

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: SHALINI D

Department: IT

Introduction:

Managing cloud resources effectively is essential for optimizing costs and ensuring your infrastructure scales according to demand. By creating a shell script, you can automate the management of your cloud virtual machines (VMs), saving time and reducing the risk of human error. This script will leverage the Command Line Interface (CLI) provided by your cloud service provider to perform essential tasks such as launching, stopping, and terminating VMs.

Objectives:

The primary objective of this shell script is to provide a convenient and automated way to manage cloud VMs using the CLI. By the end of this tutorial, you will have a script that can:

Launch new cloud VMs based on specified parameters.

Stop running VMs to save on costs when they are not in use.

Terminate VMs that are no longer needed to free up resources.

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)

```
AWS Access Key ID [*****M7OJ]: AKIA2MNVLVWF7K4MM7OJ
AWS Secret Access Key [*****SI2O]: Vkk7DU 02xxA5eC +1iTBtkWA00 Fy3KPsS 0
Default region name [ap-south-1]: ap-south-1
Default output format [None]: json
```

3. Run Instance:

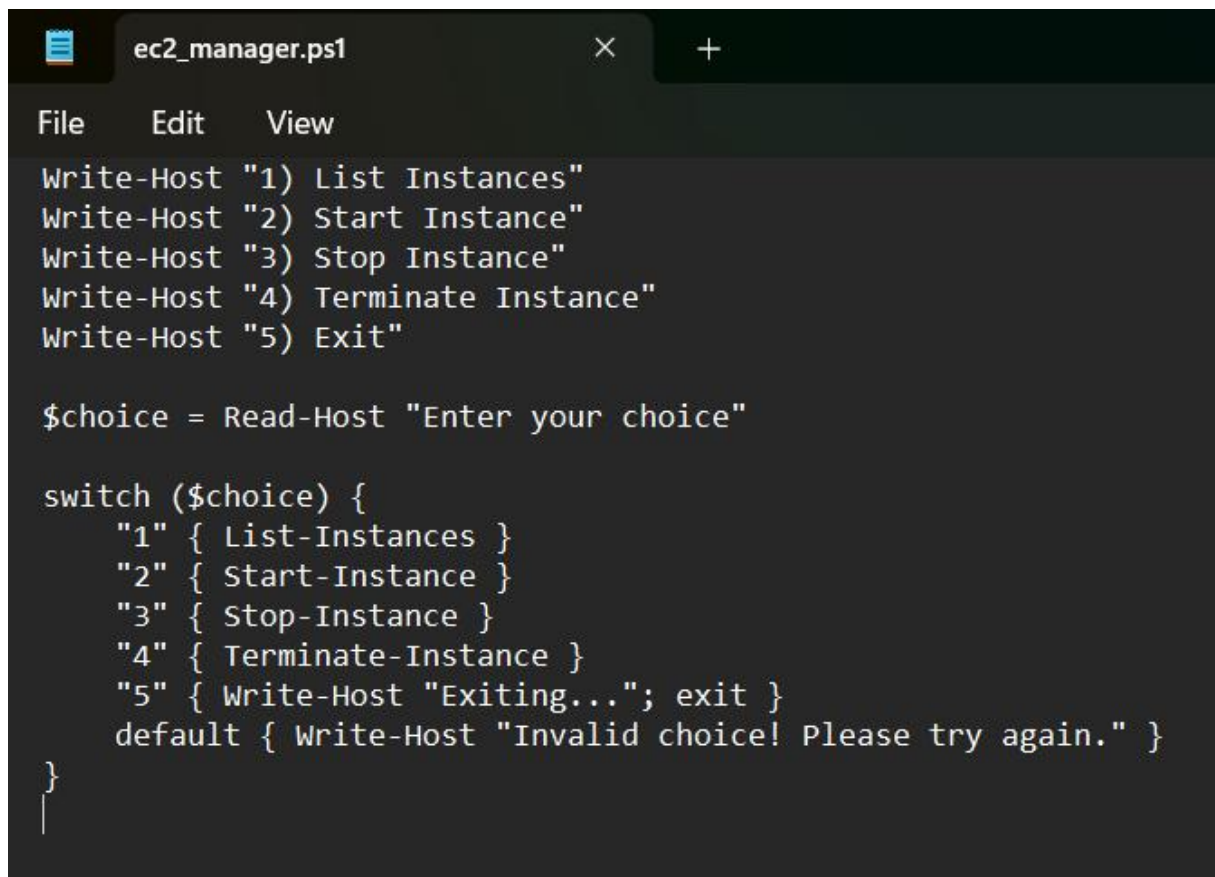
Now run the instance in the terminal by inputting the configurations of your instance

```
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.

A screenshot of a code editor window titled 'ec2_manager.ps1'. The editor has a dark theme and shows a PowerShell script. The script lists five options: 1) List Instances, 2) Start Instance, 3) Stop Instance, 4) Terminate Instance, and 5) Exit. It then prompts the user to enter a choice and uses a switch statement to execute the corresponding action. The script is as follows:

```
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"

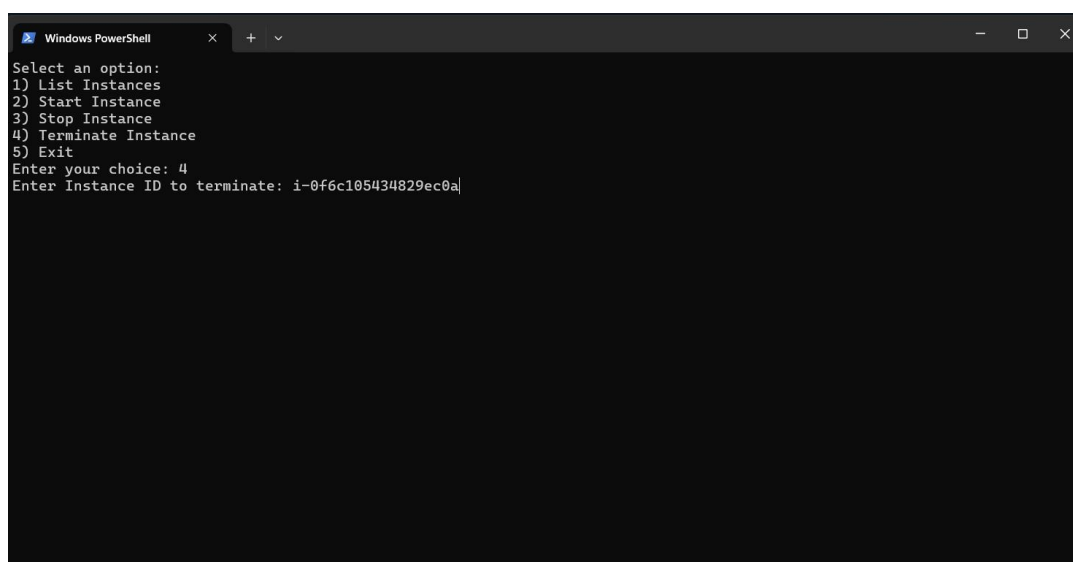
switch ($choice) {
    "1" { List-Instances }
    "2" { Start-Instance }
    "3" { Stop-Instance }
    "4" { Terminate-Instance }
    "5" { Write-Host "Exiting..."; exit }
    default { Write-Host "Invalid choice! Please try again." }
}
```

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.

A screenshot of a Windows PowerShell window. The window title is 'Windows PowerShell'. The output of the script is displayed, showing the menu options and the user's input. The user has entered '4' for the choice and 'i-0f6c105434829ec0a' for the instance ID to terminate. The output is as follows:

```
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 shows the AWS Management Console interface for an EC2 instance. The left sidebar contains navigation links for EC2, Dashboard, EC2 Global View, Events, Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, and Images. The main content area displays the 'Instance summary for i-027c56447af3fbae7 (Ec2-poo)'. The instance state is 'Shutting-down'. The public IPv4 address is 43.204.149.182. The private IPv4 address is 172.31.0.52. The public IPv4 DNS is ec2-43-204-149-182.ap-south-1.compute.amazonaws.com. The private IP DNS name (IPv4 only) is ip-172-31-0-52.ap-south-1.compute.internal. The instance type is t2.micro. The VPC ID is vpc-04b7b73f8d2523b6c. The auto-assigned IP address is 43.204.149.182. The AWS Compute Optimizer finding is 'Opt-in to AWS Compute Optimizer for recommendations'.

Instance ID	Public IPv4 address	Private IPv4 addresses
i-027c56447af3fbae7	43.204.149.182 open address	172.31.0.52

IPv6 address	Instance state	Public IPv4 DNS
-	Shutting-down	ec2-43-204-149-182.ap-south-1.compute.amazonaws.com open address

Hostname type	Private IP DNS name (IPv4 only)	Elastic IP addresses
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	AWS Compute Optimizer finding
IPv4 (A)	t2.micro	Opt-in to AWS Compute Optimizer for recommendations

Auto-assigned IP address	VPC ID
43.204.149.182 [Public IP]	vpc-04b7b73f8d2523b6c

Outcome :

By creating a shell script to manage your cloud resources, you will achieve the following outcomes:

Automation: You will have a powerful tool that automates the tasks of launching, stopping, and terminating cloud VMs, reducing the need for manual intervention and minimizing the risk of human error.

Efficiency: The script will streamline the management of your cloud infrastructure, saving you time and effort by handling repetitive tasks quickly and consistently.

Cost Optimization: By automating the stopping of VMs when they are not in use and terminating them when no longer needed, you will optimize your cloud usage and reduce unnecessary costs.

Scalability: The script will allow you to easily scale your infrastructure up or down based on demand, ensuring that your resources are used efficiently and that your application can handle varying levels of traffic.

Flexibility: You can customize the script to meet your specific needs and integrate it with other automation tools or workflows, enhancing its functionality and adaptability.

