



## **Placement Empowerment Program**

### ***Cloud Computing and DevOps Centre***

Use Cloud CLI Tools: Install the CLI for your cloud provider (e.g., AWS CLI). Use it to list resources, upload files to storage, and manage VMs.

**Name:** Jayadasan S  
**Department:** CSE

## **Introduction:**

The Cloud CLI Tools, such as the AWS Command Line Interface (CLI), empower users to interact with cloud services programmatically and efficiently. AWS CLI simplifies managing resources like storage (S3), compute (EC2), and other AWS services directly from the command line. This Proof of Concept (POC) demonstrates how to use AWS CLI on Windows to perform essential tasks like listing resources, uploading files, and managing virtual machines (VMs).

## **Overview:**

This POC is designed to showcase the capabilities of AWS CLI for managing cloud resources seamlessly. By the end of this POC, the user will have hands-on experience with:

1. Installing and configuring AWS CLI on a Windows system.
2. Performing basic operations such as listing S3 buckets and EC2 instances.

3. Uploading files to S3 storage.

4. Managing EC2 instances by starting, stopping, and terminating them.

This step-by-step process helps users understand how CLI tools can be leveraged to streamline cloud operations without relying on the AWS Management Console.

## Objectives:

The primary objectives of this POC are:

**1. Install and Configure AWS CLI:** Set up the AWS CLI on a Windows system and configure it using Access Keys.

**2. List Cloud Resources:** Learn how to retrieve details about AWS resources such as S3 buckets and EC2 instances using CLI commands.

**3. Upload Files to S3:** Upload a file from a local system to an S3 bucket using CLI commands.

**4. Manage Virtual Machines (EC2):** Start, stop, and terminate EC2 instances directly from the command line.

**5. Clean Up Resources:** Remove unused files and instances to avoid unnecessary costs.

## Importance:

The importance of this POC lies in its practicality and relevance in real-world cloud management scenarios:

- 1. Efficiency and Automation:** The AWS CLI allows users to automate tasks and manage resources more efficiently than through the web console.
- 2. Skill Development:** Hands-on experience with the CLI enhances problem-solving skills and prepares users for advanced cloud roles such as Cloud Engineer or DevOps Specialist.
- 3. Cost Optimization:** Managing resources through CLI ensures better control over resource utilization, reducing the risk of incurring unnecessary charges.

## Step-by-Step Overview:

### Step 1:

Search for **aws cli for windows** and click the first link to proceed.

Amazon AWS Documentation  
<https://docs.aws.amazon.com/cli/getting-started-install>

### Installing or updating to the latest version of the AWS CLI

This topic describes how to install or update the latest release of the AWS Command Line Interface (AWS CLI) on supported operating systems.

[Install a past release](#) · [Setting up the AWS CLI](#) · [macOS](#) · [Linux](#)

Amazon Web Services  
<https://aws.amazon.com/cli>

### AWS CLI

With just one tool to download and configure, you can control multiple AWS services from the command line and automate them through scripts. Get started.

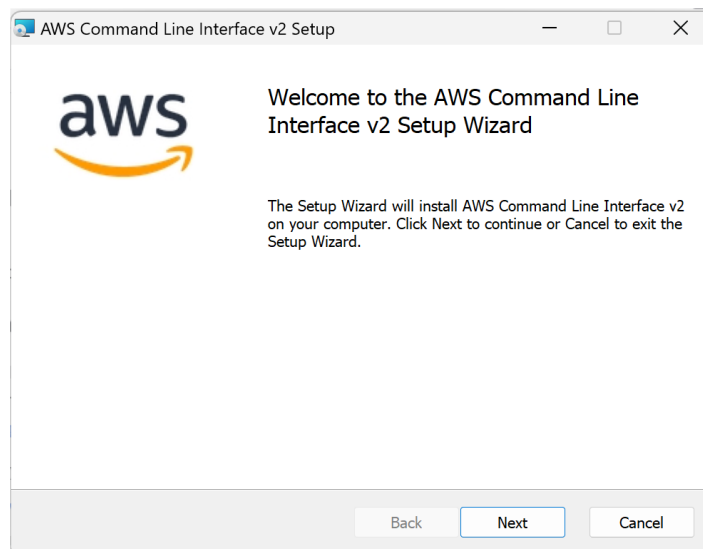
## Step 2:

In that page for windows click the download link it will start downloading.

The screenshot shows the AWS Command Line Interface documentation page for Windows. The left sidebar contains a navigation menu with links like 'About the AWS CLI', 'Get started', 'Prerequisites', 'Install/Update', 'Past releases', 'Build and install from source', 'Amazon ECR Public/Docker Setup', 'Configure the AWS CLI', 'Authentication and access credentials', 'Using the AWS CLI', 'Code examples', 'Security', 'Troubleshoot errors', 'Migration guide', 'Uninstall', and 'Document History'. The main content area is titled 'Windows' and includes sections for 'Install and update requirements' (listing support for 64-bit Windows and admin rights), 'Install or update the AWS CLI' (explaining the update process and providing a link to the AWS CLI version 2 Changelog on GitHub), and a numbered list of steps. Step 1 instructs to download and run the AWS CLI MSI installer for Windows (64-bit), providing a link to <https://awscli.amazonaws.com/AWSCLIV2.msi> and an alternative command to run the MSI installer: `C:\> msexec.exe /i https://awscli.amazonaws.com/AWSCLIV2.msi`. Step 2 instructs to confirm the installation by opening the Start menu, searching for cmd to open a command prompt window, and using the `aws --version` command. A right sidebar contains 'On this page' links (AWS CLI install and update instructions, Troubleshooting AWS CLI install and uninstall errors, Next steps) and a 'Recently added to this guide' section with a link to 'Introducing Amazon Q'.

## Step 3:

Follow up the installation wizard.



## Step 4:

Open **Command Prompt** and Type the following command and press Enter:

**aws --version**

If this appears, the AWS CLI is installed successfully.

```
C:\Users\Hi>aws --version
aws-cli/2.23.11 Python/3.12.6 Windows/11 exe/AMD64
```

## Step 4:

**Configure the AWS CLI**

Now that the AWS CLI is installed, we need to configure it with your AWS account credentials so that you can interact with AWS services like S3 and EC2.

**1. Open Command Prompt**

**2. Run the Configuration Command:**

In the command prompt, type the following command and press Enter:

```
aws configure
```

**3. Enter Your AWS Credentials:** This will prompt you for four pieces of information:

**AWS Access Key ID:**

You need to get this from your AWS account.

**How to Get It:**

1. Go to the [AWS IAM Console](#).
2. Click on **Users** on the left-hand menu.
3. Click on your username (or create a new user if needed).
  - > Under the **Security Credentials** tab, find the **Access Keys** section.
  - > Click **Create New Access Key**. It will give you an **Access Key ID** and

**Secret Access Key.** Keep this information safe!

-> Copy the **Access Key ID** and paste it when prompted.

#### 4. AWS Secret Access Key:

This is the **Secret Access Key** that is shown when you create a new access key.

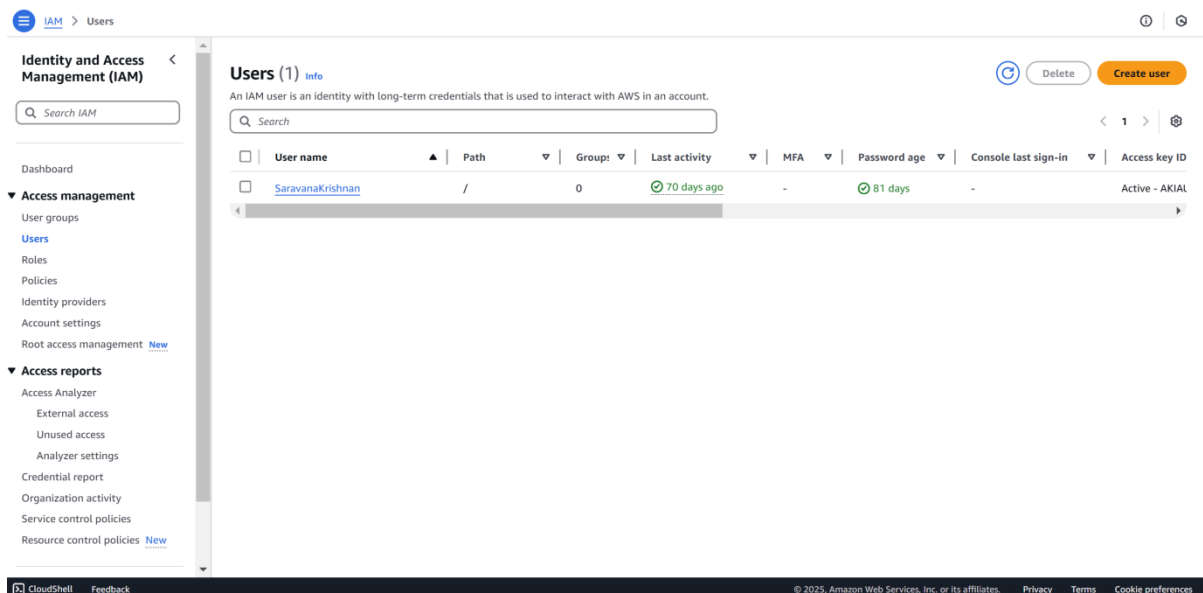
Paste this when prompted in the CLI.

#### 5. Default Region Name:

You can enter a region code like us-east-1

#### 6. Default Output Format:

Choose json



### Step 5:

### List AWS Resources Using the AWS CLI



S3 (Simple Storage Service) is where you store files in AWS. To list your S3 buckets, follow these steps:

**1. List S3 Buckets:**

Run the following command in Command Prompt

**aws s3 ls**

**2. Expected Result:**

If you have any S3 buckets, they will appear as a list.

If you don't have any, you might see an empty output like this:

A new bucket will be listed here when created.

This command helps you verify the S3 storage you currently have.

```
C:\Users\Hi>aws s3 ls
```

**Step 6:**

Launch an Ec2 instance in the console and run the following command:

**aws ec2 describe-instances --query  
"Reservations[].Instances[].{ID:InstanceId,Stat**

```
e:State.Name,Type:InstanceType,Name:Tags[?
Key=='Name'].Value | [0]}"
```

Expected Result:

This will display details about your EC2 instances.

## Step 7:

### Upload Files to S3 Using AWS CLI

If you don't already have an S3 bucket, create a new one.

Run this command to upload a file (replace file.txt with the path to the file you want to upload, and my-unique-bucket-name with your bucket name).  
Note : The whatever the file u want to store it must be in separate folder.

```
aws s3 cp C:\path\to\your\file.txt s3://my-
unique-bucket-name/ --recursive
```

## Step 8:

To check if the file is uploaded successfully, you can list the contents of your S3 bucket:

```
aws s3 ls s3://my-unique-bucket-name/
```

## Step 9:

Run the following command to list your EC2 instances, including their ID, state, type, and name (if tagged with "Name"):

```
aws ec2 describe-instances --query  
"Reservations[].Instances[].{ID:InstanceId,State:State.Name,Type:InstanceType,Name:Tags[?Key=='Name'].Value | [0]}"
```

```
C:\Users\Hi>aws ec2 describe-instances --query "Reservations[].Instances[].{ID:InstanceId,State:State.Name,Type:InstanceType,Name:Tags[?Key=='Name'].Value | [0]}"  
[  
  {  
    "ID": "i-09833c9ddf983644f",  
    "State": "running",  
    "Type": "t2.micro",  
    "Name": "Saravanan Web Server"  
  }  
]
```

## Step 10:

Now Stop the Ec2 instance you have created . If you have any stopped EC2 instances and want to start them, use this command. Replace <instance-id> with the ID of the instance you want to start (you can get it from the previous command's output):

```
aws ec2 start-instances --instance-ids  
<instance-id>
```

```
C:\Users\Hi>aws ec2 start-instances --instance-ids i-09833c9ddf983644f
{
  "StartingInstances": [
    {
      "InstanceId": "i-09833c9ddf983644f",
      "CurrentState": {
        "Code": 0,
        "Name": "pending"
      },
      "PreviousState": {
        "Code": 80,
        "Name": "stopped"
      }
    }
  ]
}
```

## Step 11:

You can verify the state of your EC2 instance (whether it's running or stopped) by using the following command:

```
aws ec2 describe-instances --instance-ids  
<instance-id> --query  
"Reservations[].Instances[].State.Name"
```

```
C:\Users\Hi>aws ec2 describe-instances --instance-ids i-09833c9ddf983644f --query "Reservations[].Instances[].State.Name"
[
  "running"
]
```

## Step 12:

To stop an EC2 instance, use this command. Again, replace <instance-id> with the actual instance ID:

**aws ec2 stop-instances --instance-ids  
<instance-id>**

```
C:\Users\Hi>aws ec2 stop-instances --instance-ids i-09833c9ddf983644f
{
  "StoppingInstances": [
    {
      "InstanceId": "i-09833c9ddf983644f",
      "CurrentState": {
        "Code": 64,
        "Name": "stopping"
      },
      "PreviousState": {
        "Code": 16,
        "Name": "running"
      }
    }
  ]
}
```

### Step 13:

Verify the state of your EC2 instance (whether it's running or stopped) by using the following command:

**aws ec2 describe-instances --instance-ids  
<instance-id> --query  
"Reservations[].Instances[].State.Name"**

```
C:\Users\Hi>aws ec2 describe-instances --instance-ids i-09833c9ddf983644f --query "Reservations[].Instances[].State.Name"
[
  "stopped"
]
```

## **Step 14:**

### **Clean Up Resources**

- **Delete a Single File:**

```
aws s3 rm s3://my-unique-bucket-name/file.pdf
```

Replace file.pdf with the actual name of the file you want to delete.

- **Delete All Files in a Bucket** (if you want to delete everything):

```
aws s3 rm s3://my-unique-bucket-name/--recursive
```

## **Step 15:**

Once the files are deleted, you can delete the bucket itself.

- **Delete an S3 Bucket:**

```
aws s3 rb s3://my-unique-bucket-name/ --force
```

Then verify it by:

```
aws s3 ls
```

It returns nothing so it means there is no bucket (deleted)

```
C:\Users\Hi>aws s3 ls
```

## Step 16:

If you no longer need the EC2 instances, you can terminate them to avoid ongoing charges.

**aws ec2 terminate-instances --instance-ids  
<instance-id>**

Then verify it by:

**aws ec2 describe-instances --query  
"Reservations[].Instances[].{ID:InstanceId,State:State.Name}"**

```
C:\Users\Hi>aws ec2 terminate-instances --instance-ids i-09833c9ddf983644f
{
  "TerminatingInstances": [
    {
      "InstanceId": "i-09833c9ddf983644f",
      "CurrentState": {
        "Code": 48,
        "Name": "terminated"
      },
      "PreviousState": {
        "Code": 80,
        "Name": "stopped"
      }
    }
  ]
}

C:\Users\Hi>aws ec2 describe-instances --query "Reservations[].Instances[].{ID:InstanceId,State:State.Name}"
[
  {
    "ID": "i-09833c9ddf983644f",
    "State": "terminated"
  }
]
```

That's the final step in this POC, You've successfully:

1. Installed and configured the AWS CLI.
2. Listed, uploaded files, and managed S3 and EC2 resources.
3. Cleaned up resources to avoid charges.

## **Outcomes:**

By completing this PoC of using AWS CLI tools, you will:

- 1. Install and Configure AWS CLI:** Successfully set up and authenticate AWS CLI on a Windows system for seamless interaction with AWS services.
- 2. List and Retrieve Cloud Resources:** Use CLI commands to list S3 buckets, EC2 instances, and other resources to understand their states and configurations.



**3. Upload Files to S3:** Demonstrate efficient file transfer to an S3 bucket from a local machine using AWS CLI commands.

**4. Manage EC2 Instances:** Programmatically start, stop, and terminate EC2 instances, showcasing practical virtual machine management skills.

**5. Optimize Cloud Resources:** Clean up resources like S3 buckets and EC2 instances to avoid unnecessary costs while maintaining a clutter-free AWS environment.