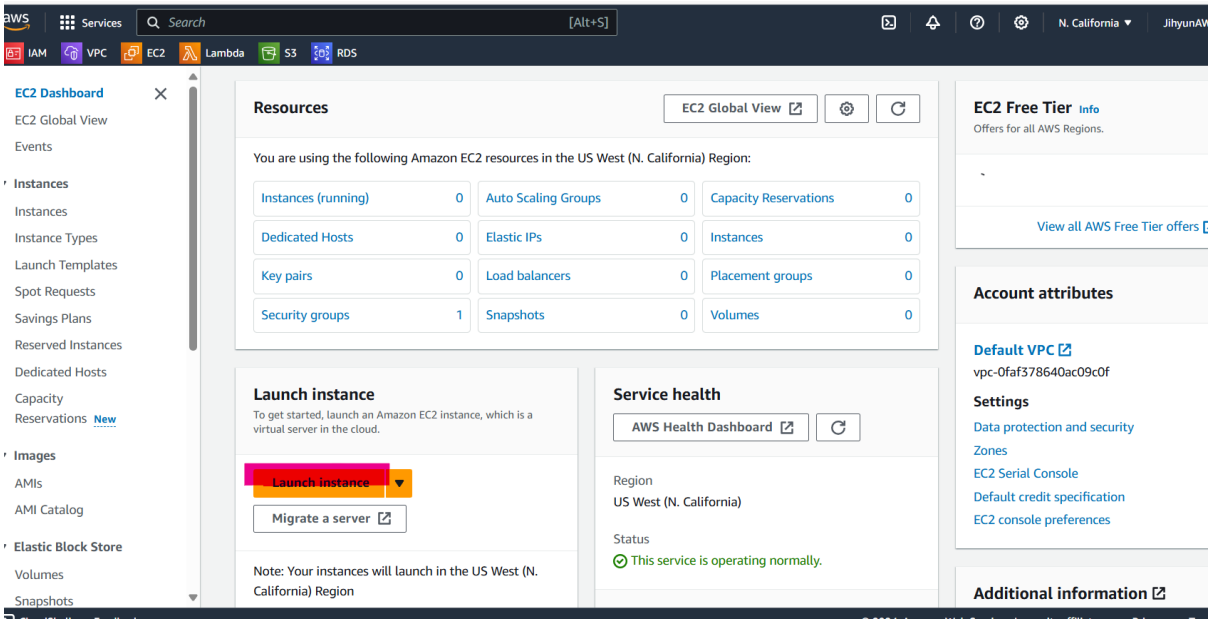
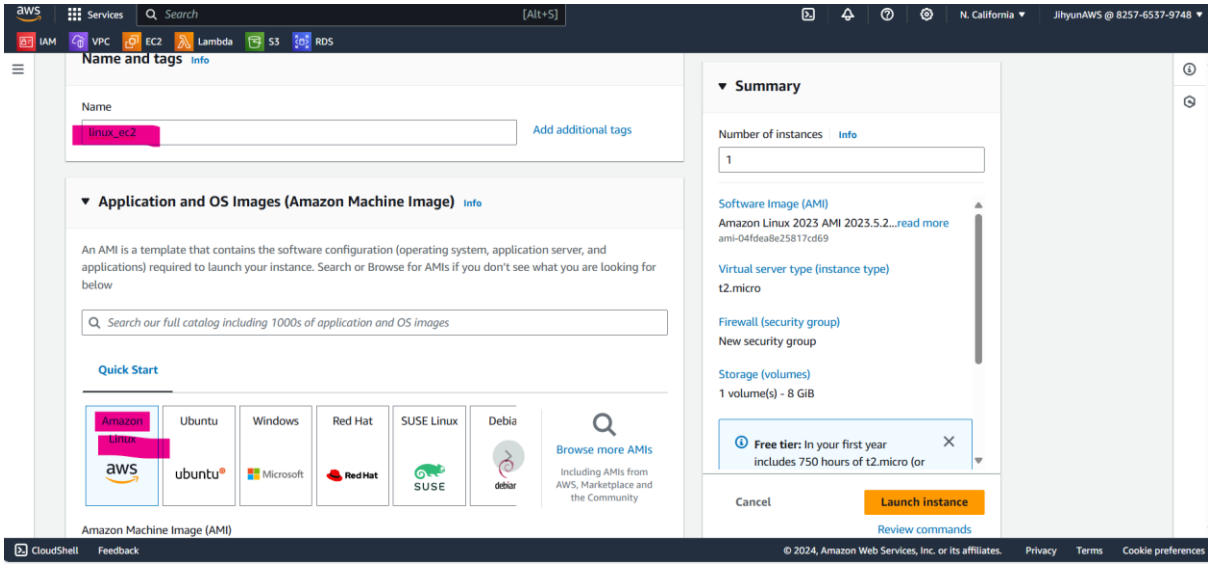
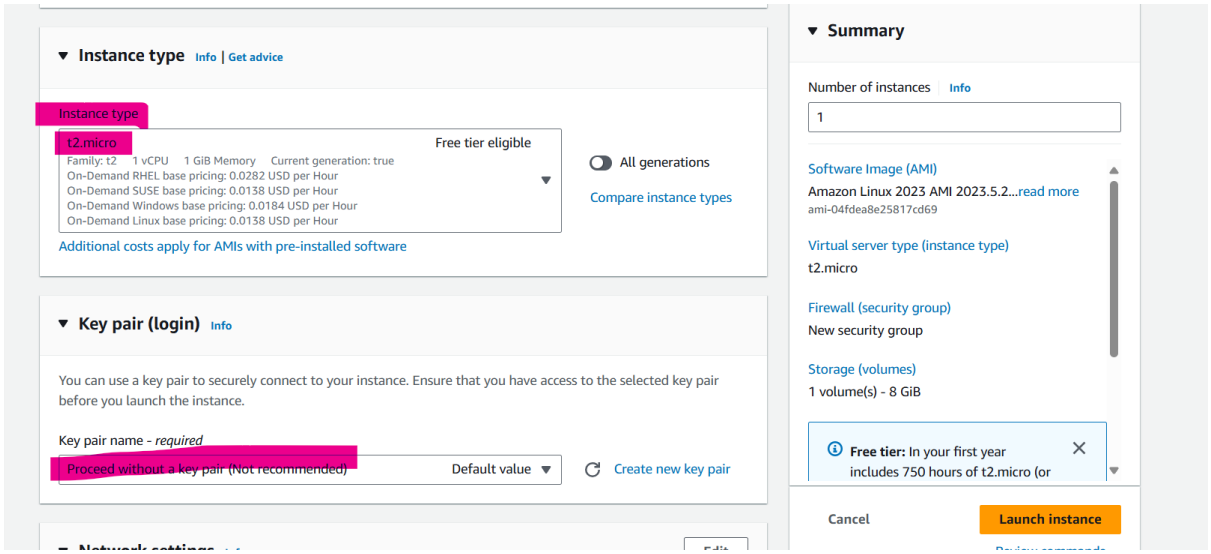
Week 4 – Linux and Bash

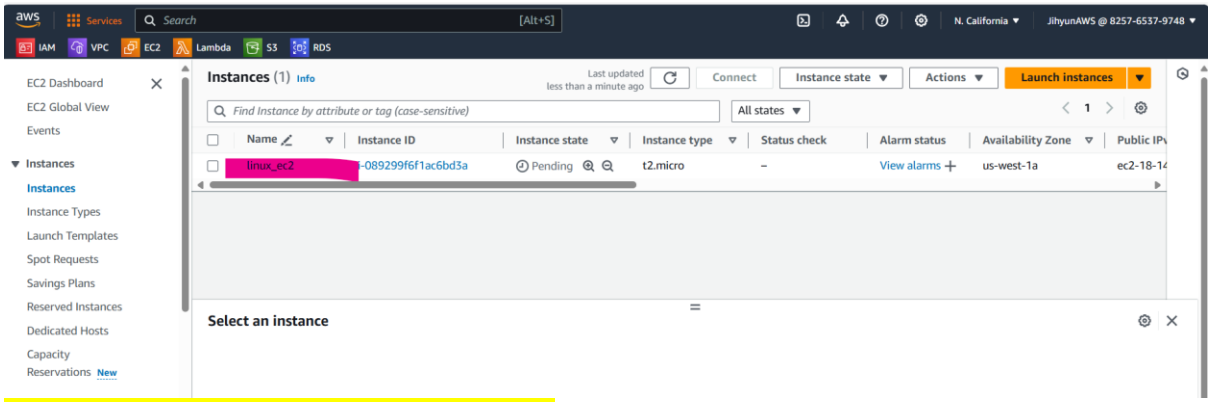
- EC2 is the elastic compute service and a virtual machine, computer running in the cloud.
- We need an operating system to run a computer (virtual machine) in the cloud, that we are going to use Linux.



- All EC2 virtual machines and operating systems are built on top of the Linux, which is the reason why it is crucial to master Linux.



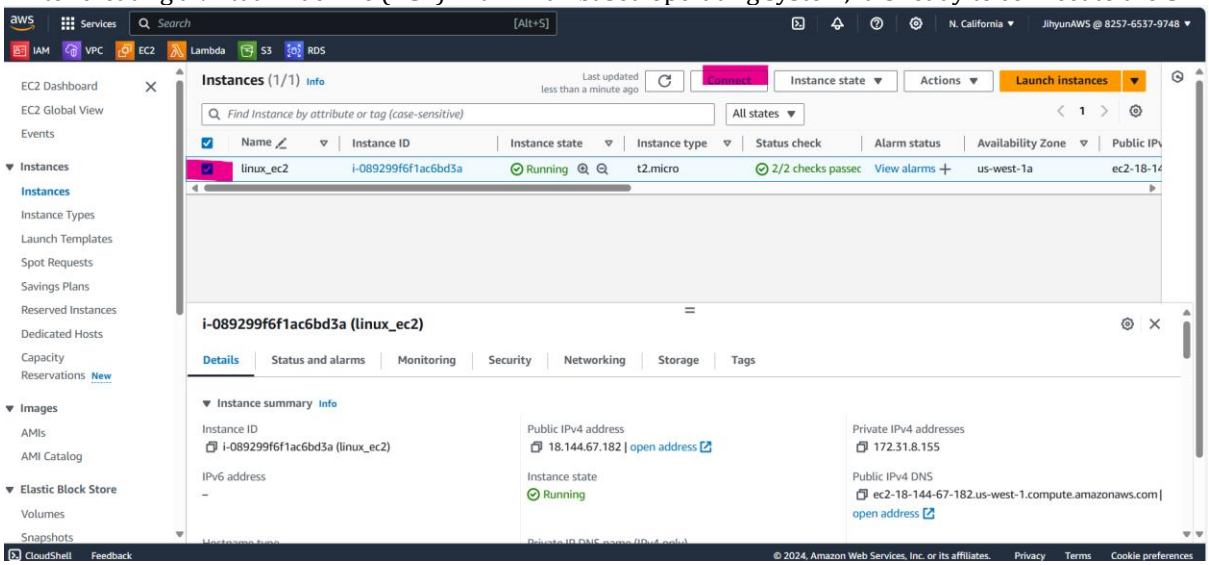


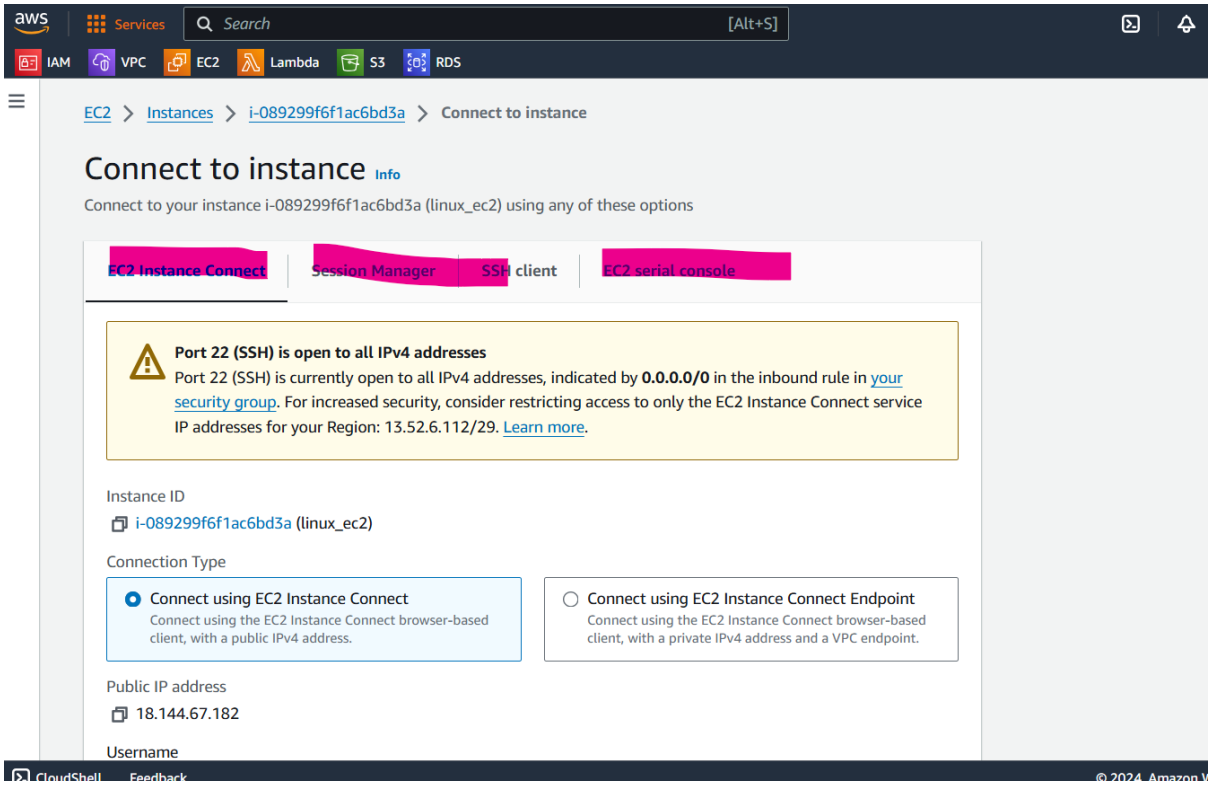


일단 모든 옵션은 default로 두고 설치하기!!!!

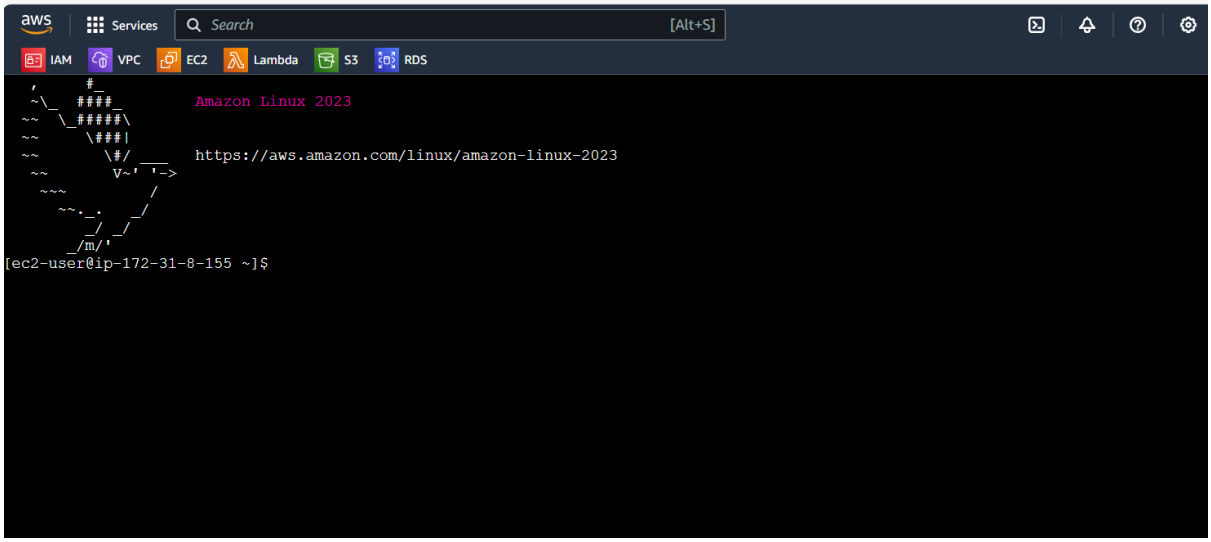
2. Navigating the Linux File system

- After creating a virtual machine (EC2) with Linux based operating system, it is ready to connect to the CLI.





- There are different ways to connect to the CLI, we are going to connect to the CLI using EC2 instance connect in this module.

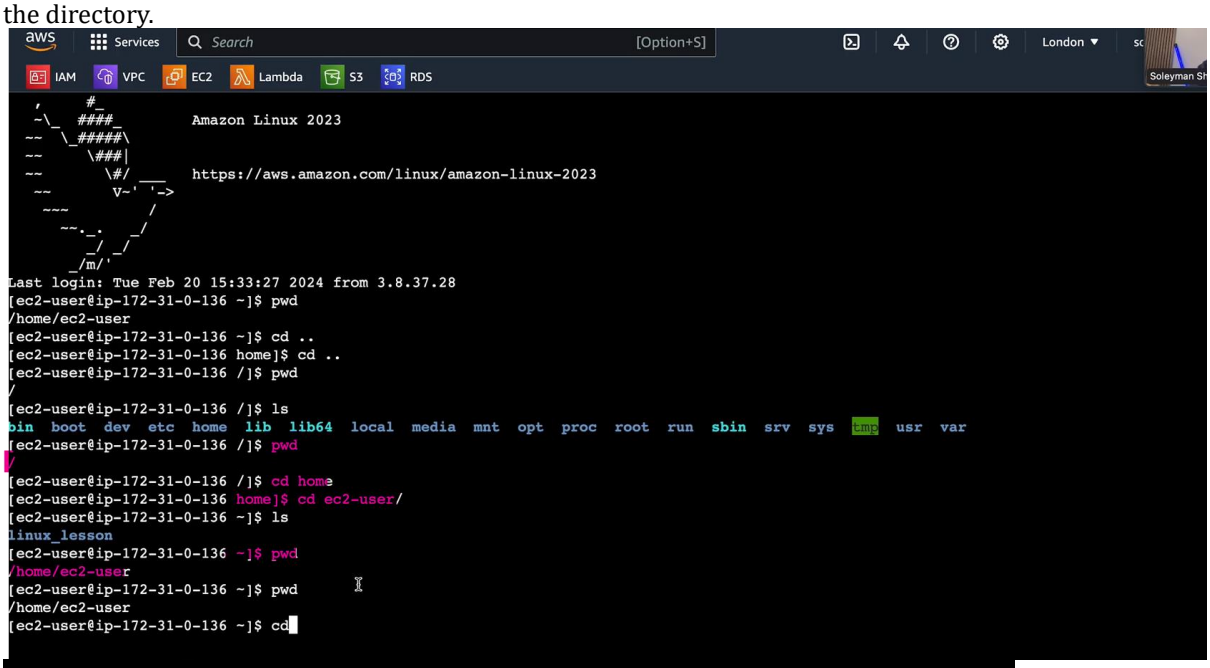


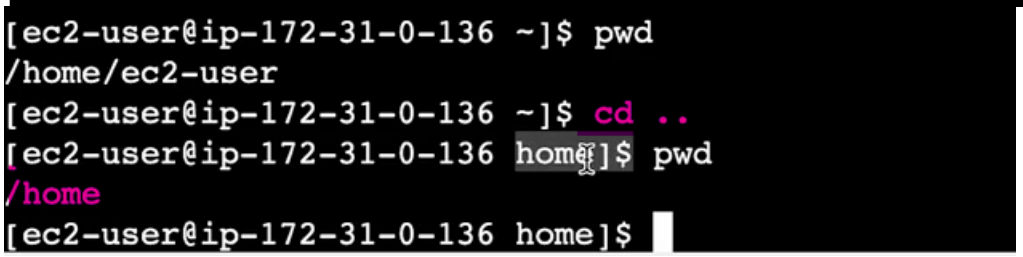
- We are connected to the Amazon Linux EC2 virtual machine.
- We are going to navigate the Linux file system which is the foundational skill for anyone working with Linux operating system.
- The Linux file system are organized in a hierarchical structure and understanding this structure helps you manage files and directories.

1. pwd
: We are now in the /home/ec2-user directory.
2. cd .. (must space between cd and the first .) two times and enter pwd.
: We can see that we are on the top directory with the result as /. Every files and directories are under this top directory.
3. ls
: List everything included in the current working directory. We can see the list of all directories under the current directory. For example, /home includes personal directory for each user. Our personal documents

and code will be stored in home directory.
- /etc directory includes system configuration files read by applications to configure any system or any other software.
- /var directory contains files such as logs and cache data which are continuously changed during normal operations.
- /bin contains executable binaries, system programs and scripts available for users.
- /lib holds a shared library needed by applications in the bin trum?
- /tmp is a temporary directory any user creates a file.

- pwd (print working directory) let us know the current working directory we are located and cd is changing the directory.





- cd .. indicates going back to the previous directory.
- cd and hit tab automatically show where we can go or just enter ls to see what directory in the home.
- ls -l(영문 l을 말하는 것!!) shows the detailed information of the file directory.
- ls -a shows all the files including the hidden files.

- Each comment that we executed is processed by bash controlling Linux file operating system environment.

3. File Operations

- We are going to add files and edit them in the Linux environment.
- Create an empty file in Linux by using ‘touch’ command and also change time stamps without modifying the file contents.
- Create an empty file naming as ‘my-file.txt’ by using ‘touch’ command but if there has already been my-file.txt in the user directory then touch command just update time stamps without modifying file contents.

```
[ec2-user@ip-172-31-8-155 /]$ pwd
/
[ec2-user@ip-172-31-8-155 /]$ ls
bin  boot  dev  etc  home  lib  lib64  local  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
[ec2-user@ip-172-31-8-155 /]$ cd
[ec2-user@ip-172-31-8-155 ~]$ pwd
/home/ec2-user
[ec2-user@ip-172-31-8-155 ~]$ touch my-file.txt
[ec2-user@ip-172-31-8-155 ~]$ ls
my-file.txt
[ec2-user@ip-172-31-8-155 ~]$ nano my-file.txt
```

- There are two ways we add the file we created, one of which is using ‘nano’ command.

```
  GNU nano 5.8                                  my-file.txt
[ Read 0 lines ]
^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo       M-A Set Mark     M-] To Bracket  M-Q Previous
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo       M-6 Copy         ^Q Where Was    M-W Next
```

- In the above shell, we can write something like ‘write something in this file’ and ctrl+O and enter but the content was not saved.
- Ctrl + x to exit this shell while saving the content in my-file.txt.

```
[ec2-user@ip-172-31-8-155 ~]$ cat my-file.txt
Writing something in this file.
[ec2-user@ip-172-31-8-155 ~]$
```

- After coming back to the main shell, we can use ‘cat’ command to see whether the contents were saved in the CLI.
- The other method to revise the file content is using ‘vi’.

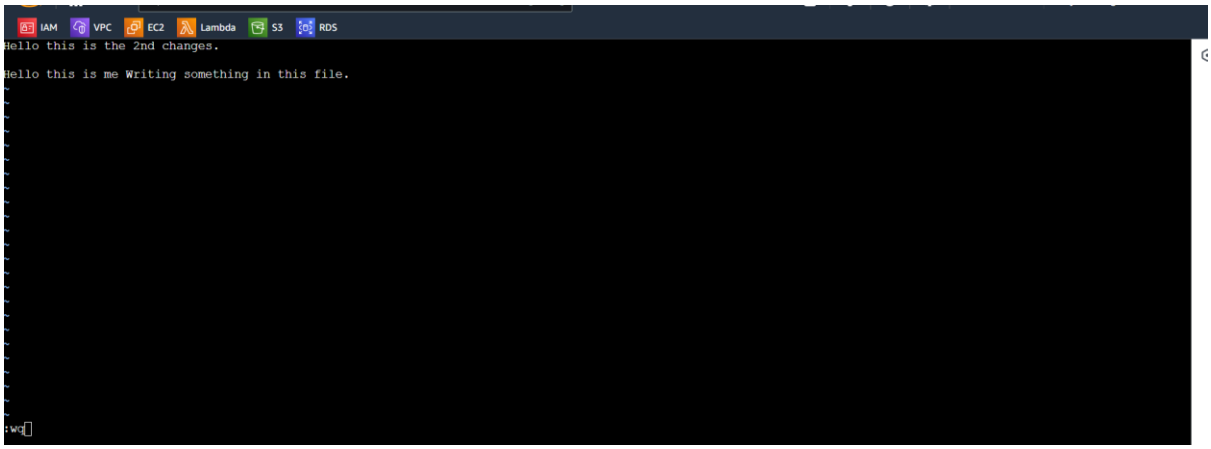
```
[ec2-user@ip-172-31-8-155 ~]$ vi my-file.txt
```

- And press ‘i’ to insert a new content and ‘:w’ to save the file and :q to exit the shell.

```
Hello this is meWriting something in this file.
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"my-file.txt" 1L, 49B written                                                                1,17          All
```

```
[ec2-user@ip-172-31-8-155 ~]$ vi my-file.txt
[ec2-user@ip-172-31-8-155 ~]$ cat my-file.txt
Hello this is me Writing something in this file.
[ec2-user@ip-172-31-8-155 ~]$
```

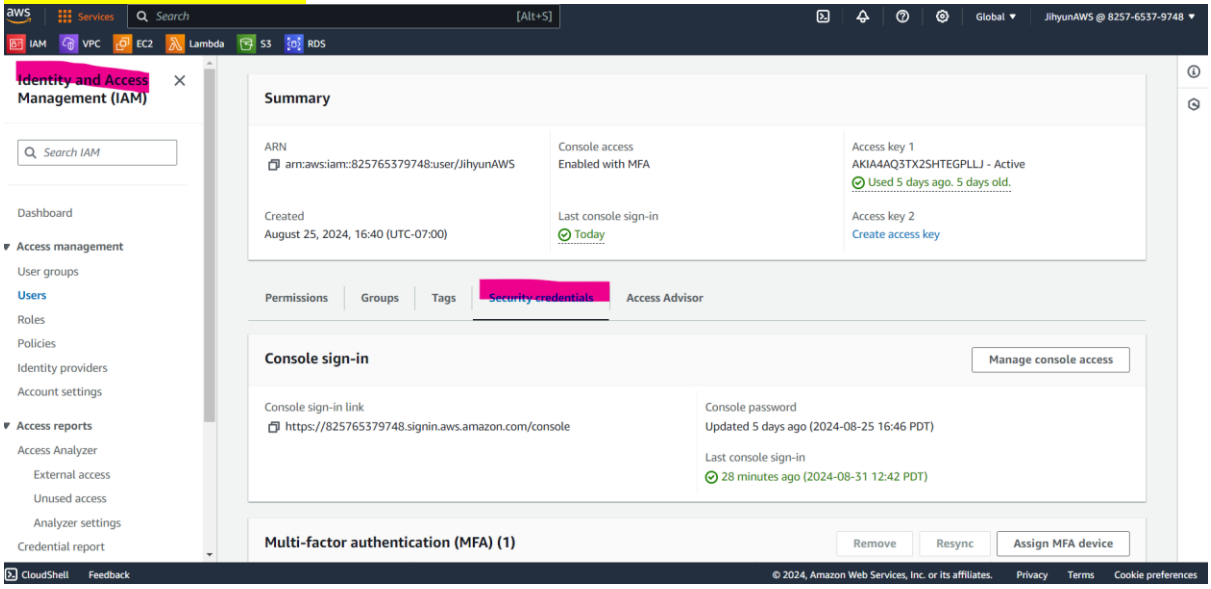
- And also put ‘cat’ command to see the contents we have added.

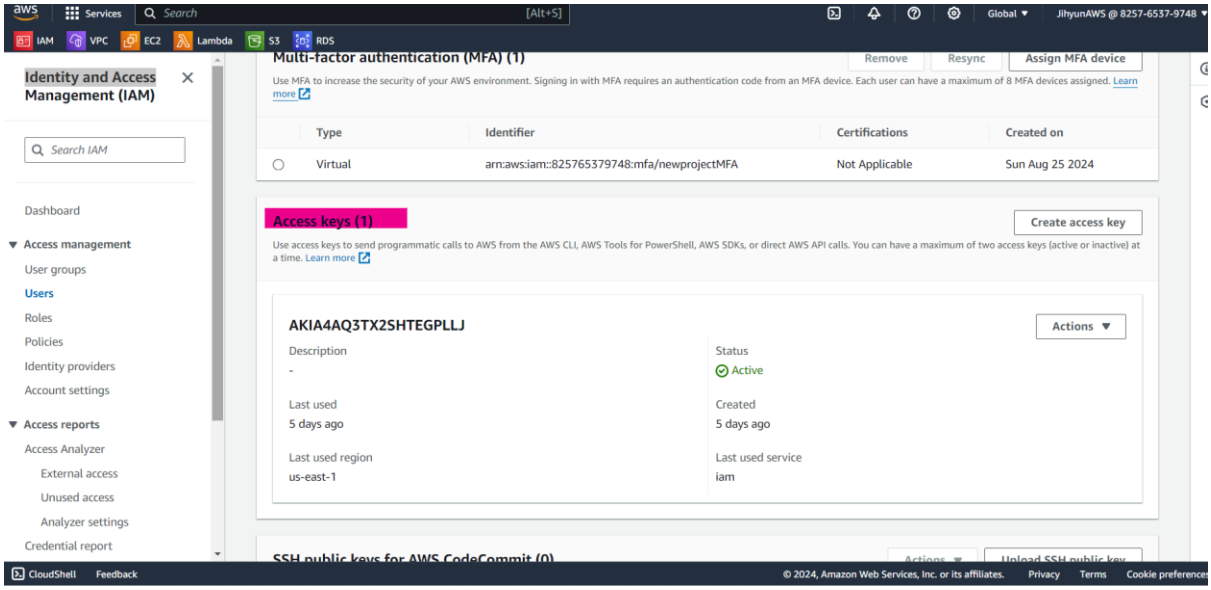


- For the 2nd change, I used ‘via my-file.txt’ and return to another text edition shell and press ‘i’ to insert something and esc to exit the insert mode and then put :wq to process saving and exiting the edition shell and get back to the original shell.

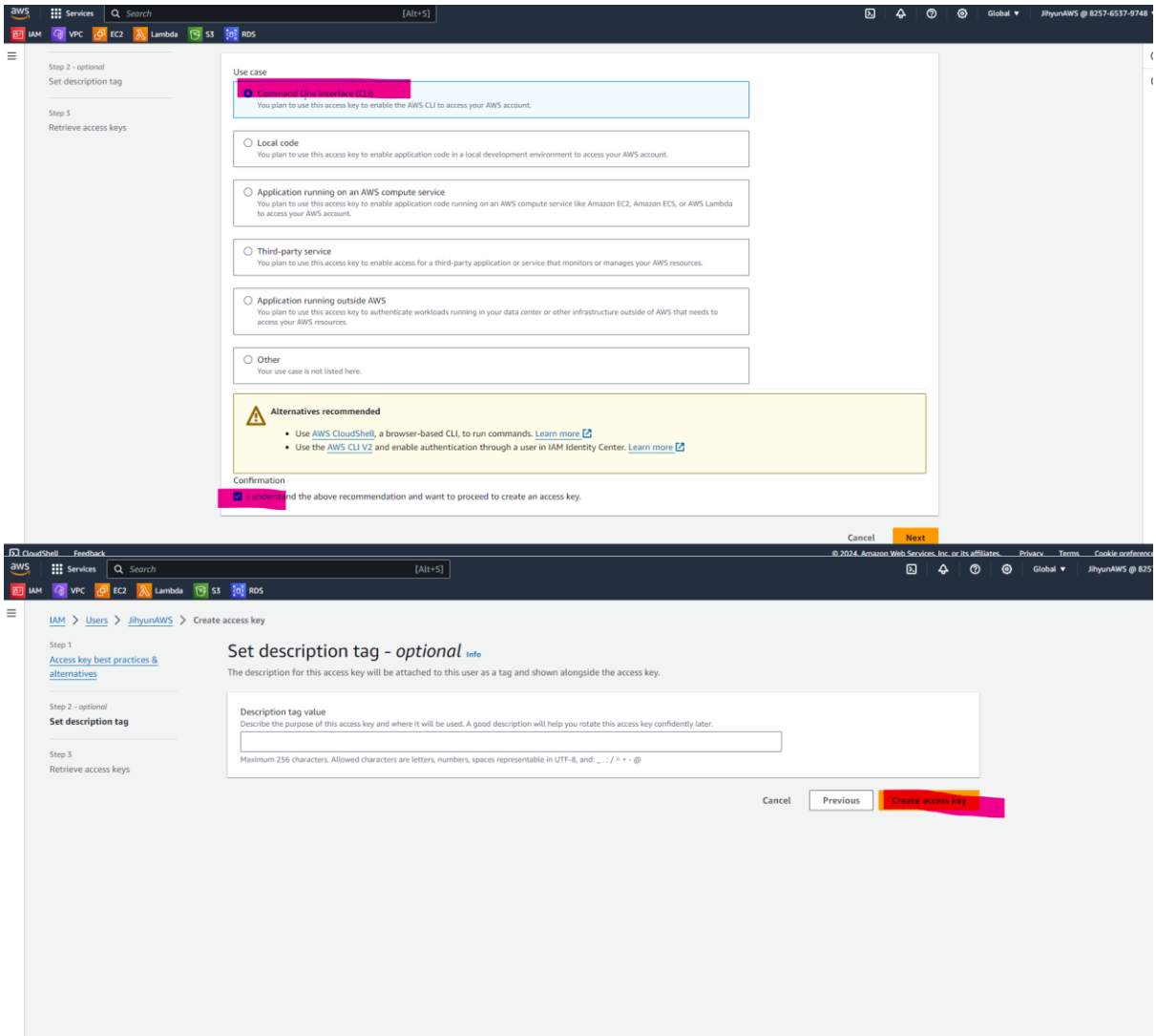
```
[ec2-user@ip-172-31-8-155 ~]$ cat my-file.txt
Hello this is the 2nd changes.

Hello this is me Writing something in this file.
```

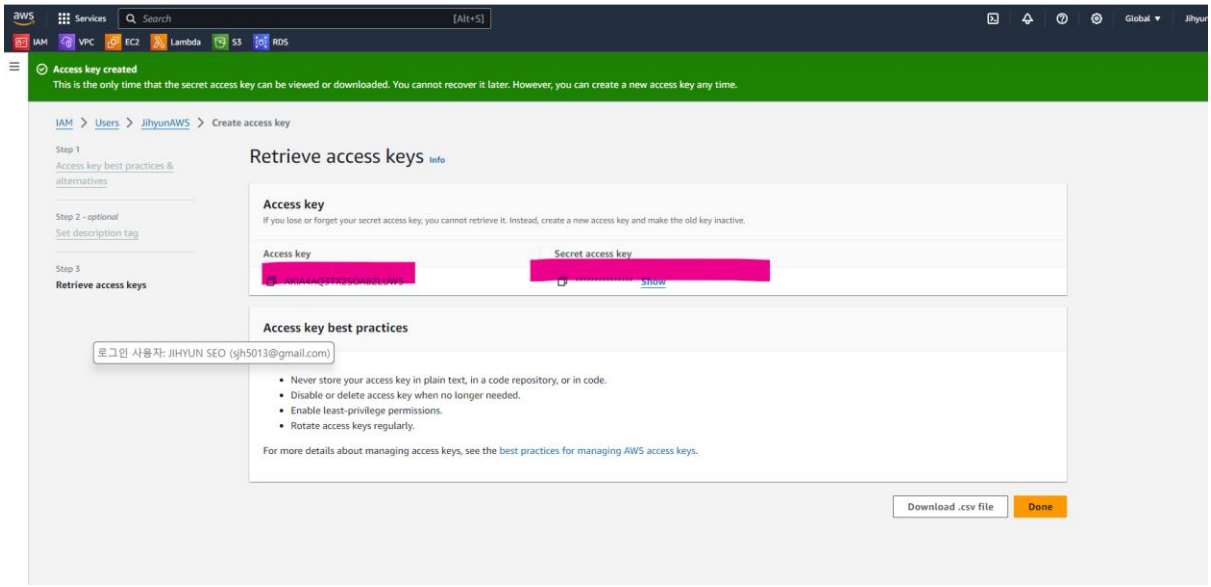
제 5강. Create access keys





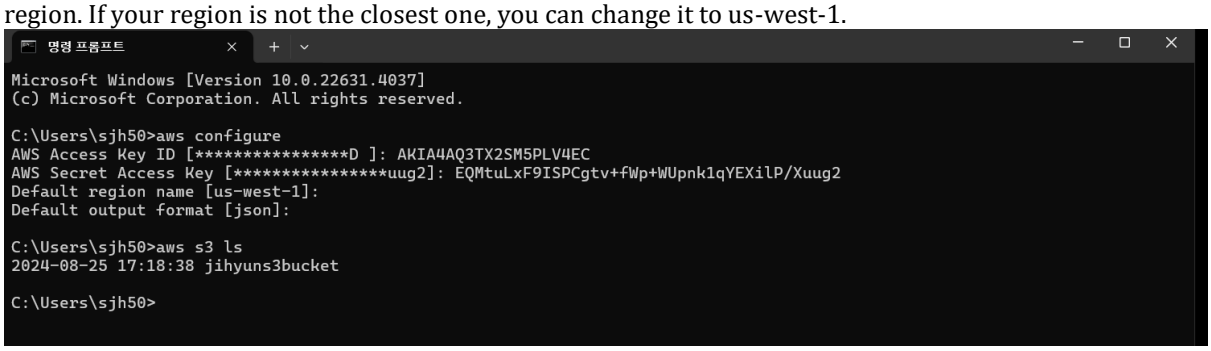
1. Click ‘Create access key’





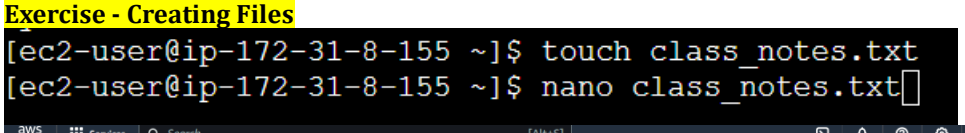
2. When your access key and secret access key are created, you can use them when accessing the CLI in AWS through your interface to configure AWS.

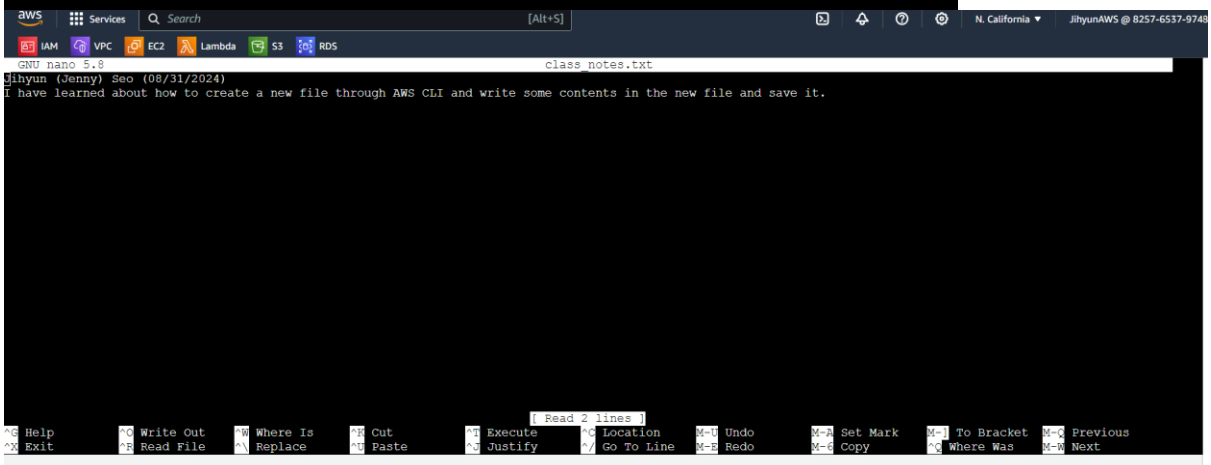
- Going back to the git bash and enter both access key ID and secret access key. After that you can see default region. If your region is not the closest one, you can change it to us-west-1.



- After doing this, you can enter 'aws s3 ls' to see the list of s3 bucket installed.

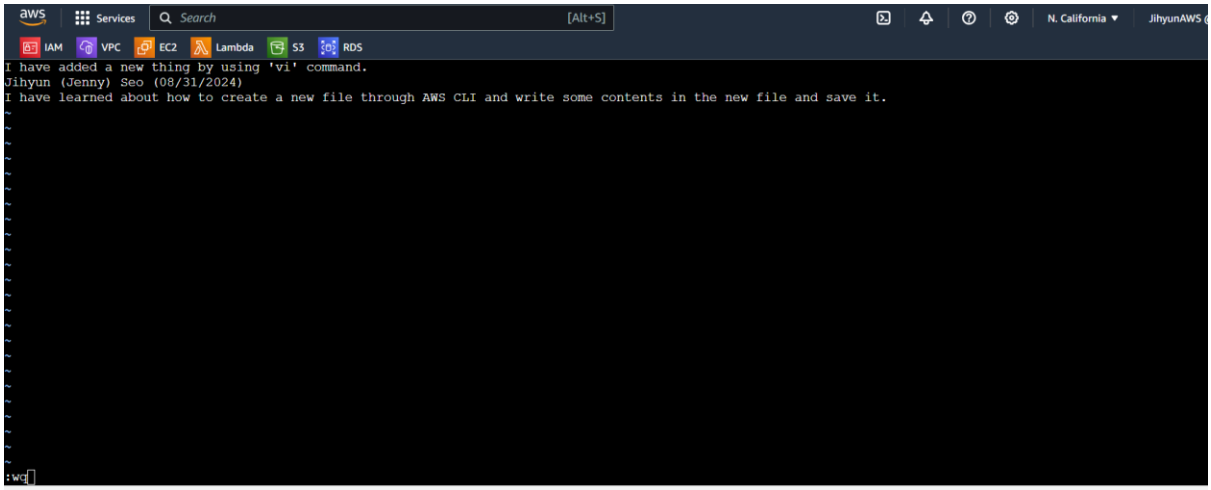
Exercise - Creating Files





In nano, press ‘i’ to insert a few lines and then save your changes with Ctrl + O and confirm it by pressing Enter and then exit with Ctrl + X.

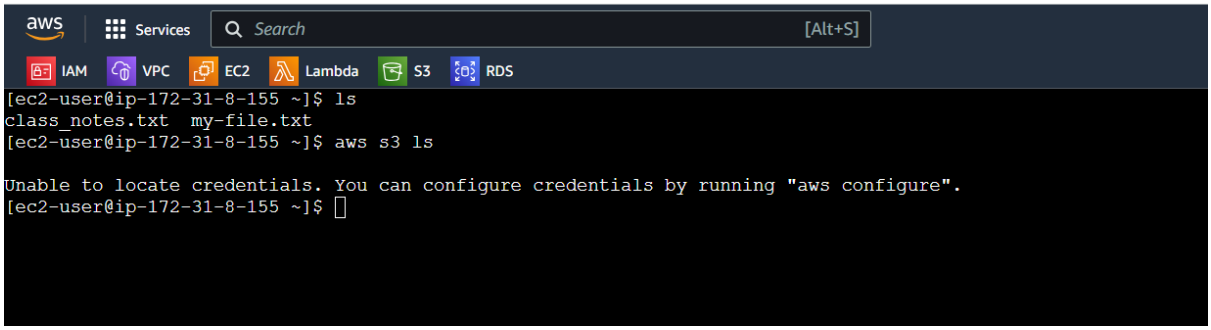
```
[ec2-user@ip-172-31-8-155 ~]$ nano class_notes.txt
[ec2-user@ip-172-31-8-155 ~]$ cat class_notes.txt
Jihyun (Jenny) Seo (08/31/2024)
I have learned about how to create a new file through AWS CLI and write some contents in the new file and save it.
```



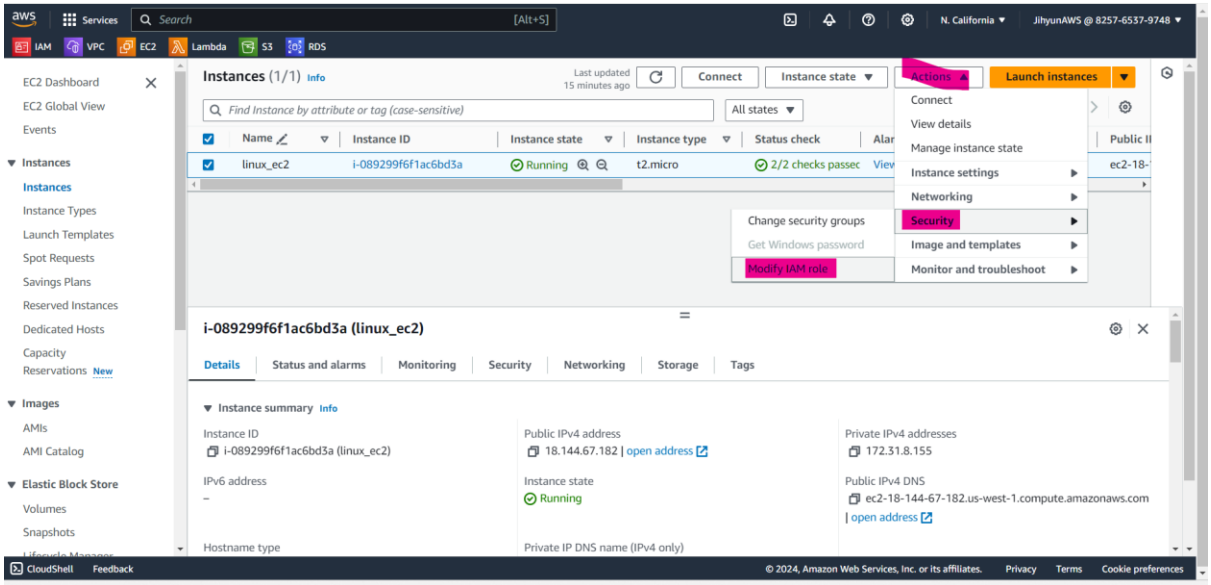
- Editing the file using ‘vi’ command and press ‘i’ to insert a new thing in the file and then press esc to return to the normal mode and then type ‘:wq’ and enter to save and exit.

```
[ec2-user@ip-172-31-8-155 ~]$ vi class_notes.txt
[ec2-user@ip-172-31-8-155 ~]$ cat class_notes.txt
I have added a new thing by using 'vi' command.
Jihyun (Jenny) Seo (08/31/2024)
I have learned about how to create a new file through AWS CLI and write some contents in the new file and save it.
[ec2-user@ip-172-31-8-155 ~]$
```

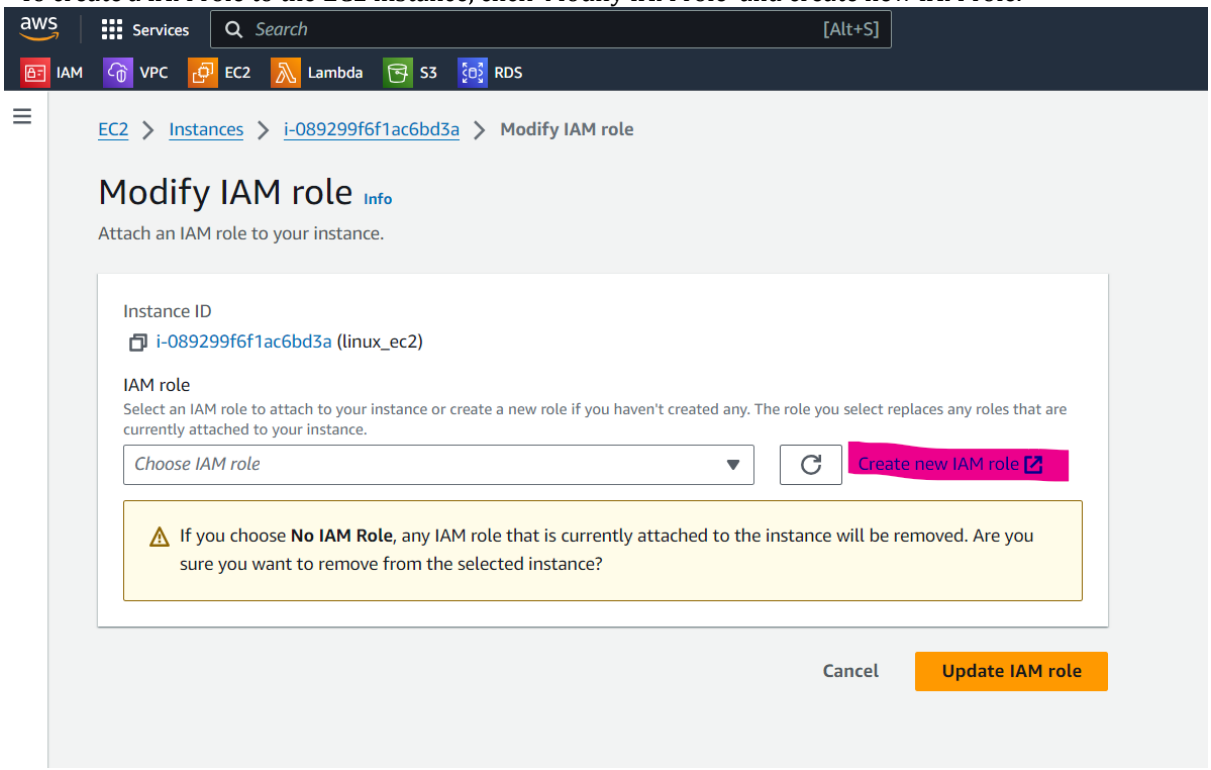
제 6강. Create IAM role and Deploy to S3

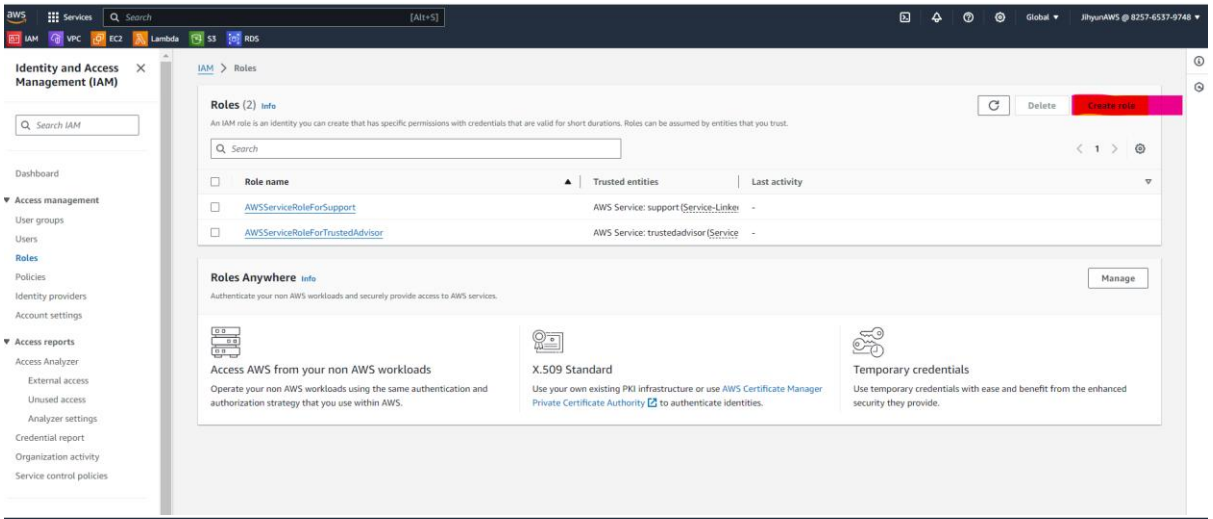


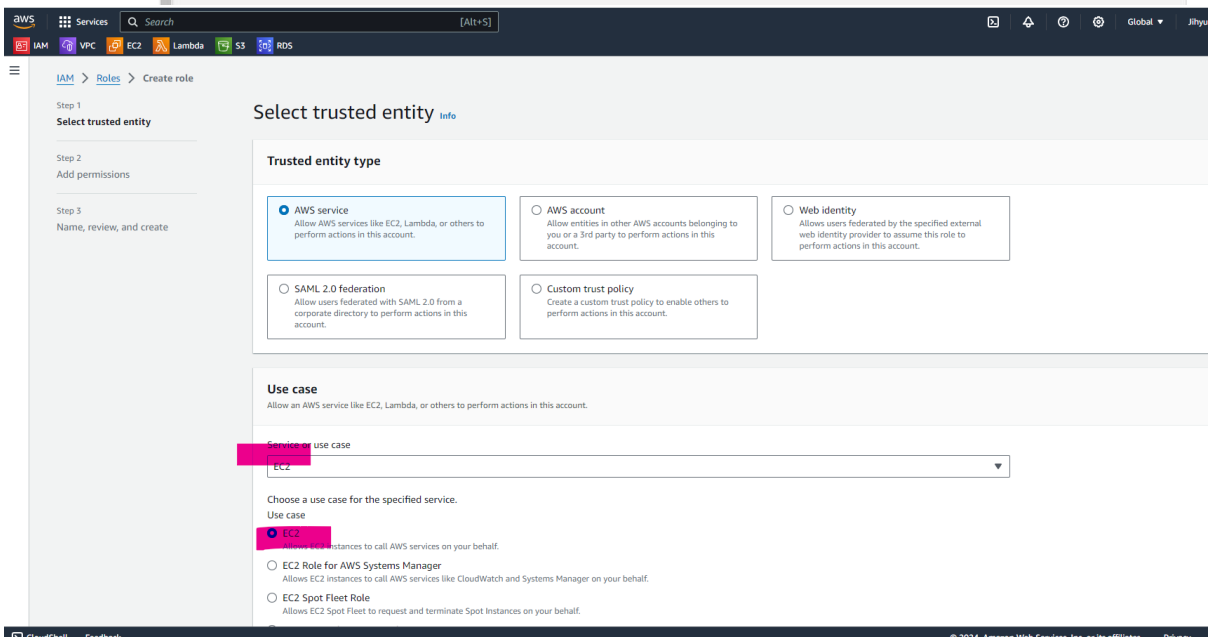
- When we try to list S3 buckets we have, there is an error because we haven’t passed IAM roles accessing the S3 buckets to the EC2 instance. EC2 instance also needs a permission to communicate with Amazon S3.

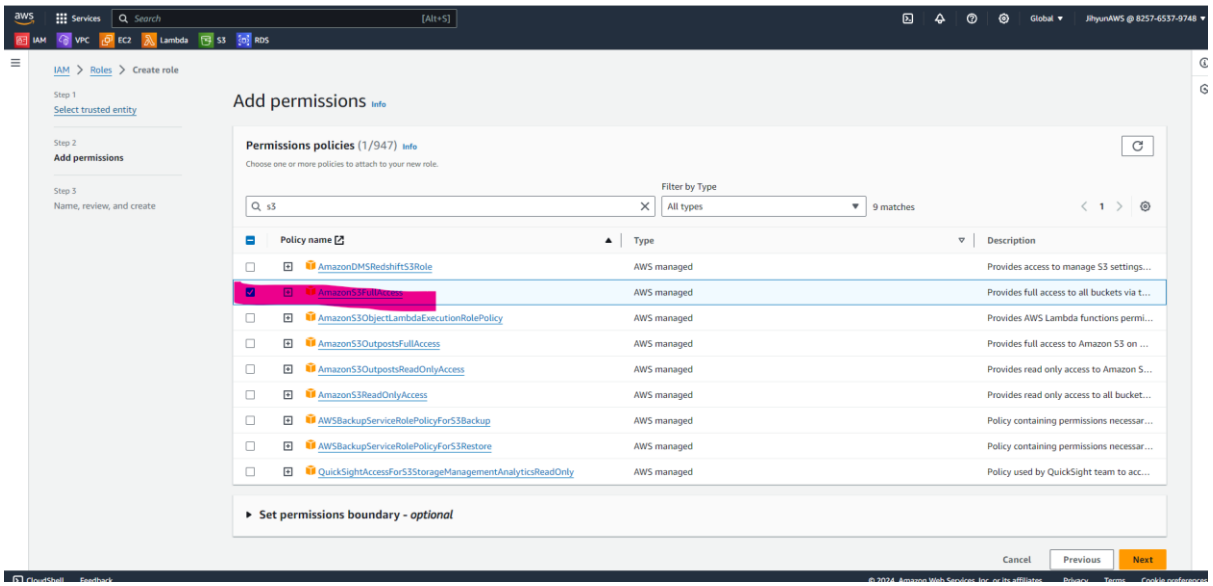


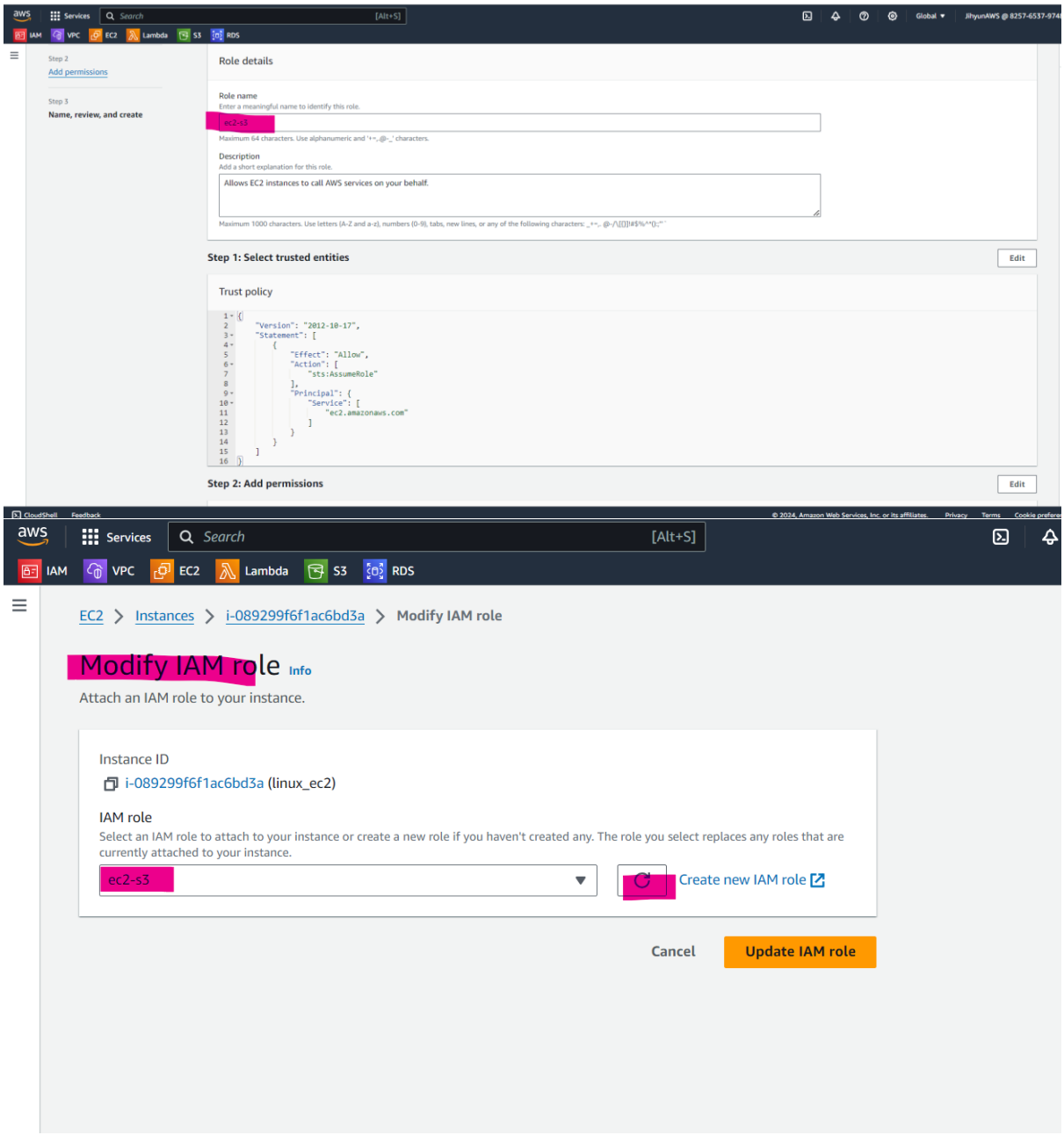
- To create a IAM role to the EC2 instance, click 'Modify IAM role' and create new IAM role.




Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Access reports
Access Analyzer
External access
Unused access
Analyzer settings
Credential report
Organization activity
Service control policies
IAM > Roles
Roles (2) Info
An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.
Search
Delete
Create role
Role name
Trusted entities
Last activity
AWSServiceRoleForSupport
AWS Service: support (Service-Linke
AWSServiceRoleForTrustedAdvisor
AWS Service: trustedadvisor (Service
Roles Anywhere Info
Authenticate your non AWS workloads and securely provide access to AWS services.
Manage
Access AWS from your non AWS workloads
Operate your non AWS workloads using the same authentication and authorization strategy that you use within AWS.
X.509 Standard
Use your own existing PKI infrastructure or use AWS Certificate Manager Private Certificate Authority to authenticate identities.
Temporary credentials
Use temporary credentials with ease and benefit from the enhanced security they provide.


IAM > Roles > Create role
Step 1
Select trusted entity
Step 2
Add permissions
Step 3
Name, review, and create
Select trusted entity Info
Trusted entity type
AWS service
Allow AWS services like EC2, Lambda, or others to perform actions in this account.
AWS account
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.
Web identity
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.
SAML 2.0 federation
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.
Custom trust policy
Create a custom trust policy to enable others to perform actions in this account.
Use case
Allow an AWS service like EC2, Lambda, or others to perform actions in this account.
Service or use case
EC2
Choose a use case for the specified service.
Use case
EC2
Allows EC2 instances to call AWS services on your behalf.
EC2 Role for AWS Systems Manager
Allows EC2 instances to call AWS services like CloudWatch and Systems Manager on your behalf.
EC2 Spot Fleet Role
Allows EC2 Spot Fleet to request and terminate Spot Instances on your behalf.


IAM > Roles > Create role
Step 1
Select trusted entity
Step 2
Add permissions
Step 3
Name, review, and create
Add permissions Info
Permissions policies (1/947) Info
Choose one or more policies to attach to your new role.
Filter by Type
s3
All types
9 matches
Policy name
Type
Description
AmazonDMSRedshiftS3Role
AWS managed
Provides access to manage S3 settings...
AmazonS3FullAccess
AWS managed
Provides full access to all buckets via t...
AmazonS3ObjectLambdaExecutionRolePolicy
AWS managed
Provides AWS Lambda functions permi...
AmazonS3OutpostsFullAccess
AWS managed
Provides full access to Amazon S3 on ...
AmazonS3OutpostsReadOnlyAccess
AWS managed
Provides read only access to Amazon S...
AmazonS3ReadOnlyAccess
AWS managed
Provides read only access to all bucket...
AWSBackupServiceRolePolicyForS3Backup
AWS managed
Policy containing permissions necessar...
AWSBackupServiceRolePolicyForS3Restore
AWS managed
Policy containing permissions necessar...
QuickSightAccessForS3StorageManagementAnalyticsReadOnly
AWS managed
Policy used by QuickSight team to acc...
Set permissions boundary - optional
Cancel
Previous
Next



- After creating a new IAM role to the EC2 instance allowing it to fully access S3 bucket, update the new IAM role and going back to the CLI.

```
[ec2-user@ip-172-31-8-155 ~]$ ls
class_notes.txt  my-file.txt
[ec2-user@ip-172-31-8-155 ~]$ aws s3 ls

Unable to locate credentials. You can configure credentials by running "aws configure".
[ec2-user@ip-172-31-8-155 ~]$ aws s3 ls
2024-08-26 00:18:38 jihyuns3bucket
[ec2-user@ip-172-31-8-155 ~]$
```

- You can access the existing S3 bucket via the CLI

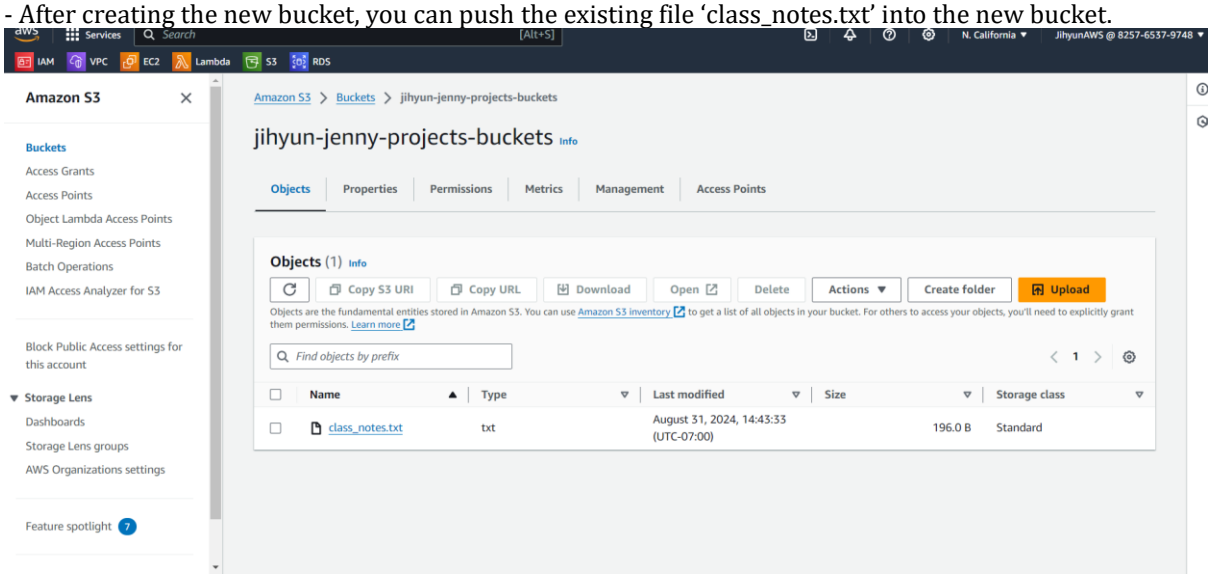
```
[ec2-user@ip-172-31-8-155 ~]$ aws s3 mb s3://jihyun-jenny-projects-buckets
make_bucket: jihyun-jenny-projects-buckets
[ec2-user@ip-172-31-8-155 ~]$
```

- And also, you can make a new bucket using CLI named as jihyun-jenny-projects-buckets, whose names

must be globally unique.

```
[ec2-user@ip-172-31-8-155 ~]$ aws s3 cp class_notes.txt s3://jihyun-jenny-projects-buckets
upload: ./class_notes.txt to s3://jihyun-jenny-projects-buckets/class_notes.txt
[ec2-user@ip-172-31-8-155 ~]$
```

- After creating the new bucket, you can push the existing file 'class_notes.txt' into the new bucket.



-After refreshing your bucket, you can see the existing file was attached to the new bucket.

제 8강. Linux File Permissions

```
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Sat Aug 31 21:19:11 2024 from 13.52.6.115
[ec2-user@ip-172-31-8-155 ~]$ pwd
/home/ec2-user
[ec2-user@ip-172-31-8-155 ~]$ ls
class_notes.txt  my-file.txt
[ec2-user@ip-172-31-8-155 ~]$ ls -l
total 8
-rw-r--r--. 1 ec2-user ec2-user 196 Aug 31 21:29 class_notes.txt
-rw-r--r--. 1 ec2-user ec2-user  81 Aug 31 20:02 my-file.txt
[ec2-user@ip-172-31-8-155 ~]$
```

1. `-rw-r--r--.`

- This is the permission that is assgined to the note, class_notes.txt.

2. '-' in the first part means that it is a file.

3. 'rw-' indicates the permission that this ec2-user has and r is for read and w is for write permission. However, it does not have any executte permission.

4. 'r--' indicates that the 2nd ec2-user has only read permission.

5. The last 'r--' means that other users who are not a file owner or the group also has read only permission.

6. No. '1' indicates it is the number of different names parts of this file…??? This file is direclty in the ec2-user file.

7. The first ec2-user is the owner of the file and the 2nd ec2-user is the group that file belongs to….?
8. 30 means that the size of the file which is calculated as bytes.
9. The date and time that the file was last modified.
10. And the last line is the name of the file in this directory.

```
drwxr-xr-x. 2 ec2-user ec2-user  6 Feb 20 15:38 linux_lesson
```

1. drwxr-xr-x indicates the permission for this directory…?
2. 'd' tells us this is a directory not a file.
3. 'rwx' is the owner of this directory which is ec-user, indicating read, write and execute permission. Execute means that we can go through the linux_lesson directory and access any sub-directory or file inside.
4. 'r-x' indicates the 2nd ec2-user group allowing them to read and access the directory and any file in it but cannot write and modify files in this directory.
5. 'r-x' means that other people who are not the owner of this directory or the group can read and execute but cannot modify files in this directory.
6. 2 means the number of hard links to this directory….? We have ec2-user directory and inside there is linux_lesson directory so we have two directories.

- In AWS EC2 environment, you might have to share files with different users or services so you should set right permissions to ensure that any sensitive file is accessed only throuhg authorized users or services.
- Any application running on EC2 that can access or generate files should be provided appropriate authorized permissions to ensure the security of AWS environment…?
- chmod means changing mode that changes the permission for files or directories. We can use chmod as symbolic mode and numeric mode.
- In the symbolic mode, chmod + means add permissions for files and chmod – means that remove permissions for files.
- In the numeric mode, 4 means read permission, 2 for write permission and 1 for execute permission.
- chmod 755 class_notes.txt provides read, write and excute permissions for the owner of the file and read and execute permission for the group and the others.

```
[ec2-user@ip-172-31-8-155 ~]$ chmod 755 class_notes.txt
```

```
[ec2-user@ip-172-31-8-155 ~]$ ls -l
total 8
-rwxr-xr-x. 1 ec2-user ec2-user 196 Aug 31 21:29 class_notes.txt
-rw-r--r--. 1 ec2-user ec2-user  81 Aug 31 20:02 my-file.txt
[ec2-user@ip-172-31-8-155 ~]$
```

- As the first line has been changed, '-rwx' indicates that read, write and execute permissions are provided for the owner of this file and r-x means that read and execute permission are provided for the group and the other users.
-chmod 444 class_notes.txt means that everyone has only read permission for this file.

```
[ec2-user@ip-172-31-8-155 ~]$ chmod 444 class_notes.txt
[ec2-user@ip-172-31-8-155 ~]$ ls -l
total 8
-r--r--r--. 1 ec2-user ec2-user 196 Aug 31 21:29 class_notes.txt
```

- chmod 644 class_notes.txt indicates that the owner can only read and write the file and other users and the group can read only. Here, 6 is the sum of both 4 (read) and 2 (write).

```
[ec2-user@ip-172-31-8-155 ~]$ chmod 644 class_notes.txt
[ec2-user@ip-172-31-8-155 ~]$ ls -l
total 8
-rw-r--r--. 1 ec2-user ec2-user 196 Aug 31 21:29 class_notes.txt
-rw-r--r--. 1 ec2-user ec2-user  81 Aug 31 20:02 my-file.txt
```

제 8강. Exercise – File Permissions

1. Start each task by reviewing the current permissions with ls -l.
2. After executing the commnad for each task, verify the changes with another ls -l.

Task 1: Making class_notes.txt Readable by Everyone

Objective: Ensure that everyone can read class_notes.txt, but no one can write to or execute it.

- Change the file permissions to read-only for all users using the numeric mode command.
- Verify the permissions using ls –l.

```
[ec2-user@ip-172-31-8-155 ~]$ ls -l
total 8
-rw-r--r--. 1 ec2-user ec2-user 196 Aug 31 21:29 class_notes.txt
-rw-r--r--. 1 ec2-user ec2-user  81 Aug 31 20:02 my-file.txt
[ec2-user@ip-172-31-8-155 ~]$ chmod 444 class_notes.txt
[ec2-user@ip-172-31-8-155 ~]$ ls -l
total 8
-r--r--r--. 1 ec2-user ec2-user 196 Aug 31 21:29 class_notes.txt
-rw-r--r--. 1 ec2-user ec2-user  81 Aug 31 20:02 my-file.txt
```

Task 2: Allowing the Owner to Write

Objective: Permit the file owner to edit to edit class_notes.txt, while keeping it readable by everyone.

- Apply the numeric mode command to modify the permissions.
- Verify the changes.

```
[ec2-user@ip-172-31-8-155 ~]$ chmod 644 class_notes.txt
[ec2-user@ip-172-31-8-155 ~]$ ls -l
total 8
-rw-r--r--. 1 ec2-user ec2-user 196 Aug 31 21:29 class_notes.txt
-rw-r--r--. 1 ec2-user ec2-user  81 Aug 31 20:02 my-file.txt
```

Task 3: Making class_notes.txt Executable by the owner

Objective: Allow the owner to run class_notes.txt as a script.

- Update the file permissions to allow execution by the owner.
- Verify the changes.

```
[ec2-user@ip-172-31-8-155 ~]$ ls -l
total 8
-rw-r--r--. 1 ec2-user ec2-user 196 Aug 31 21:29 class_notes.txt
-rw-r--r--. 1 ec2-user ec2-user  81 Aug 31 20:02 my-file.txt
[ec2-user@ip-172-31-8-155 ~]$ chmod 744 class_notes.txt
[ec2-user@ip-172-31-8-155 ~]$ ls -l
total 8
-rwxr--r--. 1 ec2-user ec2-user 196 Aug 31 21:29 class_notes.txt
-rw-r--r--. 1 ec2-user ec2-user  81 Aug 31 20:02 my-file.txt
```

Task 4: Adding Execute Permissions for the Group

Objective: Add execute permissions for the group to class_notes.txt.

- Use symbolic mode to add execute permissions for the group.
- Verify the changes.

```
[ec2-user@ip-172-31-8-155 ~]$ ls -l
total 8
-rwxr--r--. 1 ec2-user ec2-user 196 Aug 31 21:29 class_notes.txt
-rw-r--r--. 1 ec2-user ec2-user  81 Aug 31 20:02 my-file.txt
[ec2-user@ip-172-31-8-155 ~]$ chmod g+x class_notes.txt
[ec2-user@ip-172-31-8-155 ~]$ ls -l
total 8
-rwxr-xr--. 1 ec2-user ec2-user 196 Aug 31 21:29 class_notes.txt
-rw-r--r--. 1 ec2-user ec2-user  81 Aug 31 20:02 my-file.txt
```

- chmod g+x class_notes.txt
- g stands for the group
- x stands for the execute permission

Task 5. Revoking Write Access from the group

Objective: Ensure the group cannot modify class_notes.txt

- Remove write permission from the group.
- chmod g-w class_notes.txt
- Verify the changes.

```
[ec2-user@ip-172-31-8-155 ~]$ ls -l
total 8
-rwxr-xr--. 1 ec2-user ec2-user 196 Aug 31 21:29 class_notes.txt
-rw-r--r--. 1 ec2-user ec2-user  81 Aug 31 20:02 my-file.txt
[ec2-user@ip-172-31-8-155 ~]$ chmod 774 class_notes.txt
[ec2-user@ip-172-31-8-155 ~]$ ls -l
total 8
-rwxrwxr--. 1 ec2-user ec2-user 196 Aug 31 21:29 class_notes.txt
-rw-r--r--. 1 ec2-user ec2-user  81 Aug 31 20:02 my-file.txt
[ec2-user@ip-172-31-8-155 ~]$ chmod g-w class_notes.txt
[ec2-user@ip-172-31-8-155 ~]$ ls -l
total 8
-rwxr-xr--. 1 ec2-user ec2-user 196 Aug 31 21:29 class_notes.txt
-rw-r--r--. 1 ec2-user ec2-user  81 Aug 31 20:02 my-file.txt
[ec2-user@ip-172-31-8-155 ~]$
```

제 9강. Shutdown EC2