```
++++++++++
AWS CLI
++++++++++


AWS provides two ways of infrastructure configurations

1) AWS Management Web Console

2) AWS CLI (Command Line Interface)

Using the AWS web console:
++++++++++++++++++++++++++++++
It is a graphical method to connect to various AWS resources, their configuration, modification, etc.
It is simple to use and does not require knowledge of scripting.

AWS Command Line Interface:
++++++++++++++++++++++++++++++
Usually, the script provides you with the flexibility to manage multiple AWS resources,
infrastructures effectively.

For example, we can use the script to deploy multiple resources without the need to go through a
complete configuration wizard each time.

Prerequisites to Use AWS CLI
+++++++++++++++++++++++++++++

1) Create AWS Account: In order to configure AWS CLI, an AWS account needs to be created if you do
not have one. Please register for the AWS account. The new AWS account includes 12 months of free
tier access.

2) Install AWS CLI: AWS CLI is available for Windows, MAC and Linux distribution of OS.

        a) For windows : https://awscli.amazonaws.com/AWSCLIV2.msi (download and install)


        b) MAC and Linux: Please follow these steps (execute below commands)

                        $ sudo apt-get install -y python-dev python-pip
                        $ sudo pip install awscli


3) Once Installation completed then execute below commands

                        -> Open command prompt
                        -> Execute below command

                                $ aws --version (It should give AWS CLI version)
                                $ aws configure (To connect with AWS Cloud)


3) Create AWS IAM User with 'Programmatic Access' permission

User Creation Response
+++++++++++++++++++++++
On the user creation success screen, two important pieces of information are provided.

        1) Access Key ID
        2) Secret access key

-> We are going to store this information securely and will not share this information.

-> Alternatively, CSV file download option is available, CSV file contains the details.
```

-> Safely store the key or downloaded â€œCSVâ€� file, as we will not be able to retrieved Secret Access Key again.

-> Click on close and you will end up on the user dashboard. The newly created user is available now.

AWS CLI â€“ Configuration
++++++++++++++++++++++

Step1: Click on the demo user, the pertinent details corresponding to user will be shown.

Permissions
Groups
Tags
Security Credentials
Access Advisor

Step2: We want to use a Security Credentials object, click on the Security Credentials tab.

Step3: Here we are seeing Access Key ID, which was recently created with status marked as â€œActiveâ€�

Step4: In this case, the Access Key status provides an important security feature to the administrators.

Step5: Now we have the user, therefore allowing us access to AWS resources programmatically.

Configuring Terminal/Command Prompt
+++++++++++++++++++++++++++++++++++

1) Log in to the terminal window (â€œmacâ€�/ Linuxâ€�) or command prompt (â€œWindowsâ€�).

2) Before we can access the AWS resources using CLI (command-line interface), we will need to configure the CLI.

3) We will run the following command to configure AWS CLI

```
        $ aws configure
```

AWS Access Key ID : Created as part of new security credentials

AWS Secret Access Key : Corresponding to the â€œAWS Access Keyâ€� selected

Default Region name   : AWS regions, we are using ap-south-1

Default Output format : Json

4) Now we are all set with the profile.

AWS CLI syntax :   `$ aws <service-name>  <options>`

############   CLI Documentation : https://docs.aws.amazon.com/cli/latest/reference/
##################

+++++++++++++++++++++++++++++++++++++++
Working with AWS S3 Service using AWS CLI
+++++++++++++++++++++++++++++++++++++++
Step-1: In this case, we will be using AWS S3 (Simple Storage Service) as an example.

In brief, AWS S3 is an object storage service.

Step-2: Next, we are going to run â€œaws s3 ls" (to display bucket lis)

```
                $ aws s3 ls

Step-3: After listing out the content of the existing bucket, let us try to create a new s3 bucket
using AWS CLI

                $ aws s3 mb s3://ashokitbucket

Step-4: As a result of the command execution, the bucket should be created

Step-5: After the command has been executed, let us check, if the bucket has been created and what is
the region of the bucket.



+++++++++++++++++++++++++++
List out all ec2 instances
++++++++++++++++++++++++++++
$ aws ec2 describe-instances

Note : It will list down all the data in JSON format

For example, we can search for instances with a given type:

$ aws ec2 describe-instances --filters Name=instance-type,Values=t2.micro

or a tag key:

$ aws ec2 describe-instances --filters "Name=tag-key,Values="EC2VM-2"


+++++++++++++++++++++++++++
Create a New Key Pair for EC2 Instances
++++++++++++++++++++++++++++++++++++++++
-> Before launching a new EC2 instance weâ€™ll need an SSH key pair that weâ€™ll use to connect to it
securely.

$ aws ec2 create-key-pair --key-name ashokitkey --output text > ashokitkey.pem

The above command will create a new key in the AWS named ashokitkey and pipe the secret key directly
to the location we specify, in this case, ashokitkey.pem.


+++++++++++++++++++++++++++
Launch New EC2 Instances
++++++++++++++++++++++++
$ aws ec2 run-instances --image-id ami-08df646e18b182346 --instance-type t2.micro --key-name
ashokitkey


+++++++++++++++++++++++++++
Stop and Start an EC2 Instance
++++++++++++++++++++++++++++++
$ aws ec2 stop-instances --instance-ids <instance-id>

And start again:

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

+++++++++++++++++++++++++++
Terminate an Instance
++++++++++++++++++++++
$ aws ec2 terminate-instances --instance-ids <instance-id>


+++++++++++++++++++++++++++
Creating Security Group in EC2
+++++++++++++++++++++++++++
$ aws ec2 create-security-group --group-name MySecurityGroup --description "My security group"


+++++++++++++++++++++++++++
Creating RDS instance (MySQL)
+++++++++++++++++++++++++++++

$ aws rds create-db-instance --db-instance-identifier test-mysql-instance --db-instance-class
db.t3.micro --engine mysql  --master-username admin --master-user-password secret99 --allocated-
storage 20


+++++++++++++++++++++++++++
Delete RDS Instance (MySQL)
++++++++++++++++++++++++++
$ aws rds delete-db-instance ^
    --db-instance-identifier test-mysql-instance ^
    --skip-final-snapshot ^
    --no-delete-automated-backups
```