

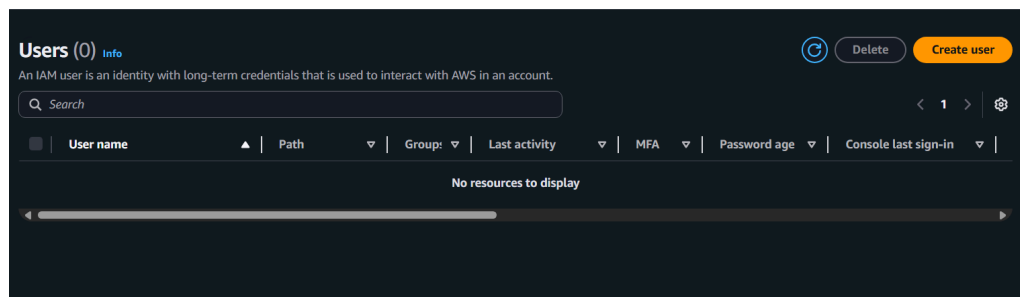


## Create an IAM user from root and connect CLI with root and IAM User

Date	@September 8, 2025
Multi-select	IAM Lab-1
Status	Done

### Create IAM User

- To create an IAM user, go to **IAM**, then under **Access Management**, click on **Users**, and select **Create user**.



- Provide the user details under **User details**. Enter the **User name** (in our case, `swinal-mlops`), then click **Next**.

Step 1: Specify user details

### Specify user details

**User details**

User name:

The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and +, -, @, \_ (hyphen).

☐ Provide user access to the AWS Management Console - optional  
If you're providing console access to a person, it's a [best practice](#) to manage their access in IAM Identity Center.

☒ If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. [Learn more](#)

[Cancel](#) [Next](#)

- Next, set permissions for the user. You have three options:

- Add user to group**

Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.

- Copy permissions**

Copy all group memberships, attached managed policies, and inline policies from an existing user.

- Attach policies directly**

Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

Here, we are creating a new policy, so we select **Attach policies directly** and attach the policy **AmazonS3FullAccess**. Then click **Next**.

Step 1: Specify user details

Step 2: Set permissions

### Set permissions

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

**Permissions options**

☐ Add user to group  
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.

☐ Copy permissions  
Copy all group memberships, attached managed policies, and inline policies from an existing user.

☒ Attach policies directly  
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

**Permissions policies (1/1400)**

Choose one or more policies to attach to your new user. [Create policy](#)

Search: S3 Filter by Type: All types 28 matches

Policy name	Type	Attached entities
<input type="checkbox"/> AmazonDMSRedshiftS3Role	AWS managed	0
<input checked="" type="checkbox"/> AmazonS3FullAccess	AWS managed	0

- Review the details, then click **Create user**.

Step 1: Specify user details

Step 2: Set permissions

Step 3: Review and create

### Review and create

Review your choices. After you create the user, you can view and download the autogenerated password, if enabled.

**User details**

User name: swinal-mlops Console password type: None Require password reset: No

**Permissions summary**

Name	Type	Used as
AmazonS3FullAccess	AWS managed	Permissions policy

**Tags - optional**

Tags are key-value pairs you can add to AWS resources to help identify, organize, or search for resources. Choose any tags you want to associate with this user.

No tags associated with the resource.

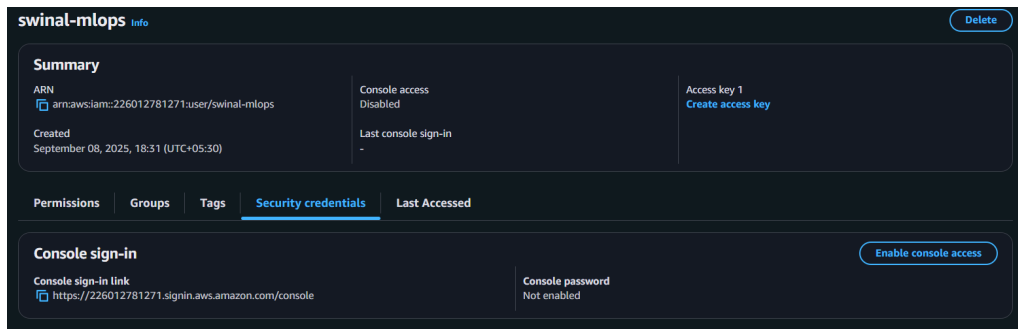
[Add new tag](#)

You can add up to 50 more tags.

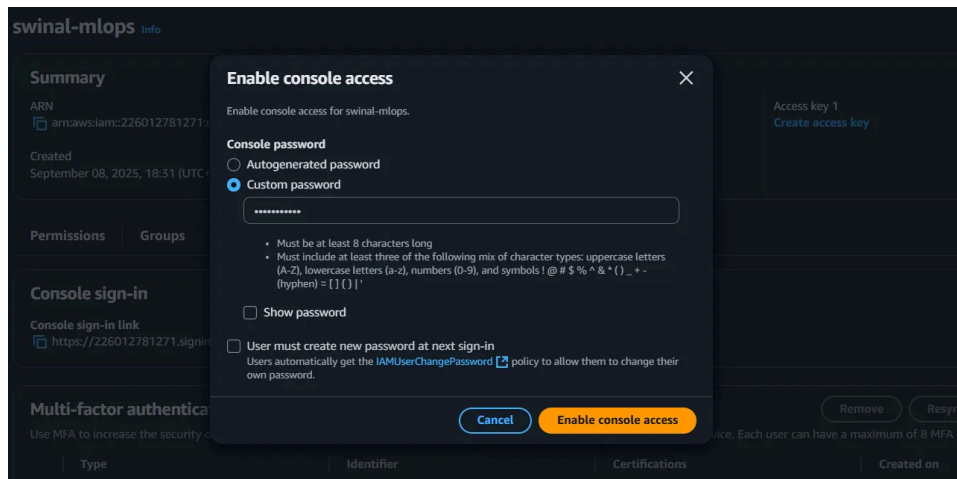
[Cancel](#) [Previous](#) [Create user](#)

## Users Console Access

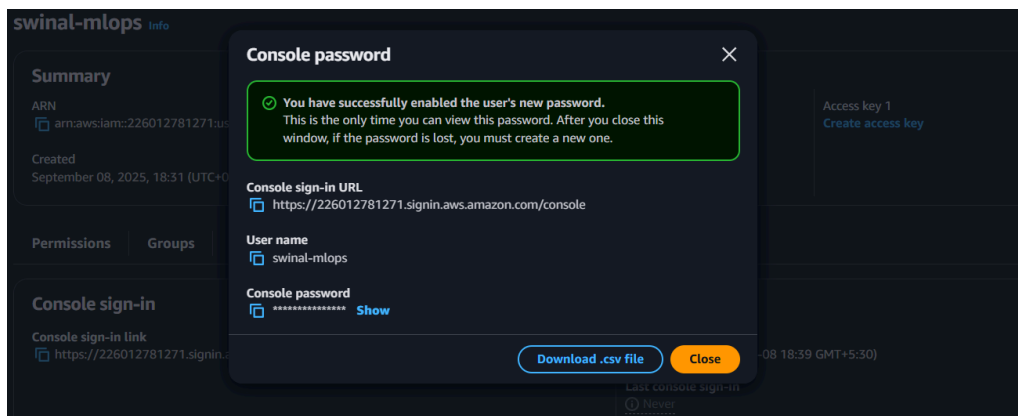
- Open the user **swinal-mlops**, go to the **Security Credentials** tab, and click **Enable console access**.



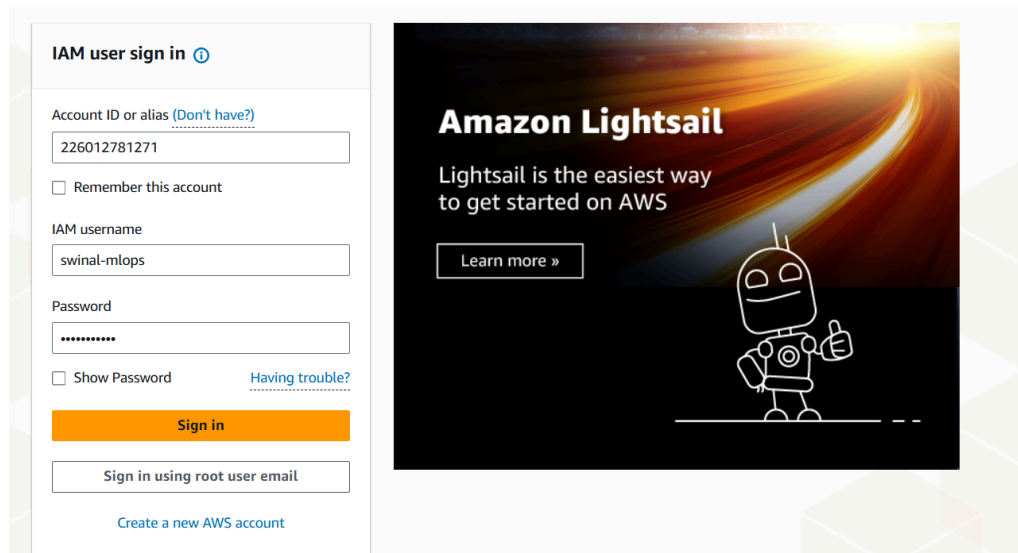
- Set a custom password for the user `swinal-mlops`, then click **Enable console access**.



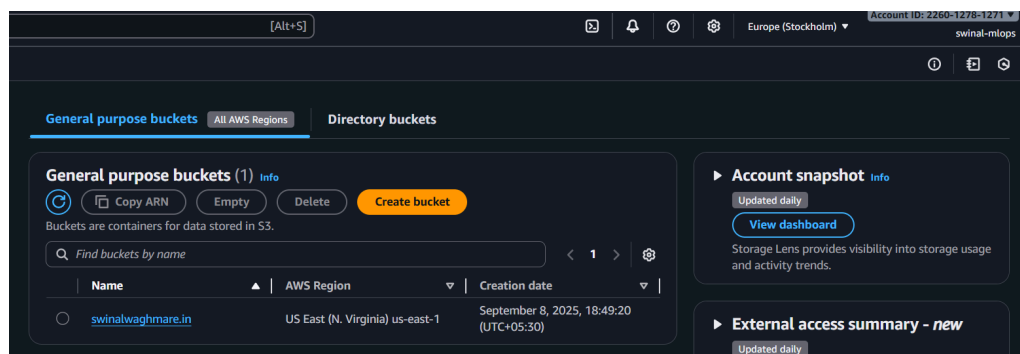
- Download the user credentials as a `.csv` file.



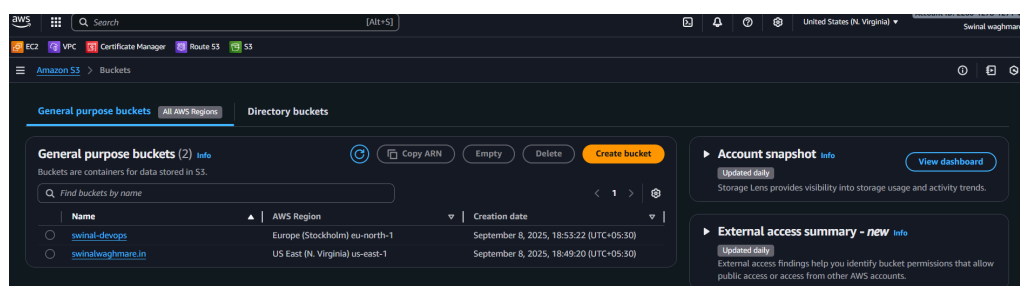
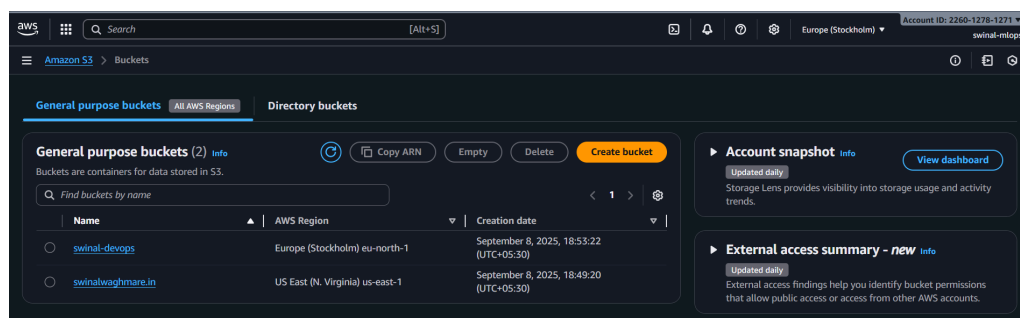
- Now, sign in as the IAM user `swinal-mlops`. Provide the **Account ID**, **IAM username**, and **Password**.
  - **Account ID** → URL account number
  - **IAM username** → `swinal-mlops`
  - **Password** → the custom password



- Since only S3 access is granted, the user can access only **Amazon S3**.



- Any changes made by the user will also reflect in the root account. For example, if the IAM user creates a new bucket ( `swinal-devops` ), it will be visible to the root user.



## Create Credentials for Root User and IAM User (CLI Access)

- To access AWS through the **Command Line Interface (CLI)**, first install the AWS CLI:

#### Installing or updating to the latest version of the AWS CLI - AWS Command Line Interface

The AWS CLI is an open source tool built using the AWS SDK for Python (Boto) that provides commands for interacting with AWS services. With minimal configuration, you can start using all of the functionality provided by the AWS Management Console from your favorite terminal program.

<https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html>

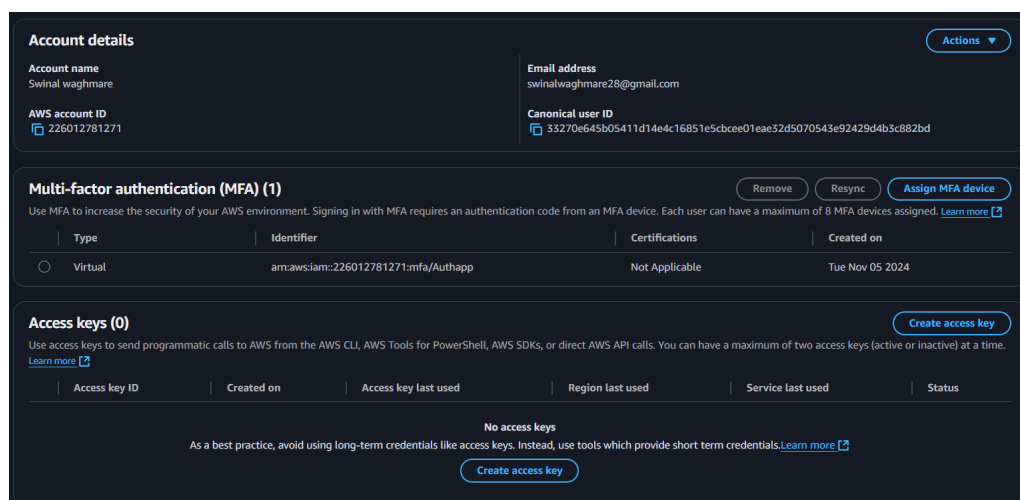
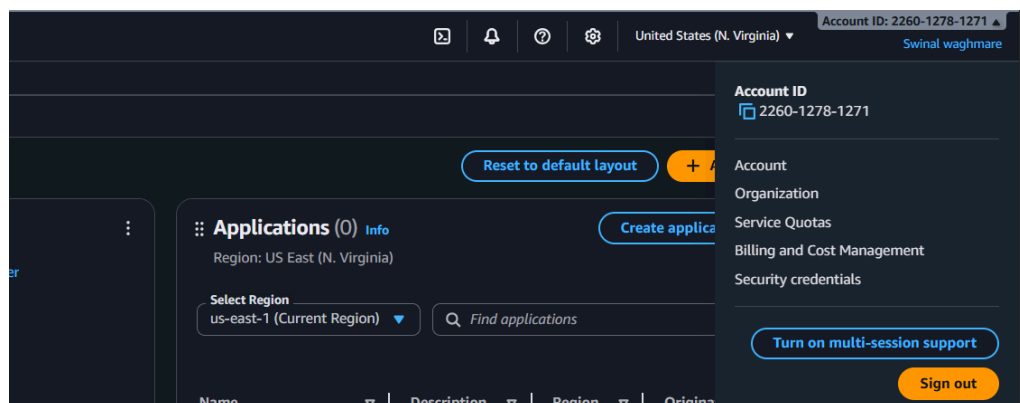


- Verify installation:

```
aws --version
```

```
ACER on ~
# aws --version
aws-cli/2.28.25 Python/3.13.7 Windows/11 exe/AMD64
ACER on ~
#
```

- First, create credentials for the **root user**.  
Go to the **root account**, click on your account ID, then select **Security credentials**.



- Create an **Access Key** and **Secret Key** by selecting **Create access key**.



### Root user access keys are not recommended

We don't recommend that you create root user access keys. Because you can't specify the root user in a permissions policy, you can't limit its permissions, which is a best practice.

Instead, use alternatives such as an IAM role or a user in IAM Identity Center, which provide temporary rather than long-term credentials. [Learn More](#)

If your use case requires an access key, create an IAM user with an access key and apply least privilege permissions for that user. [Learn More](#)

Step 1  
● Alternatives to root user access keys  
Step 2  
○ Retrieve access key

### Alternatives to root user access keys Info

**Root user access keys are not recommended**  
We don't recommend that you create root user access keys. Because you can't specify the root user in a permissions policy, you can't limit its permissions, which is a best practice.  
Instead, use alternatives such as an IAM role or a user in IAM Identity Center, which provide temporary rather than long-term credentials. [Learn More](#)  
If your use case requires an access key, create an IAM user with an access key and apply least privilege permissions for that user. [Learn More](#)

**Continue to create access key?**  
☒ I understand creating a root access key is not a best practice, but I still want to create one.

[Cancel](#) [Create access key](#)

- Download the keys, then click **Done**.

Step 1  
○ Alternatives to root user access keys  
Step 2  
● Retrieve access key

### Retrieve access key Info

**Access key**  
If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key	Secret access key
AKIATJH3JLLUEJH7F5H	***** <a href="#">Show</a>

**Access key best practices**

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the best practices for managing AWS access keys.

[Download .csv file](#) [Done](#)



**Note:** You can only create **two access keys per user** (applies to both root and IAM users).

- Configure the AWS CLI with root user credentials:

```
aws configure
```

```
AWS Access Key ID [*****QUQS]: AKIATJH3JLLUEJH7F5H
AWS Secret Access Key [*****kQBv]: x61ElkNiWoHE+zboo6oaaRiaFu6K7nfT5ky7RttM
Default region name [ap-southeast-2]: ap-south-1
Default output format [json]:
```

```

@ ACER on ~
# aws --version
aws-cli/2.28.25 Python/3.13.7 Windows/11 exe/AMD64
@ ACER on ~
# aws configure
AWS Access Key ID [*****QUQS]: AKIATJH3JLLUEJH7F5H
AWS Secret Access Key [*****kQBv]: x61EIkNiWoHE+zboo6oaaRiaFu6K7nfT5ky7RttM
Default region name [ap-southeast-2]: ap-south-1
Default output format [json]:
@ ACER on ~

```

- Verify S3 access (root user has admin privileges):

```
aws s3 ls
```

```

@ ACER on ~
# aws --version
aws-cli/2.28.25 Python/3.13.7 Windows/11 exe/AMD64
@ ACER on ~
# aws configure
AWS Access Key ID [*****QUQS]: AKIATJH3JLLUEJH7F5H
AWS Secret Access Key [*****kQBv]: x61EIkNiWoHE+zboo6oaaRiaFu6K7nfT5ky7RttM
Default region name [ap-southeast-2]: ap-south-1
Default output format [json]:
@ ACER on ~
# aws s3 ls
2025-09-08 18:53:24 swinal-devops
2025-09-08 18:49:21 swinalwaghmare.in
@ ACER on ~

```

- AWS CLI credentials are stored in the `.aws` folder:
  - `config` → region and output format
  - `credentials` → Access Key ID and Secret Access Key

```

C: > Users > ACER > .aws > config
1 [default]
2 region = ap-south-1
3 output = json
4

```

```

C: > Users > ACER > .aws > credentials
1 [default]
2 aws_access_key_id = AKIATJH3JLLUEJH7F5H
3 aws_secret_access_key = x61EIkNiWoHE+zboo6oaaRiaFu6K7nfT5ky7RttM
4

```

- Now create credentials for the IAM user `swinal-mlops` Go to the IAM user → **Security credentials** → **Create access key**

**swinal-mlops** Info Delete

**Summary**

ARN: `arn:aws:iam::226012781271:user/swinal-mlops`

Created: September 08, 2025, 18:31 (UTC+05:30)

Console access: Enabled without MFA

Last console sign-in: Today

Access key 1: Create access key

**Permissions** **Groups** **Tags** **Security credentials** **Last Accessed**

**Console sign-in** Manage console access

Console sign-in link: `https://226012781271.signin.aws.amazon.com/console`

Console password: Updated 43 minutes ago (2025-09-08 18:47 GMT+5:30)

Last console sign-in: 42 minutes ago (2025-09-08 18:48 GMT+5:30)

**Multi-factor authentication (MFA) (0)** Remove Resync Assign MFA device

Use MFA to increase the security of your AWS environment. Signing in with MFA requires an authentication code from an MFA device. Each user can have a maximum of 8 MFA devices assigned. [Learn more](#)

Type	Identifier	Certifications	Created on
No MFA devices. Assign an MFA device to improve the security of your AWS environment.			

Assign MFA device

**Access keys (0)** Create access key

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. [Learn more](#)

No access keys. As a best practice, avoid using long-term credentials like access keys. Instead, use tools which provide short term credentials. [Learn more](#)

Create access key

- Choose **Command Line Interface (CLI)** as the use case, then click **Next**.

**Step 1**

- Access key best practices & alternatives**
- Step 2 - optional: Set description tag
- Step 3: Retrieve access keys

**Access key best practices & alternatives** Info

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

**Use case**

- ☒ **Command Line Interface (CLI)**  
You plan to use this access key to enable the AWS CLI to access your AWS account.
- ☐ **Local code**  
You plan to use this access key to enable application code in a local development environment to access your AWS account.
- ☐ **Application running on an AWS compute service**  
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.
- ☐ **Third-party service**  
You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.
- ☐ **Application running outside AWS**  
You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.
- ☐ **Other**  
Your use case is not listed here.

**Alternatives recommended**

- Use [AWS CloudShell](#), a browser-based CLI, to run commands. [Learn more](#)
- Use the [AWS CLI V2](#) and enable authentication through a user in IAM Identity Center. [Learn more](#)

**Confirmation**

☒ I understand the above recommendation and want to proceed to create an access key.

Cancel Next

- Add a description, e.g., `s3-access`, then click **Create access key**.

**Step 1**

- Access key best practices & alternatives
- Step 2 - optional: Set description tag**
- Step 3: Retrieve access keys

**Set description tag - optional** Info

The description for this access key will be attached to this user as a tag and shown alongside the access key.

**Description tag value**

Describe the purpose of this access key and where it will be used. A good description will help you rotate this access key confidently later.

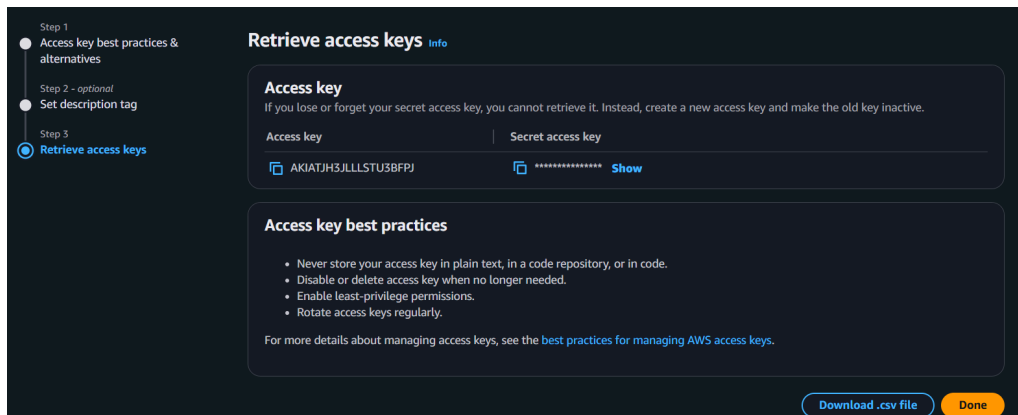
`s3-access`

Maximum 256 characters. Allowed characters are letters, numbers, spaces representable in UTF-8, and: `_.:/=+-@`

Cancel Previous Create access key

- Download the keys in `.csv` format and click **Done**.





- Configure IAM user CLI profile (to avoid overwriting root credentials):

```
aws configure --profile swinal-mlops
```

```
@ ACER on ~
# aws configure --profile swinal-mlops
AWS Access Key ID [None]: AKIATJH3JLLSTU3BFPJ
AWS Secret Access Key [None]: Ho2HAeQiVLRy/tOmCHzLFBnZ57gnYHbh5ej1BP0U
Default region name [None]: ap-south-1
Default output format [None]:
@ ACER on ~
# |
```

- Now the `.aws/credentials` file contains both users:

```
C: > Users > ACER > .aws > credentials
1 [default]
2 aws_access_key_id = AKIATJH3JLLUEJH7F5H
3 aws_secret_access_key = x61EIkNiWoHE+zboo6oaaRiaFu6K7nfT5ky7RttM
4 [swinal-mlops]
5 aws_access_key_id = AKIATJH3JLLSTU3BFPJ
6 aws_secret_access_key = Ho2HAeQiVLRy/tOmCHzLFBnZ57gnYHbh5ej1BP0U
7
```

- Verify IAM user permissions (only S3 access):

```
aws s3 ls --profile swinal-mlops
```

```
@ ACER on ~
# aws s3 ls --profile swinal-mlops
2025-09-08 18:53:24 swinal-devops
2025-09-08 18:49:21 swinalwaghmare.in
@ ACER on ~
# |
```

- Test EC2 access with **root user**:

```
aws ec2 describe-instances
```

```
# aws ec2 describe-instances
{
  "Reservations": [
    {
      "ReservationId": "r-0980349ac185e82f6",
      "OwnerId": "226012781271",
      "Groups": [],
      "Instances": [
        {
          "Architecture": "x86_64",
          "BlockDeviceMappings": [
            {
              "DeviceName": "/dev/xvda",
              "Ebs": {
                "AttachTime": "2025-09-08T14:18:00+00:00",
                "DeleteOnTermination": true,
                "Status": "attached",
                "VolumeId": "vol-033432b0b42e1d8f6"
              }
            }
          ],
          "ClientToken": "b74e4ac3-bdbd-498d-81ed-1b0f3e054ffa",
          "EbsOptimized": true,
          "EnaSupport": true,
          "Hypervisor": "xen",
          "NetworkInterfaces": [
            {
              "Association": {
                "IpOwnerId": "amazon",
                "PublicDnsName": "ec2-13-201-26-246.ap-south-1.compute.amazonaws.com",
                "PublicIp": "13.201.26.246"
              },
              "Attachment": {
                "AttachTime": "2025-09-08T14:18:00+00:00",

```

**Instance summary for i-004d5aa5f85d36d89 (sample-instance)** Info

Updated less than a minute ago

<b>Instance ID</b> i-004d5aa5f85d36d89	<b>Public IPv4 address</b> 13.201.26.246   <a href="#">open address</a>	<b>Private IPv4 addresses</b> 172.31.42.156
<b>IPv6 address</b> -	<b>Instance state</b> Running	<b>Public DNS</b> ec2-13-201-26-246.ap-south-1.compute.amazonaws.com   <a href="#">open address</a>
<b>Hostname type</b> IP name: ip-172-31-42-156.ap-south-1.compute.internal	<b>Private IP DNS name (IPv4 only)</b> ip-172-31-42-156.ap-south-1.compute.internal	<b>Elastic IP addresses</b> -
<b>Answer private resource DNS name</b> IPv4 (A)	<b>Instance type</b> t3.micro	<b>AWS Compute Optimizer finding</b> Opt-in to AWS Compute Optimizer for recommendations.   <a href="#">Learn more</a>
<b>Auto-assigned IP address</b> 13.201.26.246 [Public IP]	<b>VPC ID</b> vpc-04e69030a6e05c83c	<b>Auto Scaling Group name</b> -
<b>IAM Role</b> -	<b>Subnet ID</b> subnet-0ddd4221da6eb0ae	<b>Managed</b>
<b>IMDSv2</b> -	<b>Instance ARN</b>	

- Test EC2 access with **IAM user** ( `swinal-mlops` ):

```
aws ec2 describe-instances --profile swinal-mlops
```

→ Error (since IAM user only has S3 access).

```
© ACER on ~  
# aws ec2 describe-instances --profile swinal-mlops  
  
An error occurred (UnauthorizedOperation) when calling the DescribeInstances operation: You are not authorized  
to perform this operation. User: arn:aws:iam::226012781271:user/swinal-mlops is not authorized to perform: ec  
2:DescribeInstances because no identity-based policy allows the ec2:DescribeInstances action  
© ACER on ~  
#
```

✅ Lab Completed