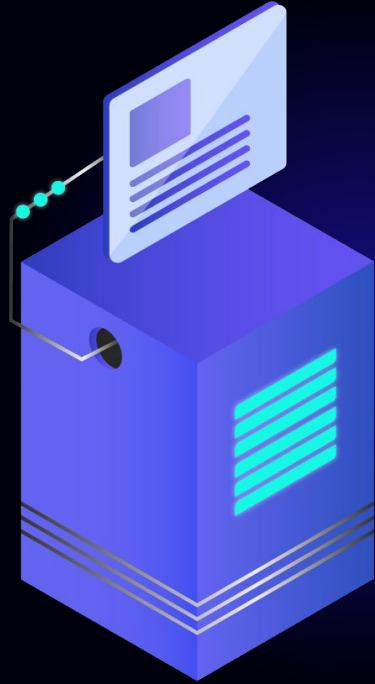




Deploying a flask app on an AWS EC2 instance

Temperature Convertor Application



INTRODUCTION

In this project a temperature conversion application was deployed on an AWS EC2 instance. The application is where you can enter a temperature in either celsius or fahrenheit and get its converted value in fahrenheit or celsius, respectively. Also the Functionality and auto scalability were evaluated.

TABLE OF CONTENTS

01

Flask App

03

Evaluation

02

***Deployment
on EC2***

04

Results



01



Flask App

In this section, we will explain our application, and how we code it.

Flask App

- A Flask application is an instance for the **Flask** class.
- Everything about the application, such as **configuration** and **URLs**, will be registered with this class.



Flask

Flask App - step 1

- First, we imported Flask and request from the flask.
- Next, we created a Flask instance using this command: `app = Flask(__name__)`.
- Then, using command `@app.route('/')` a simple route was created.
 - It creates a connection between the URL and a **function that returns a response**.

temp-convertor M X

```
1  from flask import Flask
2  from flask import request
3  app = Flask(__name__)
4  @app.route('/')

```

Flask App - step 2

HTML was used as a markup language to build our application's frontend.

```
6 def homepage():
7     website = '''
8     <!DOCTYPE html>
9     <html>
10    <body>
11        <center>
13        </center>
14        <h1><center>Welcome to our Temperature Convertor application</center></h1>
15        <br>
16        <h2><center>Enter a value in Celsius or Farenheit, and it will be converted</center></h2>
17        <center><form action="tempConversion" method="POST">
18        <label for="temp-type-select">Choose a temperature type:</label>
19        <select name="types" id="type-select">
20        <option value="">--Please choose an option--</option>
21        <option value=0>Celcius</option>
22        <option value=1>Farenheit</option>
23        </select>
24        <input type="text" name="tempInCelsius">
25        <input type="submit" value="Convert">
26        </form></center>
27    </body>
28    </html>'''
29    return website
```

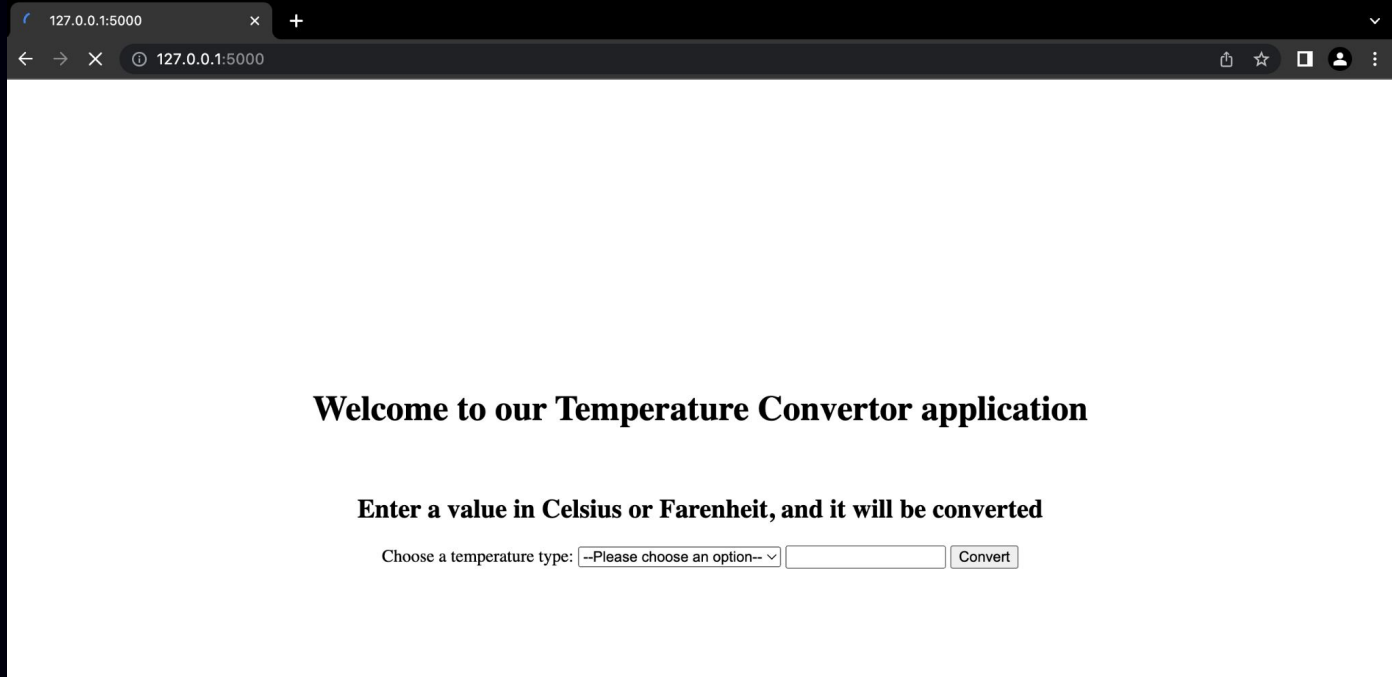
Flask App - step 3

Finally, we defined another function which allows the user to select one option between celsius and fahrenheit and enter a value. Then, the converted temperature will be returned as a result.

```
temp-converto M x
30
31 @app.route('/tempConversion', methods=['POST'])
32 def my_form_post():
33     temp = request.form['tempInCelsius']
34     option = request.form['types']
35     try:
36         if option == '0':
37             celsiusTemp = float(temp)
38             fahrenheitTemp = (celsiusTemp * 9.0/5) + 32
39             return str(fahrenheitTemp)
40         elif option == '1':
41             fahrenheitTemp = float(temp)
42             celsiusTemp = (fahrenheitTemp - 32) * 5.0/9
43             return str(celsiusTemp)
44         else:
45             return "You should first choose a temperature type. Please go back and try again."
46     except:
47         return "That's not a number!"
48
49
50
51 if __name__ == '__main__':
52     app.run()
```


Flask App - Result

By visiting <http://127.0.0.1:5000> in a browser we were running our Flask web application on a local host.



02

***Deployment on an
AWS EC2 instance***

Step1 : Launch a new EC2 instance

- Opening EC2 console
- Select Ubuntu server 20.04
- Instance type: t2.micro
- Create a security group with following inbound rules:
 - SSH
 - HTTP
 - HTTPS
- **SSH** setting allows ssh access from any IP address
- **HTTP** allows to access to the new instance from the browser
- **HTTPS** is for running secure connections

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 1: Choose an Amazon Machine Image (AMI)

SUSE Linux x86 / ami-06ec4eaf39ca724d4 (64-bit Arm)
Free tier eligible
SUSE Linux Enterprise Server 15 Service Pack 2 (HVM), EBS General Purpose (SSD) Volume Type. Public Cloud, Advanced Systems Management, Web and Scripting, and Legacy modules enabled.
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Ubuntu Server 20.04 LTS (HVM), SSD Volume Type - ami-0885b1f6bd170450c (64-bit x86) / ami-054e49cb26c2fd312 (64-bit Arm)
Free tier eligible
Ubuntu Server 20.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Cancel and Exit
Select
☒ 64-bit (x86)
☐ 64-bit (Arm)
Select
☒ 64-bit (x86)
☐ 64-bit (Arm)

Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more](#) about instance types and how they can meet your computing needs.

Filter by: All instance families Current generation Show/Hide Columns

Currently selected: t2.micro (- ECUs, 1 vCPUs, 2.5 GHz, -, 1 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GiB)	EBS-Optimized Available	Network Performance	IPv6 Support
<input type="checkbox"/>	t2	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
<input checked="" type="checkbox"/>	t2	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	t2	t2.small	1	2	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	t2	t2.medium	2	4	EBS only	-	Low to Moderate	Yes

Type	Protocol	Port Range	Source	Description
SSH	TCP	22	Custom 0.0.0.0/0	e.g. SSH for Admin Desktop
HTTP	TCP	80	Anywhere 0.0.0.0/0, ::/0	e.g. SSH for Admin Desktop
HTTPS	TCP	443	Anywhere 0.0.0.0/0, ::/0	e.g. SSH for Admin Desktop

Step1 : Launch a new EC2 instance

- Create a key pair and save it

The screenshot shows the AWS Management Console interface for launching an EC2 instance. A modal dialog box is open, titled "Select an existing key pair or create a new key pair". The dialog contains the following elements:

- Title:** Select an existing key pair or create a new key pair
- Text:** A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.
- Note:** The selected key pair will be added to the set of keys authorized for this instance. Learn more about [removing existing key pairs from a public AMI](#).
- Form:** A dropdown menu with "Create a new key pair" selected, followed by a text input field labeled "Key pair name".
- Buttons:** "Download Key Pair" and "Launch Instances".
- Message:** A blue box with a speech bubble icon containing the text: "You have to download the **private key file** (*.pem file) before you can continue. **Store it in a secure and accessible location.** You will not be able to download the file again after it's created."

In the background, the console shows the "Instance Type" section with a table:

Instance Type	ECUs
t2.small	-

Below this is the "Security Groups" section, showing a table with columns for "Security group name" and "Description". The "Type" dropdown is set to "SSH".

Step 2: Update the permission of private key

EC2 > Instances > i-0ab093ec4f8a208eb > Connect to instance

Connect to instance [Info](#)

Connect to your instance i-0ab093ec4f8a208eb using any of these options

EC2 Instance Connect



Session Manager

SSH client

EC2 serial console


Instance ID

 i-0ab093ec4f8a208eb

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is TemperatureFlaskApp.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
 **chmod 400 TemperatureFlaskApp.pem**
4. Connect to your instance using its Public DNS:
 ec2-18-207-189-233.compute-1.amazonaws.com

Example:

 ssh -i "TemperatureFlaskApp.pem" ubuntu@ec2-18-207-189-233.compute-1.amazonaws.com

 **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.

Step 3: Connect to our new instance via SSH

EC2 > Instances > i-0ab093ec4f8a208eb > Connect to instance

Connect to instance [Info](#)

Connect to your instance i-0ab093ec4f8a208eb using any of these options

EC2 Instance Connect



Session Manager

SSH client

EC2 serial console

Instance ID

 i-0ab093ec4f8a208eb

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is TemperatureFlaskApp.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
 `chmod 400 TemperatureFlaskApp.pem`
4. Connect to your instance using its Public DNS:
 `ec2-18-207-189-233.compute-1.amazonaws.com`

Example:

 `ssh -i "TemperatureFlaskApp.pem" ubuntu@ec2-18-207-189-233.compute-1.amazonaws.com`



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.

Step4: Update , upgrade and set up the environment

- Update the local package index and upgrade the system using following commands:

```
$ sudo apt-get update
```

```
$ sudo apt-get -y upgrade
```

 - This makes sure everything is up to date and prevents any errors due to deprecations
- Install Python
- Install PIP
 - It is needed to manage software packages for python

Step 5: Install and configure Apache and WSGI

- Install the Apache webserver with the **mod_wsgi** module using the command below to interface with our *Flask* app.

```
$ sudo apt-get install -y apache2
```

 - It is required to interface with our flask app because web servers do not understand python and WSGI makes the communication happen.
- By opening our instance's public DNS in a browser we saw Apache's default page, indicating the installation is working correctly.

EC2 > Instances > i-0ab093ec4f8a208eb > Connect to instance

Connect to instance [Info](#)

Connect to your instance i-0ab093ec4f8a208eb using any of these options

EC2 Instance Connect	Session Manager	SSH client	EC2 serial console
----------------------	-----------------	-------------------	--------------------

Instance ID
i-0ab093ec4f8a208eb

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is TemperatureFlaskApp.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.

```
chmod 400 TemperatureFlaskApp.pem
```
4. Connect to your instance using its Public DNS:

ec2-18-207-189-233.compute-1.amazonaws.com

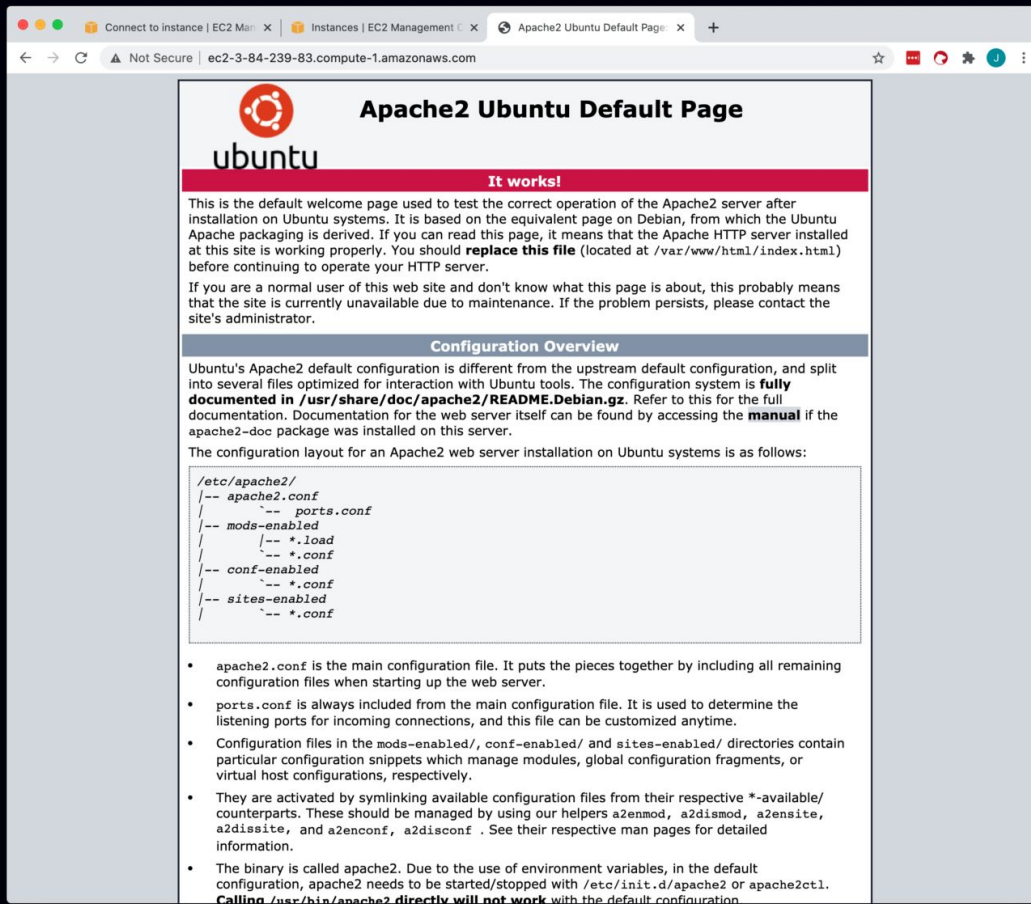
Example:

```
ssh -i "TemperatureFlaskApp.pem" ubuntu@ec2-18-207-189-233.compute-1.amazonaws.com
```

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.

Step 5: Install and configure Apache and WSGI

Apache's default page is indicating that the installation is working correctly.



Step 6: Import flask app in python file .(run.py)

temp-convertor X


```
1 from flask import Flask
2 from flask import request
3 #from flask import render_template
4
5 app = Flask(__name__)
6
7 @app.route('/')
8 def homepage():
9     website = ''
```

```
33 @app.route('/tempConversion', methods=['POST'])
34 def my_form_post():
35     temp = request.form['tempInCelsius']
36     option = request.form['types']
37     try:
38         if option == '0':
39             celsiusTemp = float(temp)
40             fahrenheitTemp = (celsiusTemp * 9.0/5) + 32
41             return str(fahrenheitTemp)
42         elif option=='1':
43             fahrenheitTemp = float(temp)
44             celsiusTemp = (fahrenheitTemp - 32) * 5.0/9
45             return str(celsiusTemp)
46         else:
47             return "You should first choose a temperature type. Please go back and try again."
48     except:
49         return "That's not a number!"
50
51
52
53 if __name__ == '__main__':
54     app.run()
```

Step 7: Configure virtual host

- Apache displays HTML pages by default but to serve dynamic content from Flask we made the following changes.
- The default Apache configuration file is located at `etc / apache2 / sites-available / 000-default.conf`. Instead of overriding that file we created a new one.
- `$ sudo vi / etc / apache2 / sites -available / microservices.com.conf`
- Add the following to the `microservices.com.conf`

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    ServerName your_domain
    ServerAlias www.your_domain
    DocumentRoot /var/www/html/microservices.com
    WSGIDaemonProcess microservices.com threads=5
    WSGIScriptAlias / /var/www/html/microservices.com/production.wsgi
    <Directory microservices.com>
        WSGIProcessGroup microservices.com
        WSGIApplicationGroup %{GLOBAL}
        Order deny,allow
        Allow from all
    </Directory>
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```



Step 8: Reload Apache , and get updated public DNS

- Disable the default Apache config by running the following command.

```
$ sudo a2dissite 000-default.conf
```



- Enable our new conf

```
$ sudo a2ensite microservices.com.conf
```



- Reload apache to implement the changes made.

```
$ sudo systemctl reload apache2
```

- **Public DNS link:**

Connect to instance [Info](#)

Connect to your instance i-0218679621747ab53 using any of these options

[EC2 Instance Connect](#) | [Session Manager](#) | **[SSH client](#)** | [EC2 serial console](#)

Instance ID
i-0218679621747ab53

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is Temperature-Flask.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
chmod 400 Temperature-Flask.pem
4. Connect to your instance using its Public DNS:
ec2-3-85-122-133.compute-1.amazonaws.com

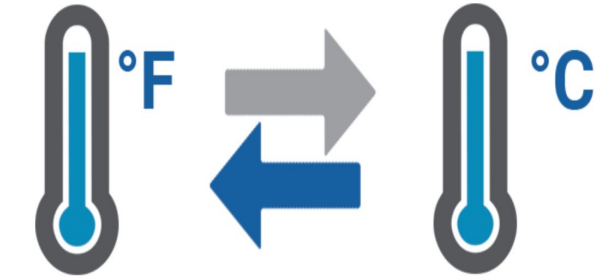
Example:
ssh -i "Temperature-Flask.pem" ubuntu@ec2-3-85-122-133.compute-1.amazonaws.com

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.

Step 9: Test and run the deployed app

← → ↻ Not Secure | ec2-18-234-50-237.compute-1.amazonaws.com

Solve Python | Ha... BBC Learning Eng... courses cloud project PDEng Expenses With Ma... library noodle CashbackBase - B... per Merito



The interface features two thermometer icons. The left thermometer is labeled with a large blue '°F' and the right with a large blue '°C'. Between them are two large, thick arrows: a grey arrow pointing right and a blue arrow pointing left, indicating the direction of conversion.

Welcome to our Temperature Convertor application

Enter a value in Celsius or Farenheit, and it will be converted

Choose a temperature type:

03

Evaluation

Scale and load balance of the Architecture

- **Elastic Load Balancing :**

Automatically distributes incoming application traffic across multiple Amazon EC2 instances. It enables us to achieve fault tolerance in our applications by seamlessly providing the required amount of load balancing capacity needed to route application traffic.

- **Auto Scaling :**

- **It helps us to maintain application availability and allows us to scale our Amazon EC2 capacity out or in automatically according to conditions we defined. You can use Auto Scaling to help ensure that you are running your desired number of Amazon EC2 instances.**

- **Auto Scaling can automatically increase the number of Amazon EC2 instances during demand spikes to maintain performance and decrease capacity during lulls to reduce costs.**

Objective

- **Create an Amazon Machine Image (AMI) from a running instance.**
- **Create a load balancer.**
- **Create a launch configuration and an Auto Scaling group.**
- **Automatically scale new instances within specific subnets.**
- **Create Amazon CloudWatch alarms and monitor performance of our infrastructure.**

Step 1 : Creating AMI for Autoscaling

- **First, confirm that the instance is running and Web Server checks passed displayed.**
- **Create an AMI based upon this instance.**
- **Select Web Server >click Image and templates > Create image, then configure:
(Image name and Image description)**
- **Click Create image and use this AMI when launching the Auto Scaling group later.**

Step 2 :Create a load balancer

- we started with:
 - choosing the Target Groups>Choose a target type: Instances>Target group name and VPC> create target group.
 - Create Load Balancer and enter our Load balancer name.
 - Network mapping section, we selected:
VPC and specified which subnets the Load Balancer should use. The subnets Us-east-1a
Us-east-1b
- Next Security groups section selection:
We created a security group named Tempreature-Group with HTTP , HTTPS and SSH
- We created a load balancer successfully created and created an AMI based upon this instance
- Finally click to create image and use this AMI when launching the Auto Scaling group later

Step 3 :Create a launch Configuration

- **First in the left navigation pane, click Launch Configurations>Create launch configuration**
- **Configure these settings:**
Launch configuration name>Amazon Machine Image (AMI) >Instance type
Monitoring: Select Enable EC2 instance detailed monitoring within CloudWatch to allows Auto Scaling to react quickly to changing utilization.
- **Security groups configuration: Choose Select an existing security group**
- **Key pair configuration: Key pair options >Existing key pair**
- **Finally ,create launch configuration**

Step 4 :Create an auto scaling group

- **We created an Auto Scaling group that uses our previous Launch Configuration.**

Enter Auto Scaling group name > Next > Network page configures>VPC Network > Subnet > Next

Attach to an existing load balancing

CloudWatch monitoring under Group size configure:

Desired capacity: 2 Minimum capacity: 2 Maximum capacity: 5

Under Scaling policies, choose Target tracking scaling policy and configure:

- **policy name>Metric type>Target value> Next**

Choose Add tag and Configure the following:

- **Key and Value>Next**

Add notification: We added an email to receive an email when the alarm is triggered.

- **Our Auto Scaling group will initially show an instance count of zero, but new instances will be launched to reach the Desired count of 2 instances.**

04

Results

1. *Verify that load balancing is working*

- We created an Auto Scaling group that uses Launch Configuration. The two new instances were created because we set the minimum number of instances to 2.

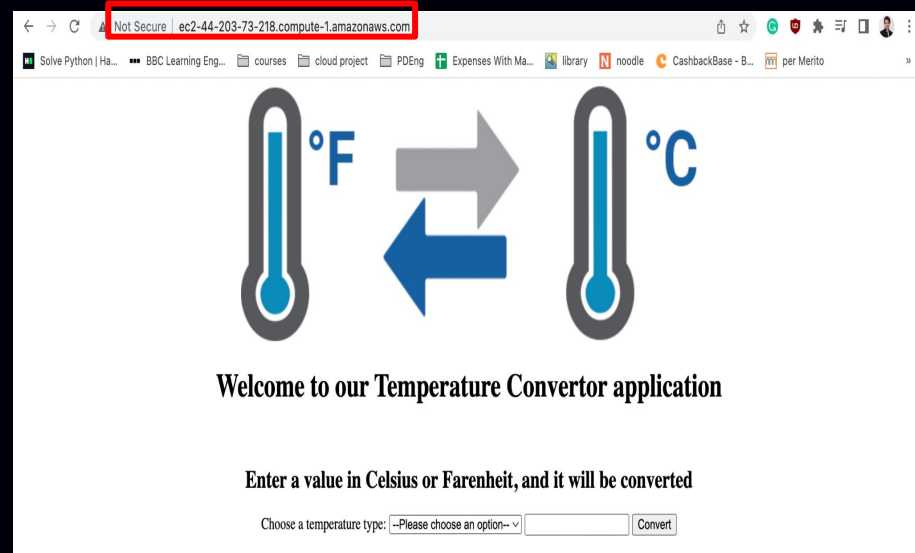
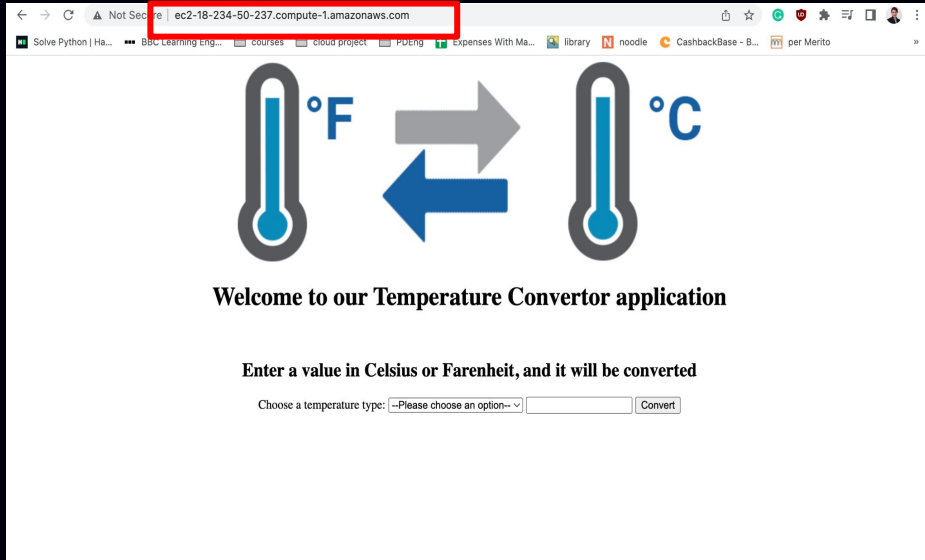
The screenshot shows the AWS Management Console for the us-east-1 region. The main content area displays the 'Instances (3)' page. The table below lists the instances:

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status
<input type="checkbox"/>	TemperatureInstance	i-031c23ab3339518d9	Running	t2.micro	-	No alarms
<input type="checkbox"/>	Temperature-Converting	i-0218679621747ab53	Running	t2.micro	2/2 checks passed	No alarms
<input type="checkbox"/>	TemperatureInstance	i-0186478dff19e895	Running	t2.micro	-	No alarms

The left sidebar shows the navigation menu with 'Instances' selected. The top navigation bar shows the user is logged in as 'voclabs/user1618430=barghimehmandari.1959831@studenti.uniroma...'.

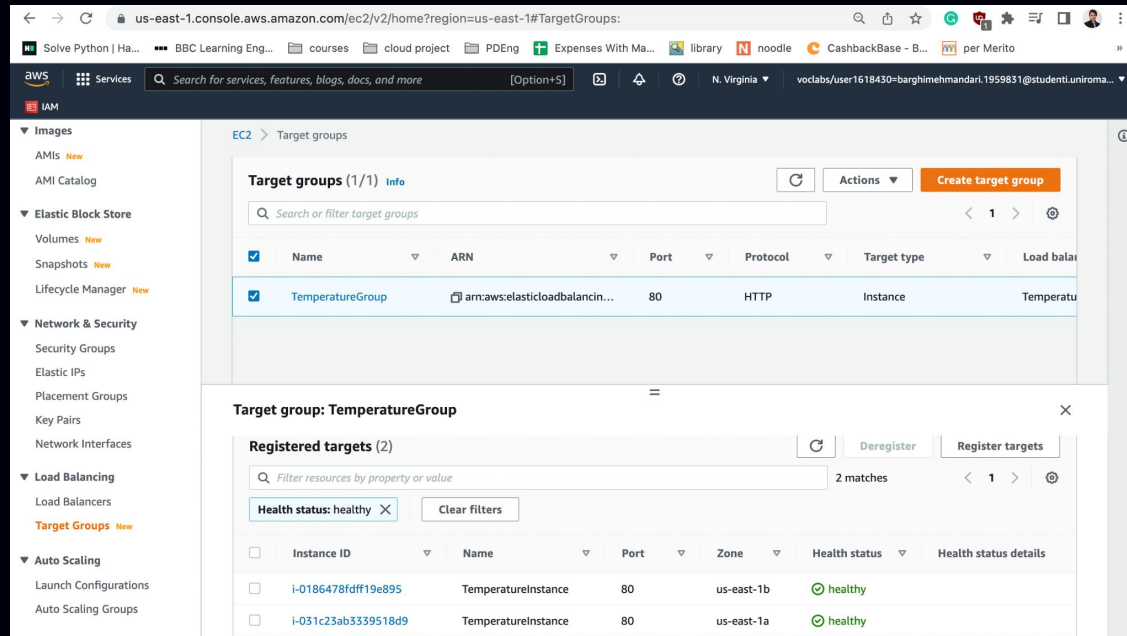
1. Verify that load balancing is working

- Now we can access the Auto Scaling group via the Load Balancer. By copy and paste the DNS after connecting of new instances that were created. The application appeared in our browser. This indicates that the Load Balancer received the request, sent it to one of the EC2 instances, then passed back the result.



1. Verify that load balancing is working

- In the Target Groups (in the Load Balancing section) two “TemperatureInstance” targets were listed for this target group. Healthy signs indicates that an instance has passed the Load Balancer's health check. This means that the Load Balancer will send traffic to the instance.



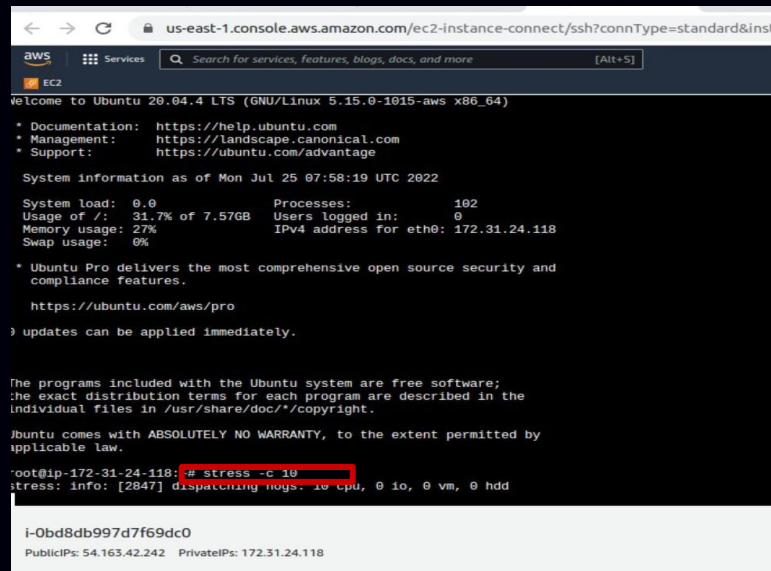
The screenshot shows the AWS Management Console interface for the 'us-east-1' region. The left sidebar contains navigation links for various AWS services, including Images, Elastic Block Store, Network & Security, Load Balancing, and Auto Scaling. The 'Load Balancing' section is expanded, showing 'Load Balancers' and 'Target Groups'. The 'Target Groups' page is displayed, showing a table with one target group, 'TemperatureGroup'. The table has columns for Name, ARN, Port, Protocol, Target type, and Load balancer. The 'TemperatureGroup' target group is listed with an ARN of 'arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/temperaturegroup/123456789012', port 80, HTTP protocol, and Instance target type. Below the table, a modal window titled 'Target group: TemperatureGroup' is open, showing 'Registered targets (2)'. The modal has a search bar and a 'Clear filters' button. It displays two registered targets, both in a 'healthy' state. The targets are listed in a table with columns for Instance ID, Name, Port, Zone, Health status, and Health status details.

Instance ID	Name	Port	Zone	Health status	Health status details
i-0186478fdff19e895	TemperatureInstance	80	us-east-1b	healthy	
i-031c23ab3339518d9	TemperatureInstance	80	us-east-1a	healthy	

2. Stress Test

- We increased the stress and create an alarm for the instance that we have. We set it to be triggered when the cpu utilization is higher than 50% and also we will receive an email when the alarm be activated.

```
ubuntu@ip-172-31-87-196:~$ sudo apt install stress
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  stress
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 18.4 kB of archives.
After this operation, 55.3 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 stress amd64 1.0.4-6 [18.4 kB]
Fetched 18.4 kB in 0s (409 kB/s)
Selecting previously unselected package stress.
(Reading database ... 97048 files and directories currently installed.)
Preparing to unpack .../stress_1.0.4-6_amd64.deb ...
Unpacking stress (1.0.4-6) ...
Setting up stress (1.0.4-6) ...
Processing triggers for install-info (6.7.0.dfsg.2-5) ...
Processing triggers for man-db (2.9.1-1) ...
ubuntu@ip-172-31-87-196:~$ stress --cpu 90
stress: info: [1392] dispatching hogs: 90 cpu, 0 io, 0 vm, 0 hdd
```



us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?connType=standard&ins...

aws Services Search for services, features, blogs, docs, and more [Alt+S]

EC2

Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.15.0-1015-aws x86_64)

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

System information as of Mon Jul 25 07:58:19 UTC 2022

System load: 0.0	Processes: 102
Usage of /: 31.7% of 7.57GB	Users logged in: 0
Memory usage: 27%	IPv4 address for eth0: 172.31.24.118
Swap usage: 0%	

* Ubuntu Pro delivers the most comprehensive open source security and compliance features.
<https://ubuntu.com/aws/pro>

updates can be applied immediately.

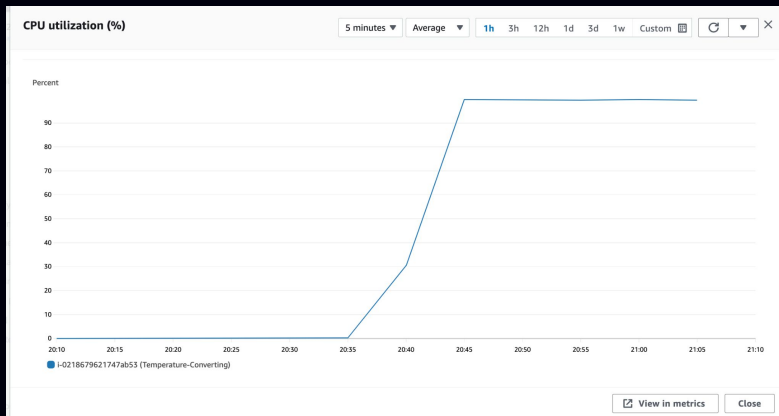
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

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

```
root@ip-172-31-24-118:~# stress -c 10
stress: info: [2847] dispatching hogs: 10 cpu, 0 io, 0 vm, 0 hdd
```

i-0bd8db997d7f69dc0
PublicIPs: 54.163.42.242 PrivateIPs: 172.31.24.118

2. Stress Test



41 of 2,087 < > []

AWS Notifications <no-reply@sns.amazonaws.com> Mon, 18 Jul, 12:13 (3 days ago) ☆ []
to me ▾

You are receiving this email because your Amazon CloudWatch Alarm "awssec2-4-02e3b6a726a88d2a-GreaterThanOrEqualToThreshold-CPUUtilization" in the US East (N. Virginia) region has entered the ALARM state, because "Threshold Crossed: 1 datapoint [86.11204070096102 (18/07/22 19:08:00)] was greater than or equal to the threshold (60.0)." at "Monday 18 July 2022 19:13:36 UTC".

View this alarm in the AWS Management Console:
<https://us-east-1.console.aws.amazon.com/cloudwatch/deeplink.js?region=us-east-1#alarmsV2:alarm/awssec2-4-02e3b6a726a88d2a-GreaterThanOrEqualToThreshold-CPUUtilization>

Alarm Details:

- Name: awssec2-4-02e3b6a726a88d2a-GreaterThanOrEqualToThreshold-CPUUtilization
- Description: Created from EC2 Console
- State Change: OK -> ALARM
- Reason for State Change: Threshold Crossed: 1 datapoint [86.11204070096102 (18/07/22 19:08:00)] was greater than or equal to the threshold (60.0).
- Timestamp: Monday 18 July 2022 19:13:36 UTC
- AWS Account: 861435167366
- Alarm Arn: arn:aws:cloudwatch:us-east-1:861435167366:alarm:awssec2-4-02e3b6a726a88d2a-GreaterThanOrEqualToThreshold-CPUUtilization

Threshold:

- The alarm is in the ALARM state when the metric is GreaterThanOrEqualToThreshold 60.0 for at least 1 of the last 1 period(s) of 300 seconds.

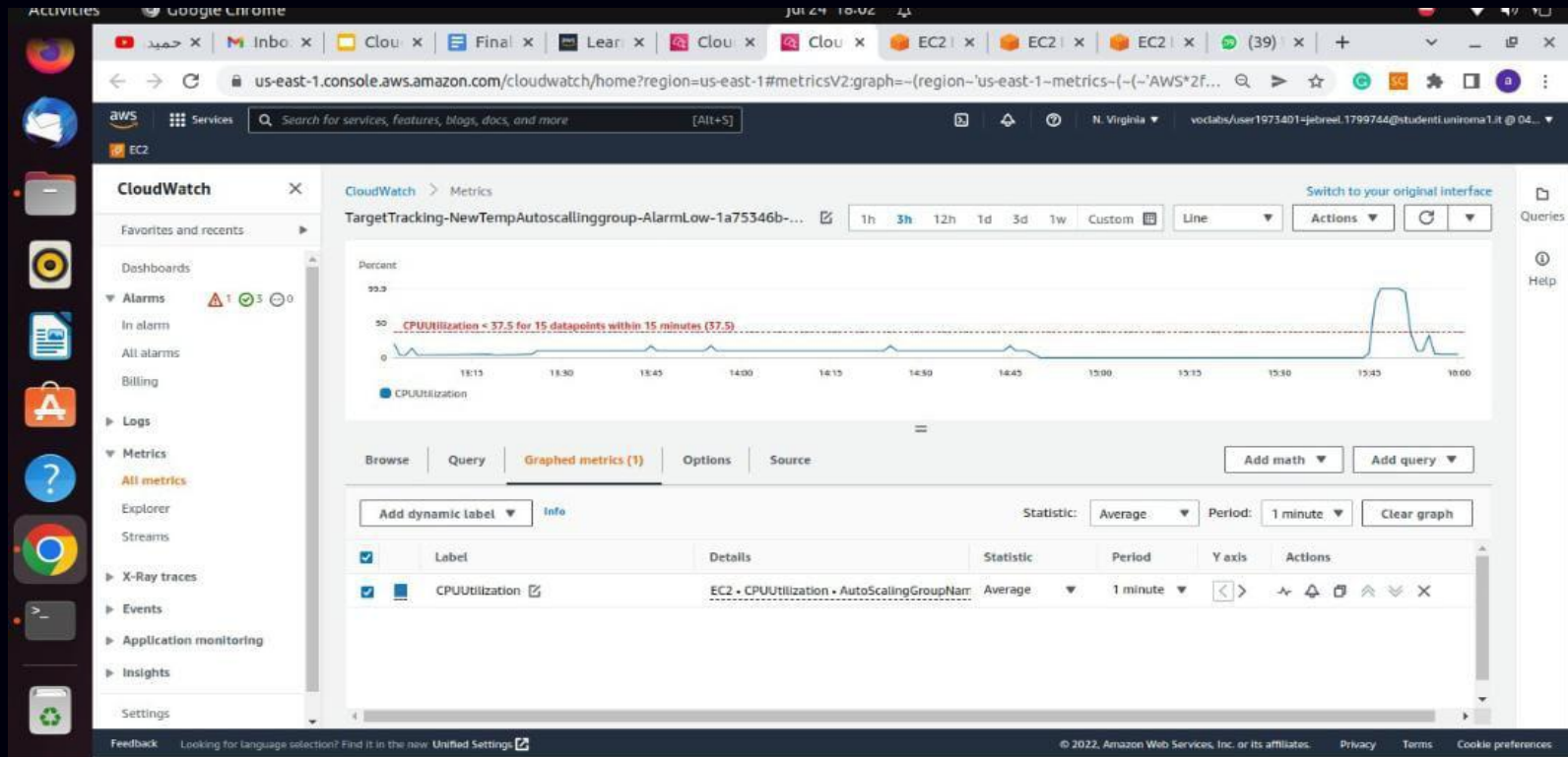
Monitored Metric:

- MetricNamespace: AWS/EC2
- MetricName: CPUUtilization
- Dimensions: [InstanceId = i-02e3b6a726a88d2a]
- Period: 300 seconds
- Statistic: Average
- Unit: not specified

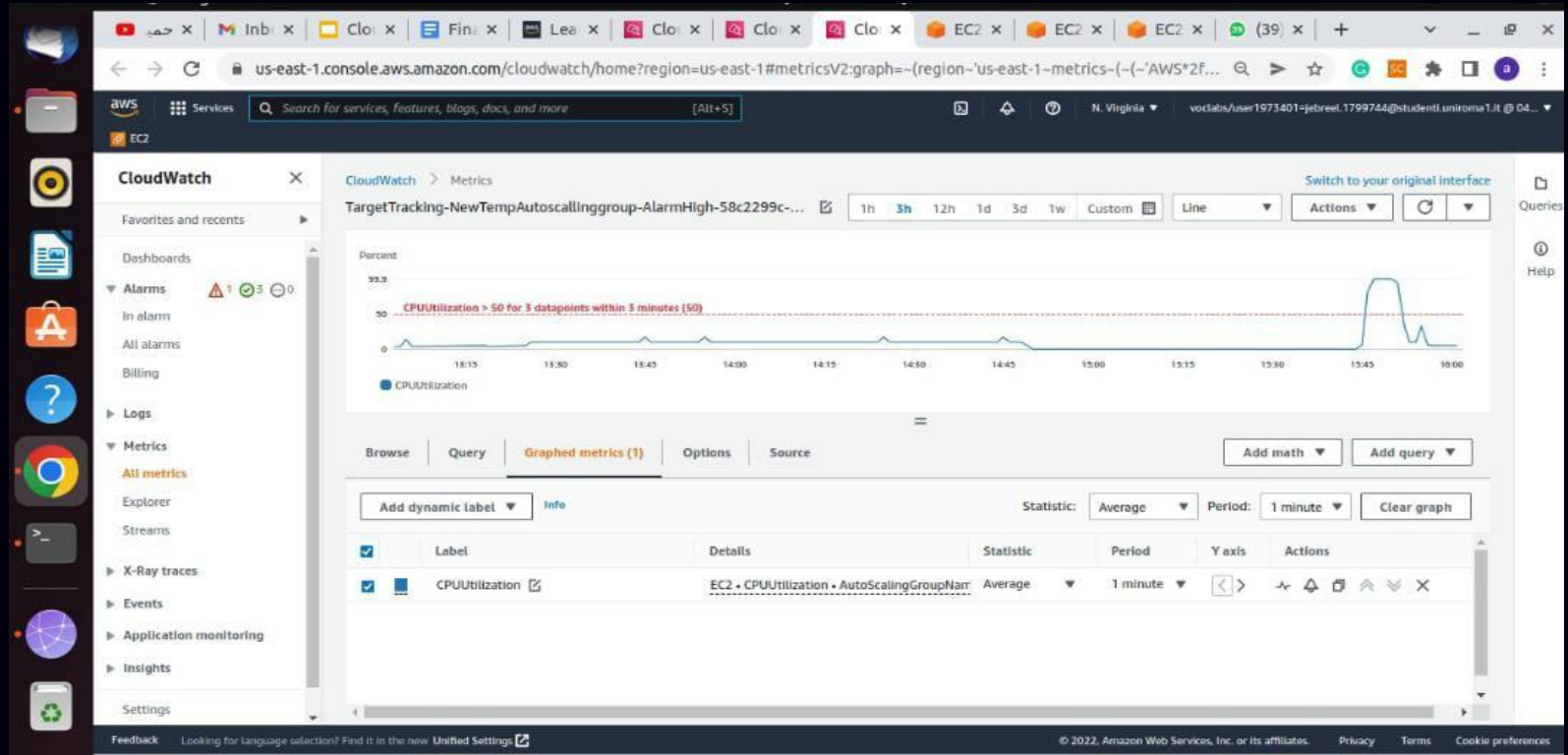
State Change Actions:

- OK
- ALARM: [arn:aws:sns:us-east-1:861435167366:Temp-converter]
- INSUFFICIENT_DATA:

2. Stress Test - Alarm-low

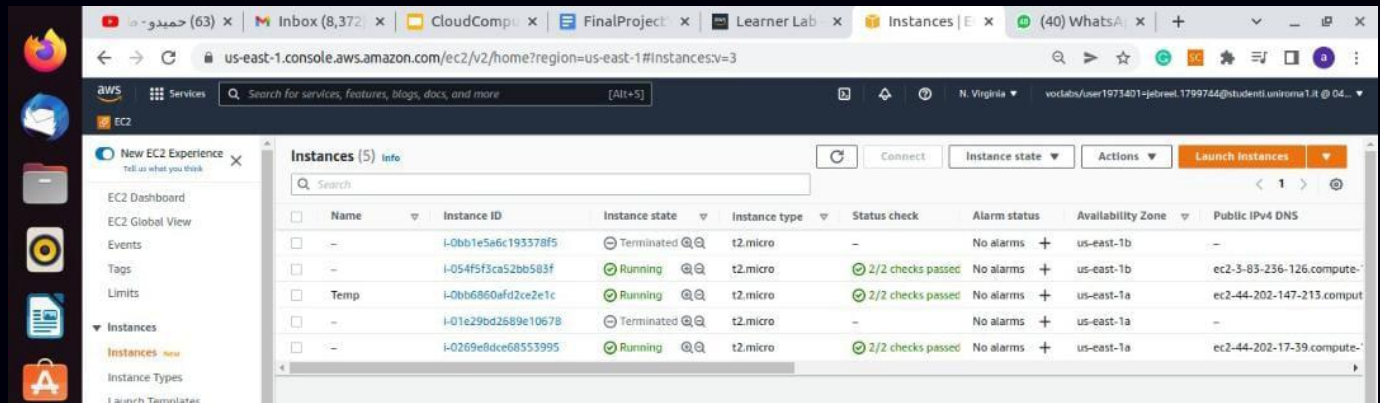
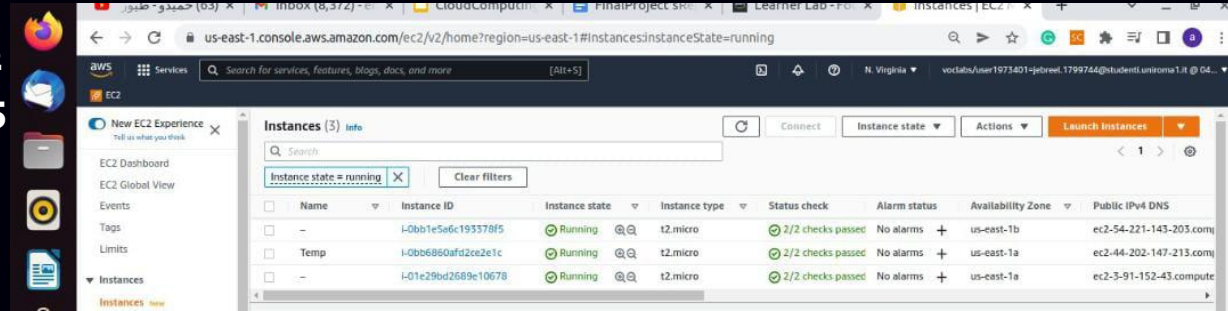


2. Stress Test - Alarm-high



3. Test Auto Scaling

- For checking Auto Scaling, we terminated two of the instances manually and, as you can see in the image below, another instance is created.
- Desired capacity = 2
- Minimum capacity = 2
- Maximum capacity = 5



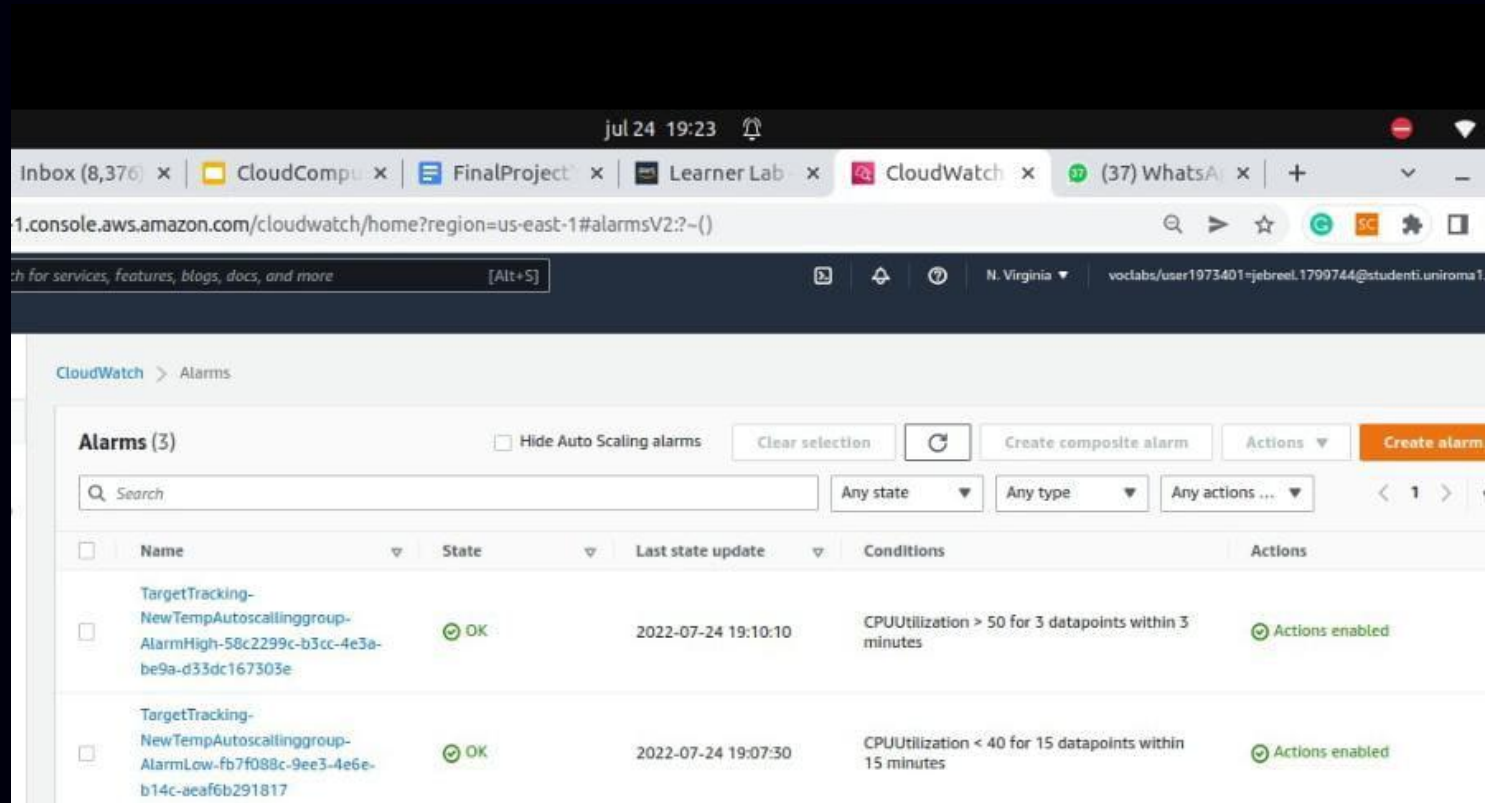
3. Test Auto Scaling with stress applied

- We increased the load on all 3 instances that we had. Since we set an alarm to be triggered when the average cpu utilization is higher than 50% we had an alarm triggered and also three new instances were created as we put that the maximum instance is five

The screenshot shows the AWS Management Console for the EC2 service. The main content area displays a list of 6 instances. The first instance is terminated, and the other five are running. The running instances are all t2.micro type and have 2/2 checks passed. The left sidebar shows the EC2 Dashboard and various navigation options.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Ava
-	I-0d673dd063a55e44c	Terminated	t2.micro	-	No alarms	US-
-	I-0768bdf563d01df70	Running	t2.micro	2/2 checks passed	No alarms	US-
-	I-0942baa8e60c8f478	Running	t2.micro	2/2 checks passed	No alarms	US-
Temp	I-0bb6860afd2ce2e1c	Running	t2.micro	2/2 checks passed	No alarms	US-
-	I-09e6685e4daf42cd	Running	t2.micro	2/2 checks passed	No alarms	US-
-	I-0e66ad3e490b4bcdb	Running	t2.micro	2/2 checks passed	No alarms	US-




Alarms' status are ok again






The screenshot shows the AWS CloudWatch Alarms console. The browser address bar indicates the URL is `1.console.aws.amazon.com/cloudwatch/home?region=us-east-1#alarmsV2?~()`. The page title is "CloudWatch > Alarms".

Alarms (3)

☐ Hide Auto Scaling alarms

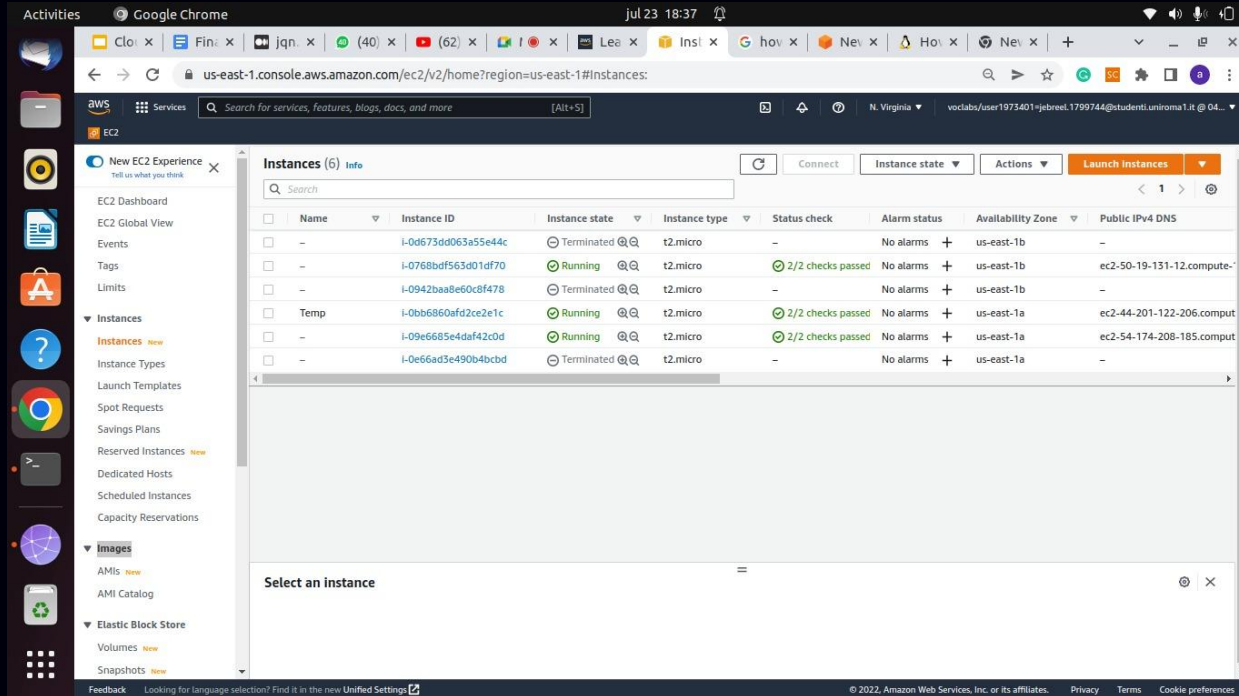
Clear selection  Create composite alarm Actions  Create alarm 

Search: Any state  Any type  Any actions ...  < 1 >

<input type="checkbox"/>	Name	State	Last state update	Conditions	Actions
<input type="checkbox"/>	TargetTracking-NewTempAutoscalinggroup-AlarmHigh-58c2299c-b3cc-4e3a-be9a-d33dc167303e	OK	2022-07-24 19:10:10	CPUUtilization > 50 for 3 datapoints within 3 minutes	Actions enabled
<input type="checkbox"/>	TargetTracking-NewTempAutoscalinggroup-AlarmLow-fb7f088c-9ee3-4e6e-b14c-aeaf6b291817	OK	2022-07-24 19:07:30	CPUUtilization < 40 for 15 datapoints within 15 minutes	Actions enabled

3. Test Auto Scaling after removing the stresses

- When we decreased the cpu utilization again, the instances were terminated since the cpu stress was removed



The screenshot shows the AWS Management Console interface for EC2 instances. The left sidebar contains navigation options like EC2 Dashboard, EC2 Global View, Events, Tags, Limits, Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Scheduled Instances, Capacity Reservations, Images, AMIs, AMI Catalog, Elastic Block Store, Volumes, and Snapshots. The main content area displays a table of instances with columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, and Public IPv4 DNS. The table shows six instances, with three in a 'Terminated' state and three in a 'Running' state. The 'Running' instances are 'Temp', 'i-09e6685e4daf42cd', and 'i-0e66ad3e490b4bcd'. The 'Terminated' instances are 'i-0d673dd063a55e44c', 'i-0768bdf563d01df70', and 'i-0942baa8e60c8f478'. The bottom of the console shows a 'Select an instance' dialog box.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
-	i-0d673dd063a55e44c	Terminated	t2.micro	-	No alarms	us-east-1b	-
-	i-0768bdf563d01df70	Terminated	t2.micro	2/2 checks passed	No alarms	us-east-1b	ec2-50-19-131-12.compute-
-	i-0942baa8e60c8f478	Terminated	t2.micro	-	No alarms	us-east-1b	-
Temp	i-0bb6860afd2ce2e1c	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a	ec2-44-201-122-206.comput
-	i-09e6685e4daf42cd	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a	ec2-54-174-208-185.comput
-	i-0e66ad3e490b4bcd	Terminated	t2.micro	-	No alarms	us-east-1a	-

Summary report

Activities Google Chrome jul 23 18:14

us-east-1.console.aws.amazon.com/ec2/v2/home?region=us-east-1#AutoScalingGroupDetails:sd=TemperatureTemplate;view=a...

Search for services, features, blogs, docs, and more [Alt+S]

EC2

Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Scheduled Instances
Capacity Reservations

Images
AMIs
AMI Catalog

Elastic Block Store
Volumes
Snapshots
Lifecycle Manager

Network & Security
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces

Load Balancing
Load Balancers
Target Groups

Auto Scaling
Launch Configurations
Auto Scaling Groups

Successful	Terminating EC2 instance: i-0942baa8e60c8f478	At 2022-07-23T16:03:30Z a monitor alarm TargetTracking-TemperatureTemplate-AlarmLow-5fa5bd2-d049-4a9d-8513-68010e6596 in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 3 to 2. At 2022-07-23T16:03:42Z an instance was taken out of service in response to a difference between desired and actual capacity, shrinking the capacity from 3 to 2. At 2022-07-23T16:03:42Z instance i-0942baa8e60c8f478 was selected for termination.	2022 July 23, 06:03:42 PM +02:00	2022 July 23, 06:09:42 PM +02:00
Successful	Terminating EC2 instance: i-0e66ad3e490b4bcdb	At 2022-07-23T16:02:04Z a monitor alarm TargetTracking-TemperatureTemplate-AlarmLow-0f6c428-f825-452b-bb0b-a9e1b255e2d in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 4 to 3. At 2022-07-23T16:02:08Z an instance was taken out of service in response to a difference between desired and actual capacity, shrinking the capacity from 4 to 3. At 2022-07-23T16:02:08Z instance i-0e66ad3e490b4bcdb was selected for termination.	2022 July 23, 06:02:08 PM +02:00	2022 July 23, 06:08:08 PM +02:00
Successful	Terminating EC2 instance: i-0d673dd063a55e44c	At 2022-07-23T16:01:14Z a monitor alarm TargetTracking-TemperatureTemplate-AlarmLow-6301b1a6-fd42-49a4-9f68-5ad9acc1855c in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 5 to 4. At 2022-07-23T16:01:20Z an instance was taken out of service in response to a difference between desired and actual capacity, shrinking the capacity from 5 to 4. At 2022-07-23T16:01:21Z instance i-0d673dd063a55e44c was selected for termination.	2022 July 23, 06:01:21 PM +02:00	2022 July 23, 06:07:09 PM +02:00
Successful	Launching a new EC2 instance: i-0768bd1f563d01df70	At 2022-07-23T15:21:51Z a monitor alarm TargetTracking-TemperatureTemplate-AlarmHigh-ce1f7be1-55ac-4d1e-a24d-21fc19d0a0f04 in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 4 to 5. At 2022-07-23T15:21:58Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 4 to 5.	2022 July 23, 05:22:00 PM +02:00	2022 July 23, 05:24:32 PM +02:00
Successful	Launching a new EC2 instance: i-0942baa8e60c8f478	At 2022-07-23T15:16:51Z a monitor alarm TargetTracking-TemperatureTemplate-AlarmHigh-ce1f7be1-55ac-4d1e-a24d-21fc19d0a0f04 in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 2 to 4. At 2022-07-23T15:16:59Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 2 to 4.	2022 July 23, 05:17:02 PM +02:00	2022 July 23, 05:20:04 PM +02:00
Successful	Launching a new EC2 instance: i-0e66ad3e490b4bcdb	At 2022-07-23T15:16:51Z a monitor alarm TargetTracking-TemperatureTemplate-AlarmHigh-ce1f7be1-55ac-4d1e-a24d-21fc19d0a0f04 in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 2 to 4. At 2022-07-23T15:16:59Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 2 to 4.	2022 July 23, 05:17:02 PM +02:00	2022 July 23, 05:20:04 PM +02:00
Successful	Launching a new EC2 instance: i-09e6685e4da42c0d	At 2022-07-23T15:04:24Z an instance was launched in response to an unhealthy instance needing to be replaced.	2022 July 23, 05:04:27 PM +02:00	2022 July 23, 05:04:59 PM +02:00
Successful	Terminating EC2 instance: i-0a176b45286a5b087	At 2022-07-23T15:04:24Z an instance was taken out of service in response to an EC2 health check indicating it has been terminated or stopped.	2022 July 23, 05:04:24 PM +02:00	2022 July 23, 05:09:45 PM +02:00
Successful	Launching a new EC2 instance: i-0d673dd063a55e44c	At 2022-07-23T15:02:22Z an instance was launched in response to an unhealthy instance needing to be replaced.	2022 July 23, 05:02:25 PM +02:00	2022 July 23, 05:02:57 PM +02:00
Successful	Terminating EC2 instance: i-059875e383ec89457	At 2022-07-23T15:02:22Z an instance was taken out of service in response to an EC2 health check indicating it has been terminated or stopped.	2022 July 23, 05:02:22 PM +02:00	2022 July 23, 05:07:29 PM +02:00

Feedback Looking for language selection? Find it in the new Unified Settings

© 2022, Amazon Web Services, Inc. or its affiliates Privacy Terms Cookie preferences

Thank You