# Deploying Application In Elastic Beanstalk using EB CLI and EB Extensions

Created by[Zeal Vora](https://deloittedevelopment.udemy.com/user/cybercorp/)

**AWS Workspace**

**60-120 minutes**

In this lab, you will be a consultant at an E-Commerce based startup that wants to create a simple solution that can quickly deploy the applications of their developers instead of having them manually create and manage underlying resources like EC2, ELB, Auto-Scaling groups,Web Server configuration, Deployment Configurations, and others. The architecture group thinks that using Elastic Beanstalk CLI and .ebextensions will allow developers to quickly automate, customize the environment and orchestrate the backend infrastructure. Your boss wants you to build a proof-of-concept for this solution.

The Engineering team has raised an issue. For the part of their everyday development and testing cycle, they would like an alternative to the Elastic Beanstalk console that can run via CLI and be automated easily via scripts. If we do this, they won’t have to manually login to the AWS console and create the Elastic Beanstalk application and environment through guided console steps. I want you to create a POC which allows developers to easily create and customize the Elastic Beanstalk environment using CLI. Developers have shared the link to download their sample application based on Docker and further customization required at the EC2 instance level within a file called **install-system.txt** file. I have provided it for you in the Resources.

All the tasks specified at the install-system.txt file should be completed by making use of Elastic Beanstalk CLI and Elastic Beanstalk Extensions (.ebextensions) scripts. The OS for installing EB CLI should be based on Amazon Linux 2. The Elastic Beanstalk application should be based on a single EC2 instance-based setup without any load balancers.

TASK

1-Create Temporary PEM private Key for EC2

2-Launche EC2 for installing Elastic Beanstalk CLI

3-Install Elastic Beanstalk cLI

4-Configure application & initialize elastic beanstalk

5-create elastic beanstalk extensions

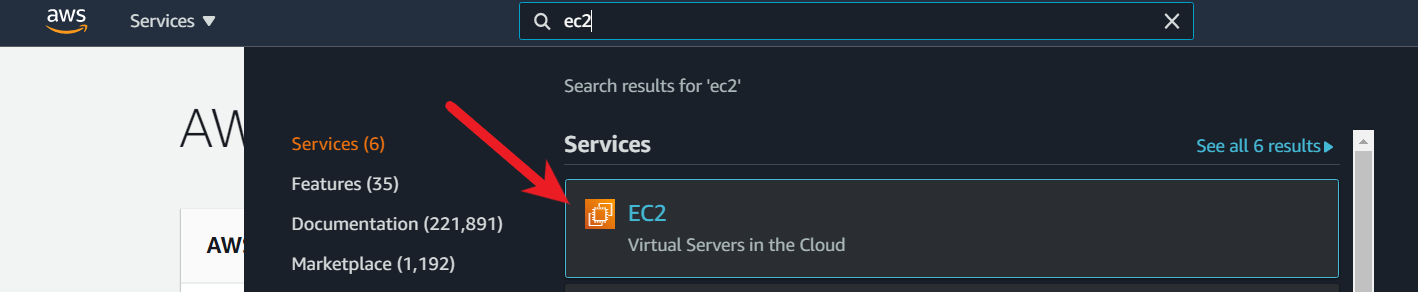
6-Deploying application

7-Verify the EB extensions deployment in EC2 and Clean Up

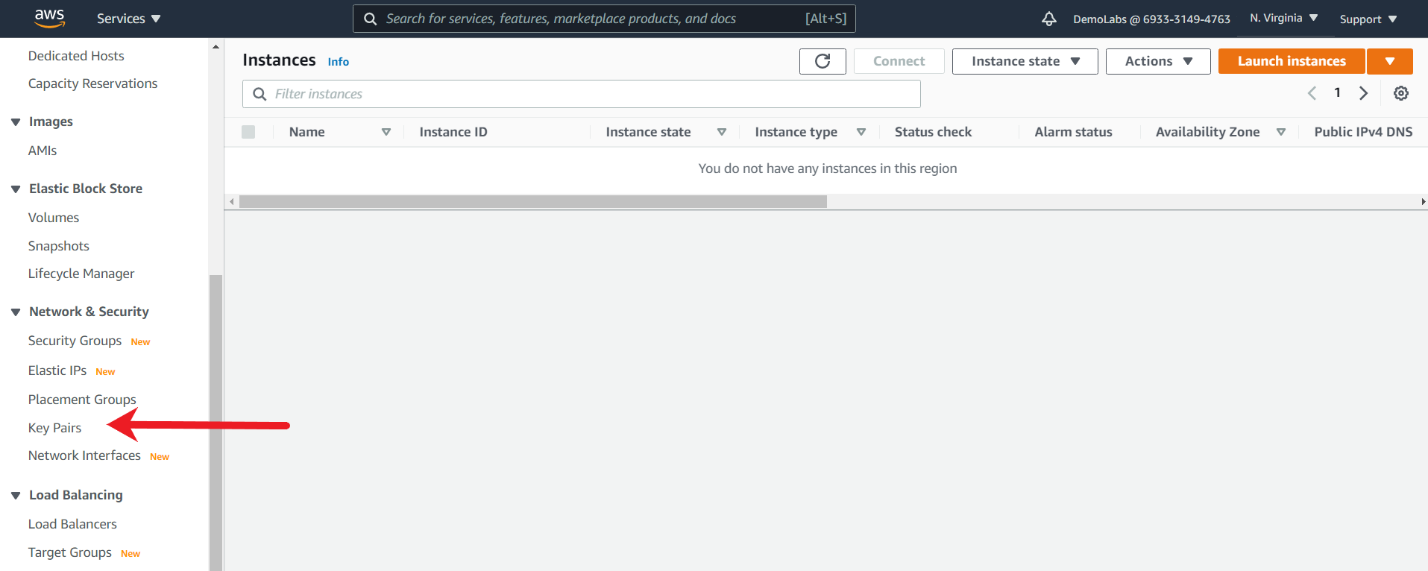
Create Temporary PEM private Key for EC2

You can make use of EC2 instances to install and configure EB CLI and Extensions. Create a new key-pair for EC2 so that you can connect over SSH from your local workstation.

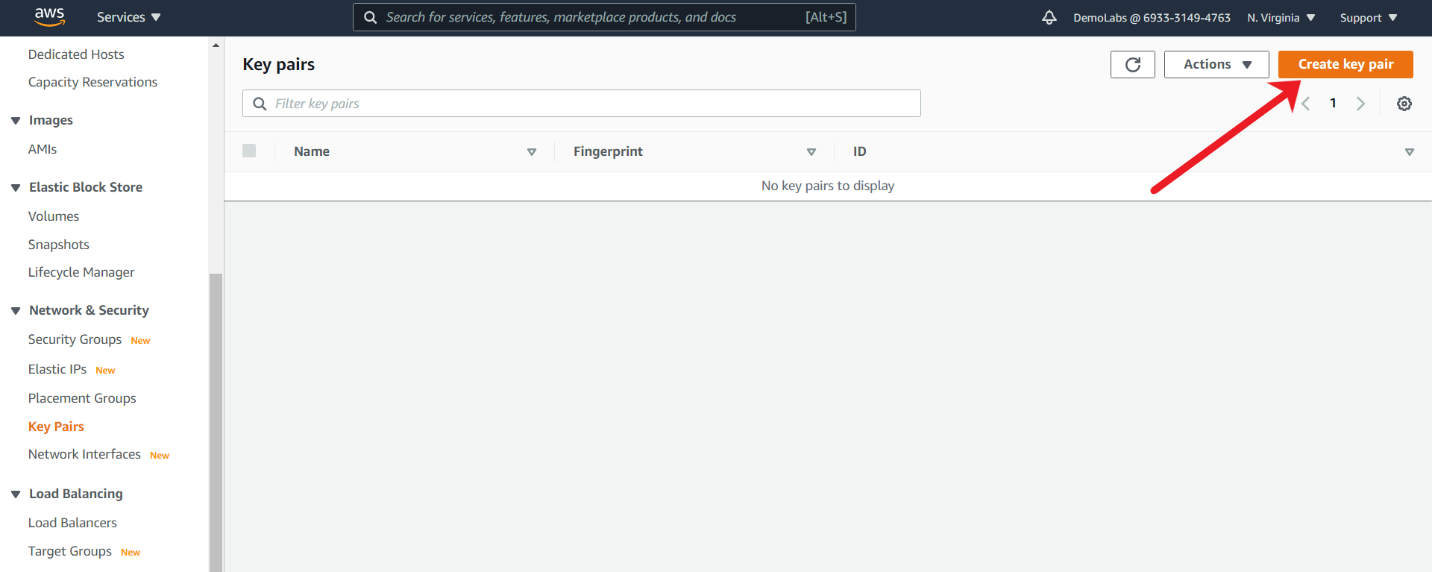
1. Open the Amazon EC2 console by searching for EC2.



1. In the navigation pane, under Network & Security, choose Key Pairs.



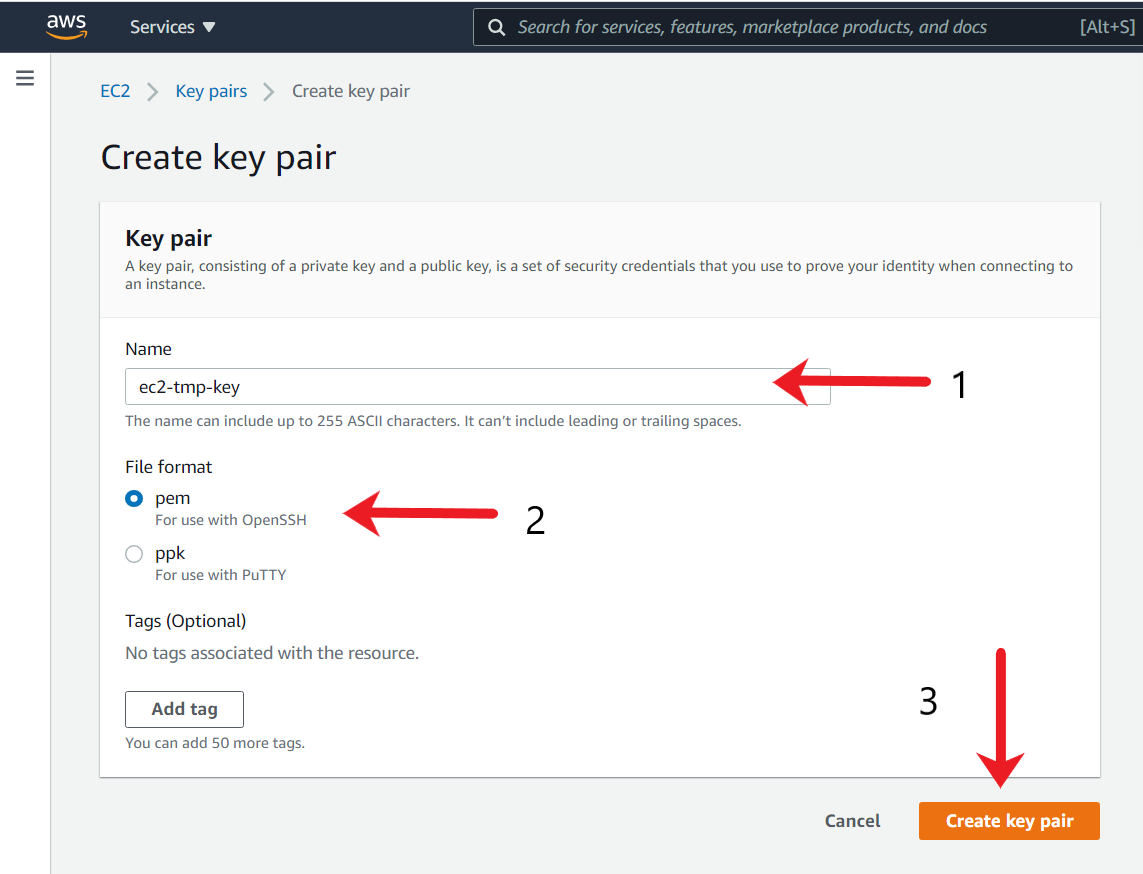
1. From the Key-Pair console, Choose Create key pair.



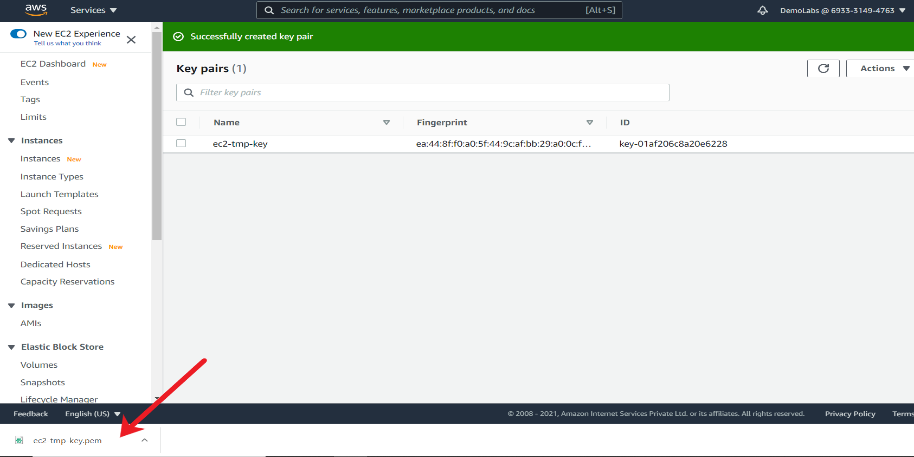
1. For Name, enter a descriptive name for the key pair.

For File format, choose the format in which to save the private key. For this use-case, I have selected PEM format, however if you are using tools like Putty, you can even select PPK.

Choose Create key pair.



The private key file is automatically downloaded by your browser. Save the private key file in a safe place.



<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html>

Link to download sample application based on Docker:

https://www.mediafire.com/file/3839yw0qfis09wu/zeal-lab-4.zip

Customization Required as Part of the EC2 instance.

i) vsftpd package needs to be installed and service should be running.

ii) A new Linux user named appuser should be created and should be added as part of Linux group named appgroup

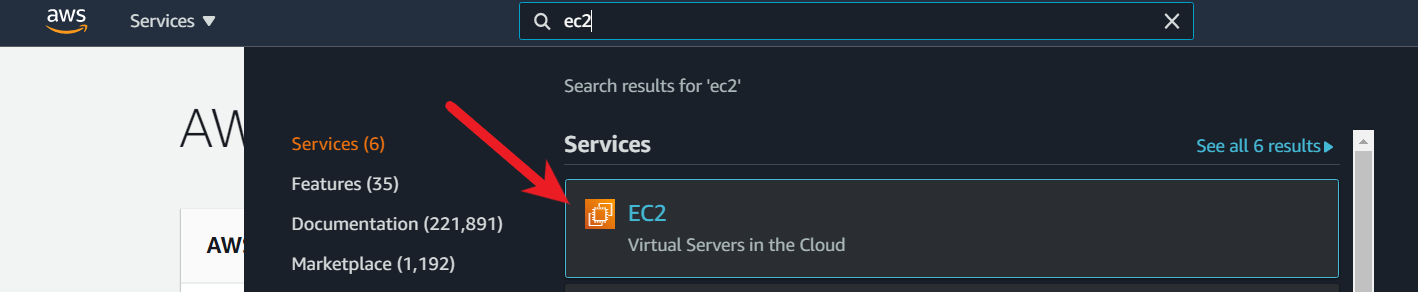
iii) A file needs to be created at /tmp directory with name deployed.txt with following contents:

Sample File deployed .ebextensions

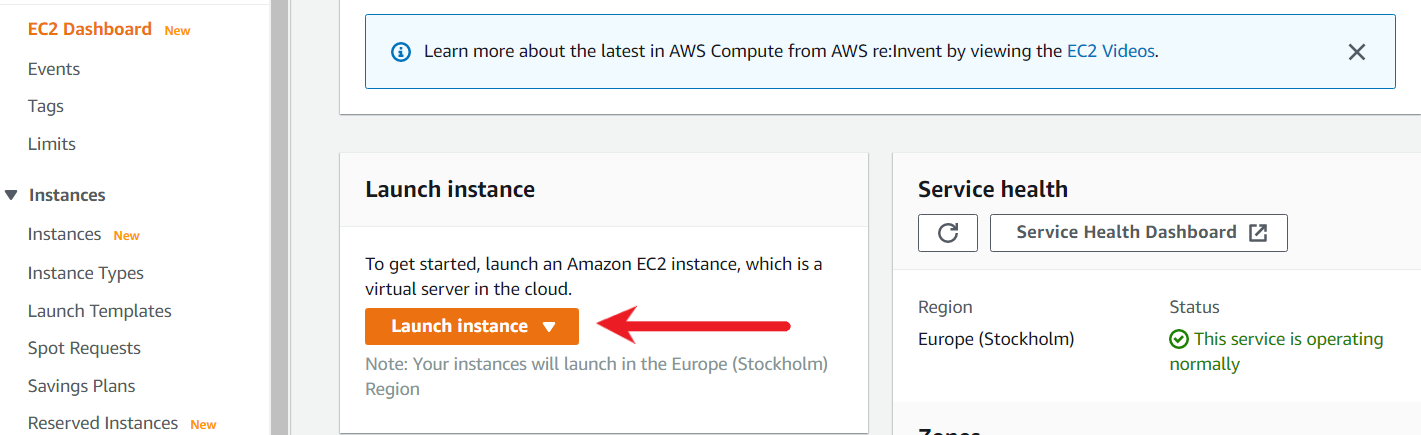
Launche EC2 for installing Elastic Beanstalk CLI

You have to create an EC2 instance based on Amazon Linux 2, which will be used for installing the Elastic Beanstalk CLI. The same instance needs to be used for deploying the Elastic Beanstalk application through the EB CLI.

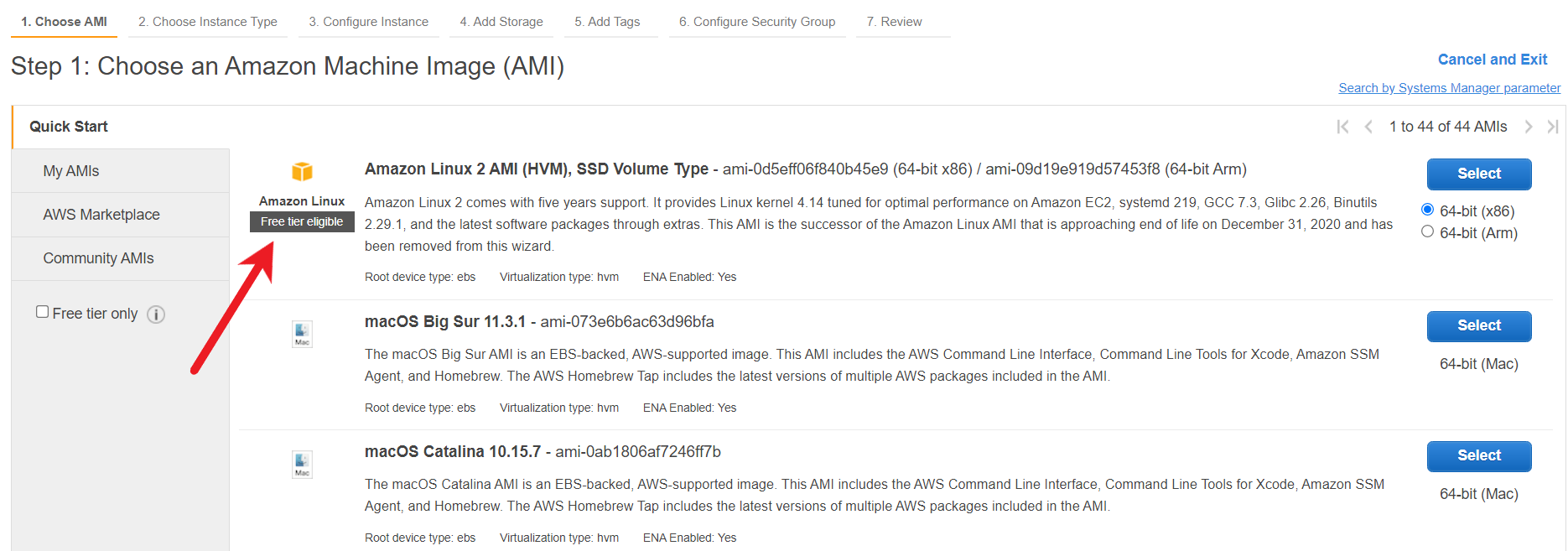
1. Open the Amazon EC2 console by searching for EC2.



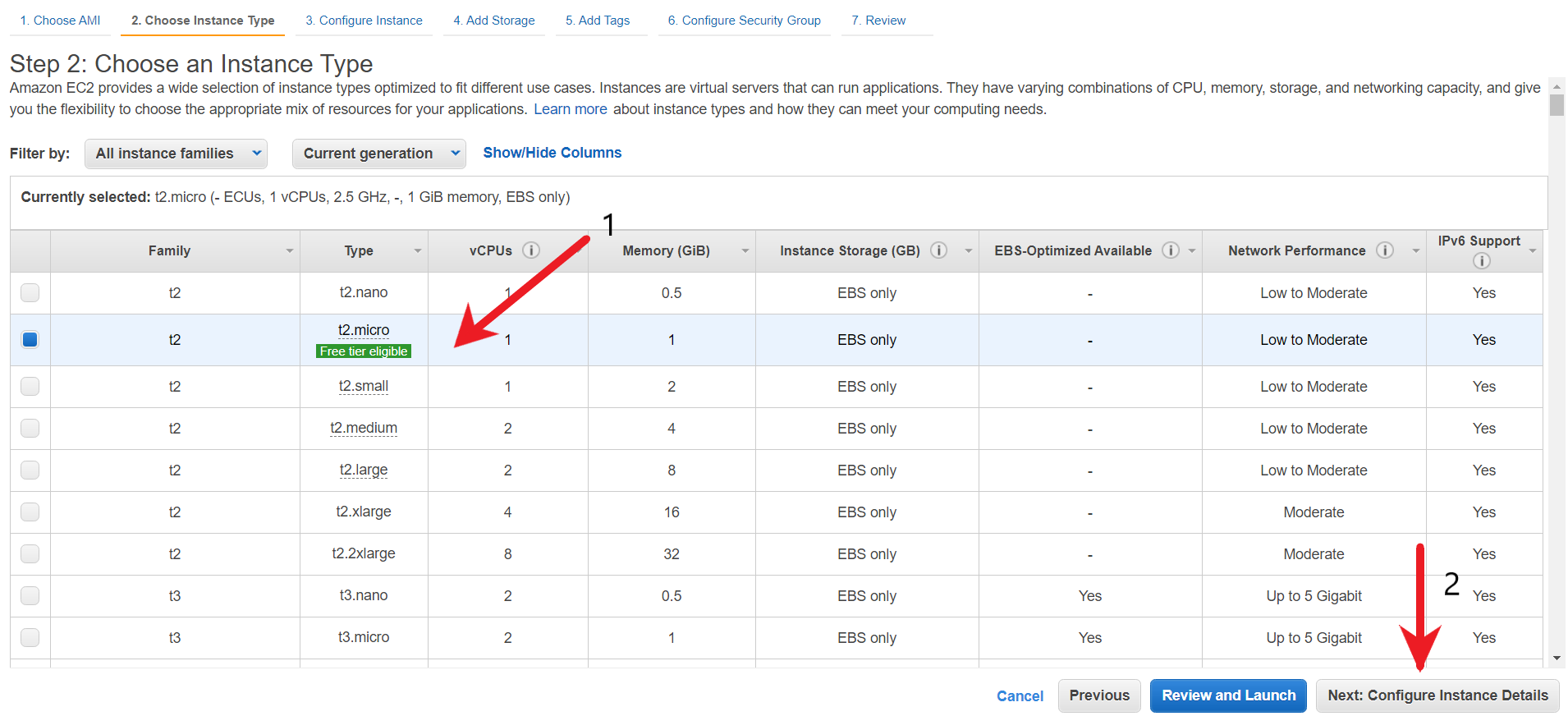
1. From the console dashboard, choose Launch Instance, and select Launch instance from the options that pop up.



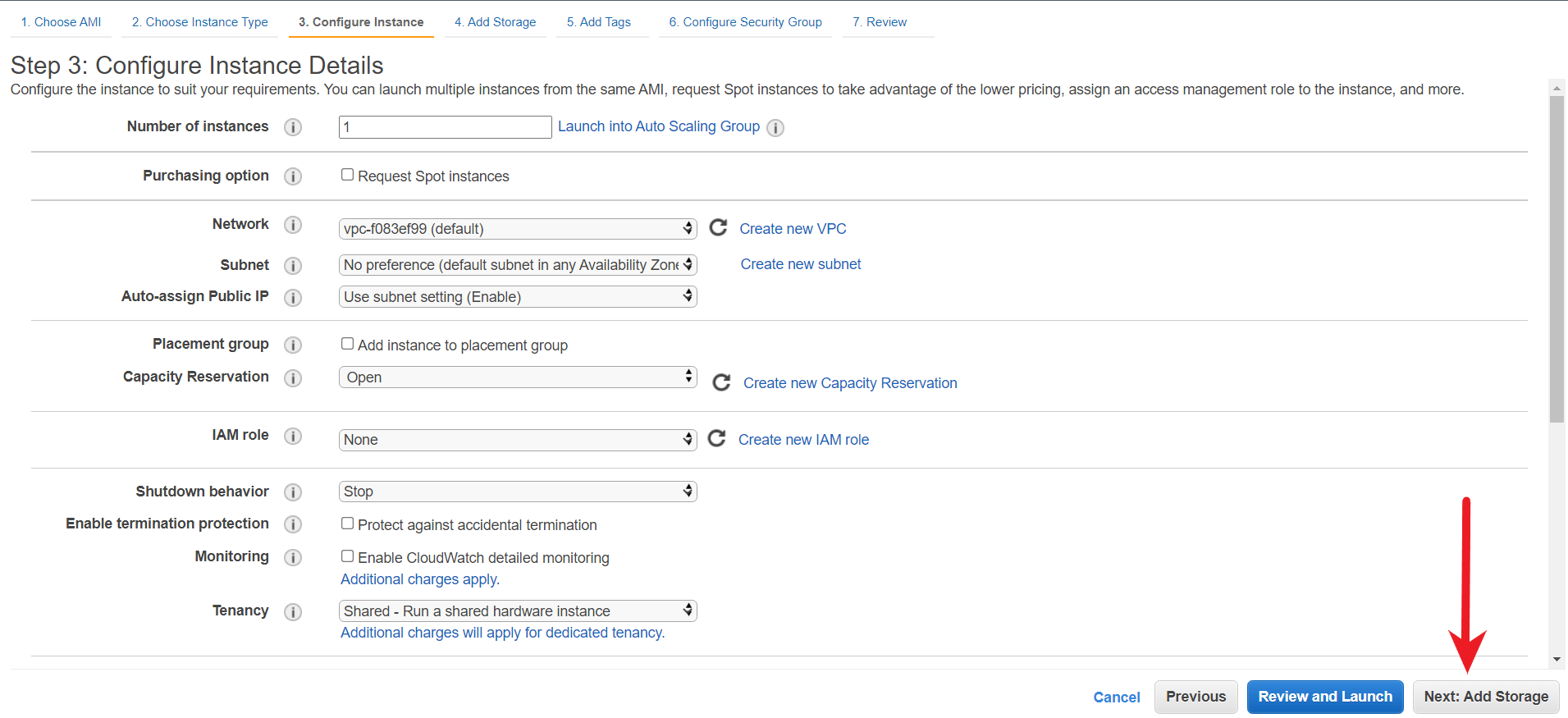
1. Choose an Amazon Machine Image (AMI), locate Amazon Linux 2 AMI (HVM), SSD Volume Type, and then choose Select. (This AMI is labeled "Free tier eligible" and can be found at the top of the list.)



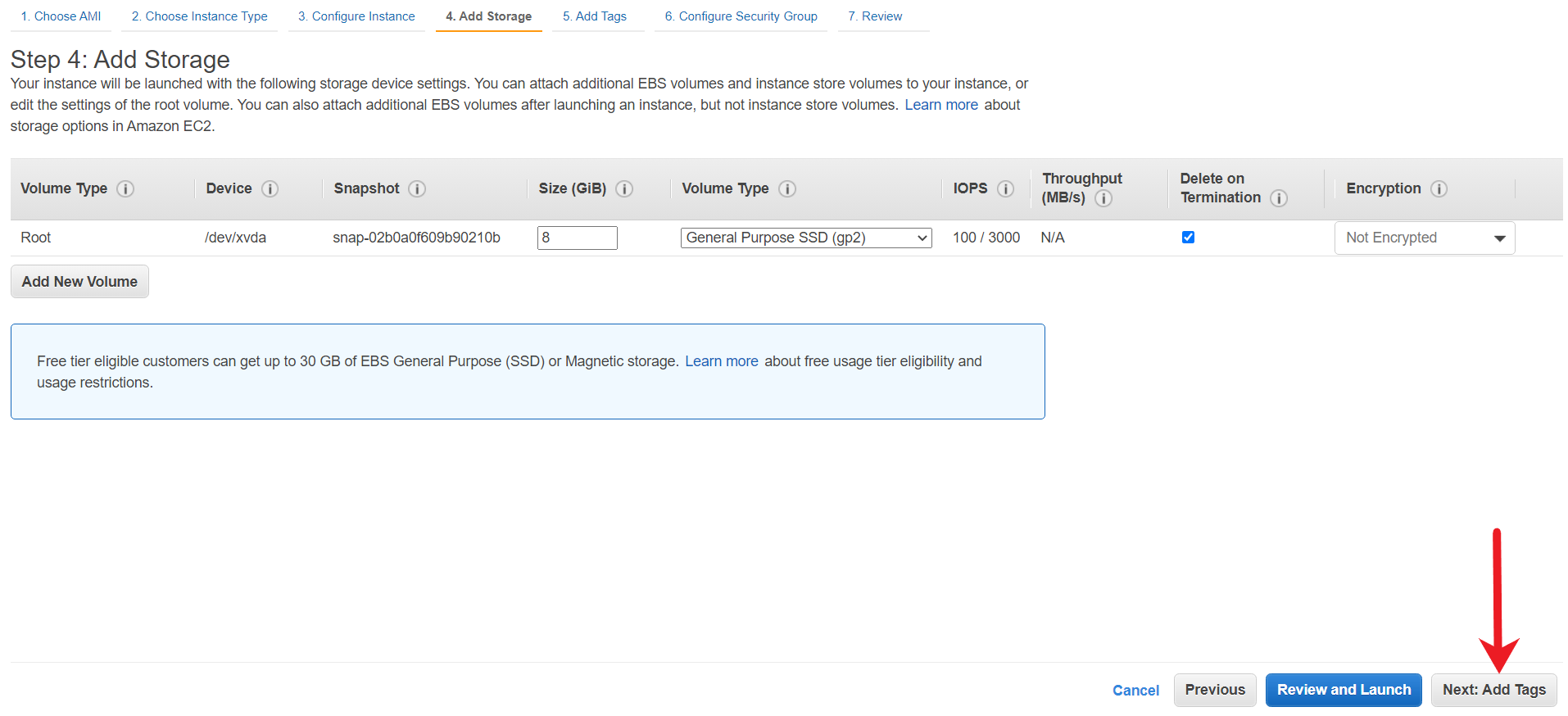
1. Choose an Instance Type page, choose the free tier eligible t2.micro type as the hardware configuration for your instance, and then choose Next: Configure Instance Details.



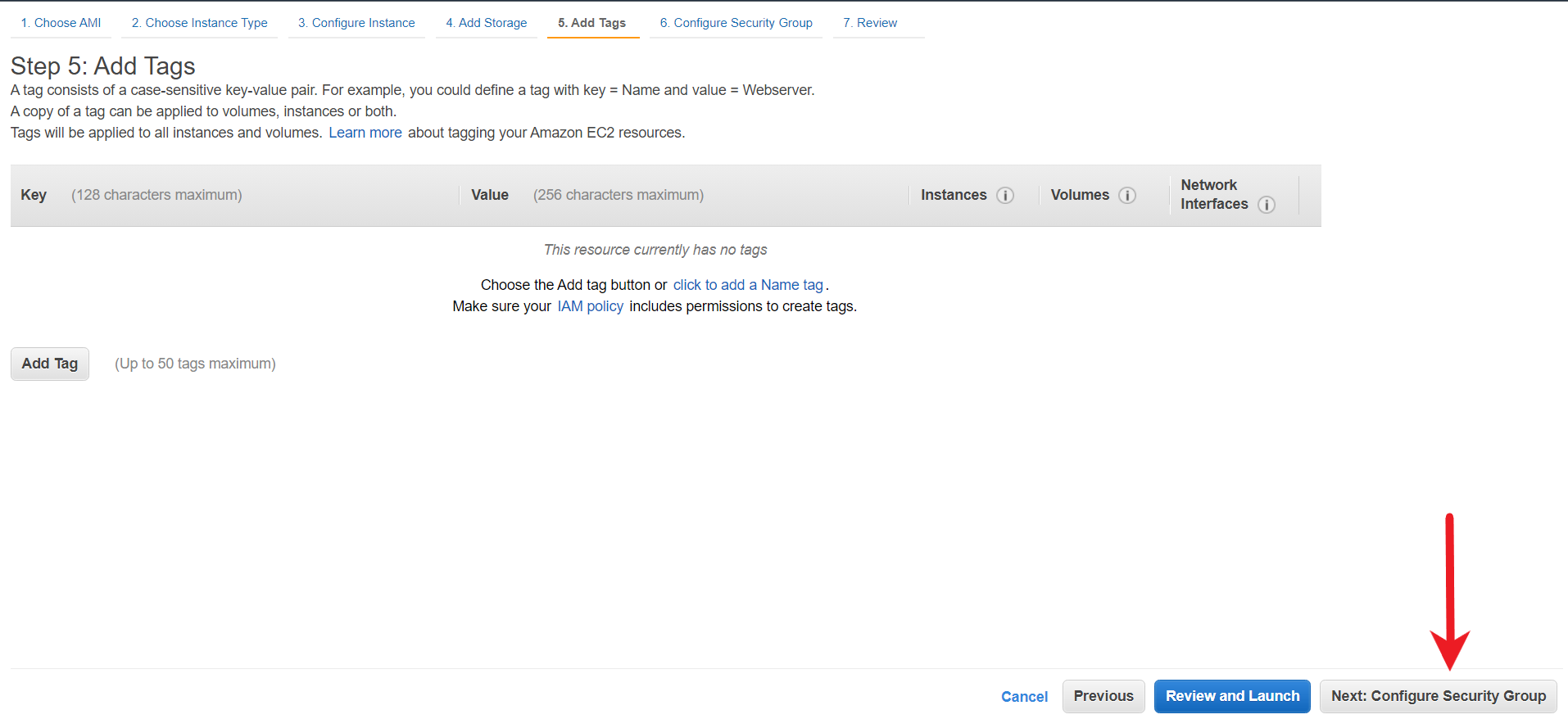
1. In the “Configure Instance Details” page, leave everything as default and click on “Next: Add Storage”.



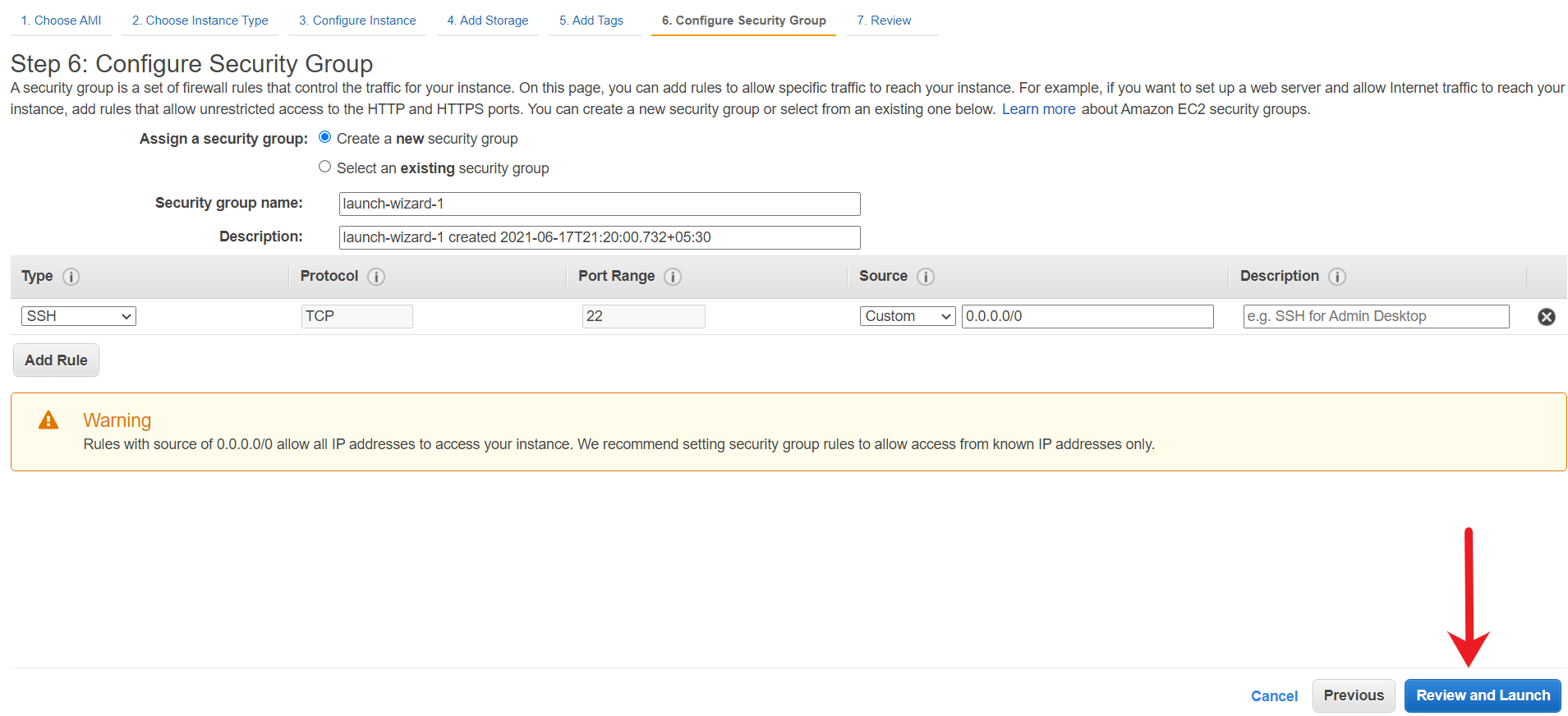
1. Add Storage page unchanged, and then choose Next: Add Tags.



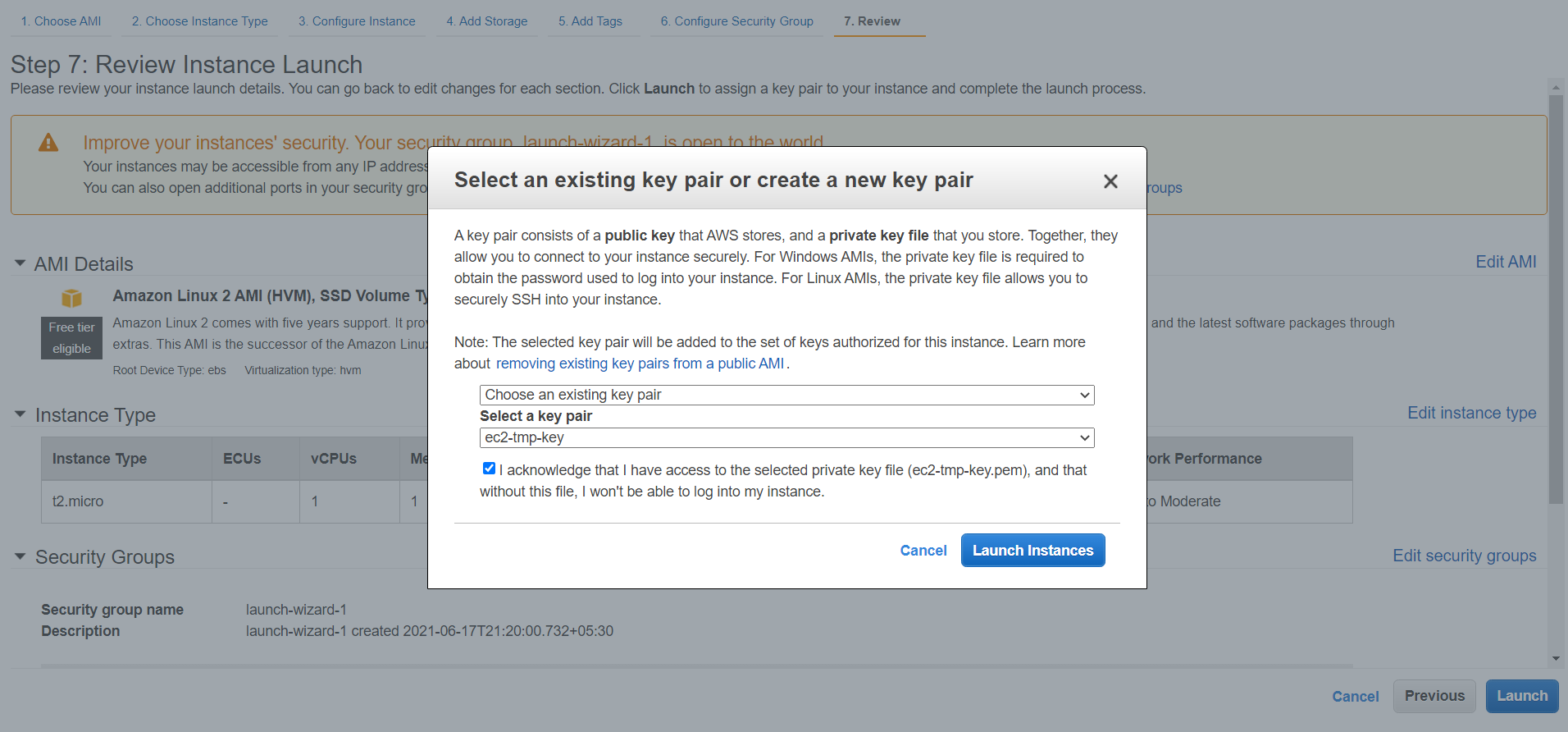
1. Leave Tags page unchanged and then choose Next: Configure Security Group.



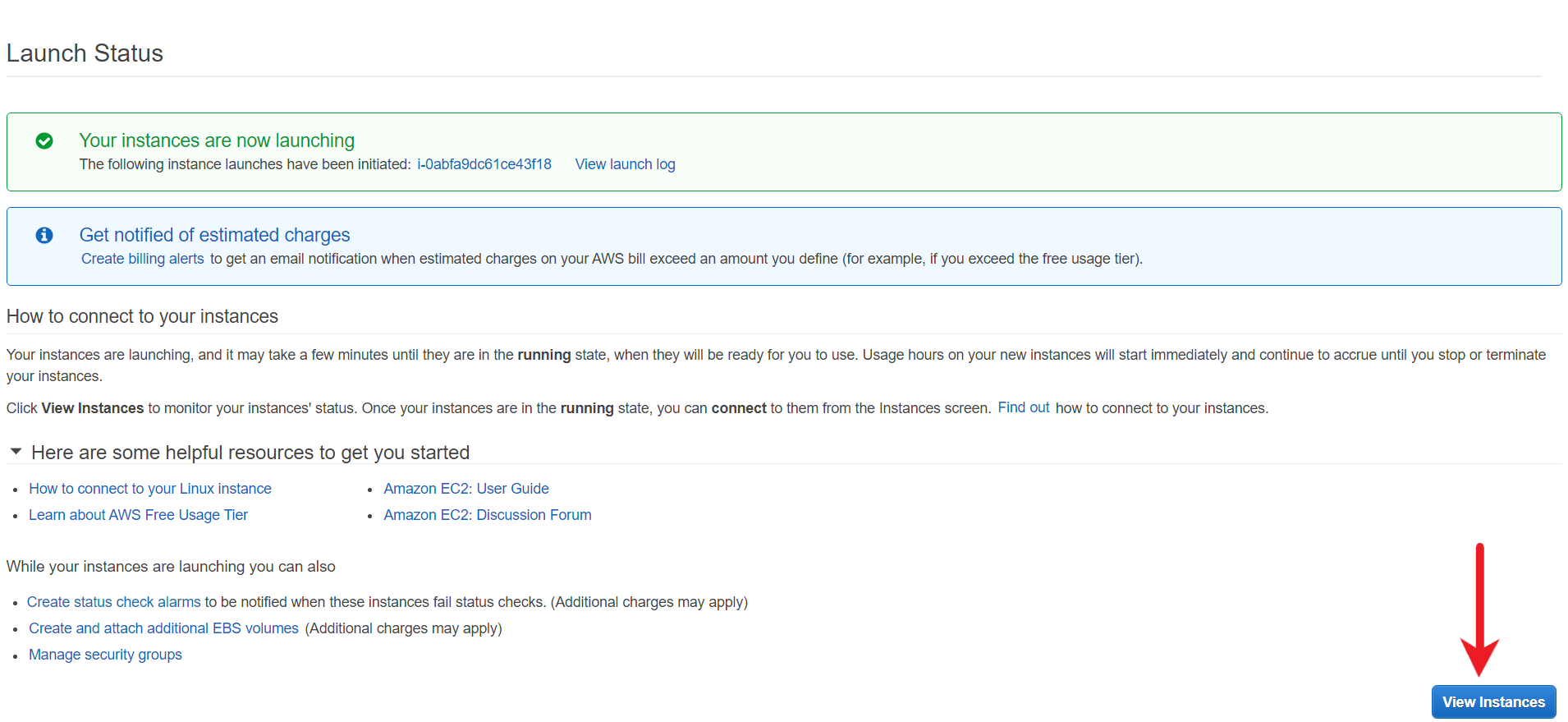
1. Leave the Security Group page unchanged and then choose Review and Launch.



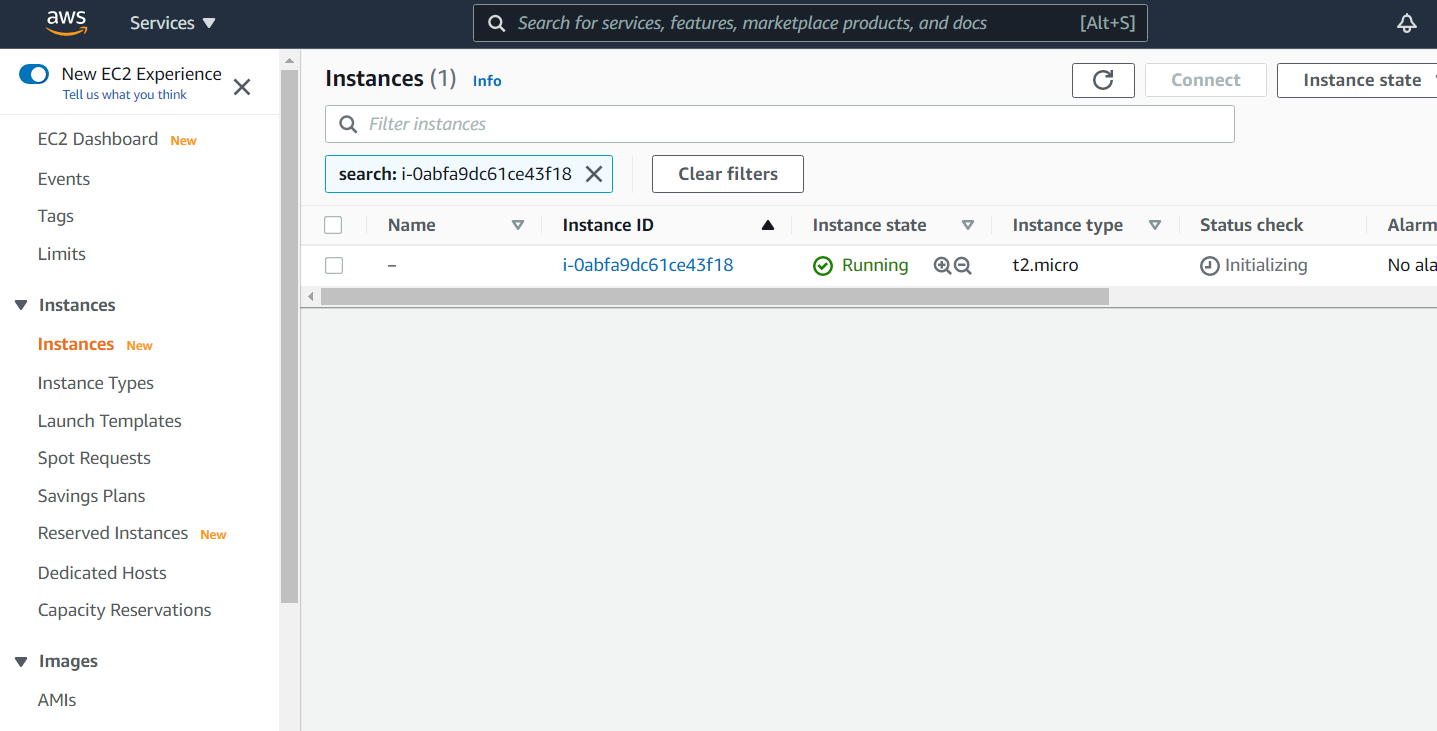
1. On the Review Instance Launch page, choose Launch. When prompted for a key pair, choose the temporary key pair that you had created. When you are ready, select the acknowledgment check box, and then choose Launch Instances.



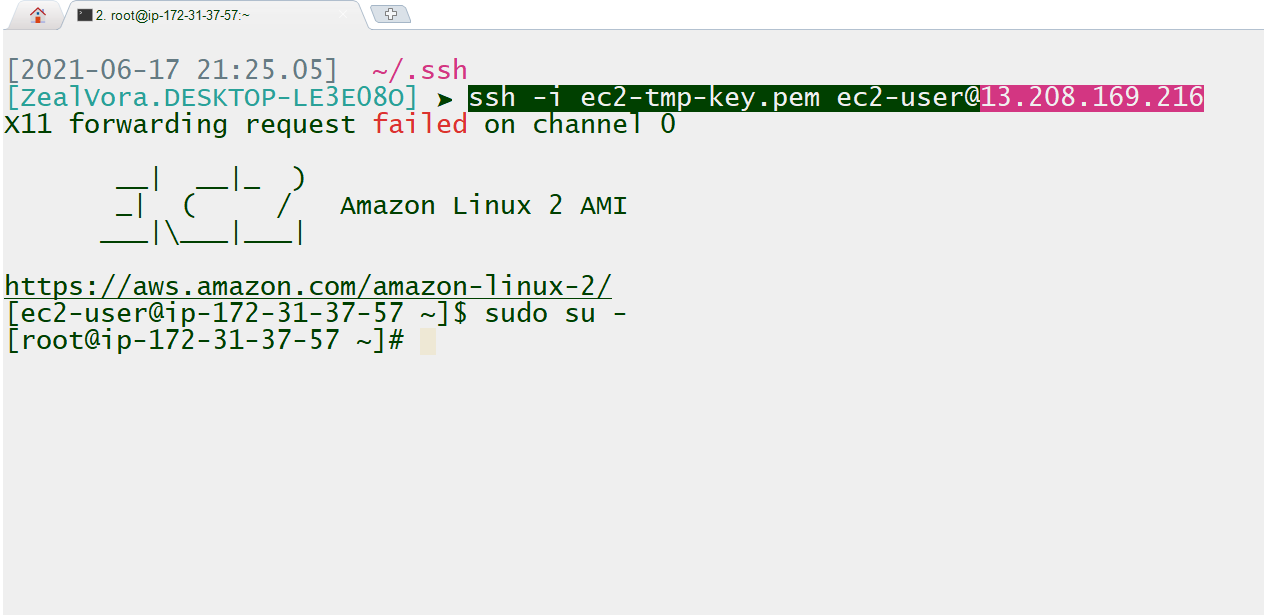
1. Choose View Instances to close the confirmation page and return to the console.



1. View the status of the launch on the Instances page. When you launch an instance, its initial state is pending. After the instance starts, its state changes to running, and it receives a public DNS name.



12. For verification, try to connect to the EC2 from your local SSH terminal. You should be able to log in with the ec2-user and the temporary key. You can also run the sudo su - command to see if you can login to root.



In-case if you get a “bad permission” error, you can change the permission of the file to 400 with the following command:

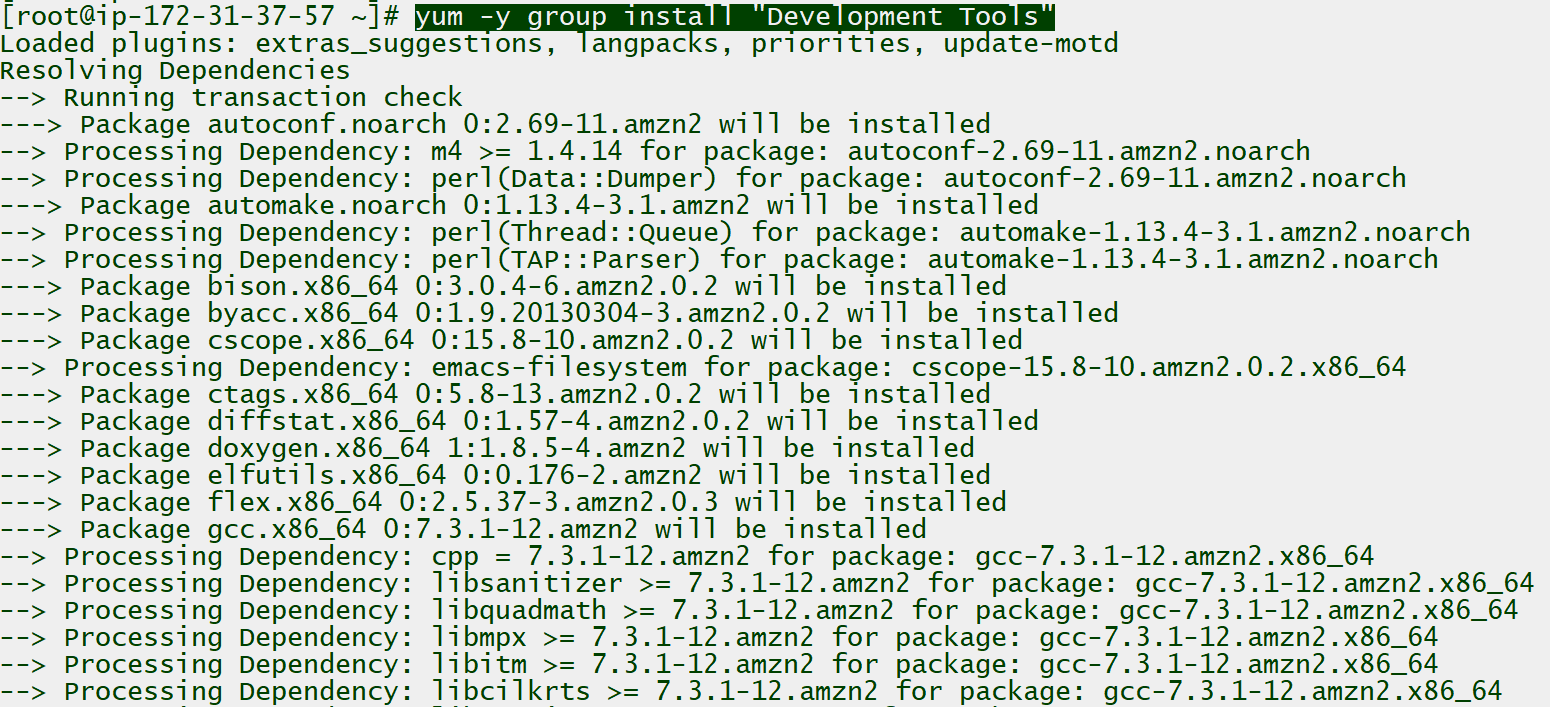
chmod 400 ec2-tmp-key.pem

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Instances.html>

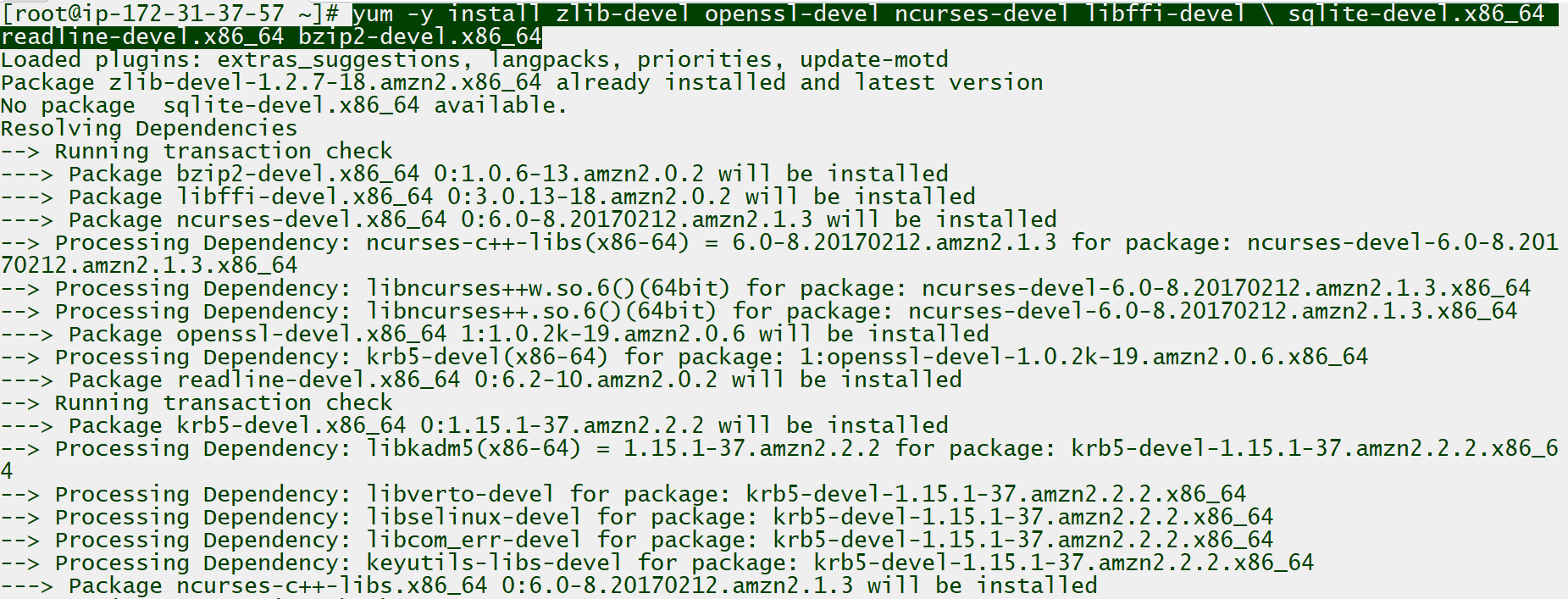
Install Elastic Beanstalk cLI

You now need to install the Elastic Beanstalk CLI which will be used for deploying the sample application.

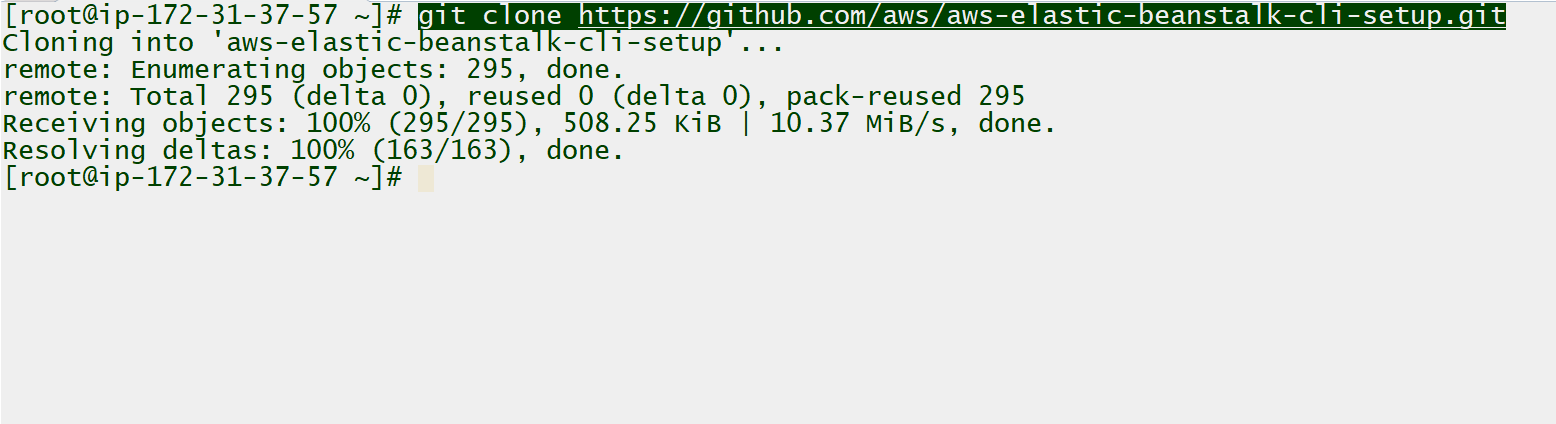
1. Install the group of Development Tools required as pre-requisites with the following command:
   1. yum -y group install "Development Tools"



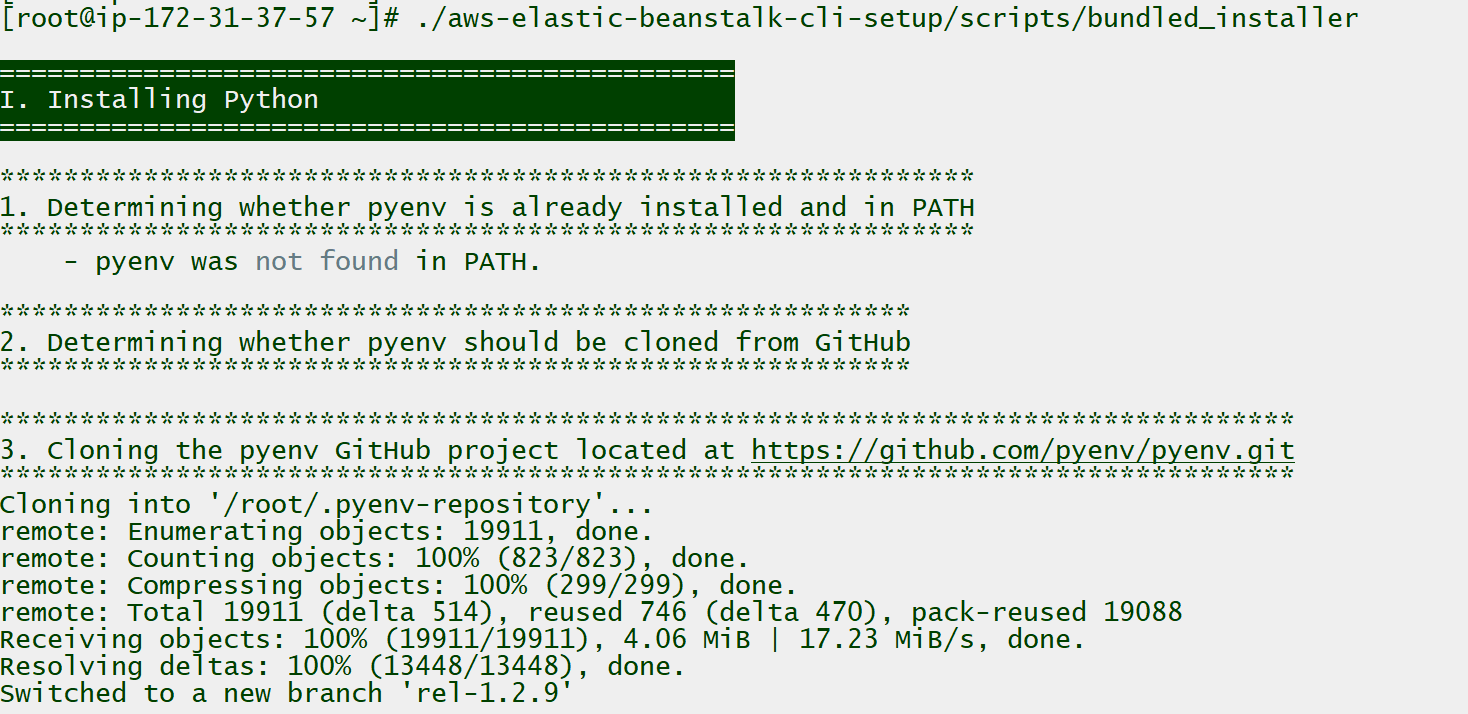
1. Install the additional packages required as pre-requisites with the following command:
   1. yum -y install zlib-devel openssl-devel ncurses-devel libffi-devel \ sqlite-devel.x86\_64 readline-devel.x86\_64 bzip2-devel.x86\_64



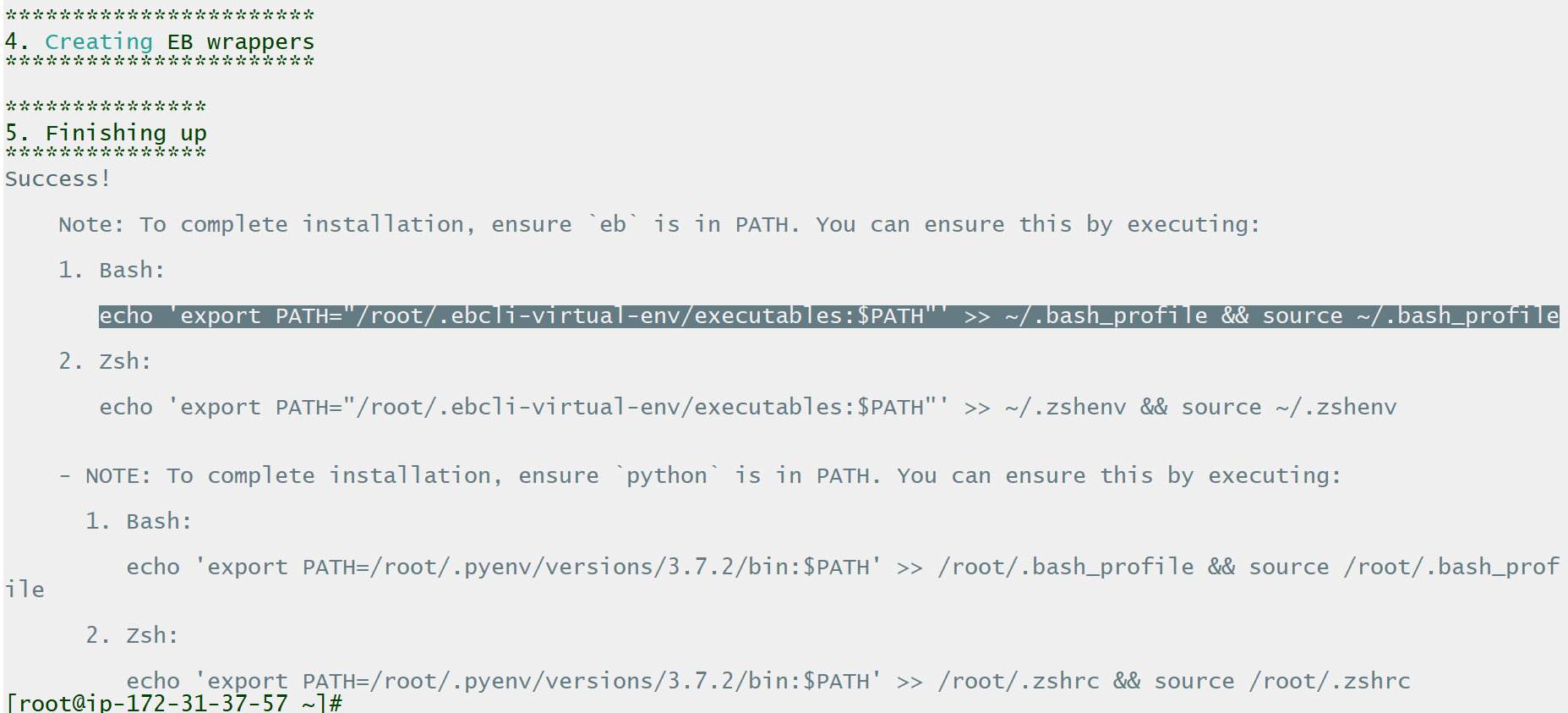
1. Clone the GIT repository provided in the resources section to proceed with the installation of EB CLI.



1. After all the pre-requisite packages have been installed, proceed with the installation of the Elastic Beanstalk CLI. Install the EB CLI with the following command:
   1. ./aws-elastic-beanstalk-cli-setup/scripts/bundled\_installer

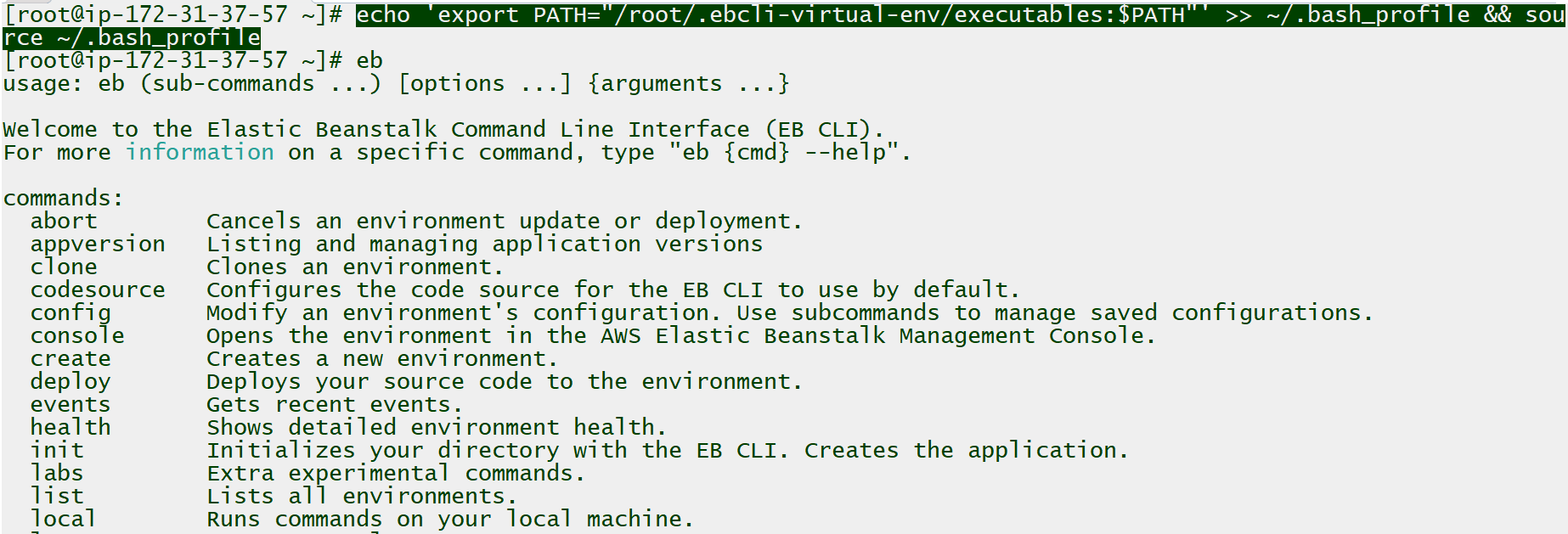


It can take 5-10 minutes for the installation to complete. Once installed, you should see a screen similar to the below screenshot:



1. Set the eb binary in the path. The command to set path is already provided as part of the “Finishing up” section in the above screenshot. Run the following command to set the path:
   1. echo 'export PATH="/root/.ebcli-virtual-env/executables:$PATH"' >> ~/.bash\_profile && source ~/.bash\_profile

After the path is set, run the eb command to see if you can see the CLI options.



<https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/eb-cli3-install.html>

<https://github.com/aws/aws-elastic-beanstalk-cli-setup>

Configure application & initialize elastic beanstalk

After the EB CLI is configured, you need to download the sample developer application and initialize the default values for the Elastic Beanstalk environment using eb init.

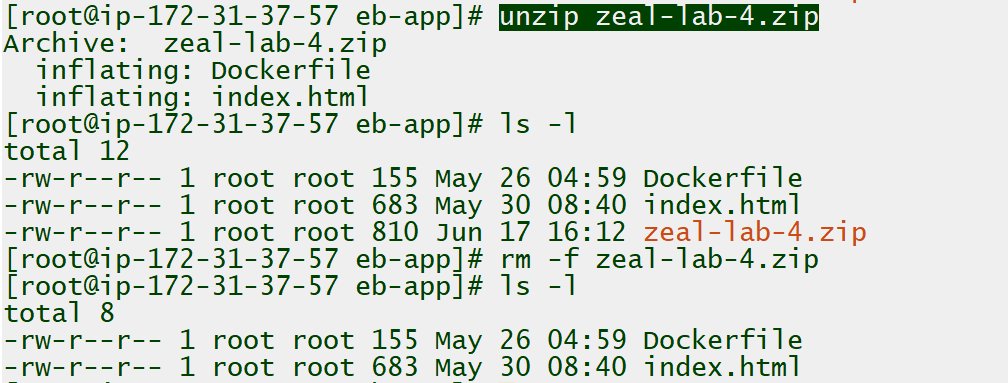
1. Create a new directory where the application file will be stored. You can run the following commands to create a new directory and go inside it:
   1. mkdir eb-app
   2. cd eb-app



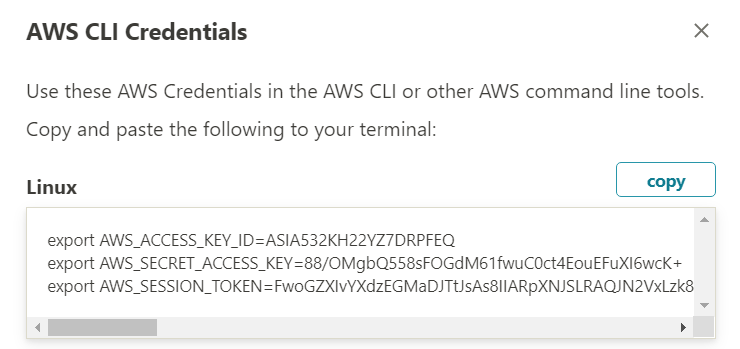
1. Download the application file zip provided by the developers from the install-system.txt file found in the Resources. We will use wget to download the zip archive to the EC2 instance.

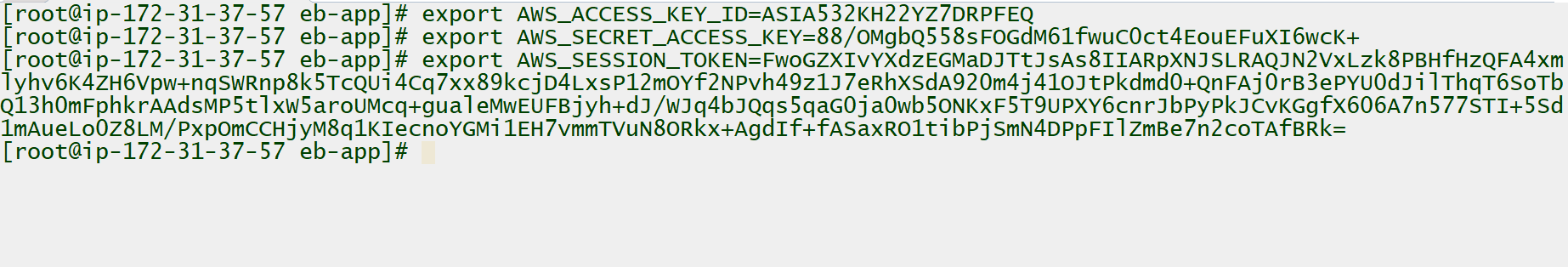


1. Extract the Archive file and remove the archive. Run the following command to unzip the archive file and remove the older archive:
   1. unzip zeal-lab-4.zip
   2. rm -f zeal-lab-4.zip



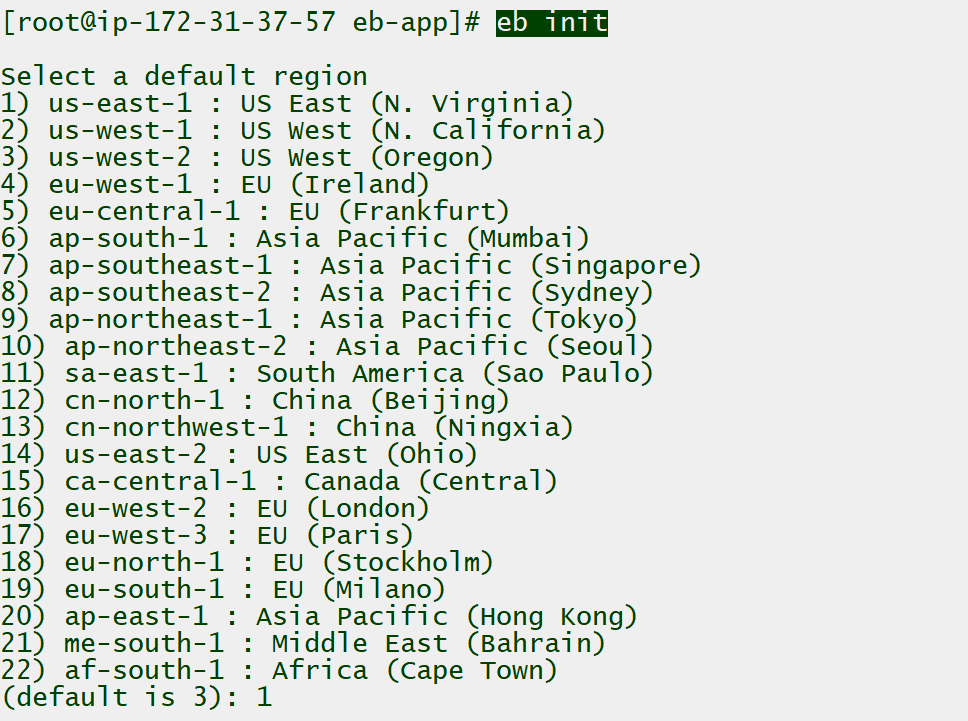
1. Export the Access/Secret keys and the session tokens. Copy these contents from the Workspace console and paste it within the SSH terminal. These keys will be used for creating and deploying applications in Elastic         Beanstalk.



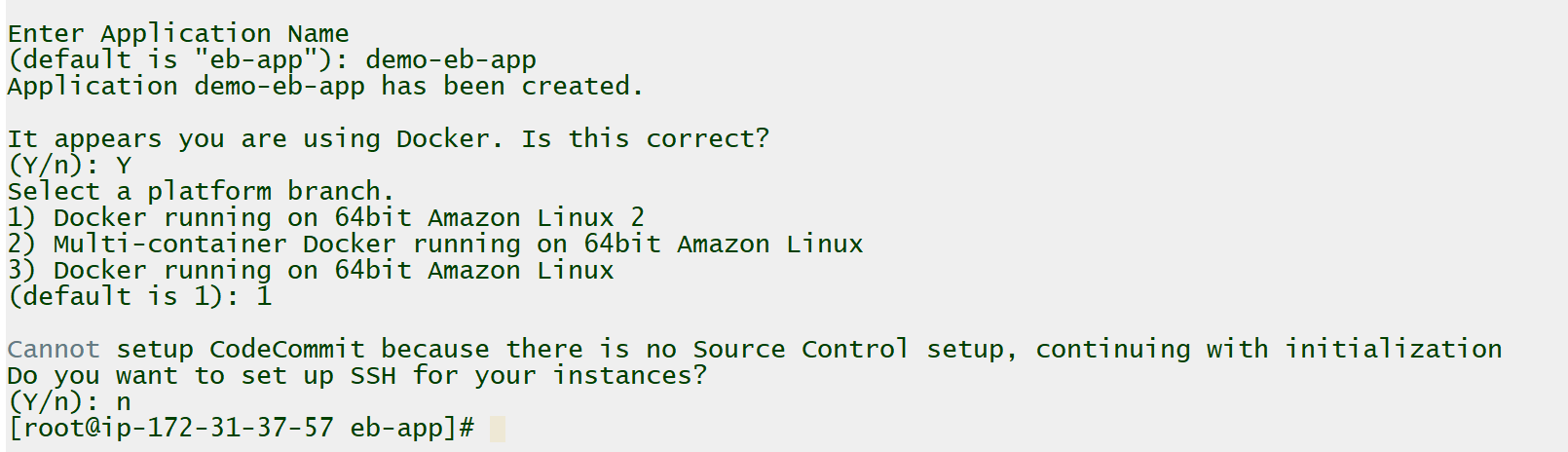


Set the default values for Elastic Beanstalk applications created with EB CLI using the eb init command. You can go ahead and select region of your choice, but some suggested values are:

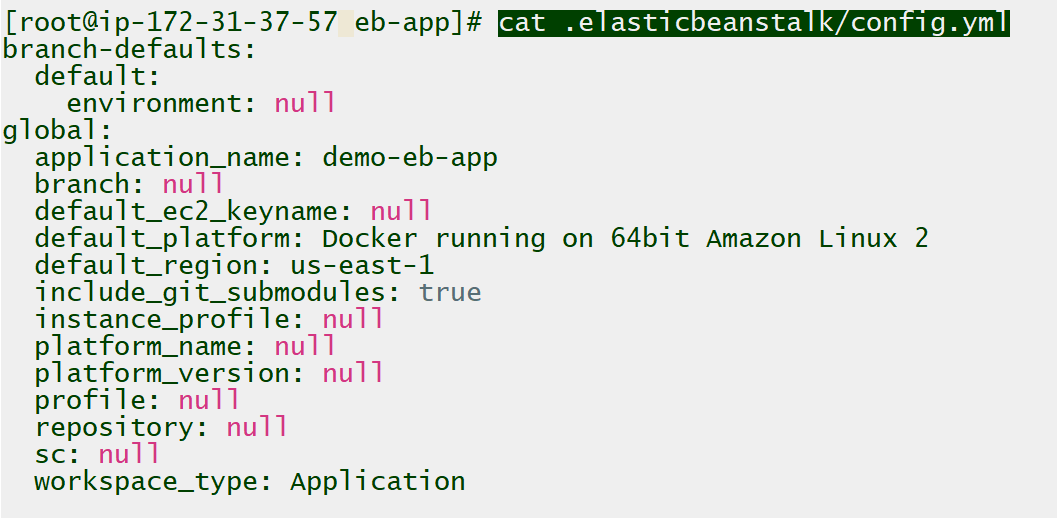
* 1. Region: 1 (North Virginia)
  2. Name: demo-eb-app
  3. Platform: Docker
  4. Platform Branch: Docker running on 64bit Amazon Linux 2
  5. SSH for instance: No



In the next step, if the prompt asks you to select an application to use, you can select [Create New Application] which is the default value. Just press Enter.



1. After the eb init command setup is completed, a new file under .elasticbeanstalk must have been created. Verify the contents of the file named config.yml using the following command:
   1. cat .elasticbeanstalk/config.yml



<https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/eb3-cmd-commands.html>

App code link

<https://www.mediafire.com/file/3839yw0qfis09wu/zeal-lab-4.zip>

create elastic beanstalk extensions

You need to create the .ebextension scripts that will configure the EC2 instances based on the requirement shared by the developers. Interpret the requirements specified in the install-system.txt file found in the Resources and configure .ebextensions accordingly.

1. Create a new directory for storing the configuration files used to set up EC2 as part of .ebextensions. You can make use of the following command to create the directory and go inside it.
   1. mkdir .ebextensions
   2. cd .ebextensions

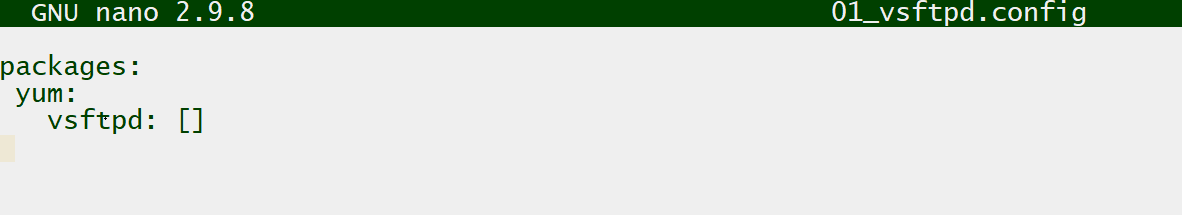


1. Create a new file for installing the vsftpd package. The name of the file for this demo is 01\_vsftpd.config. You can make use of a nano editor or any preferred choice of editor to create and edit the file.
   1. nano 01\_vsftpd.config

You can use the following code to configure the file:

* 1. packages:
  2. yum:
  3. vsftpd: []

The final configuration of the file is shown in the following screenshot.



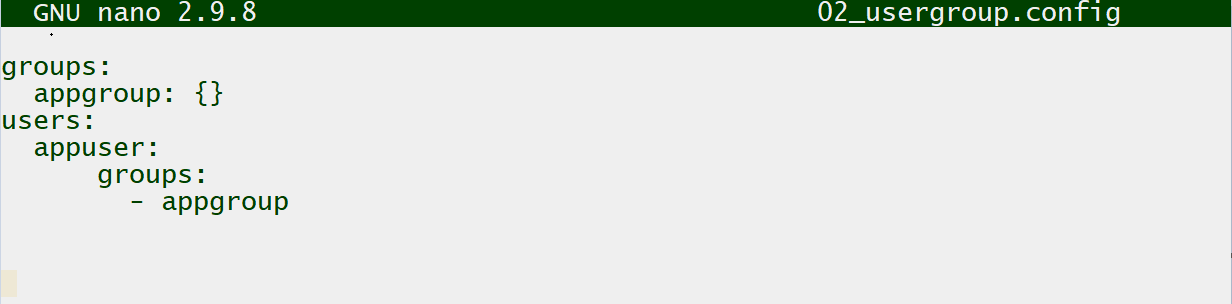
1. Create a new file for creating appropriate users and groups as per requirements. The name of the file for this demo is 02\_usergroup.config.

You can make use of a nano editor or any preferred choice of editor to create and edit the file. For this example, we have used nano editor with the following command to create the file:

* 1. nano 02\_usergroup.config

You can use the following code to configure the file:

* 1. groups:
  2. appgroup: {}
  3. users:
  4. appuser:
  5. groups:
  6. - appgroup



1. Create a new file for starting the vsftpd service. The name of the file for this demo is 03\_command.config.

You can make use of a nano editor or any preferred choice of editor to create and edit the file. For this example, we have used nano editor with the following command to create the file:

* 1. nano 03\_command.config

You can use the following code to configure the file:

* 1. commands:
  2. vsftpd\_start:
  3. command: "systemctl start vsftpd"



4. Create a new file with configuration responsible for creating a new text file in the /tmp directory. The name of the file for this demo is 04\_files.config.

You can make use of a nano editor or any preferred choice of editor to create and edit the file. For this example, we have used nano editor with the following command to create the file:

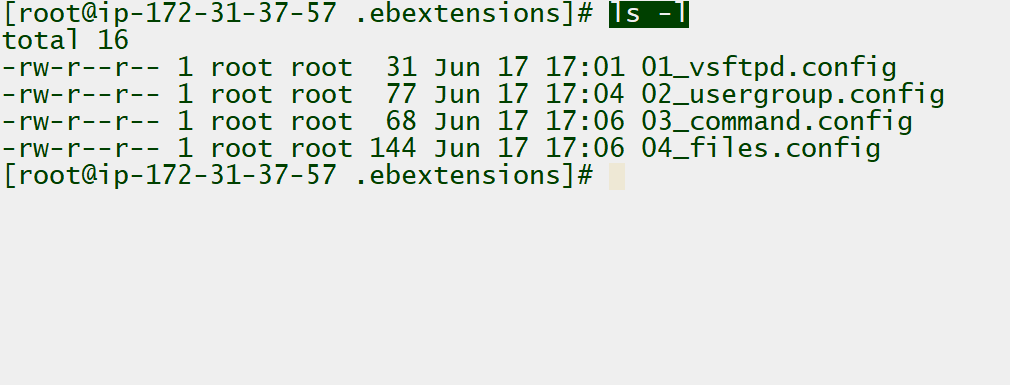
* 1. nano 04\_files.config

You can use the following code to configure the file:

* 1. files:
  2. "/tmp/deployed.txt" :
  3. mode: "000755"
  4. owner: root
  5. group: root
  6. content: |
  7. Sample File deployed .ebextensions



1. After all the files are created, verify with the ls -l command regarding the naming convention.

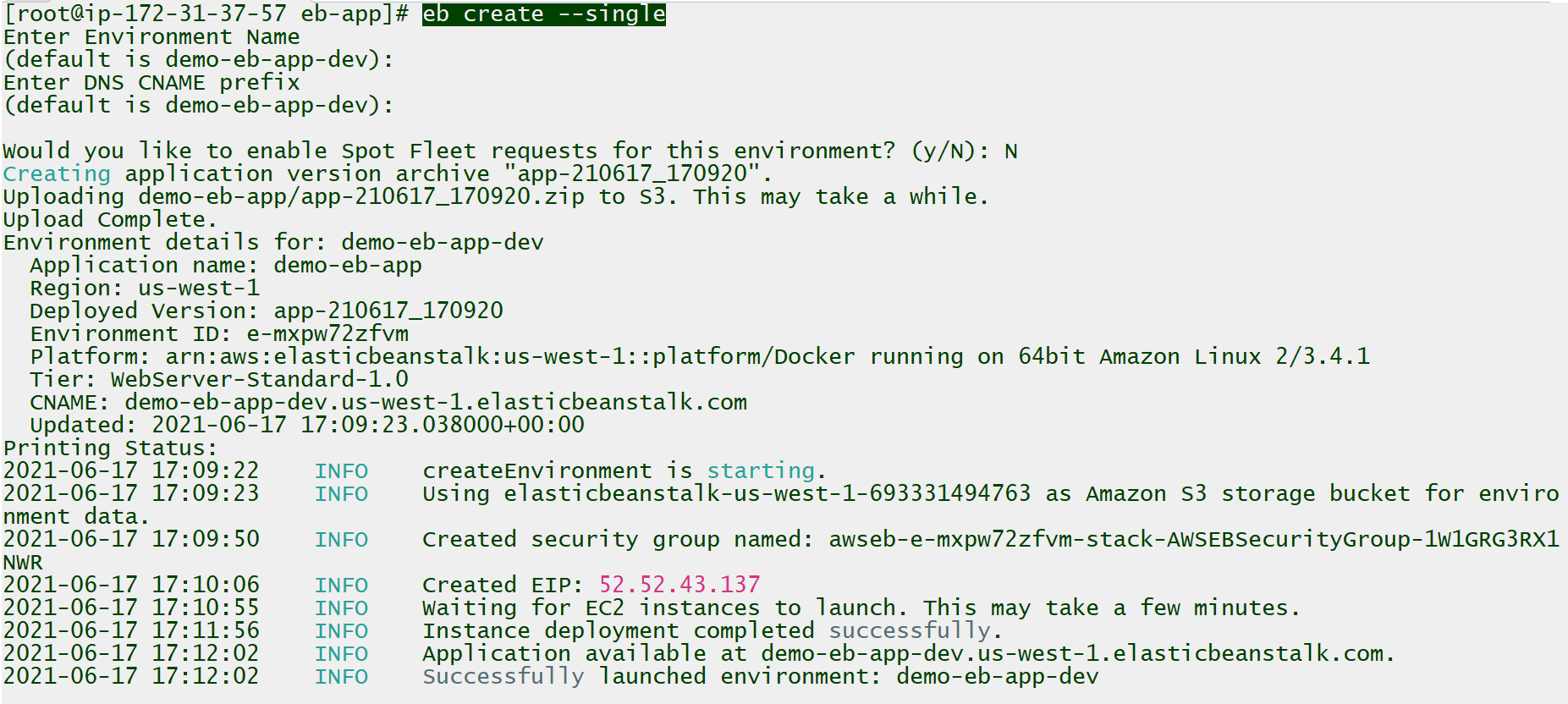


<https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/ebextensions.html>

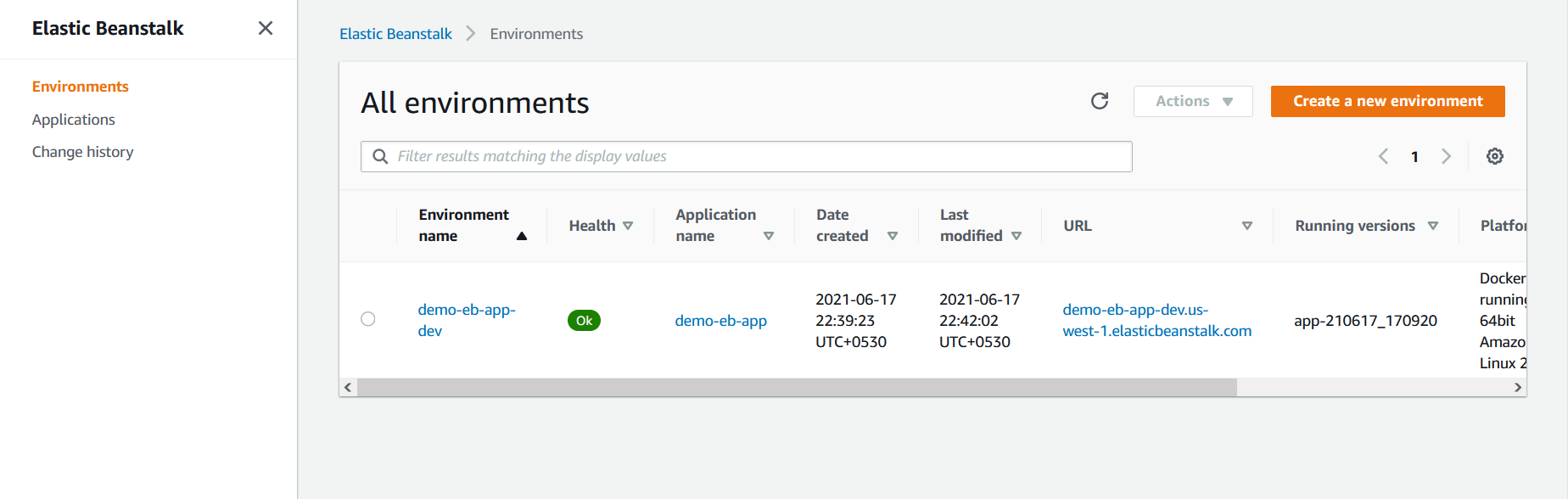
Deploying Application

After the EB CLI and Elastic Beanstalk have been configured as part of the requirement, you now can go ahead and deploy the application and create the environment.

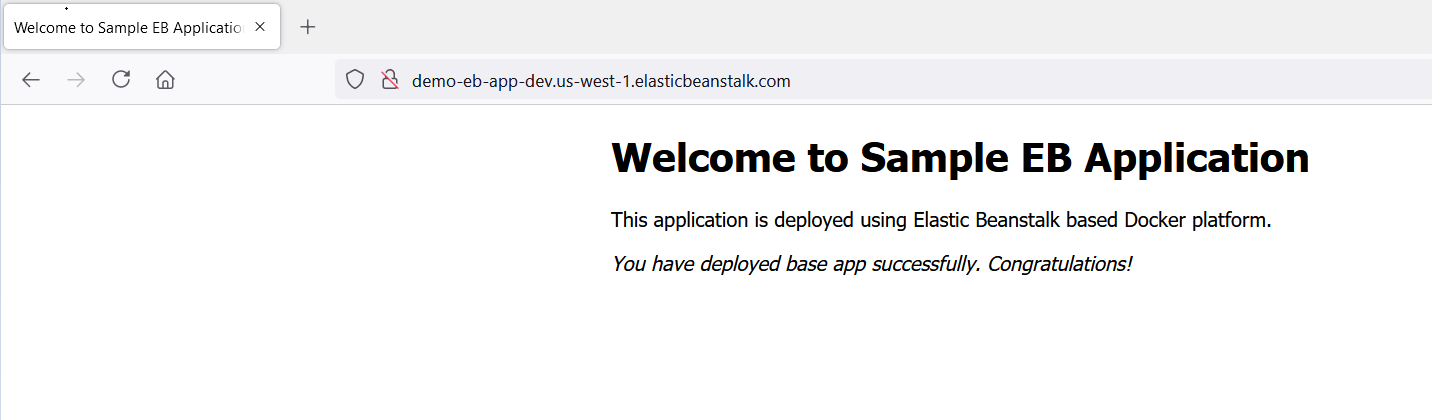
1. To create a new environment and deploying an application version to it, run the following command:
   1. eb create --single



1. After the application is created, go to the Elastic Beanstalk console to verify if the Health is Ok.



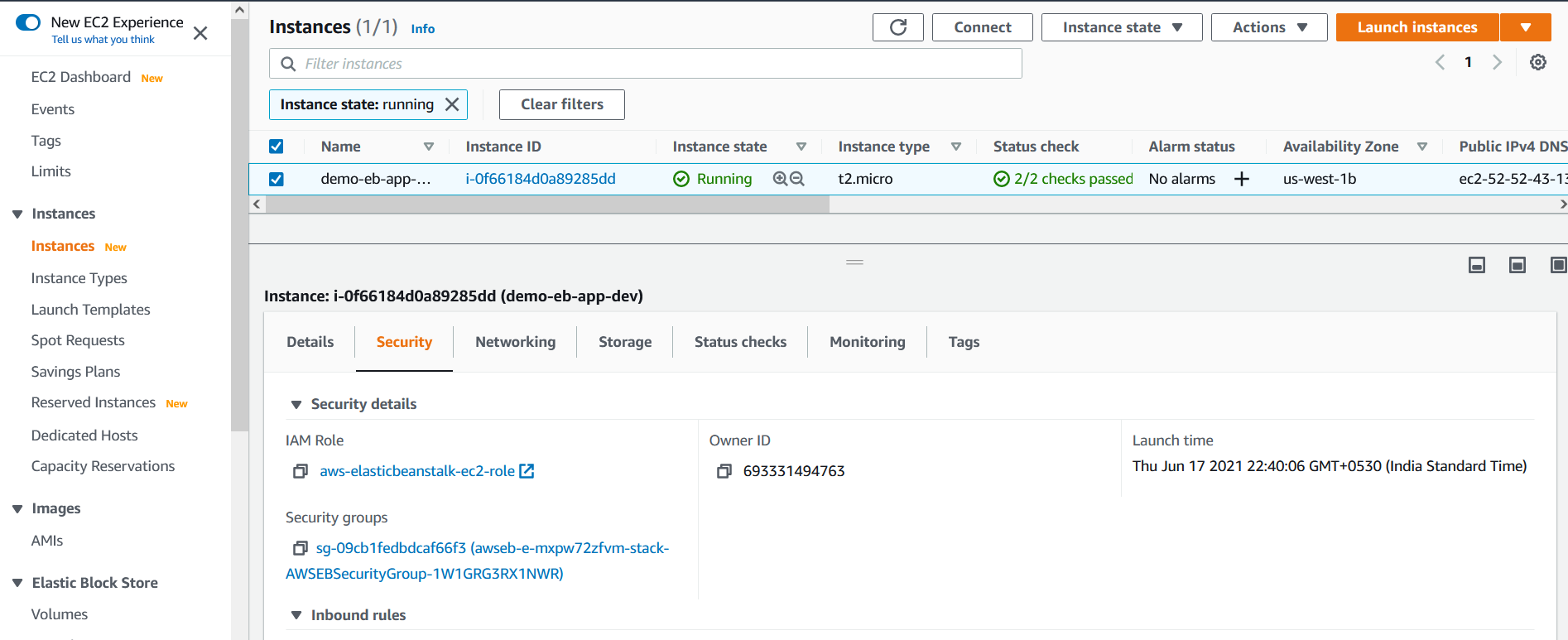
1. Click on the URL associated with the application to verify if the application is successfully deployed and is loading in the browser.



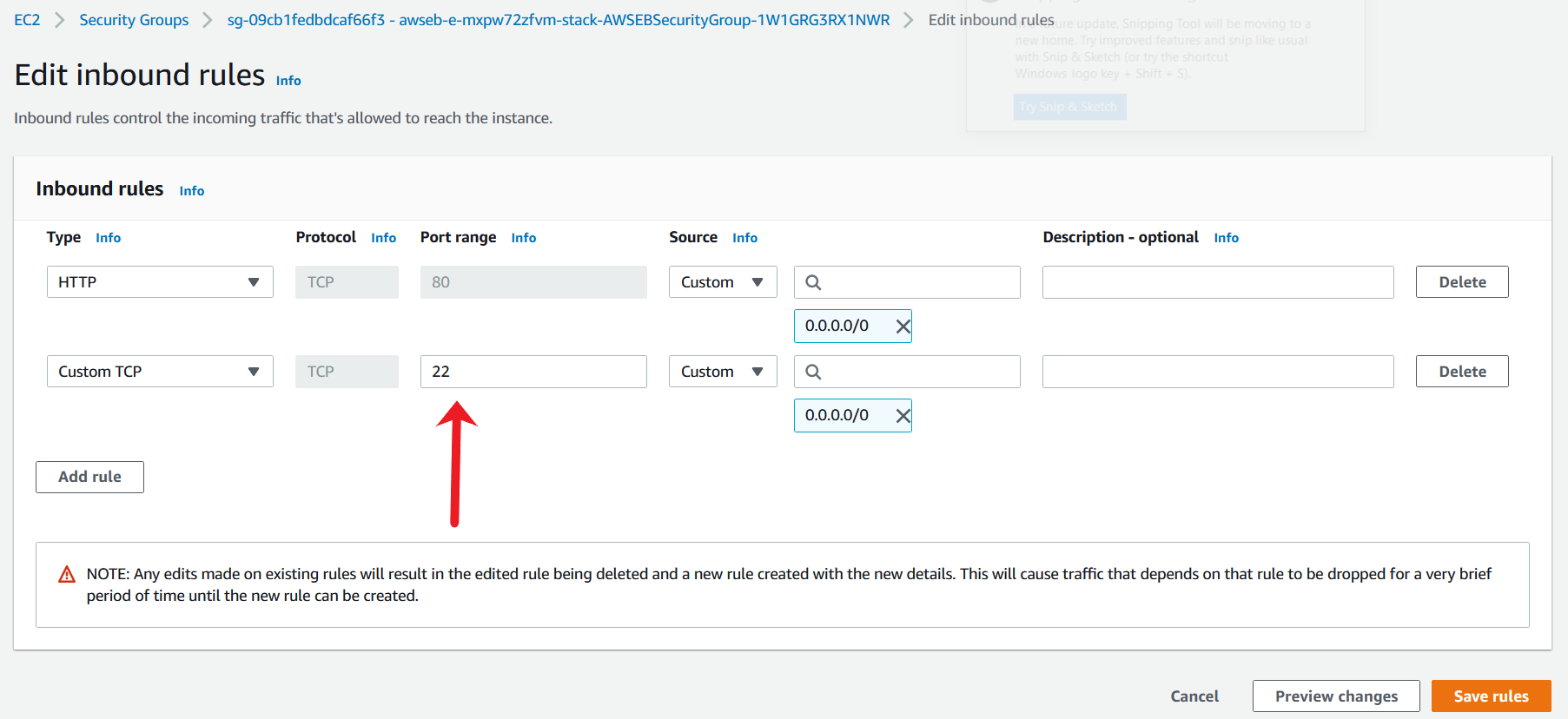
Verify the EB Extensions Deploying in EC2

Once the sample application is up and running, you need to verify if the environment customization specified using .ebextension scripts is reflecting the EC2 instance hosting the application.

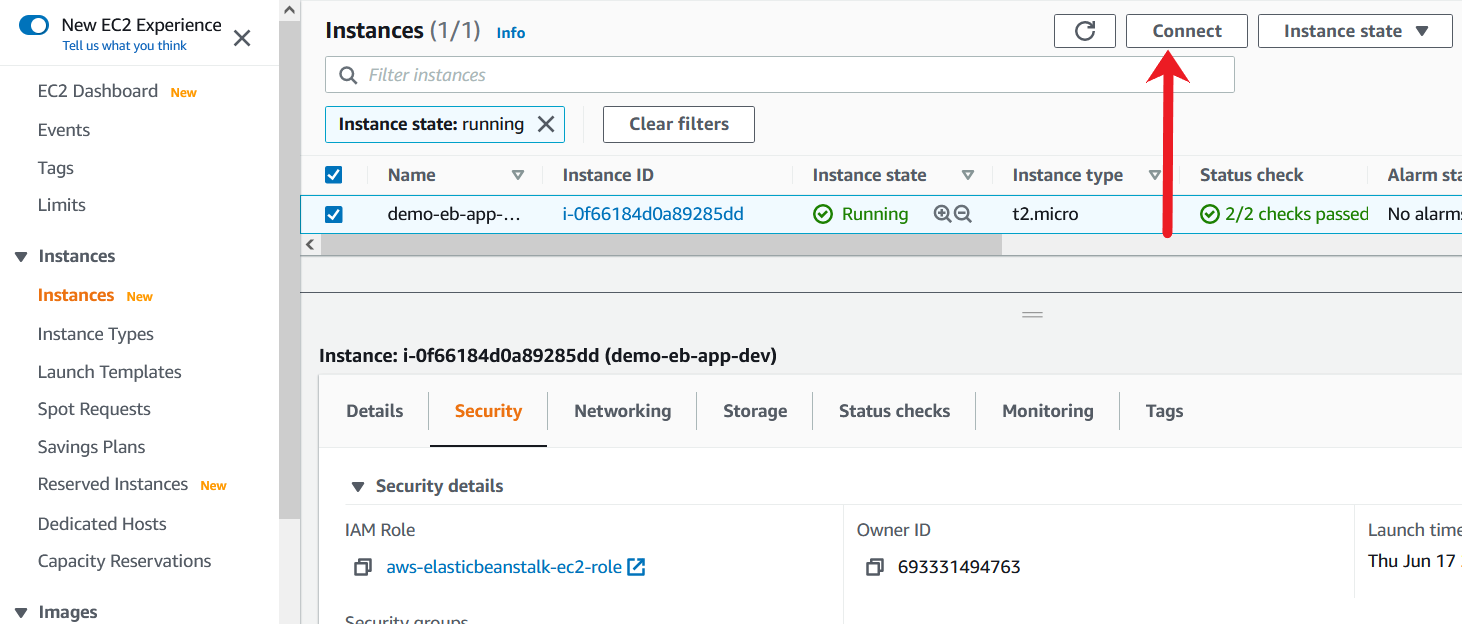
1. Go to EC2 console of the region where EB has created an environment and verify if EC2 instance is running.



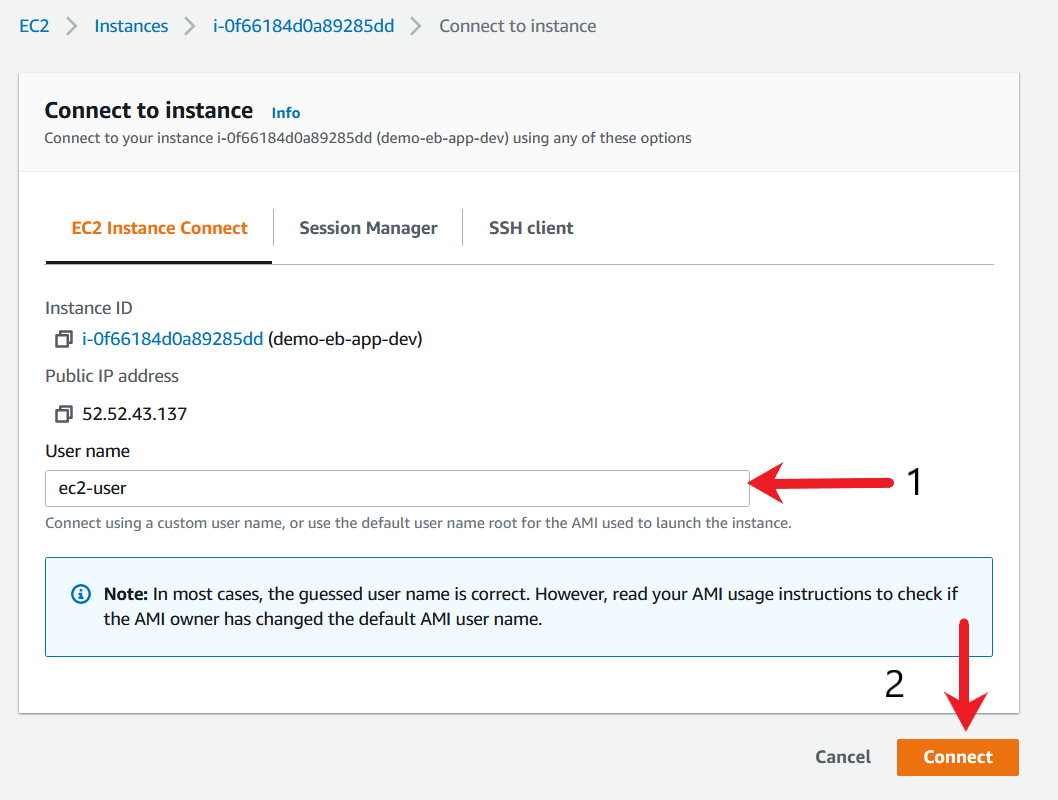
1. Modify the security group of the EC2 instance to ensure SSH Port 22 is allowed from 0.0.0.0/0 so that we can connect to it via SSH for verification.



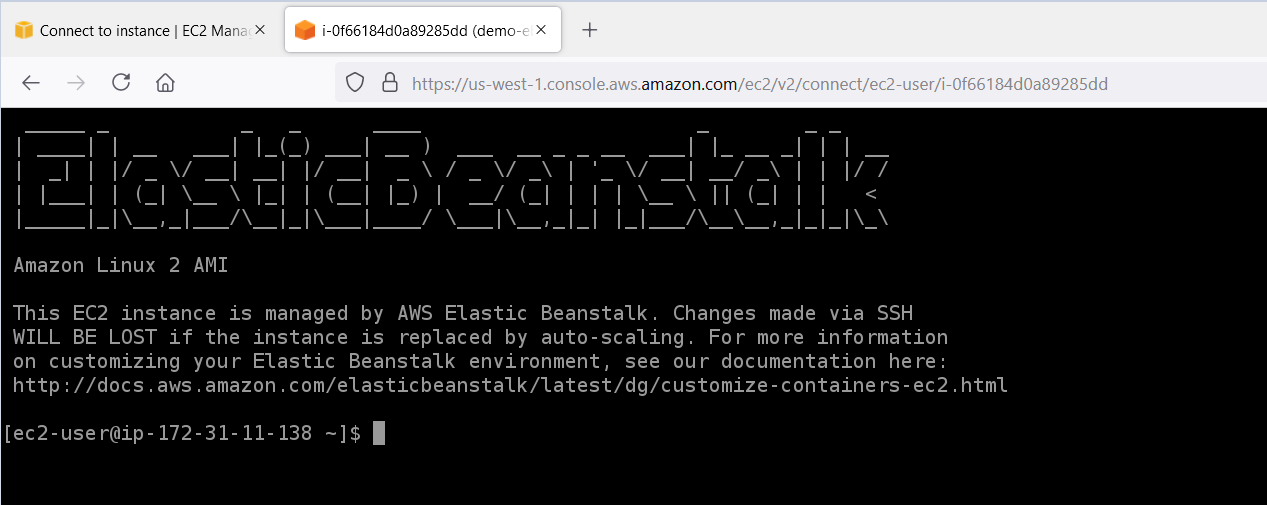
1. Connect to the EC2 instance via the EC2 connect method. Select the EC2 and click on the “Connect” button.



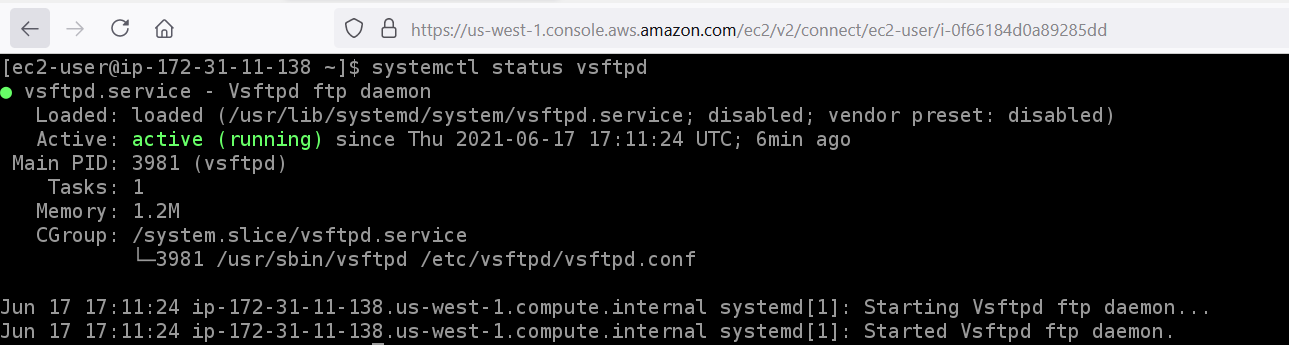
1. Specify the username as ec2-user and click on Connect.



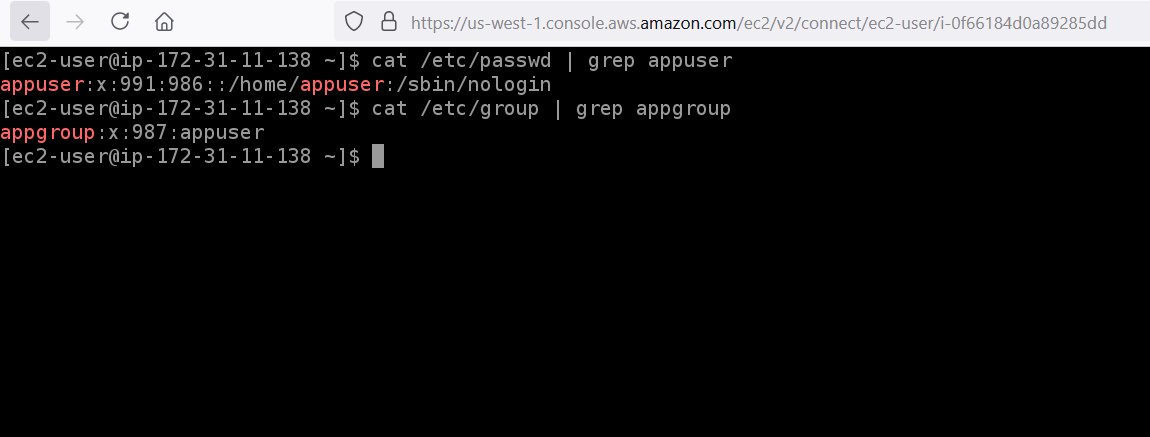
1. After a successful connection, you should see a SSH session in the browser.



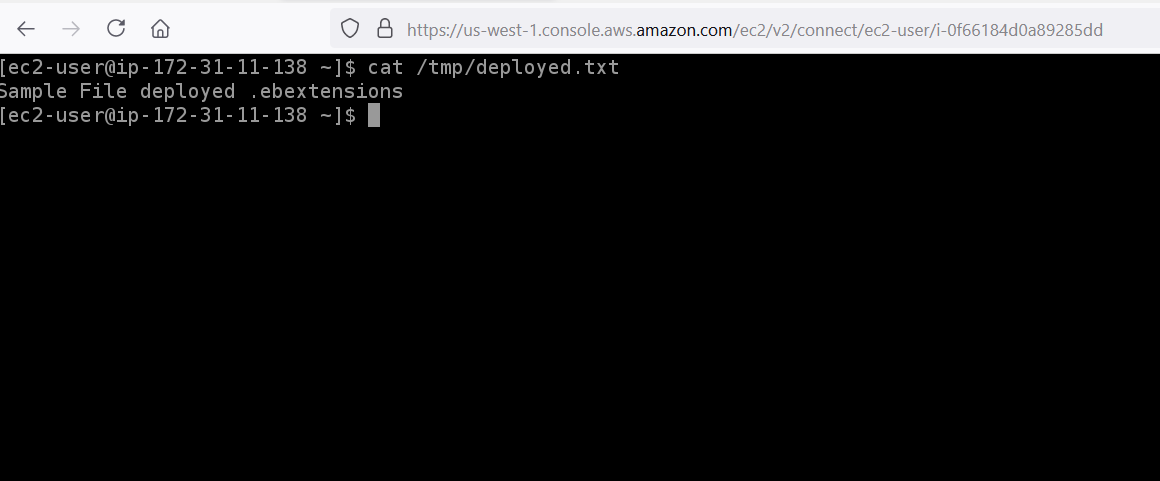
1. Verify if the vfstpd is successfully installed and running with the systemctl status vsftpd command.



1. Verify if the user and groups are successfully created, with the following command:
   1. cat /etc/passwd | grep appuser
   2. cat /etc/group | grep appgroup



1. Verify if the file is created at server using the command
   1. cat /tmp/deployed.txt

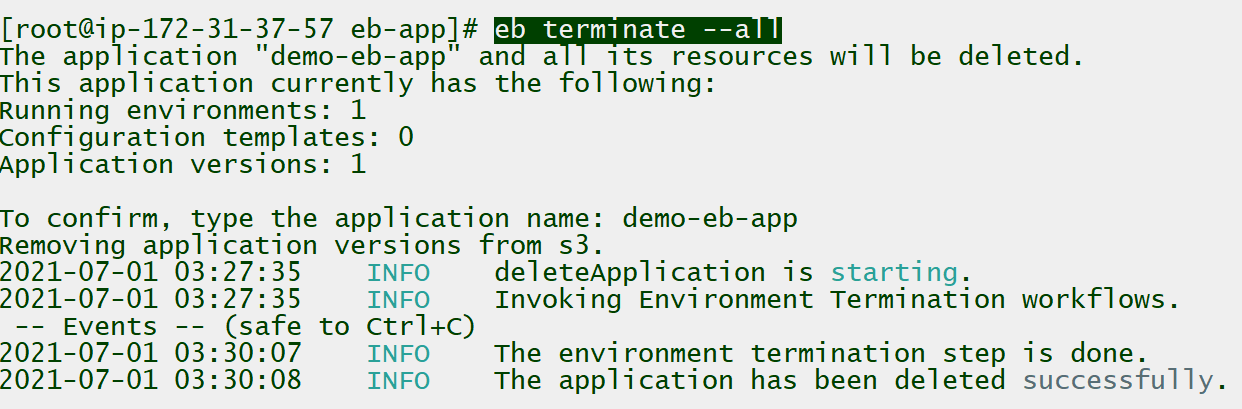


<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Connect-using-EC2-Instance-Connect.html>

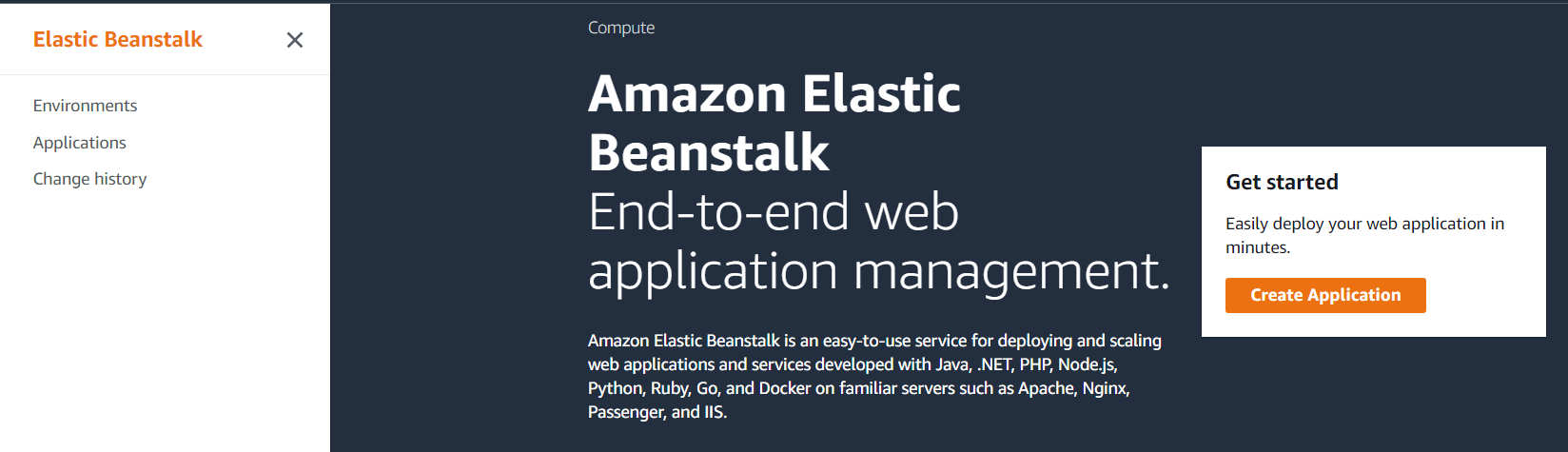
Clean Up

As we have successfully demonstrated a proof of concept for a continuous delivery pipeline, we'll clean up our AWS resources at this time. Delete all the resources created for this proof of concept.

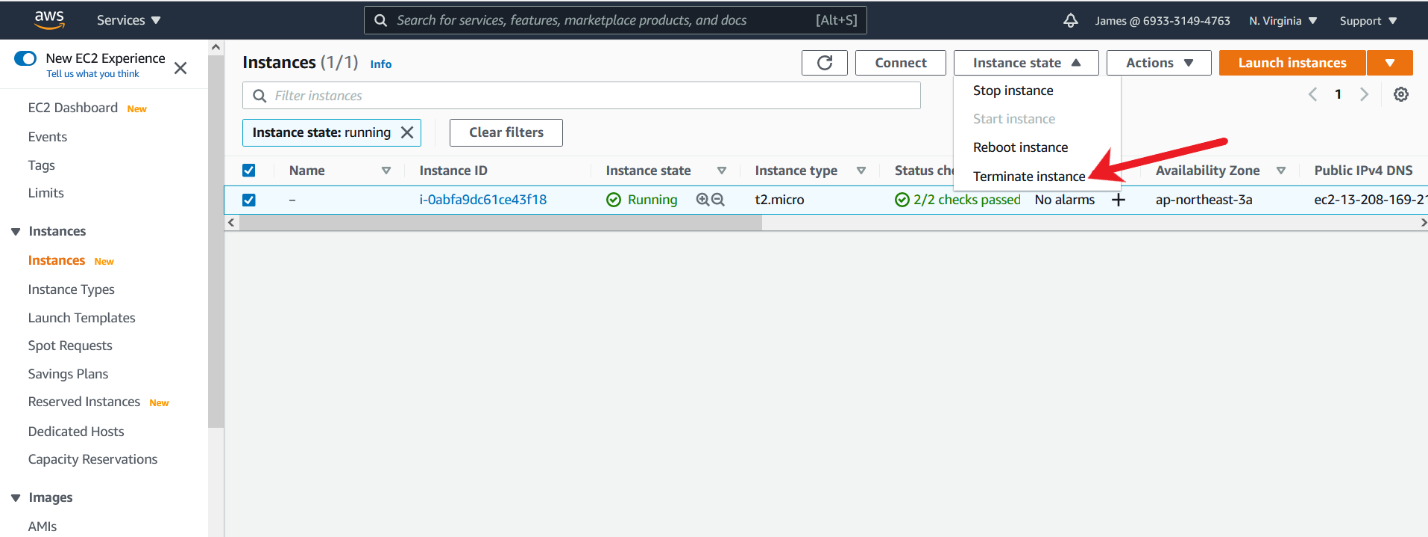
1. Terminates the running environment so that you do not incur charges for unused AWS resources with the eb terminate --all command.



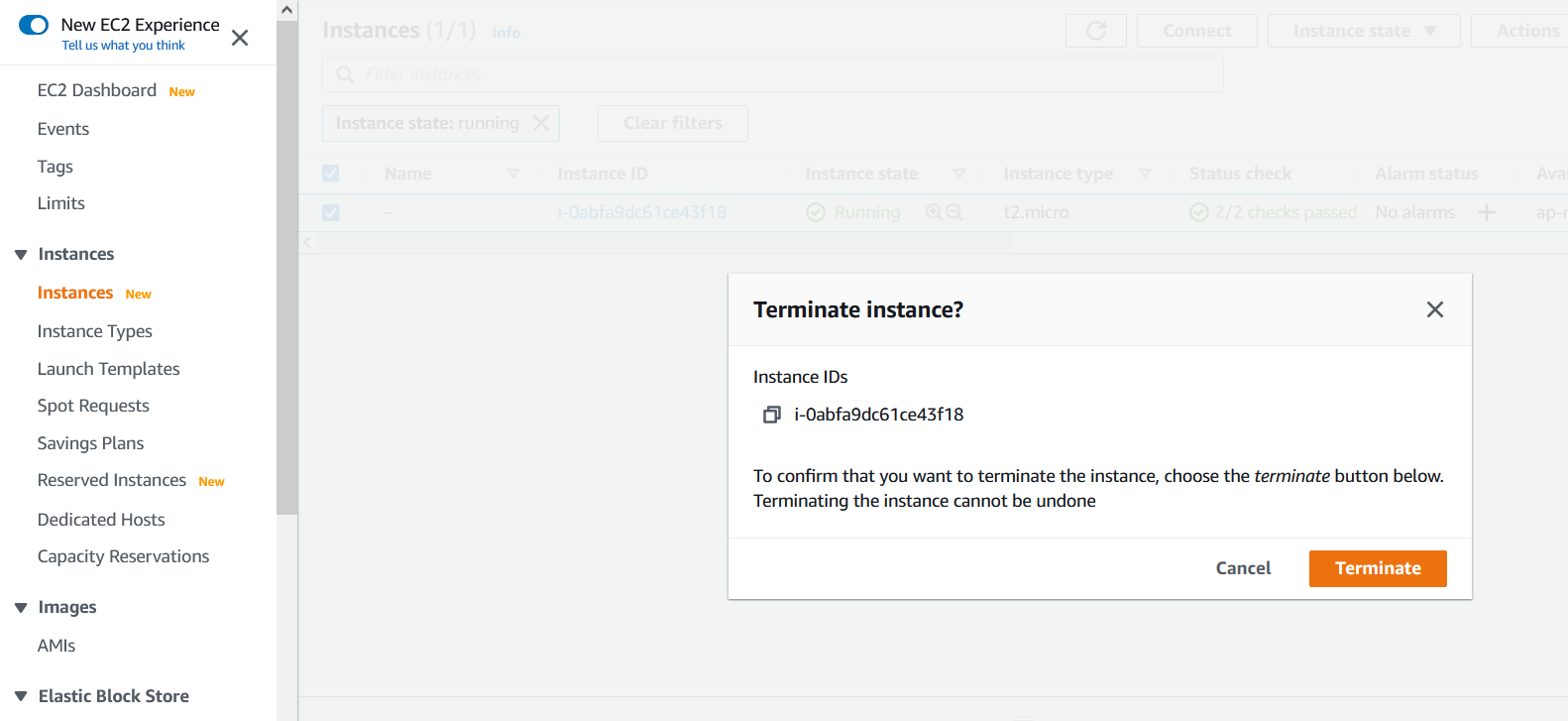
1. Go to the Elastic Beanstalk console and you should see the default page. The application and the environment are deleted.



1. Go to the EC2 console to terminate the EC2 instance which was used to deploy Elastic Beanstalk application. Select the instance, go to Action and click on Terminate.



On confirmation, click Terminate.



1. Open the Amazon EC2 console. In the navigation pane, choose Key Pairs. Select the key pair to delete and click on Action and select Delete. In the confirmation field, enter delete and then choose Delete.

