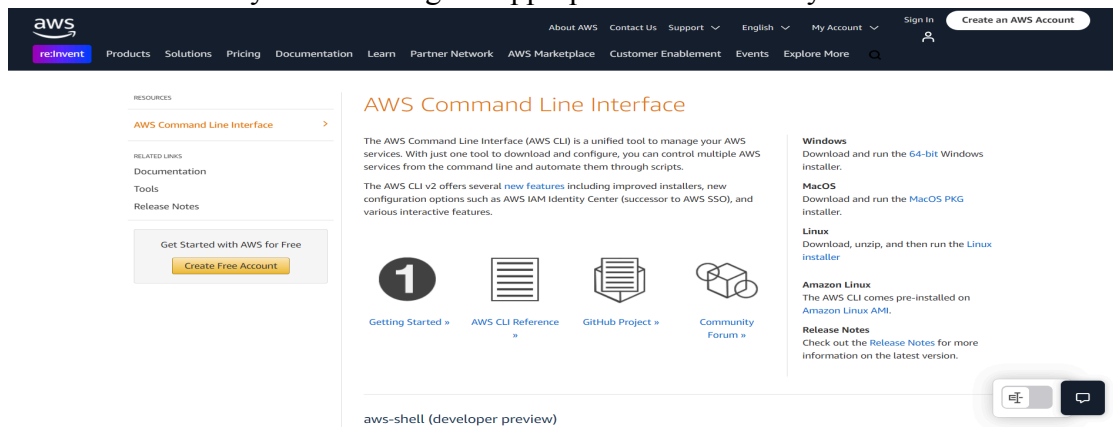


# Installing and Configuring AWS CLI

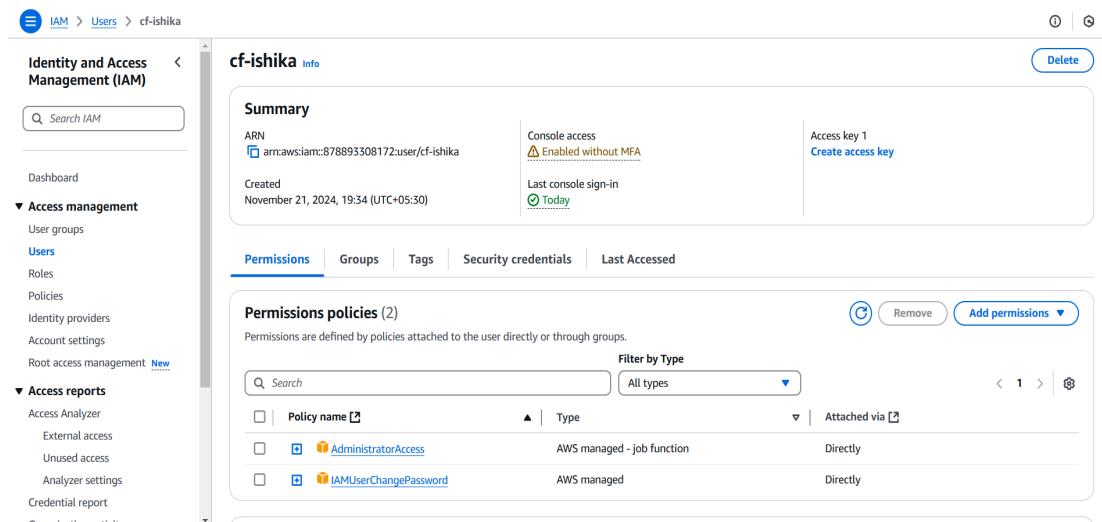
The process involves installing and configuring AWS CLI for use with AWS services. First, install Visual Studio Code and the AWS CLI appropriate for your operating system. In the AWS IAM Console, ensure the user has "AdministratorAccess" and create a new access key. Download the key as a .csv file. Open a terminal in Visual Studio Code, verify the AWS CLI installation with `aws --version`, and configure it using `aws configure` by providing the access key, secret key, and region. Test the configuration with `aws s3 ls` to ensure the AWS CLI is working. The end goal is to set up AWS CLI for stack management.

## Activity

1. Download and install Visual Studio Code for your operating system.
2. Install AWS CLI by downloading the appropriate installer for your OS.



3. Open the AWS IAM Console and ensure your user has "AdministratorAccess" permissions.



4. Navigate to the "Security credentials" tab, create a new access key, select the command line interface, tick the confirmation and download the key as a .csv file.

**Identity and Access Management (IAM)**

Search IAM

Dashboard

▼ Access management

- User groups
- Users
- Roles
- Policies
- Identity providers
- Account settings
- Root access management [New](#)

▼ Access reports

- Access Analyzer
- External access
- Unused access
- Analyzer settings
- Credential report
- Personalization dashboard

**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](#)

**SSH public keys for AWS CodeCommit (0)** [Actions](#) [Upload SSH public key](#)

Use SSH public keys to authenticate access to AWS CodeCommit repositories. You can have a maximum of five SSH public keys (active or inactive) at a time. [Learn more](#)

SSH Key ID	Uploaded	Status
No SSH public keys		

[Upload SSH public key](#)

**Use case**

☒ **Command Line Interface (CLI)**

You plan to use this access key to enable the AWS CLI to access your AWS account.

**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](#)

[Users](#) > [cf-ishika](#) > Create access key

**Access key created**

This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.

Step 1: Access key best practices & alternatives

Step 2 - optional: Set description tag

Step 3: **Retrieve access keys**

**Retrieve access keys** [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
<a href="#">AKIA4ZIQT7EGDOF7LSSW</a>	<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](#)

5. Open a terminal in Visual Studio Code or another terminal of your choice. Verify AWS CLI installation with the command `aws --version`.

```
PS C:\Users\Ishika> aws --version
aws-cli/2.22.12 Python/3.12.6 Windows/11 exe/AMD64
PS C:\Users\Ishika> aws configure
AWS Access Key ID [None]:
```

6. Configure AWS CLI using `aws configure`, providing the Access Key ID, Secret Access Key, region (eu-west-1 recommended), and leaving the default output format blank.

```
PS C:\Users\Ishika> aws --version
aws-cli/2.22.12 Python/3.12.6 Windows/11 exe/AMD64
PS C:\Users\Ishika> aws configure
AWS Access Key ID [*****LS5W]: AKIA4ZIQ7TEGDOF7LS5W
AWS Secret Access Key [*****TXzt]: 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
Default region name [eu-west-1]: eu-west-1
Default output format [None]:
PS C:\Users\Ishika> aws s3 ls
2024-11-21 01:12:20 aws500demotest0911
2024-12-01 10:51:22 awsbackupfeaturerds1
2024-11-21 01:12:27 awsbucketdkrecredemo
2024-11-21 01:12:42 awsdemoapplatest0910
2024-11-21 01:12:42 awsdemocloudformation11
2024-11-21 01:12:42 awsdeploymentconfigbucket1
2024-12-01 20:31:06 awsfiletransferdemo1
2024-11-29 07:44:43 awspurpledemobucket1
2024-11-21 01:13:12 awssimplilearnd1demo0911
2024-11-23 15:19:59 bucket-1-general
2024-11-21 01:37:38 cf-templates-kmi81w4ukk76-ap-south-1
2024-11-28 14:57:14 cf-templates-kmi81w4ukk76-eu-west-1
2024-11-29 10:10:14 nestedstackbuckets3
2024-12-06 21:51:07 riteshrotarys3cli
2024-11-19 16:29:17 s3-bucket-12010321
2024-12-01 20:03:00 s3demosgbucketdemo
PS C:\Users\Ishika> 
```

7. Test the configuration by running `aws s3 ls` to list available S3 buckets.
8. Ensure the AWS CLI is working correctly before proceeding to create stacks.