

# Placement Empowerment Program

## *Cloud Computing and DevOps Centre*

*Write a Shell Script to Manage Cloud Resources: Create a script to launch, stop, and terminate cloud VMs using the CLI.*

Name: Raichal Maria P

Department: CSE

## Introduction:

- This POC focuses on creating a shell script to automate the management of cloud virtual machines (VMs) using a Command-Line Interface (CLI). By leveraging this script, users can easily launch, stop, and terminate VMs, streamlining operations and reducing manual effort. This approach not only saves time but also ensures consistency and reliability in managing cloud infrastructures.

## Overview:

- The shell script is designed to automate cloud virtual machine (VM) management, enabling efficient interaction with cloud providers using the CLI.
- It simplifies routine cloud operations by providing options to launch, stop, and terminate VMs with minimal manual intervention.
- By automating repetitive tasks, the script reduces the risk of human error and saves time, ensuring reliable cloud resource management.
- This script is adaptable to various cloud platforms and can be integrated into larger automation workflows for managing complex infrastructures.

## Objectives:

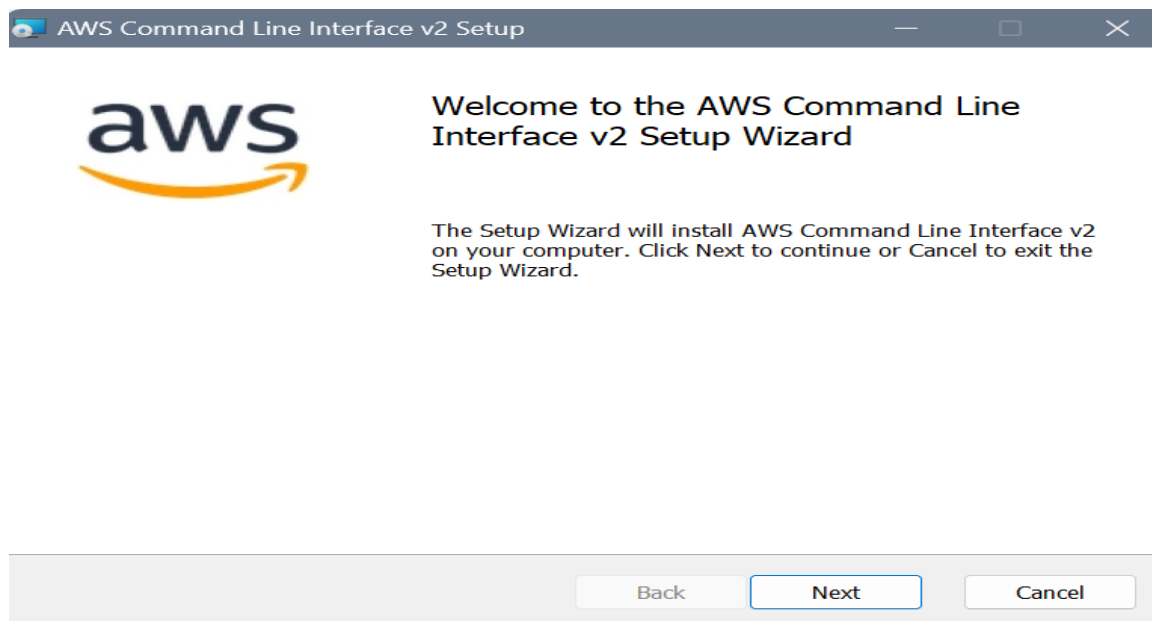
The key objectives of this task are:

- ❖ **Purpose:** Automate the management of cloud virtual machines (VMs) by creating a shell script that interacts with the cloud provider's CLI.
- ❖ **Functionality:** The script will enable users to perform key actions such as launching, stopping, and terminating VMs, simplifying routine cloud operations.
- ❖ **Efficiency:** Reduces manual effort and errors by automating repetitive tasks, ensuring consistency in managing cloud resources.
- ❖ **Scalability:** Provides a reusable and customizable solution that can be adapted to different cloud environments or integrated into larger automation workflows.

## Step-by-Step Overview:

### 1. Set Up AWS CLI

- ☐ **Install AWS CLI:** Follow the installation guide for your operating system.
- ☐ **Configure AWS CLI:** Run the following command and enter your AWS credentials:
  - First create an EC2 Instance in your console
  - You will need your AWS Access Key ID, Secret Access Key, region, and output format



### 2. Set up your AWS CLI with your credentials

You'll be prompted to enter your:

- AWS Access Key ID
- AWS Secret Access Key
- Default region name
- Default output format (e.g., json)

```
PS C:\Users\pooja\OneDrive\Desktop\Scripts> aws configure
AWS Access Key ID [*****M7OJ]: AKIA2MNVLVVF7K4MM7OJ
AWS Secret Access Key [*****SI20]: Vkk7DU 02xxA5eC +1iTBtkWA00 Fy3KPsS 0
Default region name [ap-south-1]: ap-south-1
Default output format [None]: json
```

### 3. Run Instance:

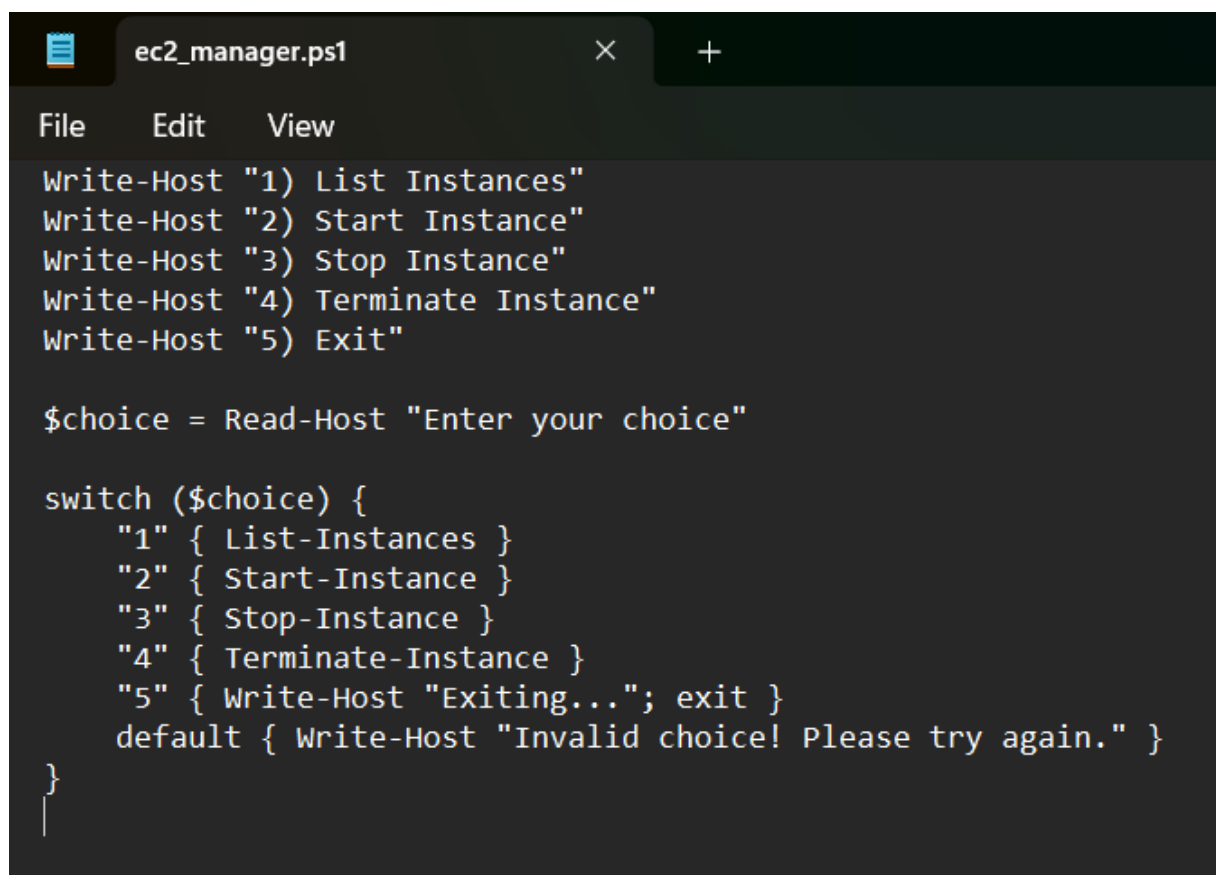
Now run the instance in the terminal by inputting the configurations of your instance

```
PS C:\Users\pooja\OneDrive\Desktop\Scripts> aws ec2 run-instances --image-id ami-0c50b6f7dc3701ddd --count 1 --instance-type t2.micro --key-name poo-key --security-group-ids sg-0102d89eacd83db9d --subnet-id subnet-009b250320823406e
{
  "ReservationId": "r-03fb5552defcacc60",
  "OwnerId": "713881791819",
  "Groups": [],
  "Instances": [
    {
      "Architecture": "x86_64",
      "BlockDeviceMappings": [],
      "ClientToken": "61221340-c906-465b-9d01-d2497ed4bf9e",
      "EbsOptimized": false,
      "EnaSupport": true,
```

### 3. Write the script

Now, write the following script in a notepad and save it as 'ec2\_manager.ps1'.

Then, save this file in a folder named script.



```
File Edit View
Write-Host "1) List Instances"
Write-Host "2) Start Instance"
Write-Host "3) Stop Instance"
Write-Host "4) Terminate Instance"
Write-Host "5) Exit"

$choice = Read-Host "Enter your choice"

switch ($choice) {
    "1" { List-Instances }
    "2" { Start-Instance }
    "3" { Stop-Instance }
    "4" { Terminate-Instance }
    "5" { Write-Host "Exiting..."; exit }
    default { Write-Host "Invalid choice! Please try again." }
}
```

### 4. Run the script

Now, right click on the file and click on run with powershell.

Now, the script is executed in a powershell window.

Select the required option and give the instance id of the ec2 instance you previously created.

```
Windows PowerShell
Select an option:
1) List Instances
2) Start Instance
3) Stop Instance
4) Terminate Instance
5) Exit
Enter your choice: 4
Enter Instance ID to terminate: i-0f6c105434829ec0a|
```

## 5. Verifying

Now, check the console to verify if the following actions are being performed on your given instance.

The screenshot displays the AWS Management Console interface for an EC2 instance. The left sidebar shows the navigation menu with 'EC2' selected. The main content area shows the 'Instance summary for i-027c56447af3fbae7 (Ec2-poo)'. The instance state is 'Shutting-down'. The summary includes details such as Instance ID, Public IPv4 address, Private IPv4 addresses, Instance state, Hostname type, Answer private resource DNS name, Auto-assigned IP address, VPC ID, and AWS Compute Optimizer finding.

Instance ID	Public IPv4 address	Private IPv4 addresses
i-027c56447af3fbae7	43.204.149.182   <a href="#">open address</a>	172.31.0.52

Instance state	Public IPv4 DNS
Shutting-down	ec2-43-204-149-182.ap-south-1.compute.amazonaws.com   <a href="#">open address</a>

Hostname type	Private IP DNS name (IPv4 only)
IP name: ip-172-31-0-52.ap-south-1.compute.internal	ip-172-31-0-52.ap-south-1.compute.internal

Answer private resource DNS name	Instance type
IPv4 (A)	t2.micro

Auto-assigned IP address	VPC ID
43.204.149.182   <a href="#">Public IP</a>	vpc-04b7b73f8d2523b6c   <a href="#">VPC</a>

Elastic IP addresses	AWS Compute Optimizer finding
-	<a href="#">Click to view AWS Compute Optimizer finding</a>

**Outcome :**

By completing this Proof of Concept (PoC) to write a Shell Script to Manage Cloud Resources

**1. Improved Automation**

- Eliminates manual steps for VM management.
- Saves time by executing predefined commands automatically.

**2. Efficient Resource Management**

- Start or stop VMs based on usage.
- Resize or reallocate resources dynamically.

**3. Enhanced Monitoring**

- Regularly check VM health, disk usage, CPU, and memory utilization.
- Send alerts when thresholds are exceeded.

**4. Security and Access Control**

- Implement SSH key-based access.
- Automate security updates and backups.

**5. Cost Optimization**

- Automatically stop unused VMs to save cloud costs.
- Schedule VM operations based on workload demands

