

Seneca

Lab 1: Setting up the environment

-Pratham Sapra

The goal of this Lab is to create a functional environment that we will use throughout this course.

We will cover the following:

1. Access to AWS Management Console via AWS Academy
2. Explore AWS Credentials and our permissions in AWS Academy environment
3. Creating Amazon Cloud9 IDE
4. Provisioning Amazon EC2 with AWS CLI
5. Securely connect to Amazon EC2 with SSH and AWS Academy provided SSH key pair
6. Optional: Configuring your local IDE to use temporary AWS Academy credentials

Task 1: Accessing AWS Management Console via AWS Academy

AWS Academy Lab environment is provisioned for you for the duration of the Deployment Automaton and Operational Security course and will be available until April 30th.

Please note, that the environment is restricted to about 40 out of more than 100 AWS services and your credit limit for the whole term is 100USD.

This means that at the end of your work, you either must clean up all the provisioned infrastructure manually or to press **“Reset”** button which will destroy all the provisioned resources.

Important: The 100USD credit is about 5 times the amount needed to complete the course, and it cannot be refilled. Please keep a close attention to your remaining credits.

To work with the AWS environment provisioned by AWS Academy, please follow the instructions below:

1. Select Modules from the course menu.

aws academy

Account

Dashboard

Courses

Calendar

Inbox

History

Help

ALLv1EN-LTI13-68759


Home

Modules

Discussions

Grades

AWS Academy Learner Lab [68759]



AWS Academy Learner Lab provides a long-running sandbox environment for ad hoc exploration of AWS services. Within this class, students will have access to **a restricted set of AWS services**. Not all AWS documentation walk-through or sample labs that operate in an AWS Production account will work in the Learner Lab environment. You will retain access to the AWS resources set up in this environment for the duration of this course. We limit your budget (\$100USD), so you should exercise caution to prevent charges that will deplete your budget too quickly. If you exceed your budget, you will lose access to your environment and lose all of your work.

Each session lasts for 4 hours by default, although you can extend a session to run longer by pressing the start button to reset your session timer. At the end of each session, any resources you created will persist. However, we automatically shut EC2 instances down. Other resources, such as RDS instances, keep running. Keep in mind that we

View Course Stream

View Course Calendar

View Course Notifications

To Do

Nothing for now

Recent Feedback

Nothing for now

2. Click through the Learner Lab link

aws academy

Account

Dashboard

Courses

Calendar

Inbox

History

Help

Home

Modules

Discussions

Grades

Learn how to effectively use the AWS Academy Learner Lab

Module Knowledge Check
100 pts | Score at least 70.0

AWS Academy Learner Lab

Launch AWS Academy Learner Lab

AWS Academy Learner Lab Resources

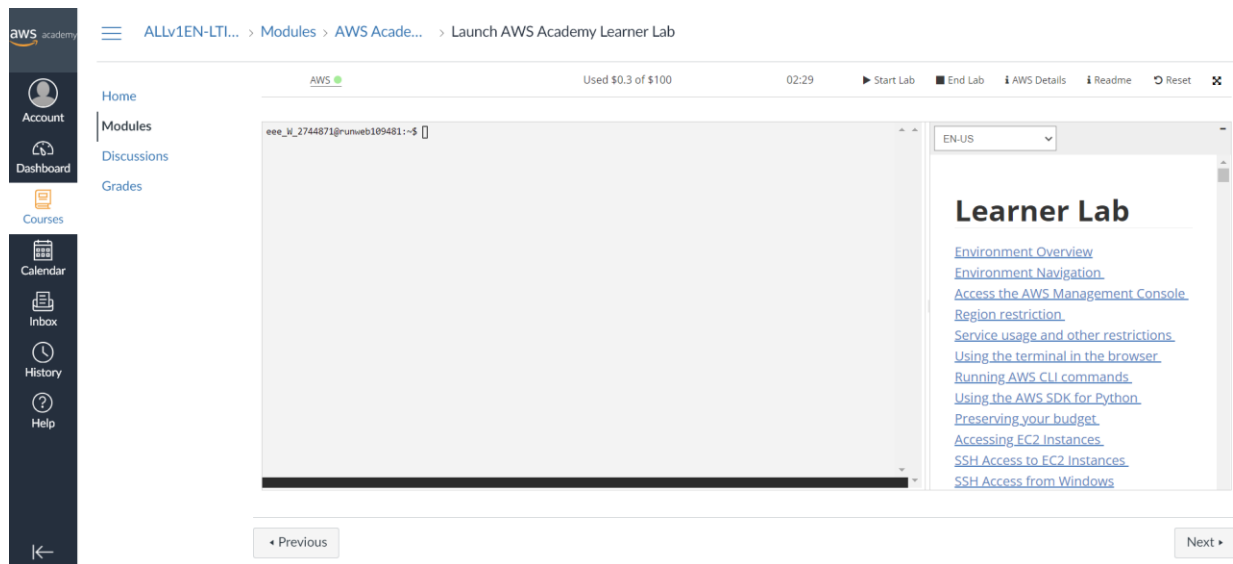
Demo - How to Access Learner Lab

Demo - General Troubleshooting Tips

Demo - How to Launch Services through AWS Console

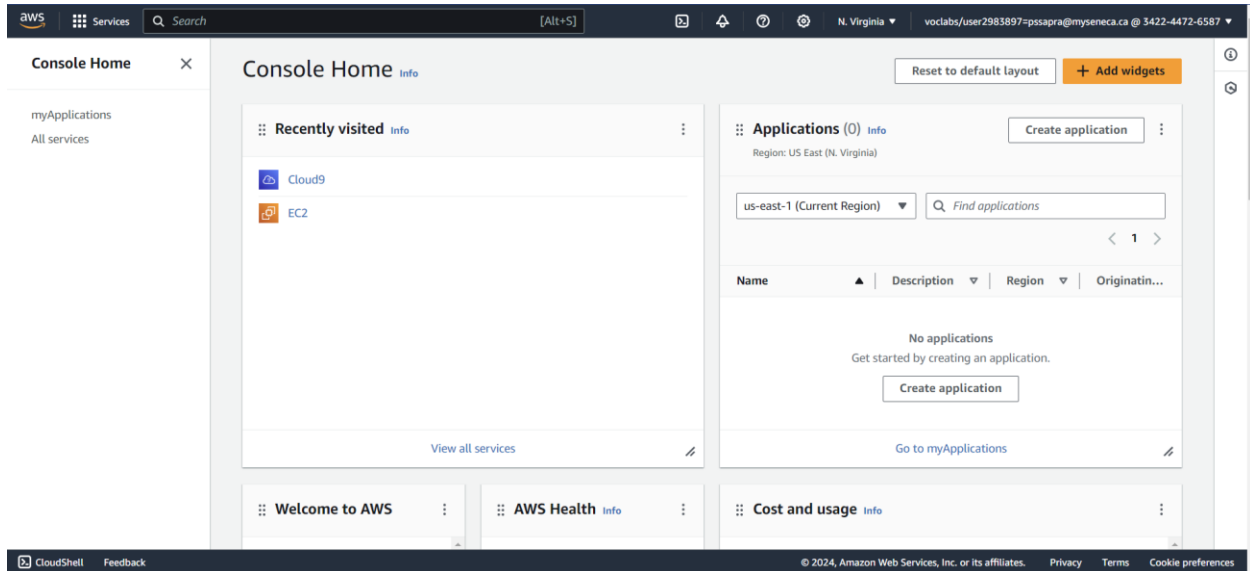
Learner Lab Activity - CodeWhisperer

3. Note: If this is your first time you may have to refresh the web page to see and accept the terms and conditions. Start the lab by pressing the “Start Lab” button.

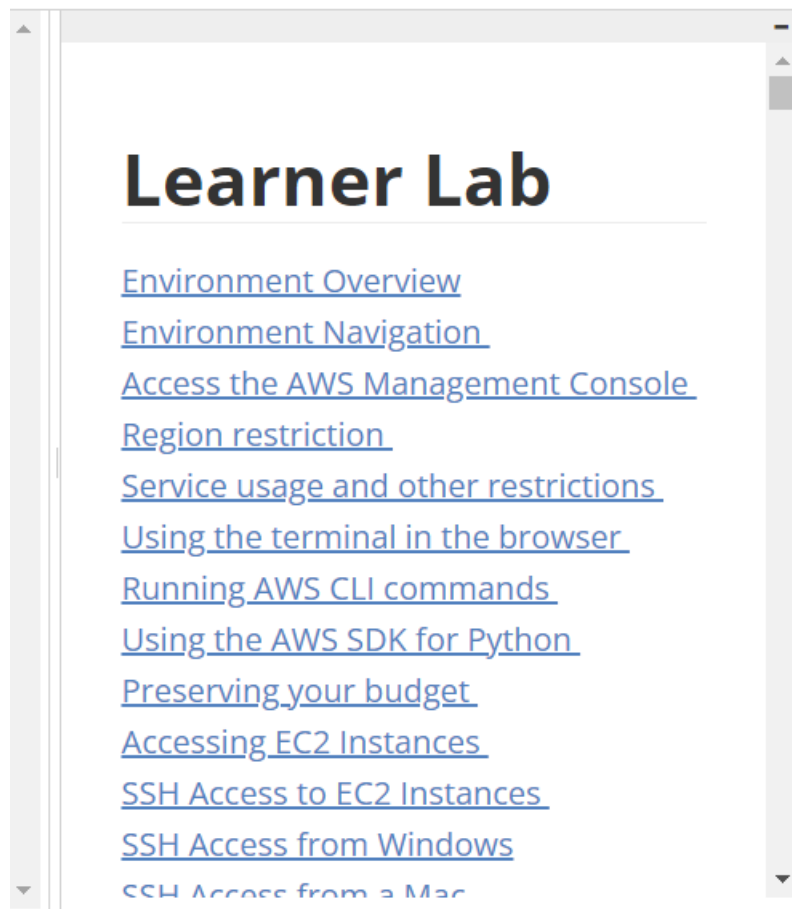



4. Once the lab is started, press the AWS button to view the AWS Management Console.


5. You will be transparently logged into a temporary AWS Account that you can use as long as the lab session timer is active (4 hours).

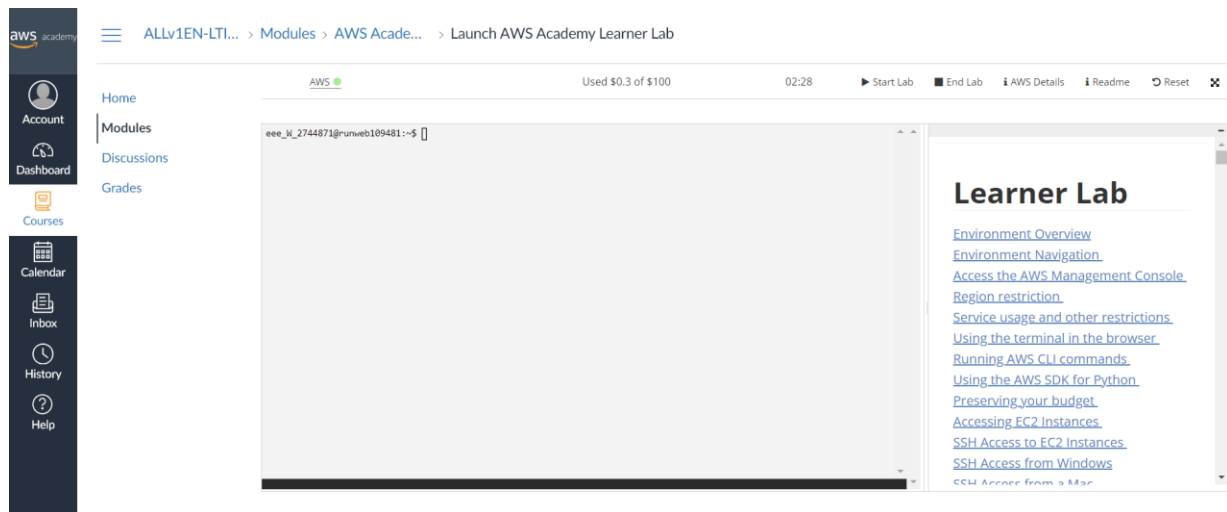


6. You can return to the LMS to see the lab instructions by using the Readme button.



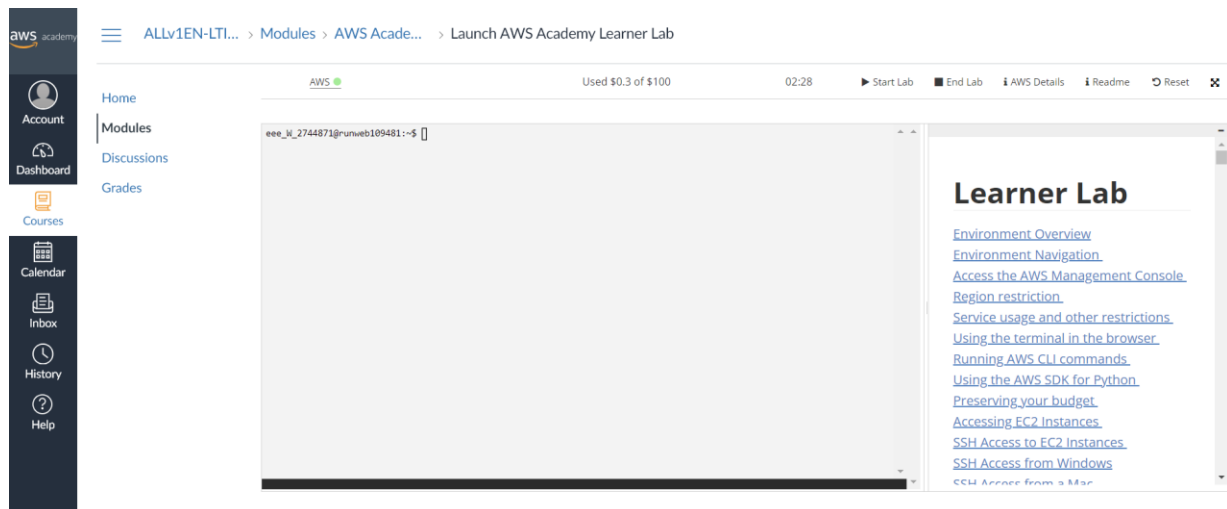
7. You can monitor your spending using the information at the top of the lab.
Note: This information is provided by AWS Budgets and may be delayed by up to 8 hours. This is an approximate view of your spending.

8. You can monitor your remaining session time at the top of the lab.
Note: If you are actively working and you need more time, you can reset your session timer by pressing “Start Lab” again.



9. If you need to wipe out all of the resources that you have configured and start with a fresh AWS Account, press the Reset Button. 

NOTE: ALL WORK WILL BE LOST AND CANNOT BE RECOVERED.



10. When you are finished with this session, press the “End Lab” button. This will stop any running EC2 instances. When you return to your lab and restart it – the

EC2 instance will restart and any other resources you've configured will still be available.

The screenshot shows the AWS Academy Learner Lab interface. On the left is a dark blue sidebar with icons for Account, Dashboard, Courses, Calendar, Inbox, History, and Help. The main content area has a breadcrumb trail: ALLv1EN-LTI... > Modules > AWS Acade... > Launch AWS Academy Learner Lab. Below the breadcrumb is a navigation menu with Home, Modules, Discussions, and Grades. The main workspace is divided into two panes. The left pane is a terminal window with the prompt 'eee_wi_2744871@runweb109481:~\$'. The right pane is titled 'Learner Lab' and contains a list of links: Environment Overview, Environment Navigation, Access the AWS Management Console, Region restriction, Service usage and other restrictions, Using the terminal in the browser, Running AWS CLI commands, Using the AWS SDK for Python, Preserving your budget, Accessing EC2 Instances, SSH Access to EC2 Instances, SSH Access from Windows, and SSH Access from a Mac. At the top of the main workspace, there is a status bar showing 'AWS' with a green dot, 'Used \$0.3 of \$100', '02:28', and buttons for Start Lab, End Lab, AWS Details, Readme, Reset, and a close button.

Task 2: Explore AWS Credentials and our permissions in AWS Academy environment

To work with AWS APIs we need to have be authenticated and authorized. When working with AWS CLI, SDK and other deployment tools, our users need to have programmatic level access and our credentials should be available in our working environment.

[Read more](#) about different ways of setting AWS credentials in your environment.

1. From your AWS Academy environment, go to AWS Academy terminal window through steps 1 to 3 in Task 1. In the terminal window, type:
`vi ~/.aws/credentials`
`vi ~/.aws/config`



1

¹ Temporary credentials provisioned by AWS Academy for the duration of the Lab session

Note: The credentials below are temporary and are invalidated when I click “End Lab”. Please do not try using these credentials.

2. Read README file on the left side panel. README clarifies limitations of the AWS Academy environment and provides the basic set of instructions.

3. Try running AWS CLI command to see if you have permissions
`aws sts get-caller-identity`

*

ALLv1EN-LTI... > Modules > AWS Acade... > Launch AWS Academy Learner Lab

[Home](#)
[Modules](#)
[Discussions](#)
[Grades](#)

AWS ● Used \$0.3 of \$100 03:38 ▶ Start Lab

```
eee_w_2744871@runweb109481:~$ vi ~/.aws/credentials
eee_w_2744871@runweb109481:~$ vi ~/.aws/config
eee_w_2744871@runweb109481:~$ aws sts get-caller-identity
{
  "UserId": "AR0AU7L3AC455NMW7BMVZ:user2983897=pssapra@myseneca.ca",
  "Account": "342244726587",
  "Arn": "arn:aws:sts::342244726587:assumed-role/voclabs/user2983897=pssapra@myseneca.ca"
}
eee_w_2744871@runweb109481:~$ █
```

4. Try running the command below. What does the error tell us?
`aws iam create-user --user-name irina`

Home
Modules
Discussions
Grades

AWS Used \$0.3 of \$100 03:36 Start Lab

```
eee_W_2744871@runweb109481:~$ vi ~/.aws/credentials
eee_W_2744871@runweb109481:~$ vi ~/.aws/config
eee_W_2744871@runweb109481:~$ aws sts get-caller-identity
{
  "UserId": "AROAU7L3AC455NMW7BMVZ:user2983897=pssapra@myseneca.ca",
  "Account": "342244726587",
  "Arn": "arn:aws:sts::342244726587:assumed-role/voclabs/user2983897=pssapra@myseneca.ca"
}
eee_W_2744871@runweb109481:~$ aws iam create-user --user-name irina
An error occurred (AccessDenied) when calling the CreateUser operation: User: arn:aws:sts::342244726587:assumed-role/voclabs/user2983897=pssapra@myseneca.ca is not authorized to perform: iam:CreateUser on resource: arn:aws:iam::342244726587:user/irina because no identity-based policy allows the iam:CreateUser action
eee_W_2744871@runweb109481:~$
```

Other resources blocked by AWS Academy include EMR and EKS clusters, ECS and everything related to IAM.

5. AWS Academy pre-created an SSH key pair in our AWS environment. The private key of the key pair is located in you AWS Academy environment in the location below:

```
vi ~/.ssh/labuser.pem
```

Home
Modules
Discussions
Grades

AWS Used \$0.3 of \$100 03:36 Start Lab

```
eee_W_2744871@runweb109481:~$ vi ~/.aws/credentials
eee_W_2744871@runweb109481:~$ vi ~/.aws/config
eee_W_2744871@runweb109481:~$ aws sts get-caller-identity
{
  "UserId": "AROAU7L3AC455NMW7BMVZ:user2983897=pssapra@myseneca.ca",
  "Account": "342244726587",
  "Arn": "arn:aws:sts::342244726587:assumed-role/voclabs/user2983897=pssapra@myseneca.ca"
}
eee_W_2744871@runweb109481:~$ aws iam create-user --user-name irina
An error occurred (AccessDenied) when calling the CreateUser operation: User: arn:aws:sts::342244726587:assumed-role/voclabs/user2983897=pssapra@myseneca.ca is not authorized to perform: iam:CreateUser on resource: arn:aws:iam::342244726587:user/irina because no identity-based policy allows the iam:CreateUser action
eee_W_2744871@runweb109481:~$ vi ~/.ssh/labuser.pem
eee_W_2744871@runweb109481:~$
```

6. To see the SSH keypair provisioned in our environment, run the command below, Note the name of the key. We will use it to provision EC2 instance.

```
aws ec2 describe-key-pairs
```

Home
Modules
Discussions
Grades

AWS
Used \$0.3 of \$100
03:35
Start Lab

```

"UserId": "AROAU7L3AC455NMW7BMVZ:user2983897=pssapra@myseneca.ca",
"Account": "342244726587",
"Arn": "arn:aws:sts::342244726587:assumed-role/voclabs/user2983897=pssapra@myseneca.ca"
}
eee_W_2744871@runweb109481:~$ aws iam create-user --user-name irina

An error occurred (AccessDenied) when calling the CreateUser operation: User: arn:aws:sts::342244726587:assumed-role/voclabs/user2983897=pssapra@myseneca.ca is not authorized to perform: iam:CreateUser on resource: arn:aws:iam::342244726587:user/irina because no identity-based policy allows the iam:CreateUser action
eee_W_2744871@runweb109481:~$ vi ~/.ssh/labuser.pem
eee_W_2744871@runweb109481:~$ aws ec2 describe-key-pairs
{
  "KeyPairs": [
    {
      "KeyPairId": "key-06333426a14701761",
      "KeyFingerprint": "60:4c:b9:3b:da:ec:87:69:12:95:c7:23:ee:3f:b2:67:47:66:d4:30",
      "KeyName": "vockey",
      "Tags": []
    },
    {
      "KeyPairId": "key-0ac4e8da1776dd8d7",
      "KeyFingerprint": "63:de:15:a3:b0:1e:7f:13:06:04:13:d8:19:86:9d:b0:67:5e:86:da",
      "KeyName": "MYEC2SSHPS",
      "Tags": []
    }
  ]
}
eee_W_2744871@runweb109481:~$

```

Task 3: Creating Amazon Cloud9 IDE

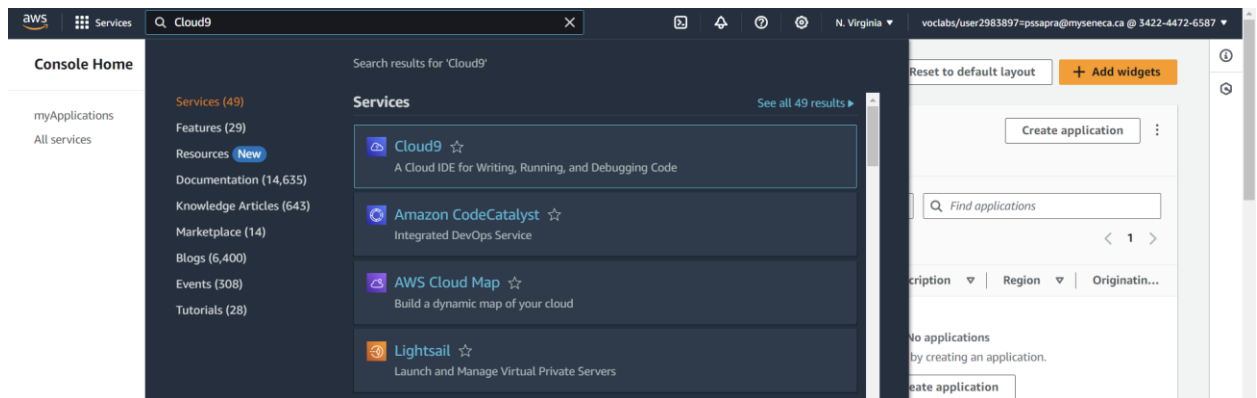
We will use Amazon Cloud9 as our IDE throughout the course. The Cloud9 environment is pre-provisioned with AWS CLI, AWS credentials, Terraform CLI, python, git and other tools that we will use throughout the course.

Please note that you can use other IDE, e.g. Visual Source code, but you will have to renew your credentials manually every few hours. We will discuss the local environment setup in Task 6.

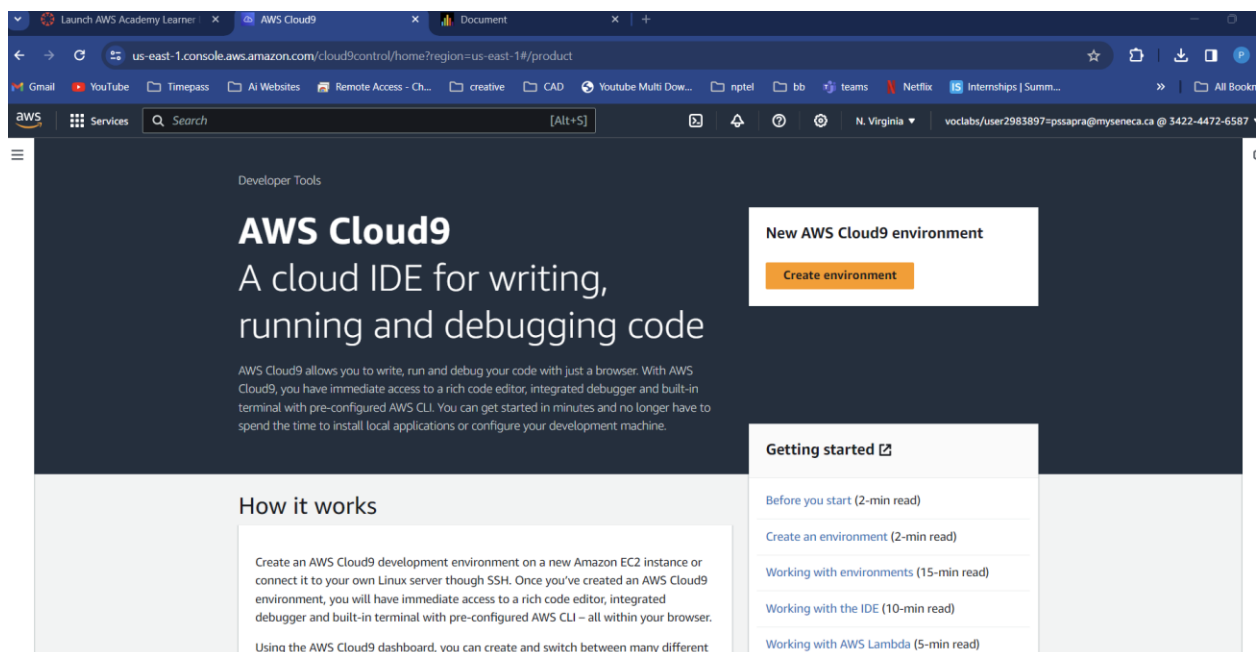
1. Open AWS Management Console by clicking on AWS with a green dot

```
eee_w_2744871@runweb109481:~$
```

2. Type Cloud9 in the search bar and click on Cloud9



3. Click on Create environment button.



4. Name the environment and click on Next step

us-east-1.console.aws.amazon.com/cloud9control/home?region=us-east-1#/create/

Services Search [Alt+S]

N. Virginia voclabs/user2983897=pssapra@myseneca.ca @ 3422-4472-6587

AWS Cloud9 > Environments > Create environment

Create environment [Info](#)

Details

Name

acs730-week1-irina

Limit of 60 characters, alphanumeric and unique per user.

Description - optional

Limit 200 characters.

Environment type [Info](#)

Determines what the Cloud9 IDE will run on.

☒ **New EC2 instance**

Cloud9 creates an EC2 instance in your account. The configuration of your EC2 instance cannot be changed by Cloud9 after creation.

☐ **Existing compute**

You have an existing instance or server that you'd like to use.

New EC2 instance

5. Leave all the defaults and click on the Next step again, then click Create environment

Services Search [Alt+S]

N. Virginia voclabs/user2983897=pssapra@myseneca.ca @ 3422-4472-6587

New EC2 instance

Instance type [Info](#)

The memory and CPU of the EC2 instance that will be created for Cloud9 to run on.

☒ **t2.micro (1 GiB RAM + 1 vCPU)**

Free-tier eligible. Ideal for educational users and exploration.

☐ **t3.small (2 GiB RAM + 2 vCPU)**

Recommended for small web projects.

☐ **m5.large (8 GiB RAM + 2 vCPU)**

Recommended for production and most general-purpose development.

☐ **Additional instance types**

Explore additional instances to fit your needs.

Platform [Info](#)

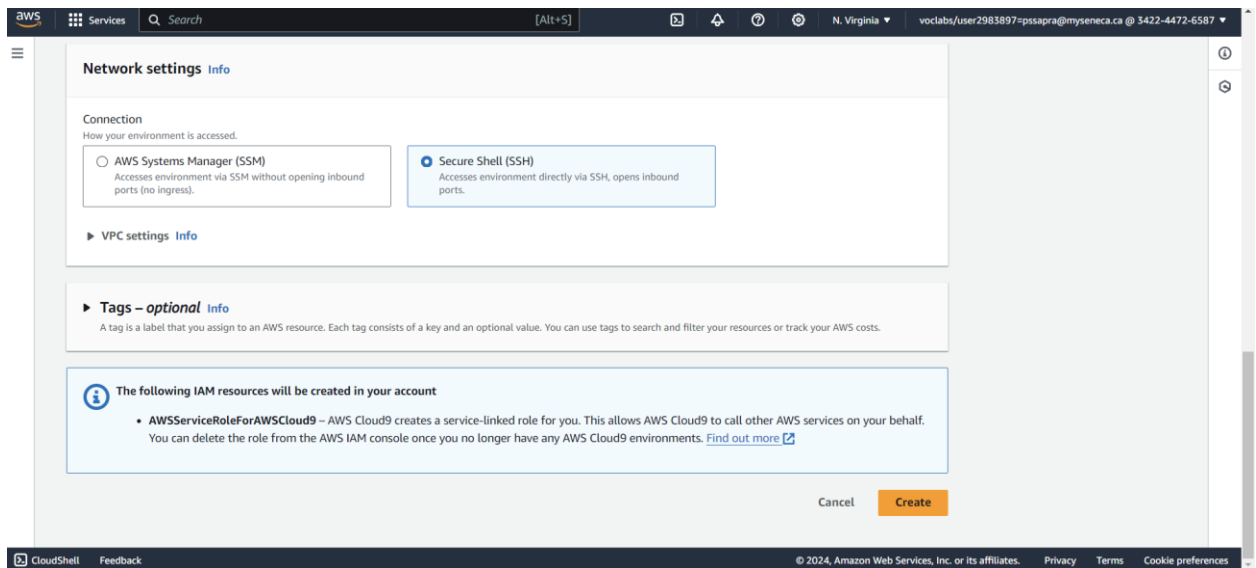
This will be installed on your EC2 instance. We recommend Amazon Linux 2023.

Amazon Linux 2023

Timeout

How long Cloud9 can be inactive (no user input) before auto-hibernating. This helps prevent unnecessary charges.

30 minutes

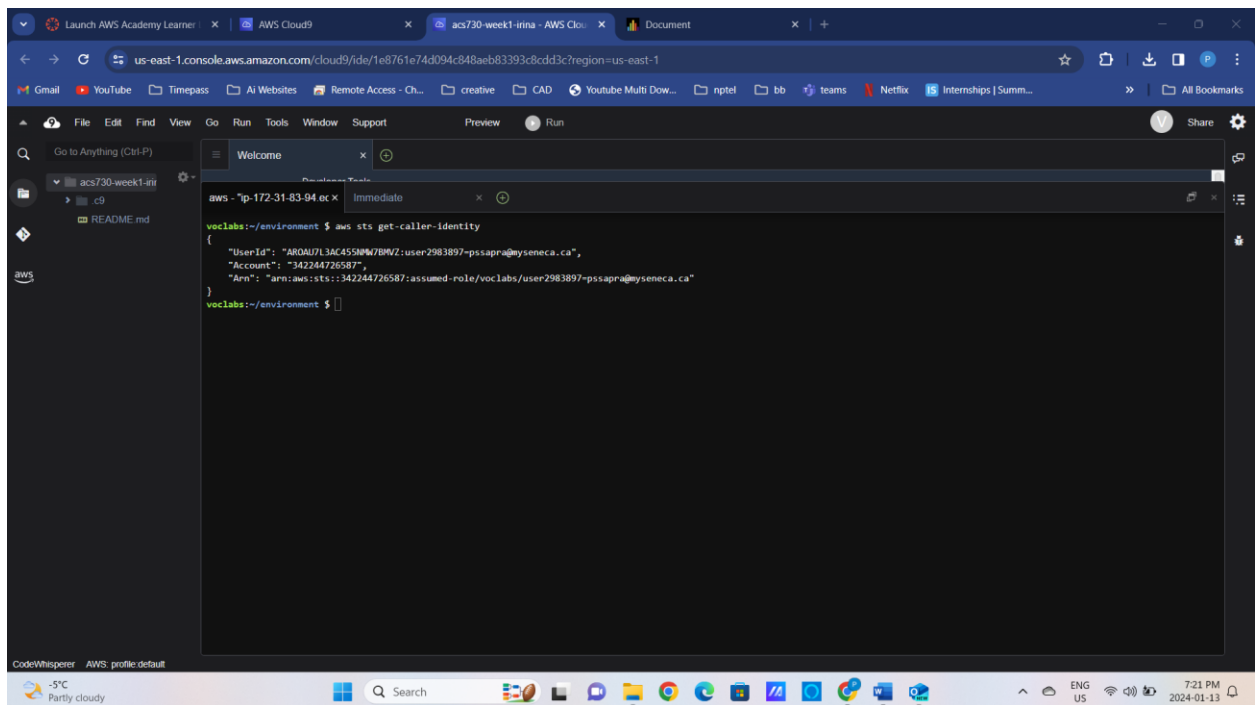


6.

The Cloud9 environment will take a couple of minutes to provision.

Try running AWS CLI command in the Cloud9 Terminal. Do you see any differences compared to Task 2, section 3?

```
aws sts get-caller-identity
```

7. Run `aws --v`, `terraform version` and `git version` to ensure the tools are installed.

Install Terraform from the steps given in the link:

<https://sysadminxpert.com/steps-to-install-terraform-on-amazon-linux/#:~:text=Steps%20to%20Install%20Terraform%20on%20Amazon%20Linux%201,...%205%20Step%205%3A%20Terraform%20help%20command%20>

```
sudo yum install wget unzip
```

```
curl -O
```

```
https://releases.hashicorp.com/terraform/0.12.2/terraform_0.12.2_linux_amd64.zip
```

```
sudo unzip ./terraform_0.12.2_linux_amd64.zip -d /usr/local/bin
```

```
terraform -v
```

The screenshot shows a terminal window in the AWS Cloud9 IDE. The terminal is running a series of commands to install Terraform on a Linux instance. The commands and their outputs are as follows:

```
voelabs:~/environment $ aws sts get-caller-identity
{
  "UserId": "AROAIU713AC4580M70WZ:user2983897-pssapra@myseneca.ca",
  "Account": "342244726587",
  "Arn": "arn:aws:sts::342244726587:assumed-role/voelabs/user2983897-pssapra@myseneca.ca"
}

voelabs:~/environment $ aws --v
aws-cli/2.15.8 Python/3.11.6 Linux/6.1.66-93.164.amzn2023.x86_64 exe/x86_64.amzn.2023 prompt/off

voelabs:~/environment $ sudo yum install wget unzip
Last metadata expiration check: 0:02:21 ago on Sun Jan 14 00:20:25 2024.
Package wget-1.21.3-1.amzn2023.0.3.x86_64 is already installed.
Package unzip-6.0-57.amzn2023.0.2.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!

voelabs:~/environment $ curl -O https://releases.hashicorp.com/terraform/0.12.2/terraform_0.12.2_linux_amd64.zip
% Total % Received % Xferd Average Speed Time Time Time Current
% Total % Received % Xferd Average Speed Time Time Time Current
100 15.2M 100 15.2M 0 0 45.9M 0 --:--:-- --:--:-- --:--:-- 45.8M

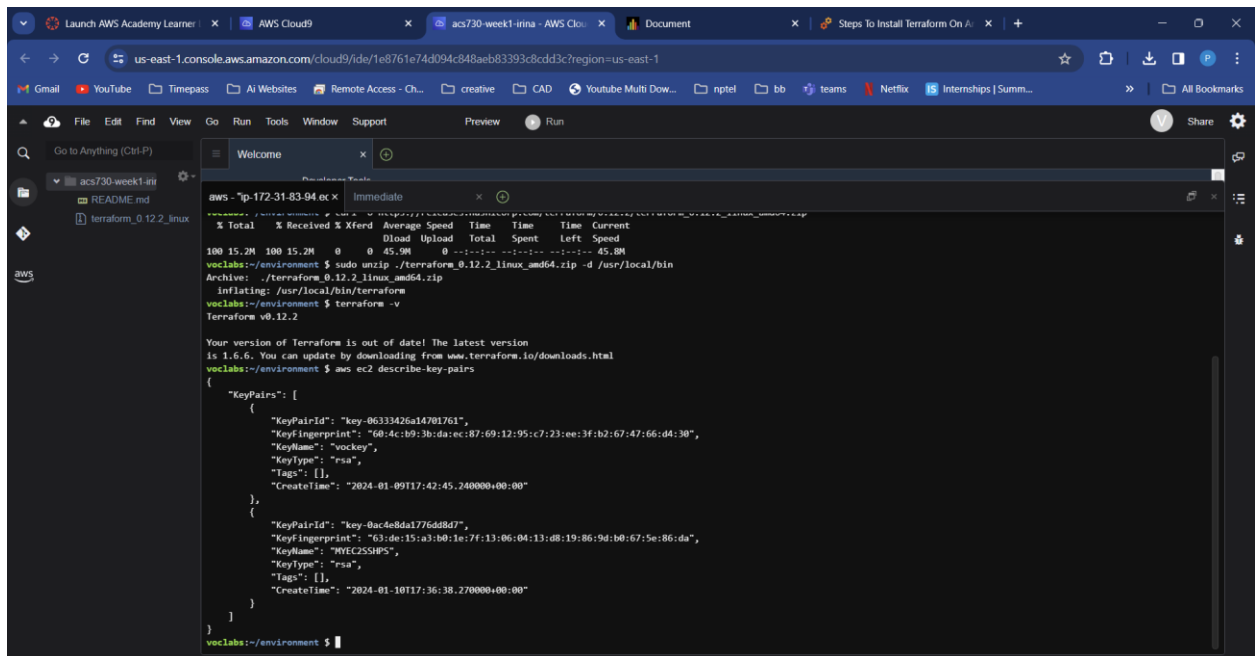
voelabs:~/environment $ sudo unzip ./terraform_0.12.2_linux_amd64.zip -d /usr/local/bin
Archive: ./terraform_0.12.2_linux_amd64.zip
Inflating: /usr/local/bin/terraform
voelabs:~/environment $ terraform -v
Terraform v0.12.2

Your version of Terraform is out of date! The latest version
is 1.6.6. You can update by downloading from www.terraform.io/downloads.html
voelabs:~/environment $
```

Task 4: Provisioning Amazon EC2 with AWS CLI

1. Get the key pair name provisioned by AWS Academy by running the AWS CLI command below.

```
aws ec2 describe-key-pairs
```



```
aws - sp-172-31-83-94.ec x Immediate x
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 15.2M 100 15.2M 0 0 45.9M 0 --:--:-- --:--:-- --:--:-- 45.8M
voclabs:~/environment $ sudo unzip -j terraform_0.12.2_linux_amd64.zip -d /usr/local/bin
Archive: ./terraform_0.12.2_linux_amd64.zip
inflating: /usr/local/bin/terraform
voclabs:~/environment $ terraform -v
Terraform v0.12.2

Your version of Terraform is out of date! The latest version
is 1.6.6. You can update by downloading from www.terraform.io/downloads.html
voclabs:~/environment $ aws ec2 describe-key-pairs
{
  "KeyPairs": [
    {
      "KeyPairId": "key-06333426a14701761",
      "KeyFingerprint": "60:4c:b9:3b:da:ec:87:69:12:95:c7:23:ee:3f:b2:67:47:66:d4:30",
      "KeyName": "vockey",
      "KeyType": "rsa",
      "Tags": [],
      "CreateTime": "2024-01-09T17:42:45.240000+00:00"
    },
    {
      "KeyPairId": "key-0ac4e8da1776dd8d7",
      "KeyFingerprint": "63:de:15:a3:b0:1e:7f:13:06:04:13:d8:19:86:9d:b0:67:5e:86:da",
      "KeyName": "vockey",
      "KeyType": "rsa",
      "Tags": [],
      "CreateTime": "2024-01-10T17:36:38.270000+00:00"
    }
  ]
}
```

2. Provision and Amazon EC2 instance with the SSH key pair from the previous command. Notice dynamic resolution of the AMI id (VM template).

```
aws ec2 run-instances --image-id resolve:ssm:/aws/service/ami-  
amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2 --instance-type  
t2.micro --key-name vockey
```

The screenshot shows a terminal window with the AWS CLI command `aws --profile default ec2 describe-instances --instance-ids i-04bef30b98ab34e3c` and its output. The output is a JSON object containing details about the instance, including its state (pending), placement (us-east-1a), and various identifiers.

```
{
  "Groups": [],
  "Instances": [
    {
      "AmiLaunchIndex": 0,
      "ImageId": "ami-04c09ef2f505b099",
      "InstanceId": "i-04bef30b98ab34e3c",
      "InstanceType": "t2.micro",
      "KeyName": "vockey",
      "LaunchTime": "2024-01-14T00:26:36+00:00",
      "Monitoring": {
        "State": "disabled"
      },
      "Placement": {
        "AvailabilityZone": "us-east-1a",
        "GroupName": "",
        "Tenancy": "default"
      },
      "PrivateDnsName": "ip-172-31-22-228.ec2.internal",
      "PrivateIpAddress": "172.31.22.228",
      "ProductCodes": [],
      "PublicDnsName": "",
      "State": {
        "Code": 0,
        "Name": "pending"
      },
      "StateTransitionReason": "",
      "SubnetId": "subnet-02f49470e9abb971e",
      "VpcId": "vpc-03e35b696af88e0d2",
      "Architecture": "x86_64",
      "BlockDeviceMappings": [],
      "ClientToken": "73b1f689-f123-49af-af8e-79399434ae62",
    }
  ]
}
```

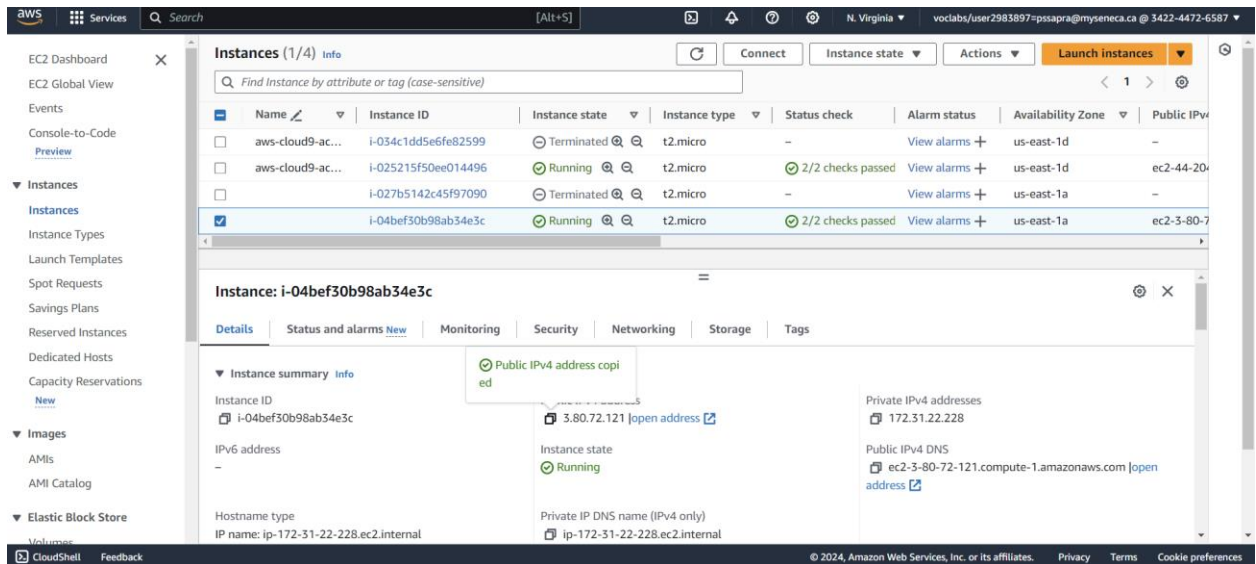
3. In the AWS Management console, verify that the instance got provisioned as expected. Save the instance id i-xxxxxxxxxx for the future use

The screenshot shows the AWS Management console's 'Instances' page. It displays a table of instances with columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, and Public IP. The table shows four instances, with two in a 'Running' state and two in a 'Terminated' state.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
aws-cloud9-ac...	i-034c1dd5e6fe82599	Terminated	t2.micro	-	View alarms	us-east-1d	-
aws-cloud9-ac...	i-025215f50ee014496	Running	t2.micro	2/2 checks passed	View alarms	us-east-1d	ec2-44-20...
	i-027b5142c45f97090	Terminated	t2.micro	-	View alarms	us-east-1a	-
	i-04bef30b98ab34e3c	Running	t2.micro	2/2 checks passed	View alarms	us-east-1a	ec2-3-80-7...

Task 5: Securely connecting Amazon EC2 with SSH (Secure Shell)

1. Select the provisioned instance and mark it's public IP.



2. Attempt connecting to the EC2 from the Cloud9 terminal using the command below: `ssh ec2-user@3.83.46.187`

```
ssh ec2-user@3.83.46.187
```

```
ssh ec2-user@3.80.72.121
```

```
bash - "p-172-31-83-94 o x" Immediate x
{
  "CapacityReservationReference": "open",
  "MetadataOptions": {
    "State": "pending",
    "HttpTokens": "optional",
    "HttpPutResponseHopLimit": 1,
    "HttpEndpoint": "enabled",
    "HttpProtocolIpv6": "disabled",
    "InstanceMetadataTags": "disabled"
  },
  "EnclaveOptions": {
    "Enabled": false
  },
  "PrivateDnsNameOptions": {
    "HostnamesType": "ip-name",
    "EnableResourceNameDnsRecord": false,
    "EnableResourceNameDnsAAAARecord": false
  },
  "MaintenanceOptions": {
    "AutoRecovery": "default"
  },
  "CurrentInstanceBootMode": "legacy-bios"
},
{
  "OwnerId": "342244726587",
  "ReservationId": "r-0d3f4379433f4be56"
}
}

voclabs:~/environment $
voclabs:~/environment $
voclabs:~/environment $ ssh ec2-user@3.80.72.121
ssh: connect to host 3.80.72.121 port 22: Connection timed out
voclabs:~/environment $
```

3. Attempt connecting to the EC2 from the AWS Academy Terminal using the command below. What is the result?

```
ssh -i ~/.ssh/labsuser.pem ec2-user@3.83.46.187
```

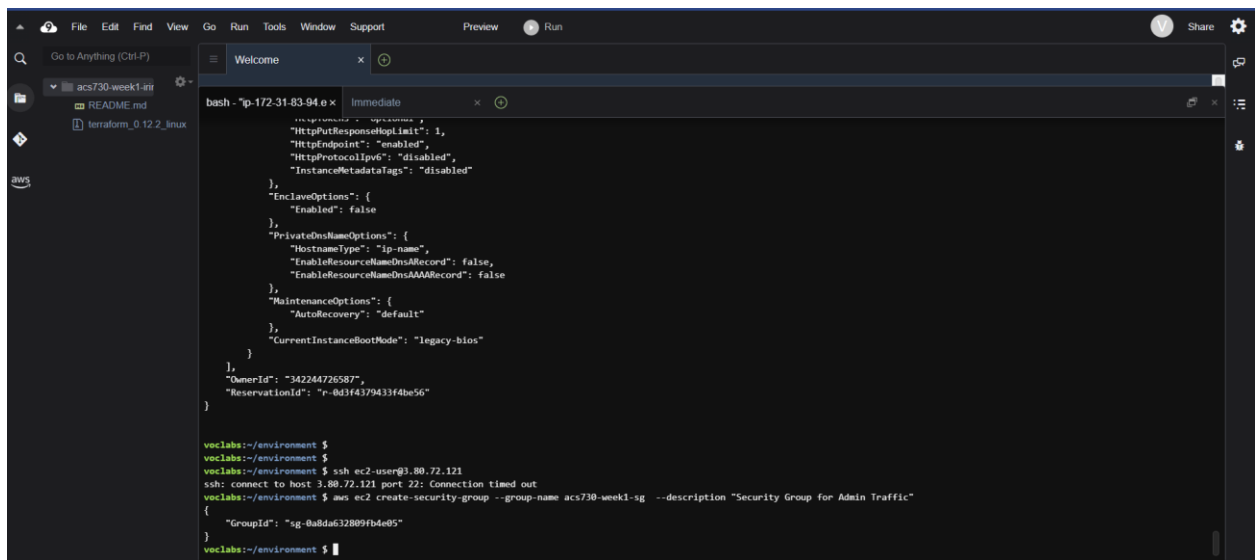
```
ssh -i ~/.ssh/labsuser.pem ec2-user@3.80.72.121
```

```
AWS Used $0 of $100, Dec, 2021 03:53

ddd_v1_w_Mabl_985915@runweb45880:~$ ssh -i ~/.ssh/labsuser.pem ec2-user@3.83.46.187
```

4. Create firewall rules for our AWS EC2 to allow ingress on port 22.

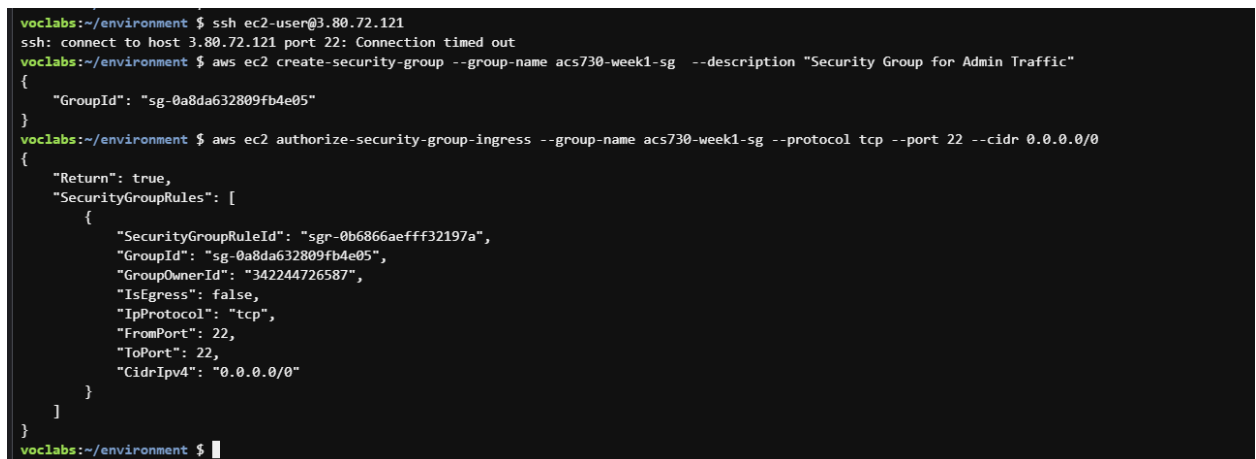
```
aws ec2 create-security-group --group-name acs730-week1-sg --description "Security Group for Admin Traffic"
```



```
bash - "p-172-31-83-94 o x" Immediate
{
  "HttpPutResponseLimit": 1,
  "HttpEndpoint": "enabled",
  "HttpProtocolIpv6": "disabled",
  "InstanceMetadataTags": "disabled"
},
  "EnclaveOptions": {
    "Enabled": false
  },
  "PrivateDnsNameOptions": {
    "HostnameType": "ip-name",
    "EnableResourceNameDnsRecord": false,
    "EnableResourceNameDnsAAAARecord": false
  },
  "MaintenanceOptions": {
    "AutoRecovery": "default"
  },
  "CurrentInstanceBootMode": "legacy-bios"
}
},
  "OwnerId": "342244726587",
  "ReservationId": "r-0d3f4379433f4be56"
}

voclabs:~/environment $
voclabs:~/environment $
voclabs:~/environment $ ssh ec2-user@3.80.72.121
ssh: connect to host 3.80.72.121 port 22: Connection timed out
voclabs:~/environment $ aws ec2 create-security-group --group-name acs730-week1-sg --description "Security Group for Admin Traffic"
{
  "GroupId": "sg-0a8da632809fb4e05"
}
voclabs:~/environment $
```

5. Authorize ingress into EC2 instance from everywhere in the world:
- ```
aws ec2 authorize-security-group-ingress --group-name acs730-week1-sg --protocol tcp --port 22 --cidr 0.0.0.0/0
```



```
voclabs:~/environment $ ssh ec2-user@3.80.72.121
ssh: connect to host 3.80.72.121 port 22: Connection timed out
voclabs:~/environment $ aws ec2 create-security-group --group-name acs730-week1-sg --description "Security Group for Admin Traffic"
{
 "GroupId": "sg-0a8da632809fb4e05"
}
voclabs:~/environment $ aws ec2 authorize-security-group-ingress --group-name acs730-week1-sg --protocol tcp --port 22 --cidr 0.0.0.0/0
{
 "Return": true,
 "SecurityGroupRules": [
 {
 "SecurityGroupRuleId": "sgr-0b6866aefff32197a",
 "GroupId": "sg-0a8da632809fb4e05",
 "GroupOwnerId": "342244726587",
 "IsEgress": false,
 "IpProtocol": "tcp",
 "FromPort": 22,
 "ToPort": 22,
 "CidrIpv4": "0.0.0.0/0"
 }
]
}
voclabs:~/environment $
```

Note the group id (sg-XXXXXXX) and use it as input for the next command. We will add this security group to our EC2 instance.

```
export INSATANCE_ID=i-039846ca048ac228a
```

```
export INSATANCE_ID=i-04bef30b98ab34e3c
```

```
export SG_IG=sg-03c8aacfc095f4070
```

```
export SG_IG=sg-05921ae694ad50992
```

```
}
voclabs:~/environment $ export INSATANCE_ID=i-04bef30b98ab34e3c
voclabs:~/environment $ export SG_IG=sg-05921ae694ad50992
voclabs:~/environment $
```

```
aws ec2 modify-instance-attribute --instance-id ${INSATANCE_ID} --groups
${SG_IG}
```

```
aws ec2 modify-instance-attribute --instance-id i-04bef30b98ab34e3c --groups sg-
05921ae694ad50992
```

```
voclabs:~/environment $ export INSATANCE_ID=i-04bef30b98ab34e3c
voclabs:~/environment $ export SG_IG=sg-05921ae694ad50992
voclabs:~/environment $ aws ec2 modify-instance-attribute --instance-id i-04bef30b98ab34e3c --groups sg-05921ae694ad50992
voclabs:~/environment $
```

6. Verify that the security group is associated with the instance.

```
aws ec2 describe-instances --instance-ids ${INSTANCE_ID} --query
"Reservations[].Instances[].SecurityGroups[].GroupId[]" --output text
```

```
aws ec2 describe-instances --instance-ids i-04bef30b98ab34e3c --query
"Reservations[].Instances[].SecurityGroups[].GroupId[]" --output text
```

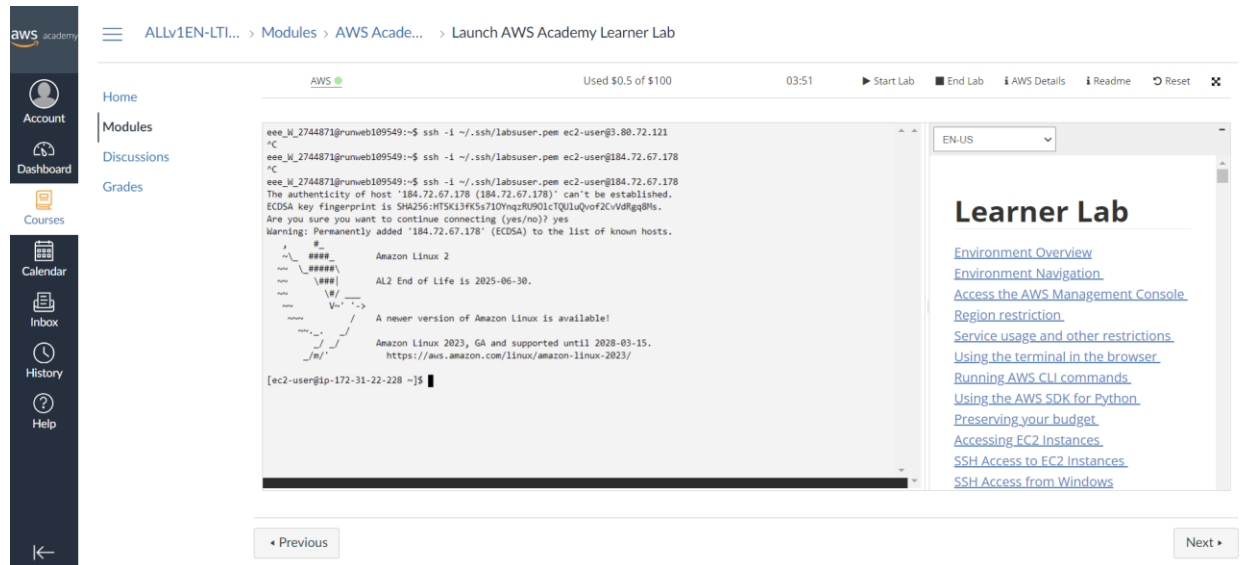
```
}
voclabs:~/environment $ export INSATANCE_ID=i-04bef30b98ab34e3c
voclabs:~/environment $ export SG_IG=sg-05921ae694ad50992
voclabs:~/environment $ aws ec2 modify-instance-attribute --instance-id i-04bef30b98ab34e3c --groups sg-05921ae694ad50992
voclabs:~/environment $ aws ec2 describe-instances --instance-ids i-04bef30b98ab34e3c --query "Reservations[].Instances[].SecurityGroups[].GroupId[]" --output text
sg-05921ae694ad50992
voclabs:~/environment $
```



7. Attempt connecting to the EC2 from the AWS Academy Terminal using the command below.

```
ssh -i ~/.ssh/labsuser.pem ec2-user@3.83.46.187
```

```
ssh -i ~/.ssh/labsuser.pem ec2-user@184.72.67.178
```



**Note:** The public IP address might change between AWS Academy sessions. Make sure to verify the public IP address of the running instance.

8. Read more about [connecting to Amazon EC2](#) from your local environment with SSH key pair and other options