1.0 Introduction to AWS:

In 2006, Amazon Web Services (AWS) began offering IT infrastructure services to businesses as web services-now commonly known as cloud computing. One of the key benefits of cloud computing is the opportunity to replace upfront capital infrastructure expenses with low variable costs that scale with your business. With the cloud, businesses no longer need to plan for and procure servers and other IT infrastructure weeks or months in advance. Instead, they can instantly spin up hundreds or thousands of servers in minutes and deliver results faster. Today, AWS provides a highly reliable, scalable, low-cost infrastructure platform in the cloud that powers hundreds of thousands of businesses in 190 countries around the world.

1.1 AWS Services:

- AWS IAM (identity and access management)
- Amazon S3 (Simple Storage Service)
- Amazon KMS (Key Management Service)
- Amazon EC2 (Elastic Compute Cloud)
- Amazon EBS (Elastic Block Storage)
- Amazon VPC (Virtual Private Cloud)
- Amazon RDS (Relational Database Service)
- Amazon Lambda

1.1.1 AWS IAM (Identity Access Management):

AWS Identity and Access Management provides secure access and management of resources in a secure and compliant manner. By leveraging IAM, you can create and manage users and groups by allowing and denying their permissions for individual resources. There are no additional costs, people only get charged for the use of other services by their users.

1.1.2 Amazon S3 (Simple Storage Service):

Amazon S3, at its core, facilitates object storage, providing leading scalability, data availability, security, and performance. Businesses of vast sizes can leverage S3 for storage and protect large sums of data for various use cases, such as websites, applications, backup, and more. Amazon S3's intuitive management features enable the frictionless organization of data and configurable access controls.

1.1.3 Amazon KMS (Key Management Service):

AWS Key Management Service (KMS) gives you centralized control over the cryptographic keys used to protect your data. The service is integrated with other AWS services making it easier to encrypt data you store in these services and control access to the keys that decrypt it.

1.1.4 Amazon EC2 (Elastic Compute Cloud):

EC2 is a cloud platform provided by Amazon that offers secure, and resizable compute capacity. Its purpose is to enable easy access and usability to developers for web-scale cloud computing, while allowing for total control of your compute resources. Deploy applications rapidly without the need for investing in hardware upfront; all the while able to launch virtual servers as-needed and at scale.

1.1.5 Amazon EBS (Elastic Block Storage):

Amazon Elastic Block Store (Amazon EBS) is a web service that provides block level storage volumes for use with Amazon Elastic Compute Cloud instances. Amazon EBS volumes are highly available and reliable storage volumes that can be attached to any running instance and used like a hard drive.

1.1.6 Amazon VPC (Virtual Private Cloud):

Amazon VPC enables you to set up a reasonably isolated section of the AWS Cloud where you can deploy AWS resources at scale in a virtual environment. VPC gives you total control over your environment, which includes the option to choose your own IP address range, creation of subsets, and arrangement of route tables and network access points.

1.1.7 Amazon RDS (Relational Database Service):

Amazon Relational Database Service (Amazon RDS) makes database configuration, management, and scaling easy in the cloud. Automate tedious tasks such as hardware provisioning, database arrangement, patching, and backups – cost-effectively and proportionate to your needs. RDS is available on various database instances which are optimized for performance and memory, providing six familiar database engines including Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle. database, and SQL server.

1.1.8 Amazon Lambda:

Lambda permits you to run code without owning or managing servers. Users only pay for the compute time consumed. Operate code for nearly any application or backend utility without administration. Users just upload the code, and Lambda rest, which provides precise software scaling and extensive availability.

2.0 Cloud Computing:

Cloud computing is the on-demand delivery of compute power, database, storage, applications, and other IT resources through a cloud services platform via the Internet with pay-as-you-go pricing. Whether you are running applications that share photos to millions of mobile users or you're supporting the critical operations of your business, a cloud services platform provides rapid access to flexible and low-cost IT resources. With cloud computing, you don't need to make large upfront investments in hardware and spend a lot of time on the heavy lifting of managing that hardware. Instead, you can provision exactly the right type and size of computing resources you need to power your newest bright idea or operate your IT department. You can access as many resources as you need, almost instantly, and only pay for what you use. Cloud computing provides a simple way to access servers, storage, databases and a broad set of application services over the Internet. A cloud services platform such as Amazon Web Services owns and maintains the network-connected hardware required for these application services, while you provision and use what you need via a web application.

2.1 Types of Cloud Computing:

Cloud computing provides developers and IT departments with the ability to focus on what matters most and avoid undifferentiated work such as procurement, maintenance, and capacity planning. As cloud computing has grown in popularity, several different models and deployment strategies have emerged to help meet specific needs of different users. Each type of cloud service and deployment method provides you with different levels of control, flexibility, and management. Understanding the differences between Infrastructure as a Service, Platform as a Service, and Software as a Service, as well as what deployment strategies you can use, can help you decide what set of services is right for your needs.

Cloud computing models

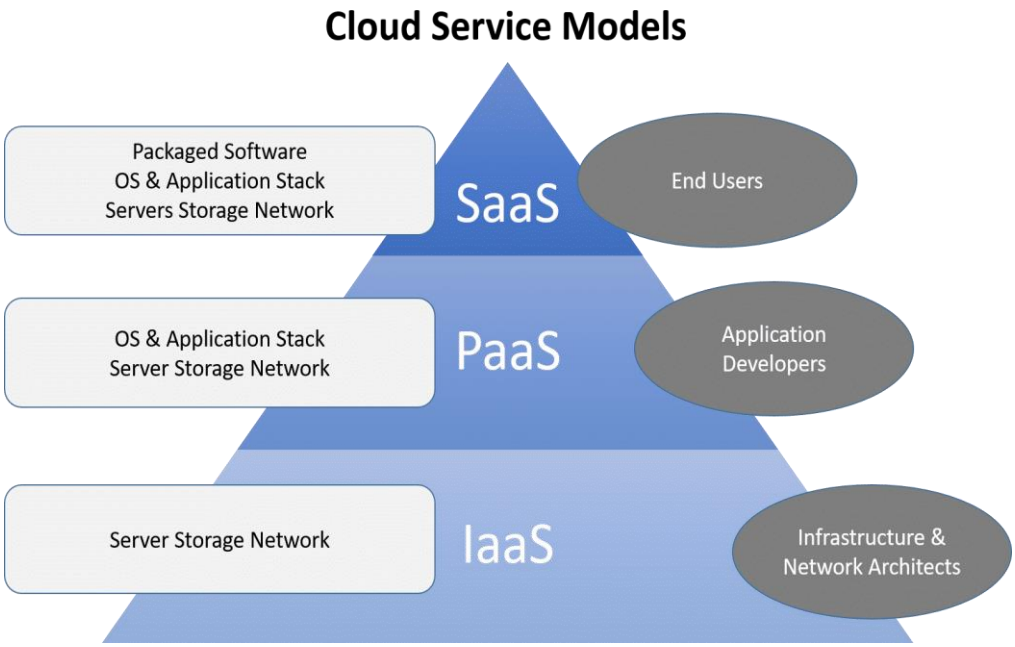

Fig 2.1 Cloud Service Model

2.1.1 Infrastructure as a Service (IaaS):

Infrastructure as a Service (IaaS) contains the basic building blocks for cloud IT and typically provides access to networking features, computers (virtual or on dedicated hardware), and data storage space. IaaS provides you with the highest level of flexibility and management control over your IT resources and is most similar to existing IT resources that many IT departments and developers are familiar with today.

2.1.2 Platform as a Service (PaaS):

Platform as a Service (PaaS) removes the need for your organization to manage the underlying infrastructure (usually hardware and operating systems) and allows you to focus on the deployment and management of your applications. This helps you be more efficient as you don't need to worry about resource procurement, capacity planning, software maintenance, patching, or any of the other undifferentiated heavy lifting involved in running your application.

2.1.3 Software as a Service (SaaS):

Software as a Service (SaaS) provides you with a completed product that is run and managed by the service provider. In most cases, people referring to Software as a Service are referring to end-user applications. With a SaaS offering you do not have to think about how the service is maintained or how the underlying infrastructure is managed; you only need to think about how you will use that particular piece of software. A common example of a SaaS application is web-based email which you can use to send and receive email without having to manage feature additions to the email product or maintain the servers and operating systems that the email program is running on.

3.0 Cloud:

A cloud-based application is fully deployed in the cloud and all parts of the application run in the cloud. Applications in the cloud have either been created in the cloud or have been migrated from an existing infrastructure to take advantage of the benefits of cloud computing. Cloud-based applications can be built on low-level infrastructure pieces or can use higher level services that provide abstraction from the management, architecting, and scaling requirements of core infrastructure.

3.1 Hybrid:

A hybrid deployment is a way to connect infrastructure and applications between cloud-based resources and existing resources that are not located in the cloud. The most common method of hybrid deployment is between the cloud and existing on-premises infrastructure to extend, and grow, an organization's infrastructure into the cloud while connecting cloud resources to the internal system. For more information on how AWS can help you with your hybrid deployment, visit our Hybrid Cloud with AWS page.

3.2 On-Premises:

The deployment of resources on-premises, using virtualization and resource management tools, is sometimes called the “private cloud.” On-premises deployment doesn’t provide many of the benefits of cloud computing but is sometimes sought for its ability to provide dedicated resources. In most cases this deployment model is the same as legacy IT infrastructure while using application management and virtualization technologies to try and increase resource utilization. For more information on how AWS can help, see Use case: Cloud services on-premises.

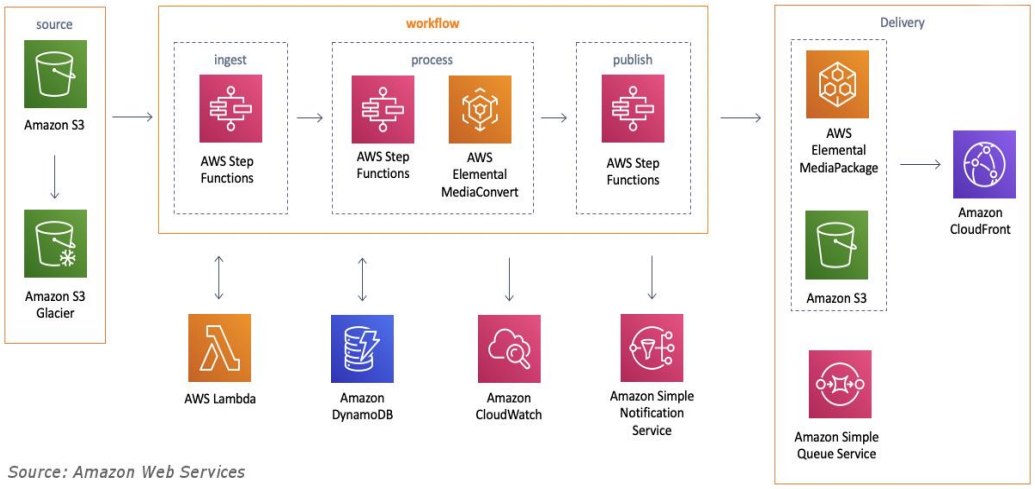

Fig 3.1 Amazon Web Services

3.3 AWS Users:

The root user is the account owner and is created when the AWS account is created. Other types of users, including IAM users, and AWS IAM Identity Center (successor to AWS Single Sign-On) (IAM Identity Center) users are created by the root user or an administrator for the account. All AWS users have security credentials.

3.3.1 Root User Credentials:

The credentials of the account owner allow full access to all resources in the account. You can't use IAM policies to explicitly deny the root user access to resources. You can only use an AWS Organizations service control policy (SCP) to limit the permissions of the root user of a member account. Because of this, we recommend that you create an administrative user in IAM Identity Center to use for everyday AWS tasks. Then, safeguard the root user credentials and use them to perform only those few account and service management tasks that require you to sign in as the root user. For the list of those tasks, see Tasks that require root user credentials. To learn how to set up an administrator for daily use in IAM Identity Center, see Getting started in the AWS IAM Identity Center (successor to AWS Single Sign-On) User Guide.

3.3.2 IAM Credentials:

An IAM user is an entity you create in AWS that represents the person or service that uses the IAM user to interact with AWS resources. These users are identities within your AWS account that have specific custom permissions. For example, you can create IAM users and give them permissions to create a directory in IAM Identity Center. IAM users have long-term credentials that they can use to access AWS using the AWS Management Console, or programmatically using the AWS CLI or AWS APIs. For step-by-step instructions for how IAM users sign in to the AWS Management Console, see Signing in as an IAM user in the AWS Sign-In User Guide.

4.0 Creating AWS Account:

Anyone who has root user credentials for your AWS account has unrestricted access to all the resources in your account, including billing information. When you create an AWS account, you begin with one sign-in identity that has complete access to all AWS services and resources in the account.

- Open the Amazon Web Services home page .
- Choose **Create an AWS account**.
- Enter your account information, and then choose **Verify email address**. This will send a verification code to your specified email address.

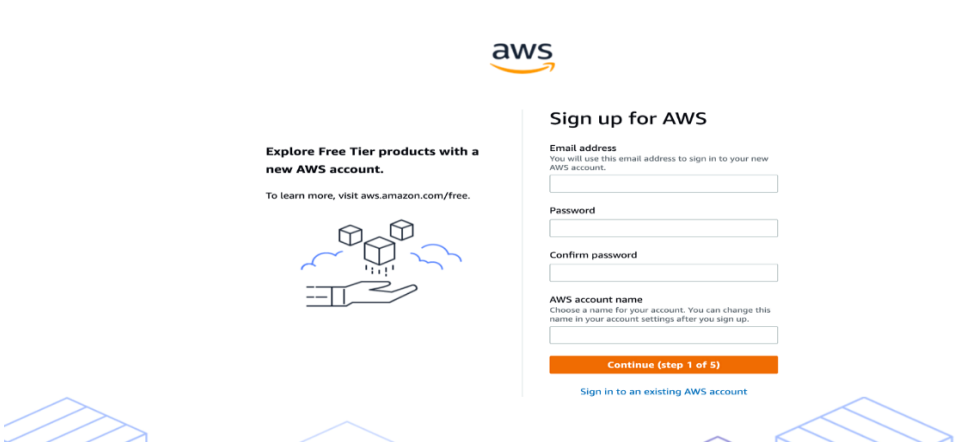

Fig 4.1 AWS Account Creation Step 1

- Enter your verification code, and then choose **Verify**.
- Enter a strong password for your root user, confirm it, and then choose **Continue**. AWS requires that your password meet the following conditions:
- It must have a minimum of 8 characters and a maximum of 128 characters.
- It must include a minimum of three of the following mix of character types: uppercase, lowercase, numbers, and ! @ # $ % ^ & * () <> [] {} | _+-= symbols.
- It must not be identical to your AWS account name or email address.
- Choose **Personal**.
- Enter your personal information.
- Click on accept by reading all the terms and condition.

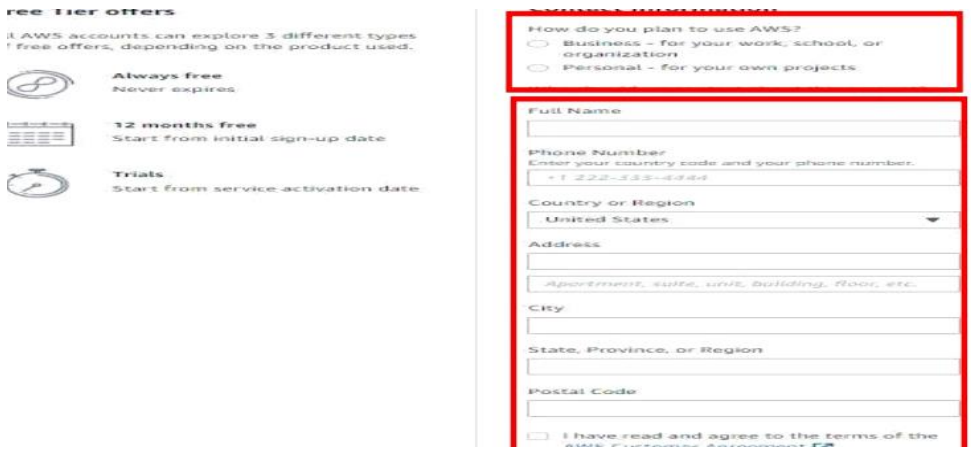

Fig 4.2 AWS Creation Step 2

- Choose **Continue**. At this point, you'll receive an email message to confirm that your AWS account is ready to use. You can sign in to your new account by using the email address and password you provided during sign up. However, you can't use any AWS services until you finish activating your account.

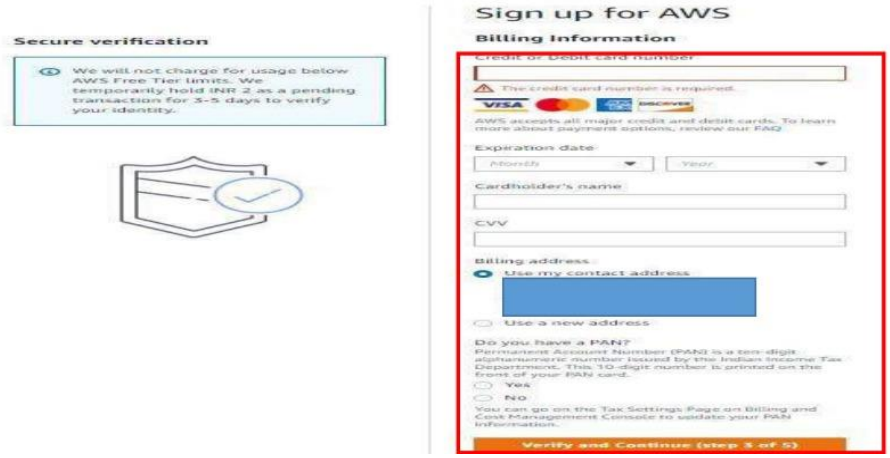

Fig 4.3 AWS Creation Step3

Fig 4.4 AWS Creation Last Step

- Enter the information about your payment method, and then choose **Verify and Continue**.
- Enter your country or region code from the list, and then enter a phone number where you can be reached in the next few minutes.
- Enter the code displayed in the CAPTCHA, and then submit.
- When the automated system contacts you, enter the PIN you receive and then submit.
- Select one of the available AWS Support plans. For a description of the available Support plans and their benefits, see Compare AWS Support plans.
- Choose **Complete sign up**. A confirmation page appears that indicates that your account is being activated.
- Check your email and spam folder for an email message that confirms your account was activated. Activation usually takes a few minutes but can sometimes take up to 24 hours.

Fig 4.5 AWS Creation Success

After you receive the activation message, you have full access to all AWS services.

5.0 Steps to Create User:

The IAM user represents the human user or workload who uses the IAM user to interact with AWS. A user in AWS consists of a name and credentials.

- Go to AWS console page, then login to your AWS root user account.
- On the **Console Home** page, select the IAM service.
- In the navigation pane, select **Users** and then select **Add users**.
- Provide user access to AWS console page.
- On the **Specify user details** page, under **User details**, in **User name**, enter the name for the new user. This is their sign-in name for AWS.
- select **I want to create an IAM user** and continue following this procedure.
- For **Console password**, select one of the following:
- **Autogenerated password** – The user gets a randomly generated password that meets the account password policy. You can view or download the password when you get to the **Retrieve password** page.
- **Custom password** – The user is assigned the password that you enter in the box.
- (Optional) **Users must create a new password at next sign-in (recommended)** is selected by default to ensure that the user is forced to change their password the first time they sign in.
- Select **Next**.
- On the **Set permissions** page, specify how you want to assign permissions for this user. Select one of the following three options:
- **Add user to group** – Select this option if you want to assign the user to one or more groups that already have permissions policies. IAM displays a list of the groups in your account, along with their attached policies. You can select one or more existing groups, or select **Create group** to create a new group.
- **Copy permissions** – Select this option to copy all of the group memberships, attached managed policies, embedded inline policies, and any existing permissions boundaries from an existing user to the new user. IAM displays a list of the users in your account. Select the one whose permissions most closely match the needs of your new user.
- **Attach policies directly** – Select this option to see a list of the AWS managed and customer managed policies in your account. Select the policies that you want to attach to the user or select **Create policy** to open a new browser tab and create a new policy. add the policy to the user.
- select **Next**.
- On the **Review and create** page, select next.
- Review all of the choices you made up to this point. When you are ready to proceed, select **Create user**.

- Select **Download .csv** to download the user's sign in credentials as a .csv file that you can save to a safe location.
- Then click on **return to user list.**
- Copy the console sign in link. Login to IAM user by entering user name and password.

Fig 5.1 IAM Creation

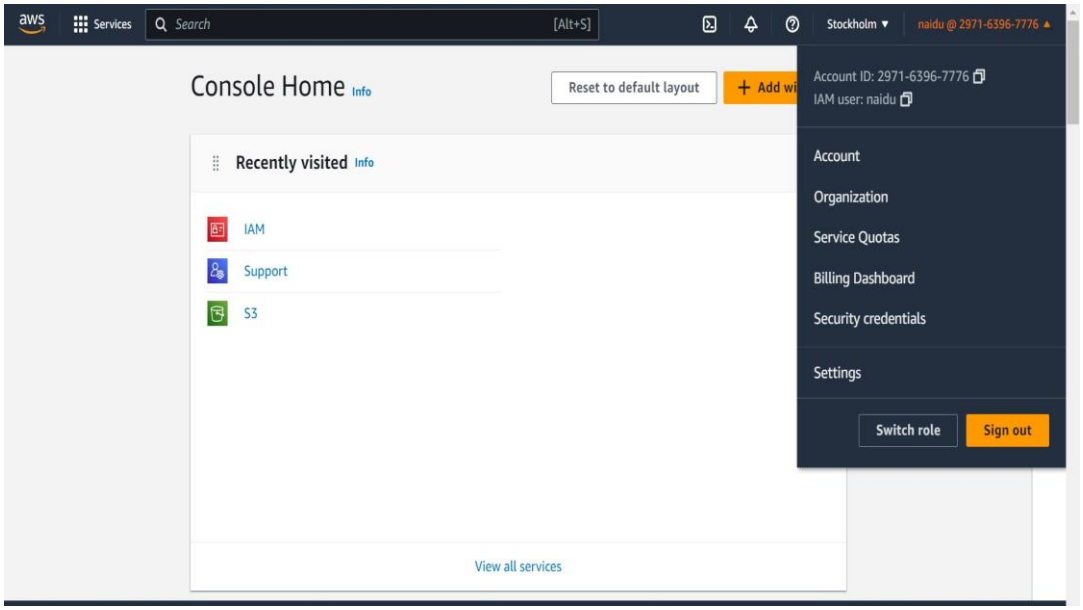

Fig 5.2 IAM User Console Page

6.0 Creating a User Group

An IAM user group is a collection of IAM users. User groups let you specify permissions for multiple users, which can make it easier to manage the permissions for those users.

- On the **Console Home** page, select the IAM service.
- Open the navigation menu and click **user groups**.
- The Groups screen is shown.
- Click **Create Group**.
- In the Create Group screen, assign a name to the group that differentiates it from the IDCS group, and enter a description(optional).
- Select policies that you want to give access to your user.
- Click **Create**.
- Open your group and then click on **add users** and then add users to your group.

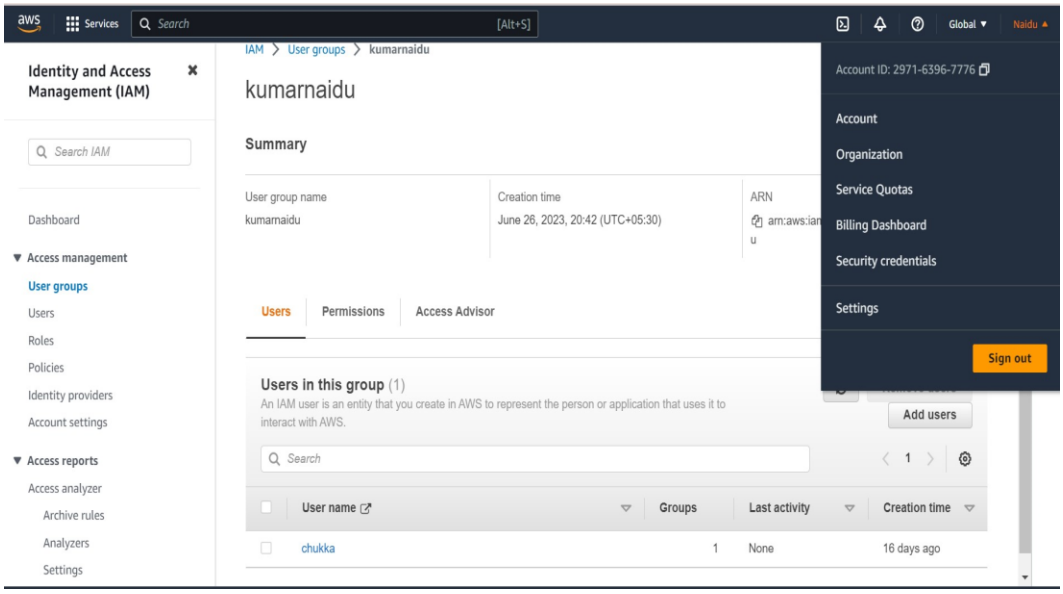

Fig 6.1 AWS User Group

7.0 Creating Policy:

A policy is an object in AWS that, when associated with an entity or resource, defines their permissions. AWS evaluates these policies when a principal, such as a user, makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents.

IAM policies define permissions for an action regardless of the method that you use to perform the operation.

- Go to AWS console page, then login to your AWS root user account.
- On the **Console Home** page, select the IAM service.
- Choose **Create policy**.
- In the **Policy editor** section, find the **Select a service** section, and then choose an AWS service. You can use the search box at the top to limit the results in the list of services. You can choose only one service within a visual editor permission block.
- In **Actions allowed**, choose the actions to add to the policy. You can choose actions in the following ways:
- Select the check box for all actions.
- Select one of the **Access level** groups to choose all actions for the access level (for example, **Read**, **Write**, or **List**).
- Expand each of the **Access level** groups to choose individual actions.
- For **Resources**,
- Choose **Add ARNs** to specify resources by their Amazon Resource Names (ARN). You can use the visual ARN editor or list ARNs manually. For more information about ARN syntax, see Amazon Resource Name (ARN) in the *AWS General Reference Guide*. For information about using ARNs in the Resource element of a policy, see IAM JSON policy elements: Resource.
- Choose **Any in this account** next to a resource to grant permissions to any resources of that type.
- Choose **All** to choose all resources for the service.
- (Optional) Choose **Request conditions - *optional*** to add conditions to the policy that you are creating.
- To add more permission blocks, choose **Add more permissions**
- When you are finished adding permissions to the policy, choose **Next**.

- On the **Review and create** page, type a **Policy Name** and a **Description** (optional) for the policy that you are creating. Review

the **Permissions defined in this policy** to make sure that you have granted the intended permissions

- Choose **Create policy** to save your new policy.

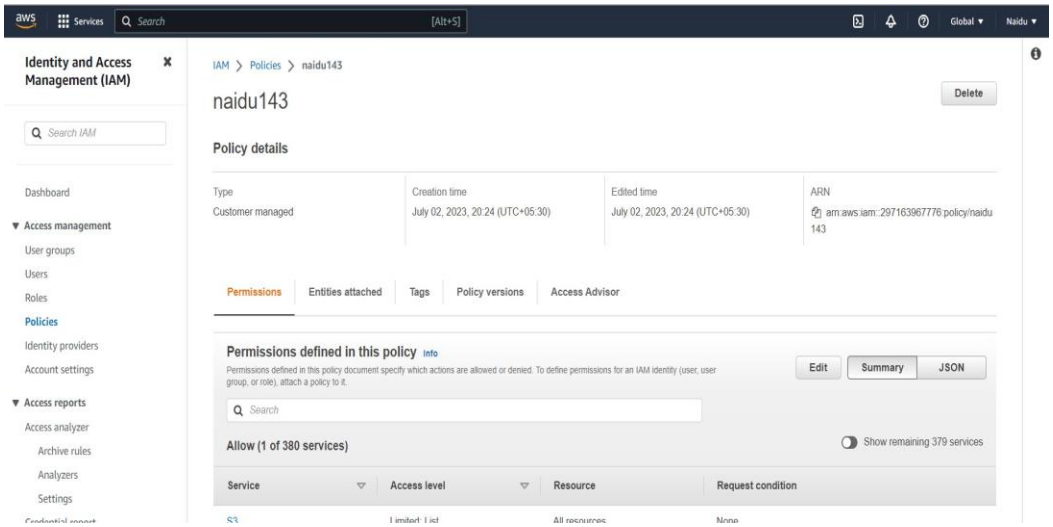

Fig 7.1 AWS Policy

7.1 Creating an Inline Policy:

An inline policy is a policy created for a single IAM identity (a user, group, or role). Inline policies maintain a strict one-to-one relationship between a policy and an identity. They are deleted when you delete the

Identity. There is no need to add the policy it will be directly attached while creating.

- In the navigation pane, choose Users or Roles.
- In the list, choose the name of the user or role to embed a policy in.
- Choose the Permissions tab.
- Choose Add permissions and then choose Create inline policy.
- Follow the above steps as creating a policy.

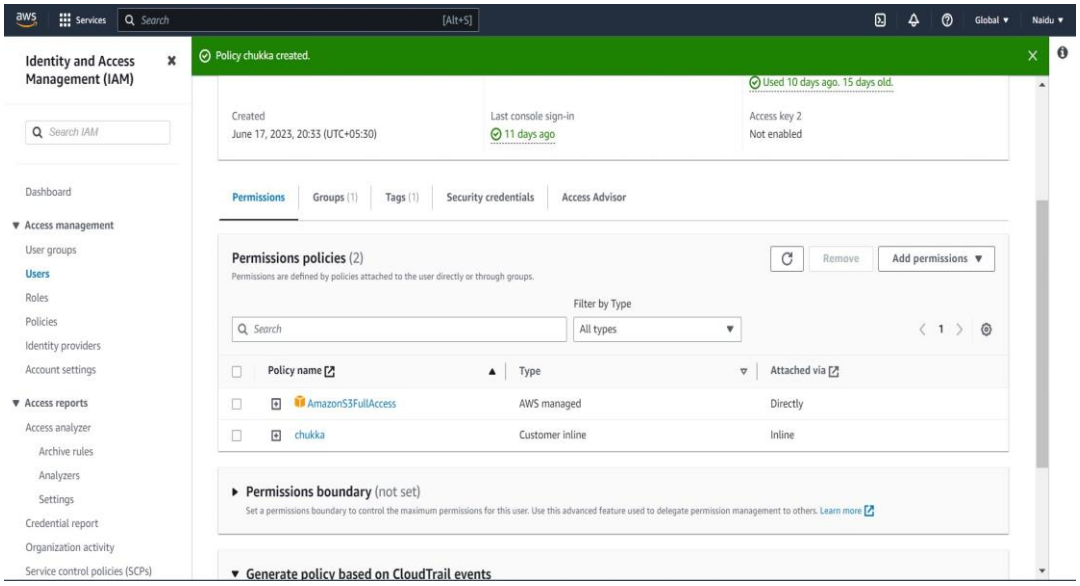

Fig 7.2 AWS Inline Policy

7.2 Copy Permissions:

Select this option to copy all of the group memberships, attached managed policies, embedded inline policies, any attached groups, and any existing permissions boundaries from an existing user to the new user. IAM displays a list of the users in your account. Select the one whose permissions most closely match the needs of your new user.

- Select the option **copy permissions** while creating the user.
- Now attach the user from user you want to copy permissions.
- Click on **create.**

7.3 Permission Boundaries:

A permissions boundary controls the maximum permissions that a role can have. Permissions boundaries are an advanced AWS feature. AWS supports *permissions boundaries* for IAM entities (users or roles). A permissions boundary is an advanced feature for using a managed policy to set the maximum permissions that an identity-based policy can grant to an IAM entity. An entity's permissions boundary allows it to perform only the actions that are allowed by both its identity-based policies and its permissions boundaries.

- Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
- In the navigation pane, choose Users.

- Choose the name of the user whose permissions boundary you want to change.
- Choose the Permissions tab. If necessary, open the Permissions boundary section and then choose Change boundary.
- Select the policy that you want to use for the permissions boundary.
- Choose Set boundary.

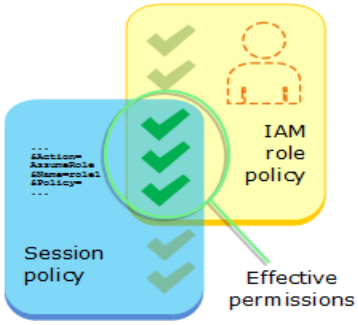

Fig 7.3 Representation of Permission Boundaries

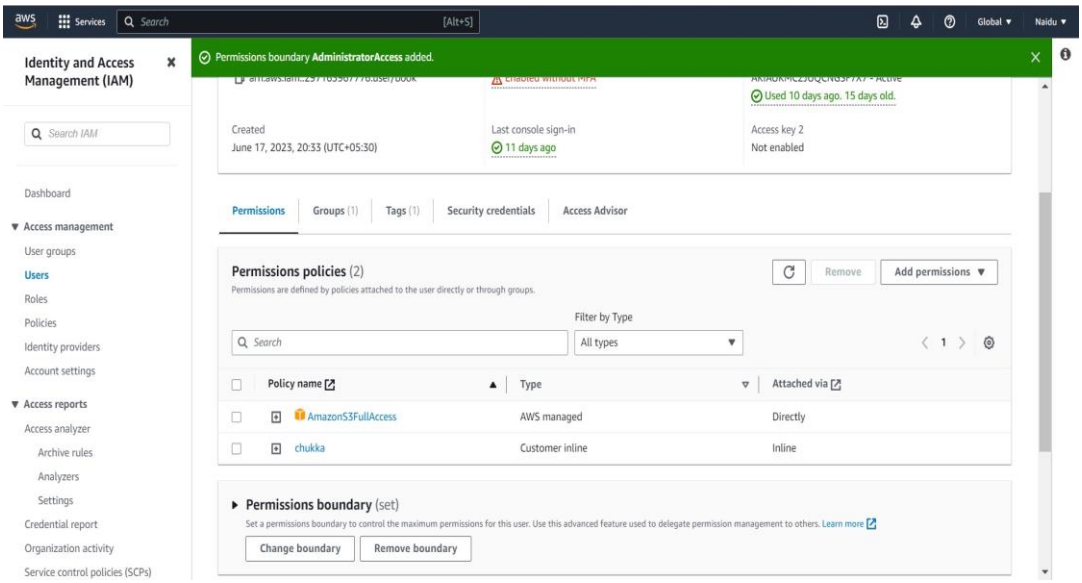

Fig 7.4 Set Permission Boundaries

8.0 Creating Roles:

- In the navigation pane, choose Roles.
- Create a role: **Create Role**
  - For the role's trust policy, you can specify a file location (**AWS account)**.
- Attach a managed permission policy to the role: **Attach Role Policy**

(or)

- Create an inline permission policy for the role: **Put Role Policy**

- (Optional) Add custom attributes to the user by attaching tags: Tag Role
- (Optional) Set the permissions boundary for the role: Put Role Permissions Boundary
- Click on **create**

An IAM *role* is an IAM identity that you can create in your account that has specific permissions. An IAM role is similar to an IAM user, in that it is an AWS identity with permission policies that determine what the identity can and cannot do in AWS. However, instead of being uniquely associated with one person, a role is intended to be assumable by anyone who needs it. Also, a role does not have standard long-term credentials such as a password or access keys associated with it. Instead, when you assume a role, it provides you with temporary security credentials for your role session.

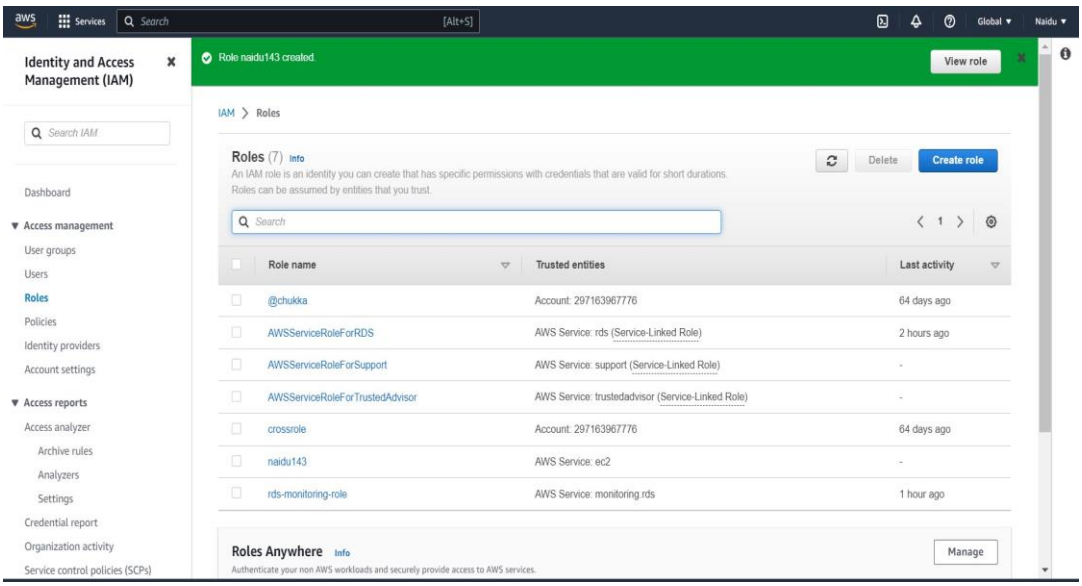

Fig 8.1 Role Created Successfully

8.1 Creating Switch Role:

When you switch to a role, you temporarily give up your user permissions and work with the permissions that are assigned to the role. When you exit the role, your user permissions are automatically restored. A role specifies a set of permissions that you can use to access AWS resources that you need. In that sense, it is similar to an IAM. When you sign in as a user, you get a specific set of permissions. However, you don't sign in to a role, but once signed in you can switch to a role. This temporarily sets aside your original user permissions and

instead gives you the permissions assigned to the role. The role can be in your own account or any other AWS account.

- Sign in to the AWS Management Console as an IAM user and open the IAM console at https://console.aws.amazon.com/iam/.
- In the IAM console, choose your user name on the navigation bar in the upper right. It typically looks like this: ***username@account_ID_number_or_alias***.
- Choose **Switch Role**. If this is the first time choosing this option, a page appears with more information. After reading it, choose **Switch Role**. If you clear your browser cookies, this page can appear again.
- On the **Switch Role** page, type the account ID number or the account alias and the name of the role that was provided by your administrator.
- (Optional) Choose a **Display name**. Type text that you want to appear on the navigation bar in place of your user name when this role is active.
- Choose **Switch Role**. The display name and color replace your user name on the navigation bar, and you can start using the permissions that the role grants you.

Choose **Back to *username***. The role and its permissions are deactivated, and the permissions associated with your IAM user and groups are automatically restored.

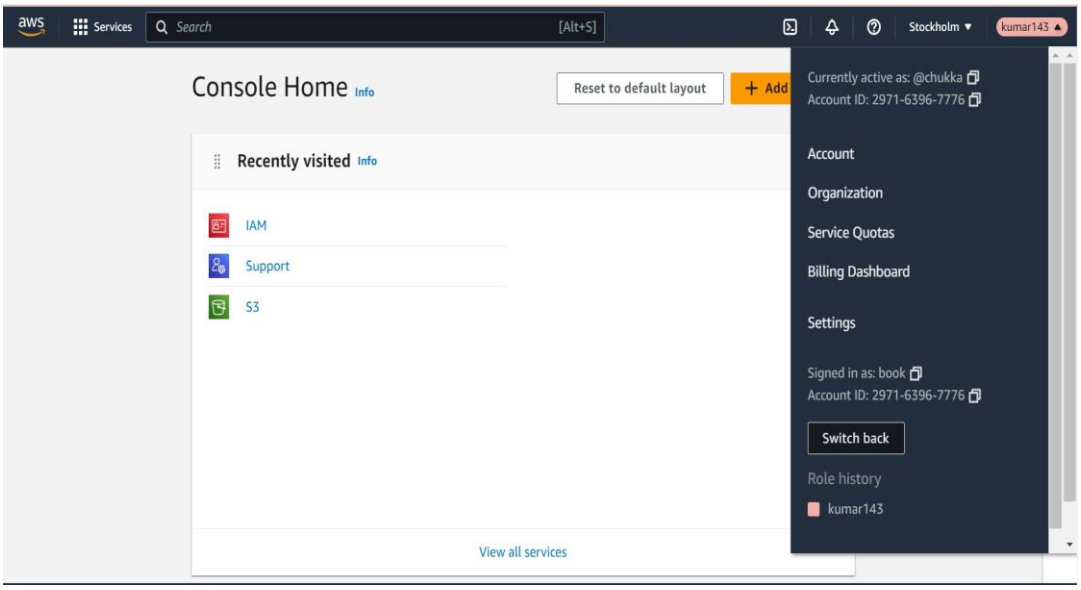

Fig 8.2 Switch Role

8.2 Flowchart for IAM:

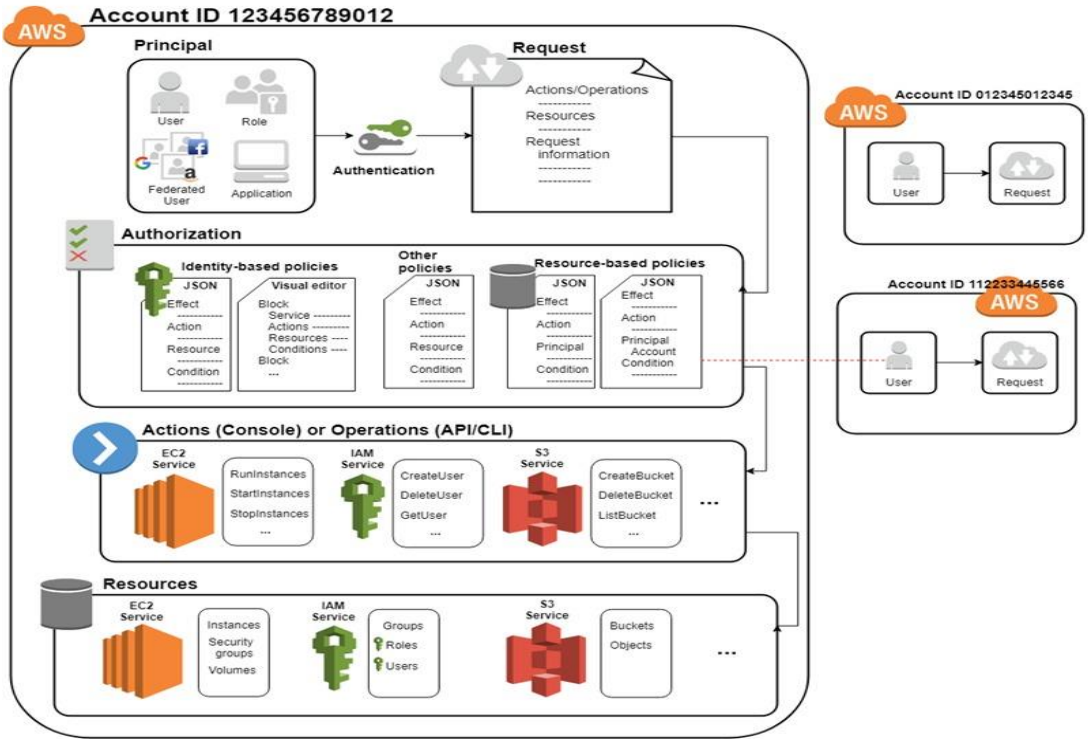

Fig 8.3 Flowchart for IAM

9.0 Simple Storage Service (S3) Introduction:

Amazon S3 (Simple Storage Service) provides object storage, which is built for storing and recovering any amount of information or data from anywhere over the internet. It provides this storage through a web services interface.

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. Customers of all sizes and industries can use Amazon S3 to store and protect any amount of data for a range of use cases, such as data lakes, websites, mobile applications, backup and restore, archive, enterprise applications, IoT devices, and big data analytics. Amazon S3 provides management features so that you can optimize, organize, and configure access to your data to meet your specific business, organizational, and compliance requirements.

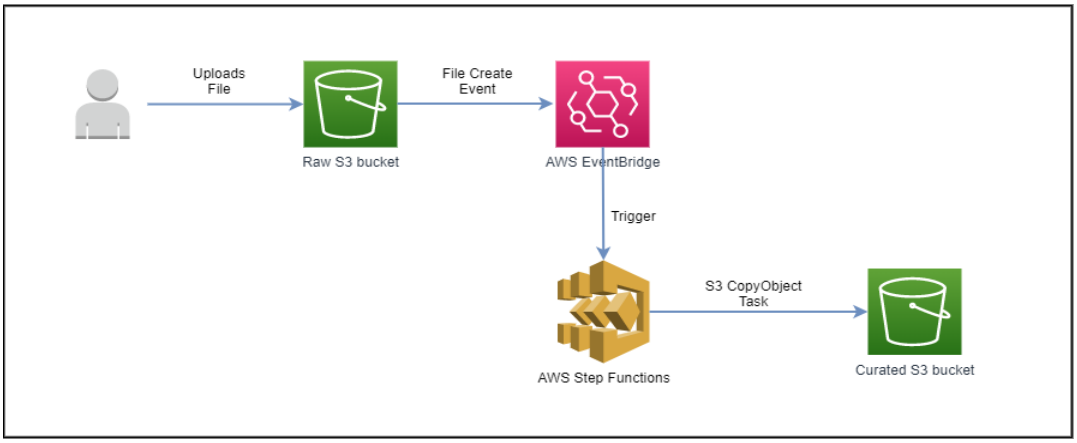

Fig 9.1 S3 Bucket

9.1 Storage Classes:

Amazon Simple Storage Service (S3) is used for storing data in the form of objects S3 is quite different from any other file storage device or service. Amazon S3 also provides industry-leading scalability, data availability, security, and performance. The data which is uploaded by the user in S3, that data is stored as objects and provided an ID. Moreover, they store in shapes like buckets and can upload the maximum file size is of 5 Terabyte (TB). This service is basically designed for the online backup and archiving of data and applications on Amazon Web Services (AWS).

9.2 Amazon S3 Storage Classes:

This storage maintains the originality of data by inspecting it. Types of storage classes are as follows:

- Amazon S3 Standard
- Amazon S3 Intelligent-Tiering
- Amazon S3 Standard-Infrequent Access
- Amazon S3 One Zone-Infrequent Access
- Amazon S3 Glacier Instant Retrieval
- Amazon S3 Glacier Flexible Retrieval
- Amazon S3 Glacier Deep Archive

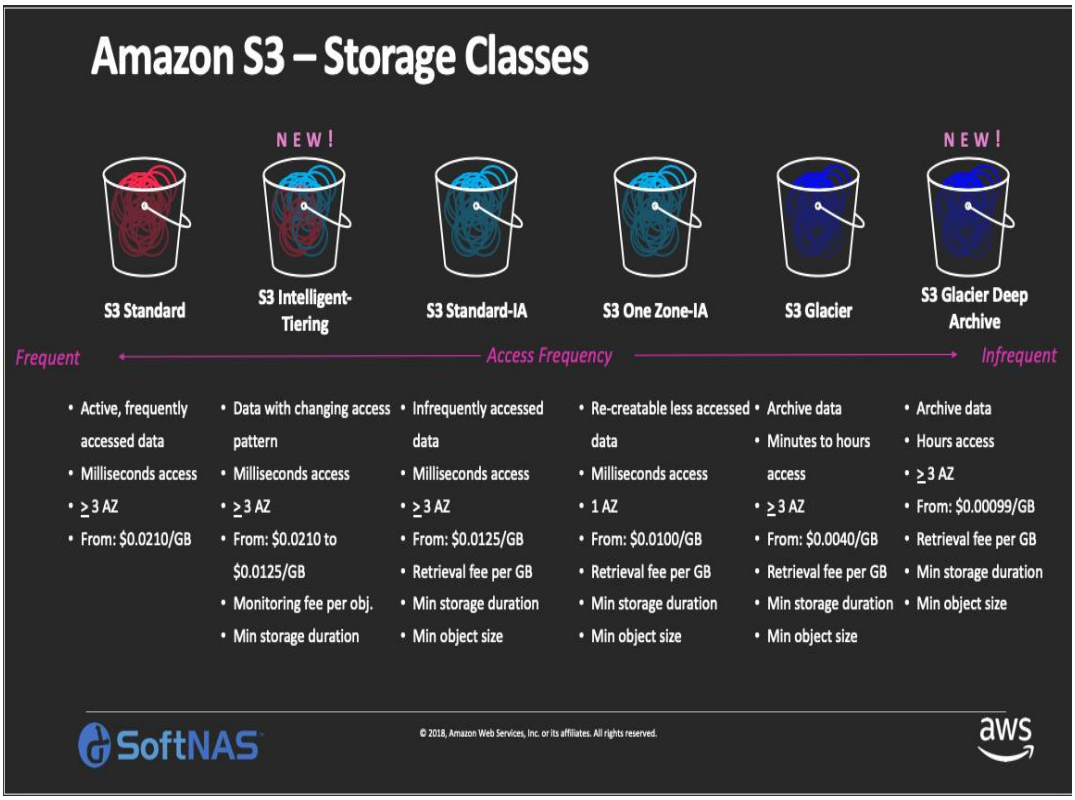

Fig 9.2 Storage Classes

9.2.1 Amazon S3 Standard:

- Availability criteria are quite good like 99.9%.
- Improves the recovery of an object file.
- It is against the events which are a little bit tough that can affect an entire Availability Zone.
- Durability of S3 standard is 99.999999999%.

9.2.2 Amazon S3 Intelligent-Tiering:

- Required less monitoring and automatically tier charge.
- No minimum storage duration and no recovery charges are required to access the service.
- Availability criteria are quite good like 99.9%.
- Durability of S3 Intelligent- Tiering is 99.999999999%.

9.2.3 Amazon S3 Standard-Infrequent Access:

- High performance and same action rate.
- Very Durable in all AZs.
- Availability is 99.9% in S3 Standard-IA.
- Durability is of 99.999999999%.

9.2.4 Amazon S3 One Zone-Infrequent Access:

- Supports SSL (Secure Sockets Layer) for data in transferring and encryption of data.
- Availability Zone destruction can damage the data.
- Availability is 99.5% in S3 one Zone- Infrequent Access.
- Durability is of 99.999999999%.

9.2.5 Amazon S3 Glacier Instant Retrieval:

- It just takes milliseconds to recover the data.
- The minimum object size should be 128KB.
- Availability is 99.9% in S3 glacier Instant Retrieval.
- Durability is of 99.999999999%.

9.2.6 Amazon S3 Glacier Flexible Retrieval:

- Free recoveries in high quantity.
- AZs destruction can lead to difficulty in accessing data.
- when you have to retrieve large data sets, then S3 glacier flexible retrieval is best for backup and disaster recovery use cases.
- Availability is 99.99% in S3 glacier flexible retrieval.
- Durability is of 99.999999999%.

9.2.7 Amazon S3 Glacier Deep Archive:

- More secured storage.
- Recovery time is less required.
- Availability is 99.99% in S3 glacier deep archive.
- Durability is of 99.999999999%.

9.3 Creating an S3 Bucket:

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. You can use Amazon S3 to store and retrieve any amount of data at any time, from anywhere.

- Sign in to the AWS Management Console and open the Amazon S3 console at https://console.aws.amazon.com/s3/.
- In the left navigation pane, choose **Buckets**.

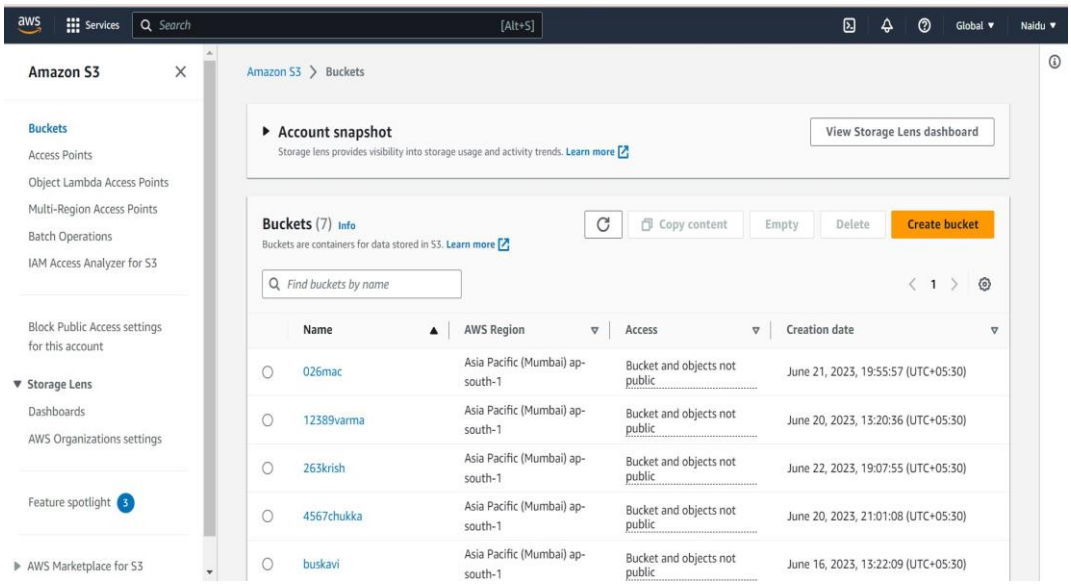

Fig 9.3 S3 Bucket Console Page

- Choose **Create bucket**.
  - The **Create bucket** page opens.
- For **Bucket name**, enter a name for your bucket.
  - The bucket name must:
  - Be unique within a partition.
- Consist only of lowercase letters, numbers, dots (.), and hyphens (-). For best compatibility, we recommend that you avoid using dots (.) in bucket names, except for buckets that are used only for static website hosting.
- Begin and end with a letter or number.
  - After you create the bucket, you cannot change its name.
- For **Region**, choose the AWS Region where you want the bucket to reside.
- Under **Object Ownership**, to disable or enable ACLs and control ownership of objects uploaded in your bucket, choose one of the following settings:
  - **ACLs disabled**
- **Bucket owner enforced (default)** – ACLs are disabled, and the bucket owner automatically owns and has full control over every object in the bucket. ACLs no longer affect access permissions to data in the S3 bucket. The bucket uses policies exclusively to define access control.
  - **ACLs enabled**
- **Bucket owner preferred** – The bucket owner owns and has full control over new objects that other accounts write to the bucket with the bucket-owner-full-control canned ACL.
- Under **Block Public Access settings for this bucket**, choose the Block Public Access settings that you want to apply to the bucket.
- By default, all four Block Public Access settings are enabled. We recommend that you keep all settings enabled, unless you know that you need to turn off one or more of them for your specific use case.
- (Optional) Under **Bucket Versioning**, you can choose if you wish to keep variants of objects in your bucket. For more information about versioning, see Using versioning in S3 buckets.
  - To disable or enable versioning on your bucket, choose either **Disable** or **Enable**.
- (Optional) Under **Tags**, you can choose to add tags to your bucket. Tags are key-value pairs used to categorize storage.
  - To add a bucket tag, enter a **Key** and optionally a **Value** and choose **Add Tag**.
- Under **Default encryption**, choose **Edit**.
  - To use S3 Bucket Keys, under **Bucket Key**, choose **Enable**.
- Choose **Create bucket**.

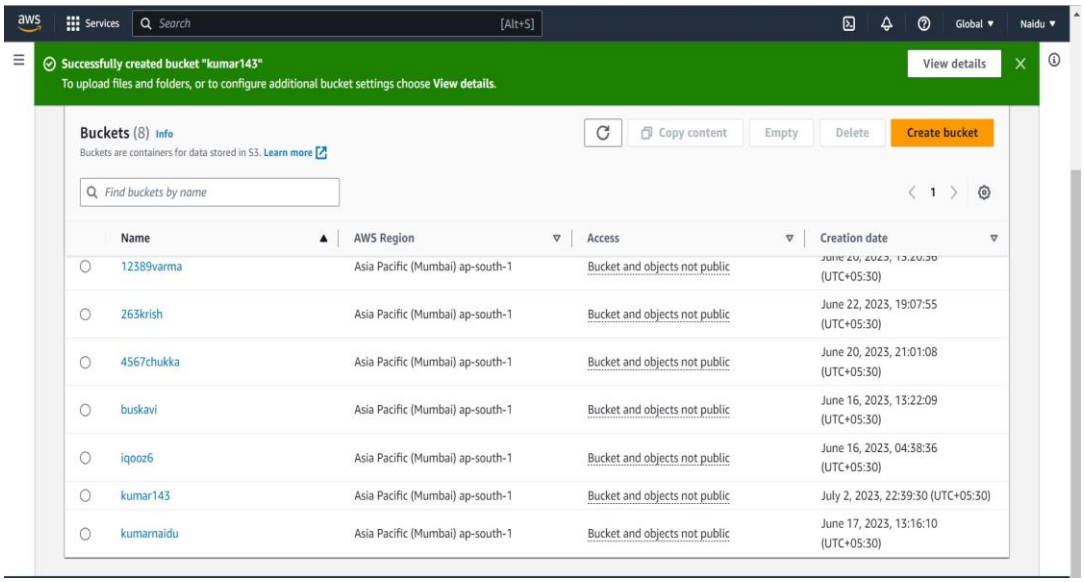

Fig 9.4 Bucket created successfully

You've created a bucket in Amazon S3.

9.3.1 Uploading Data into Buckets:

After creating a bucket in Amazon S3, you're ready to upload an object to the bucket. An object can be any kind of file: a text file, a photo, a video, and so on.

- Open the Amazon S3 console at https://console.aws.amazon.com/s3/.
- In the **Buckets** list, choose the name of the bucket that you want to upload your object to.
- On the **Objects** tab for your bucket, choose **Upload**.
- Under **Files and folders**, choose **Add files**.
- Choose a file to upload, and then choose **Open.**
- Choose **Upload**.

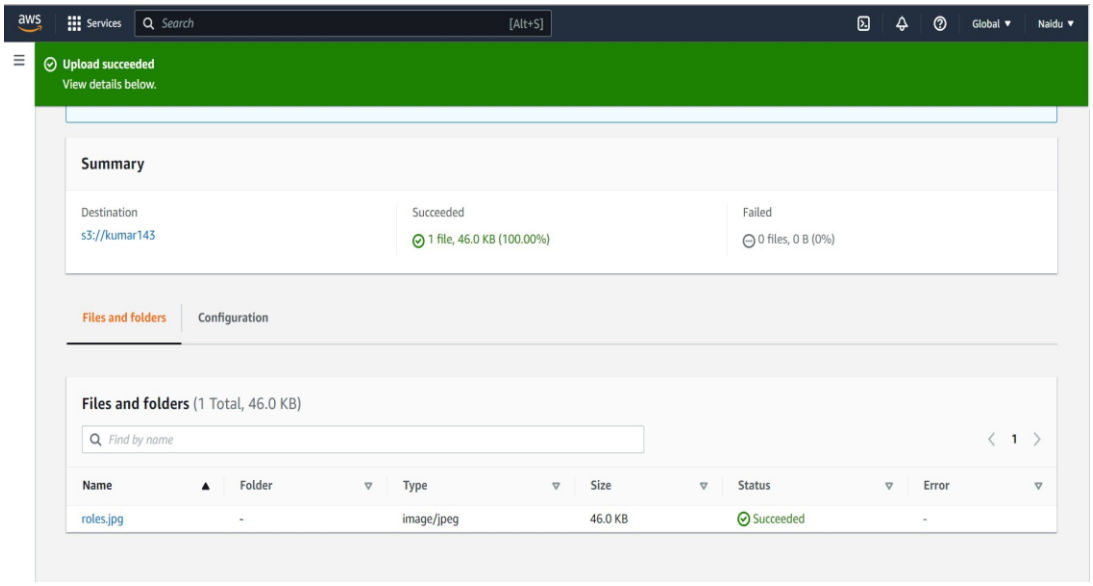

Fig 9.5 File Uploaded

You've successfully uploaded an object to your bucket.

9.3.2 Downloading an Object:

- In the **Buckets** list, choose the name of the bucket that you want to download.
- You can download an object from an S3 bucket in any of the following ways:
- Select the object and choose **Download** or choose **Download as** from the **Actions** menu if you want to download the object to a specific folder.
- If you want to download a specific version of the object, select the **Show versions** button. Select the version of the object that you want and choose **Download** or choose **Download as** from the **Actions** menu if you want to download the object to a specific folder.

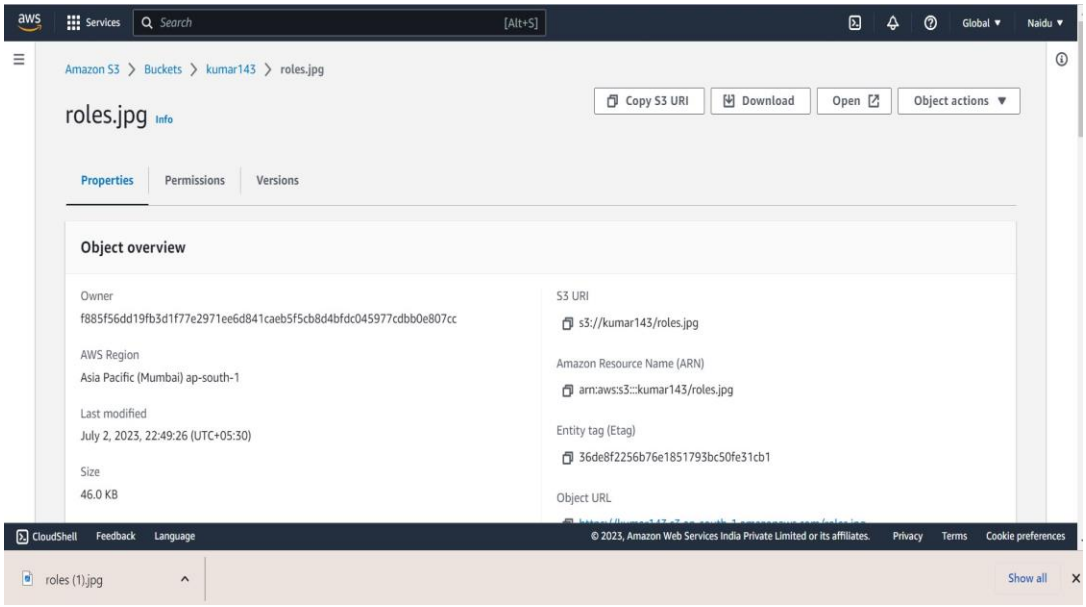

Fig 9.6 File Downloaded Successfully

You've successfully downloaded your object.

9.3.3 Deleting Object and Bucket:

- In the **Buckets** list, choose the name of the bucket that you want to delete an object from.
- Select the object that you want to delete.
- Choose **Delete** from the options in the upper right.
- On the **Delete objects** page, type **delete** to confirm deletion of your objects.
- Choose **Delete objects**.

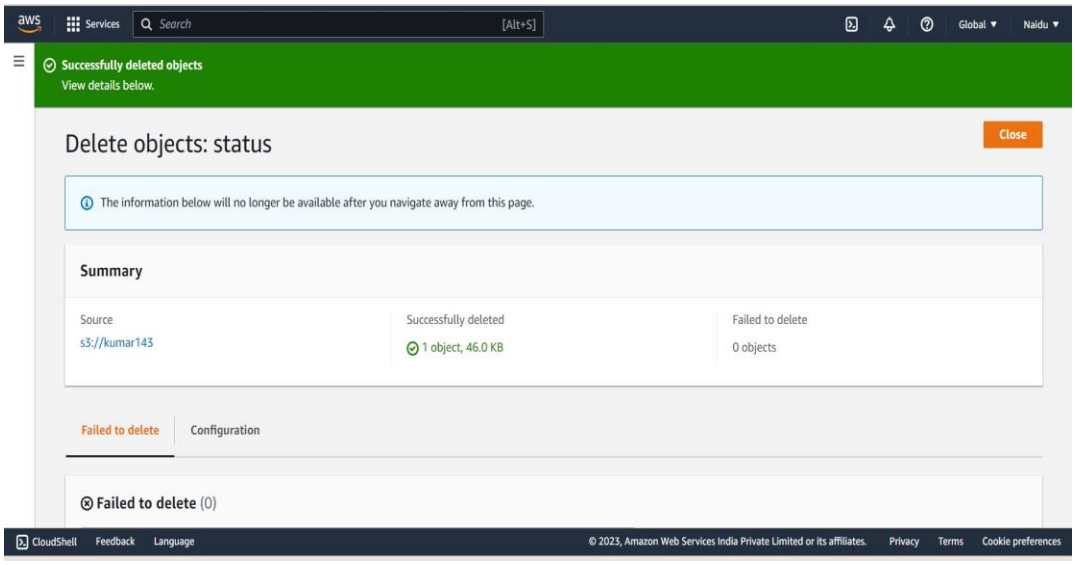

Fig 9.7 File Deleted Successfully

Objects deleted successfully.

- In the **Buckets** list, select the bucket that you want to empty, and then choose **Empty**.
- To confirm that you want to empty the bucket and delete all the objects in it, in **Empty bucket**, type **permanently delete**.
- To empty the bucket and delete all the objects in it, and choose **Empty**.
  - An **Empty bucket: Status** page opens that you can use to review a summary of failed and successful object deletions.
- To return to your bucket list, choose **Exit**.

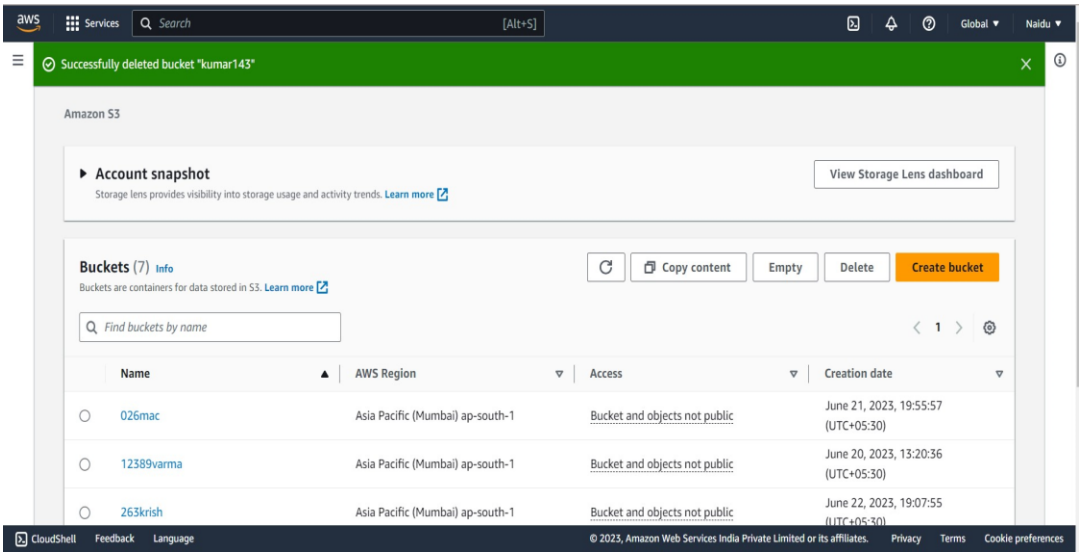

Fig 9.8 Deleting Bucket

Bucket deleted successfully.

9.4 S3 List All:

To get access for only the list of all buckets using an IAM user. We create a policy with S3 list all access and allocate it to the user.

- Login to AWS root user account.
- Go to IAM dashboard, go to any one of the user and, click on **add permissions** and, click on **create inline policy**.
- select s3 as a service, in actions select **list all my buckets** click on **review policy.**
- Give a name to your policy and then click on **create policy**.

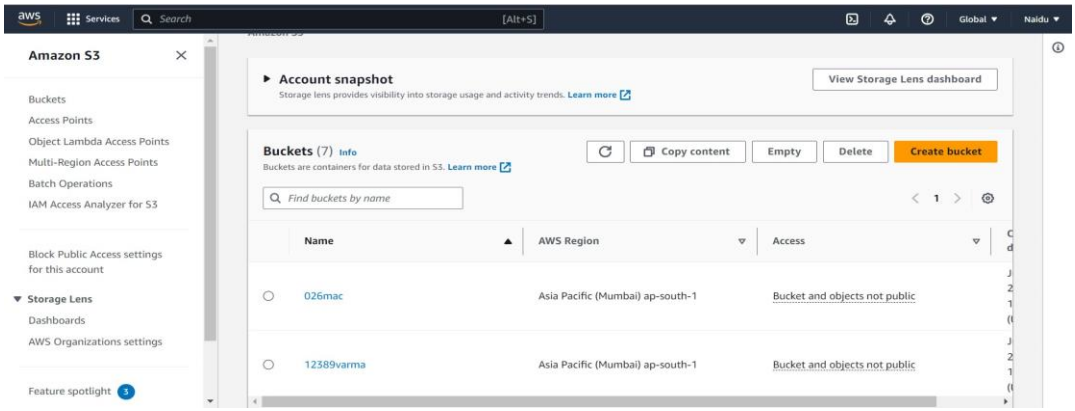

Fig 9.9 List Bucket

The user with this policy can only access the list of the buckets.

9.5 S3 List Specific Bucket:

To get access to get list of all the objects inside the bucket we use option list specific bucket. For this we add a policy with list all and list specific to the user.

- Go to IAM dashboard, go to user and then click on **add permissions** and then click **create inline policy**.

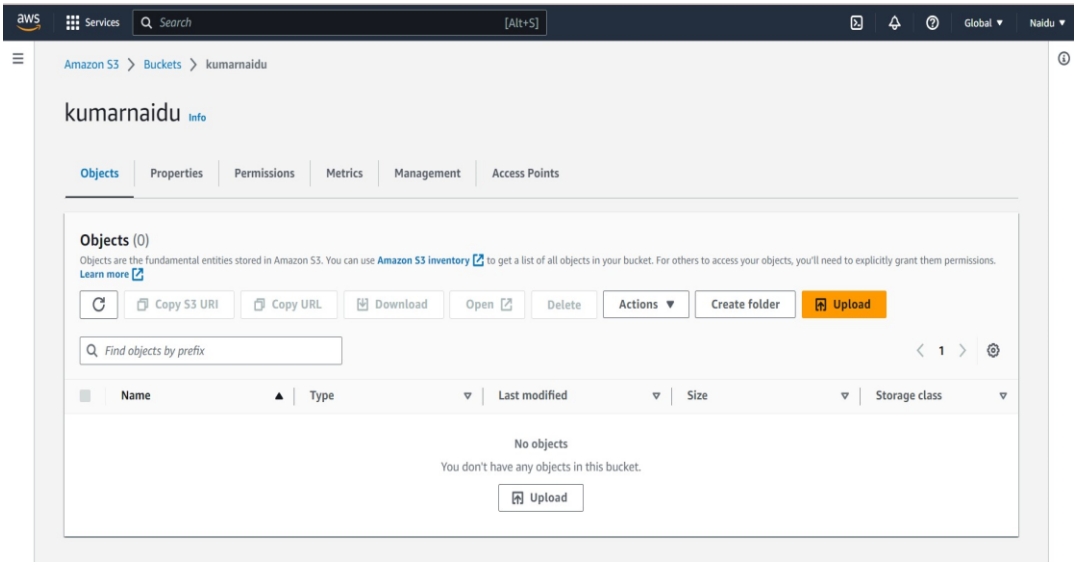

Fig 9.10 Permission for Specific Bucket

- Select s3 as a service and then in actions select **listallmybucket** and **list bucket**.
- Now in resources click on **add ARN** and give a name of your bucket then click on **add**.
- Or else go to JSON copy the code in resource and paste it and change the bucket name to add another bucket using JSON mode keep all the arn in array and click on **review policy.**
- Click on **create policy**.

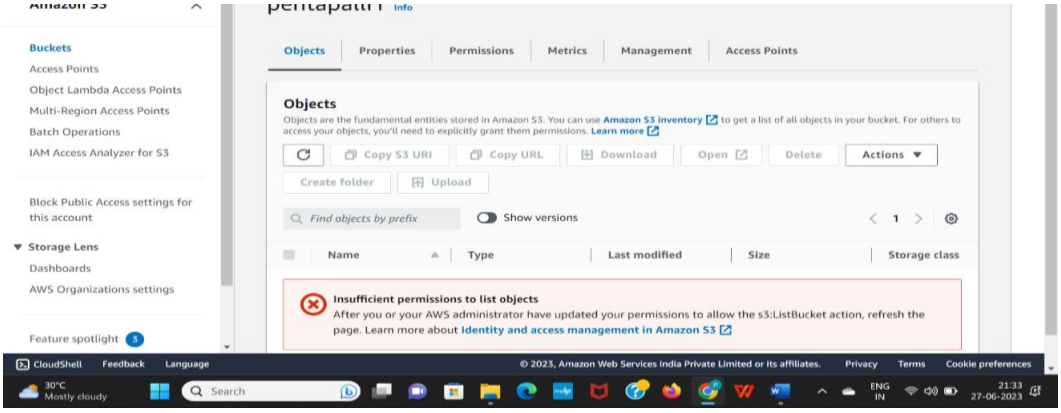

Fig 9.11 Files Cannot be Accessed Without Permission

Permissions to access list of objects in specific buckets kept successfully.

9.6 To Make Object URL Public:

- Sign in to Amazon Web Services and go to your S3 Management Console.
- Click on the name of the S3 bucket from the list. If it's still in its default access state, it should say "Buckets and objects not public" next to it.
- Go to the Permissions tab. The first sub-tab, which is open by default, is Block Public Access, and the "Block *all* public access option will be On. Click on the Edit button at the right.
- Uncheck the "Block *all* public access" option, and then click the Save button.
- You'll then be asked to confirm the change by typing in the word confirm.
- Go to permissions, go to ACL's, click on bucket owner enforced, select ACL's enabled click on **save changes**.
- Select object then go to actions and then click on make public using ACL then click on make public.
- Click on the file and copy the **object URL** in properties paste in new tab the file is opened.

Fig 9.12 Object Opened in Public URL Successfully

9.7 Public Access to The Downloaded Template in Bucket:

In AWS, buckets refer to Amazon Simple Storage Service (S3) buckets, which are containers for storing objects such as files and templates. When you download a template from an S3 bucket, the default access permissions for that template are determined by the bucket's access control configuration and the permissions granted to the IAM user or role accessing the bucket.

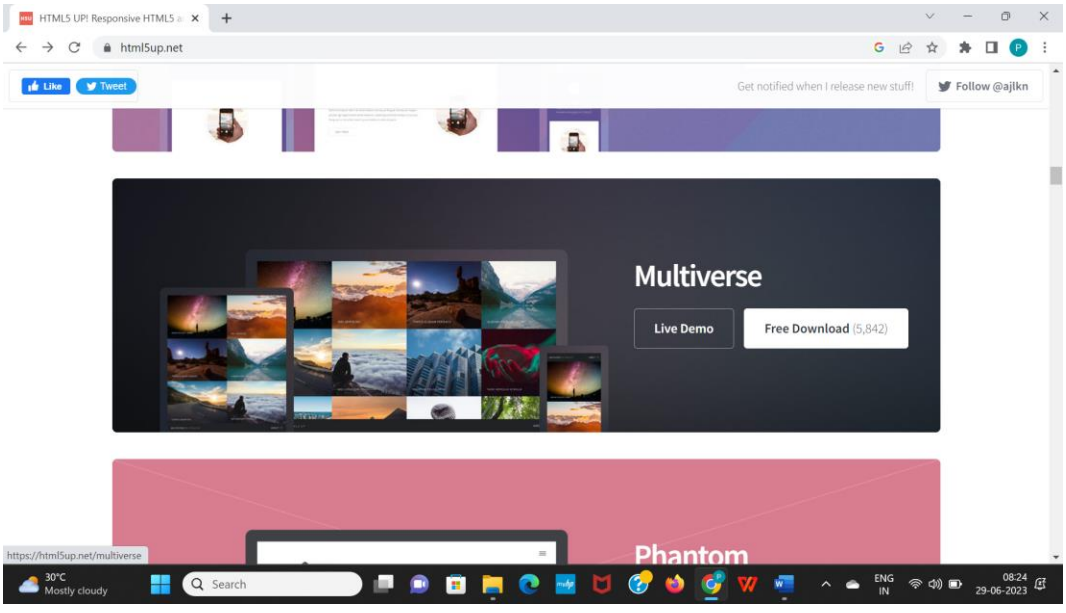

Fig 9.13 Html5 Template

Here's how you can grant public access to a downloaded template in an S3 bucket:

- Sign in to the AWS Management Console and open the Amazon S3 service.
- Locate and select the S3 bucket containing the downloaded template.
- Navigate to the object (template) you want to make public.
- Select the object by clicking the checkbox next to its name.
- Click on the "Actions" dropdown menu and choose "Make Public" or "Make Publicly Accessible."
- Confirm the action and save the changes.
- Enable static website hosting.

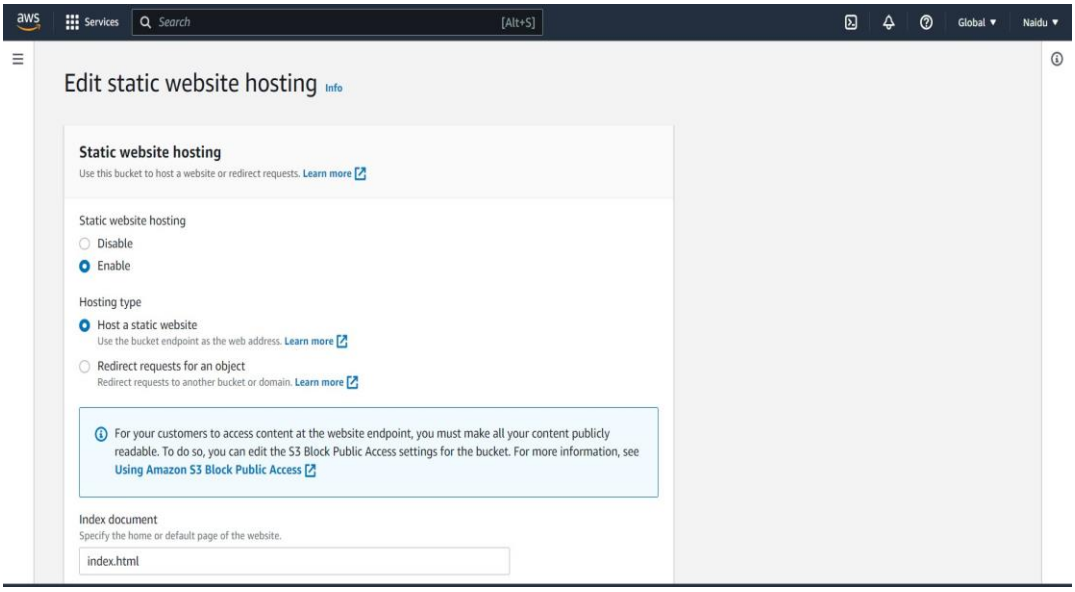

Fig 9.14 Enable Static Website Hosting

- After making the template public, it will be accessible to anyone who has the object's URL. They can either download it directly or use it in their applications or processes.
- Please note that making an object public in an S3 bucket removes any restrictions on access, allowing anyone with the object's URL to download it. Make sure to consider the sensitivity of the template and the potential implications of making it accessible to the public before granting public access.

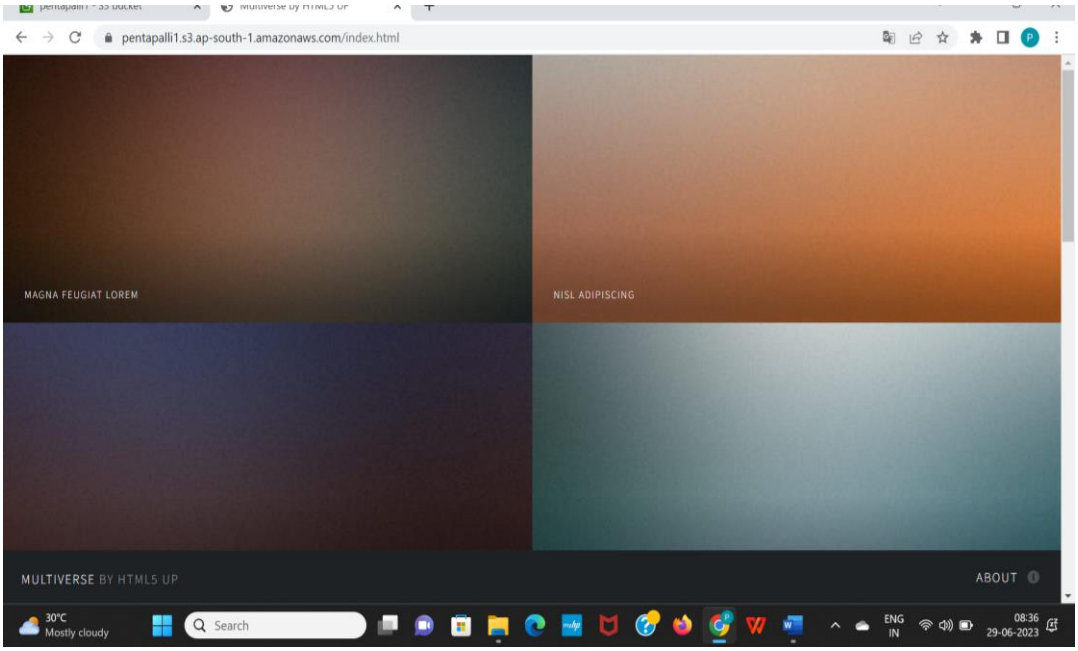

Fig 9.15 Template Opened Successfully

10.0 How to Download CLI (Command Line Interface):

The AWS Command Line Interface (AWS CLI) is a unified tool to manage your AWS services. With just one tool to download and configure, you can control multiple AWS services from the command line and automate them through scripts.

- Search for **AWS CLI Install**. Click on "AWS.amazon.com", download 64-bit windows installer.
- Extract the files from the package by using unzip. ...
- Run the install program. ...
- Ensure the directory that the AWS CLI version 1 is part of your PATH variable. ...
- Verify that the AWS CLI installed correctly.
- Enter command "AWS --Version".

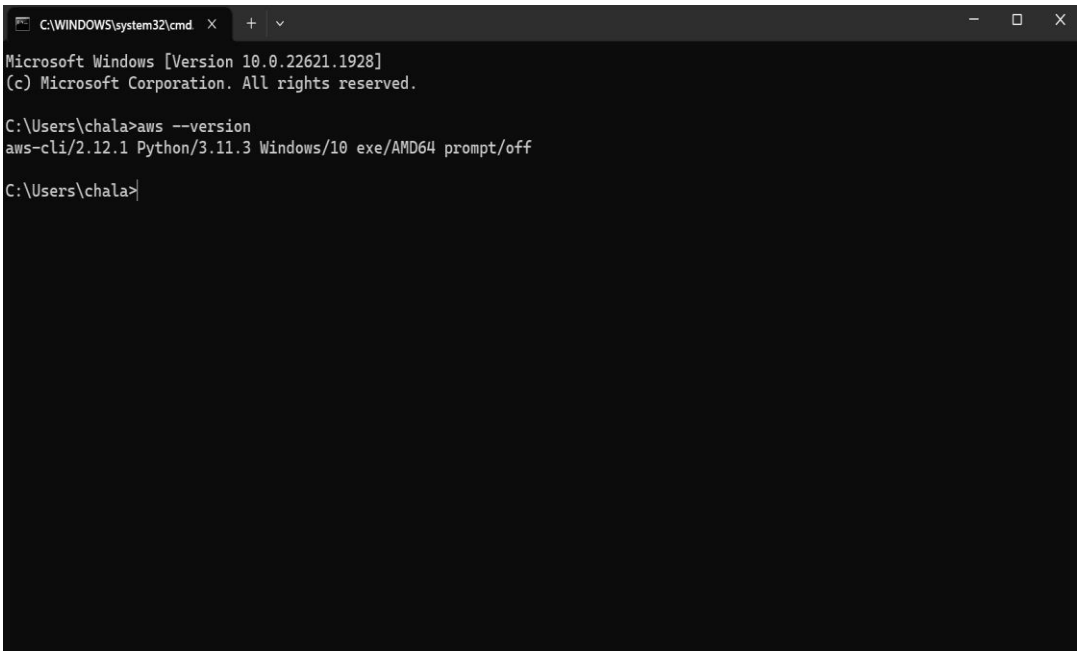

Fig 10.1 AWS CLI Installed

10.1 Login to IAM User in CLI:

Access keys are long-term credentials for an IAM user or the AWS account root user. You can use access keys to sign programmatic requests to the AWS CLI or AWS API (directly or using the AWS SDK).

- Create an IAM user with access key, download the access and secret access key, and add policy with s3 full access.
- Check that you've completed the Prerequisites.

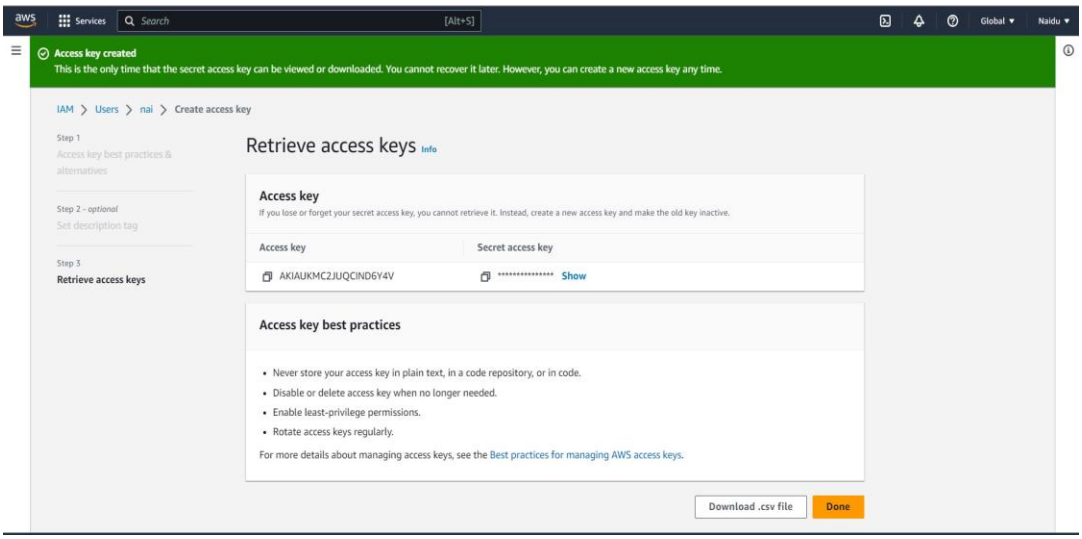

Fig 10.2 Access Key Created Successfully

- If you're signing in for the first time, configure your profile with the "aws configure" command in cli.
- After you configure your profile, run the following commands, to work on cli.
- Enter command "aws sts get-caller-identity" to check whether login to user is successful or not.

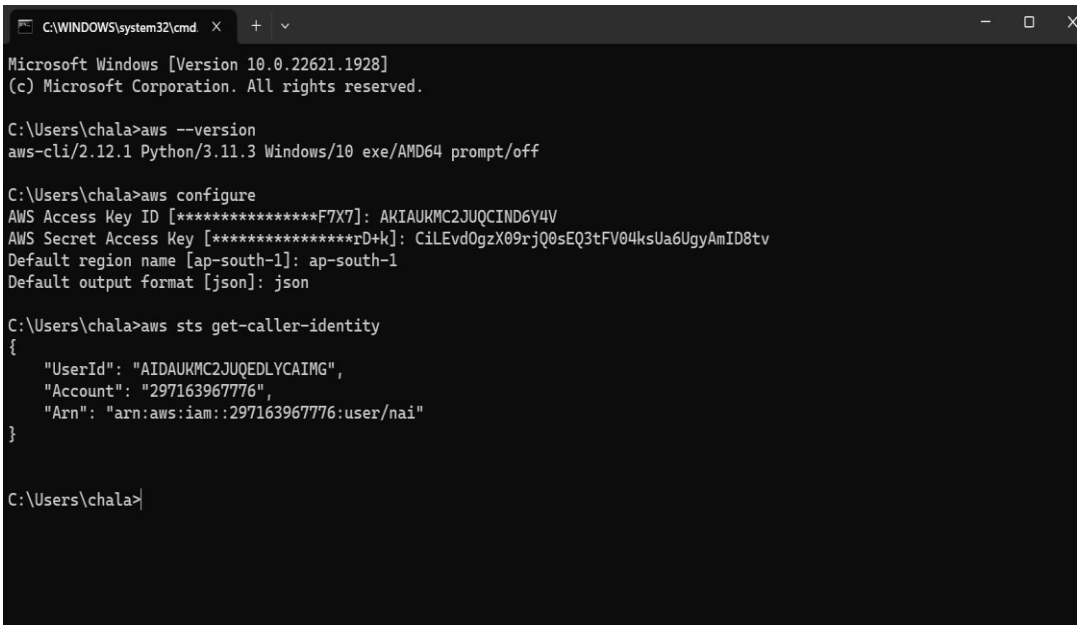

Fig 10.3 Login to IAM User

10.2 To create and Upload Single File into The Bucket Through CMD:

- To get list of buckets, enter command “aws s3 ls”.
- To create a bucket, enter command
  - “aws s3 mb s3://bucketname “.
- To get list of files and directories, enter command “dir”.

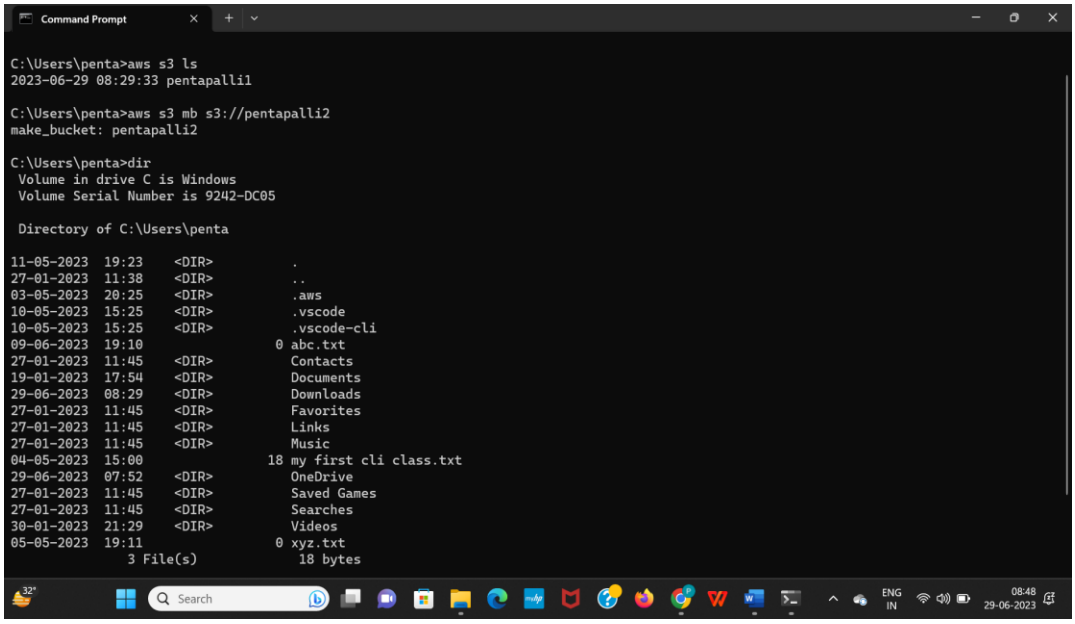

Fig 10.4 AWS Make Bucket and Directory

- To create a file through cmd enter command “aws cat > file name “.
- To upload a file into bucket, enter command “aws s3 cp filename s3://buckename “.
- To get list of files in the bucket, enter command “aws s3 ls s3://bucketname”.

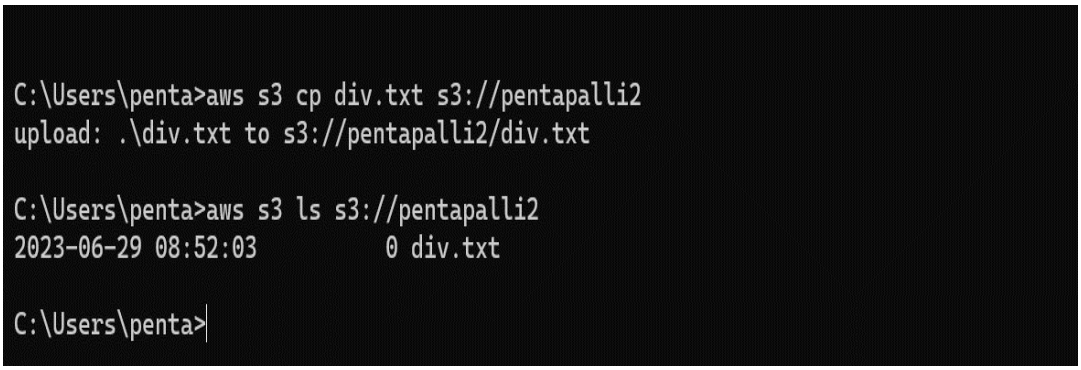

Fig 10.5 Upload Single File

10.3 To Upload Multiple Files in a Bucket:

- To upload multiple files or folder to a bucket through cmd, enter command "aws s3 cp "folder name" s3://bucketname –recursive".
- To check whether all the files uploaded or not enter command "aws s3 ls s3://bucket name".

```
C:\Users\penta>aws s3 cp "C:\Users\penta\Downloads\divya" s3://pentapalli2 --recursive
upload: Downloads\divya\div.txt to s3://pentapalli2/div.txt
upload: Downloads\divya\flower.webp to s3://pentapalli2/flower.webp
upload: Downloads\divya\TABLERDS.txt to s3://pentapalli2/TABLERDS.txt
upload: Downloads\divya\P3.pdf to s3://pentapalli2/P3.pdf

C:\Users\penta>aws s3 ls s3://pentapalli2
2023-06-29 08:56:06     422104 P3.pdf
2023-06-29 08:56:06        559 TABLERDS.txt
2023-06-29 08:56:06         79 div.txt
2023-06-29 08:56:06      27178 flower.webp

C:\Users\penta>
```

Fig 10.6 Uploading Multiple Files

10.4 To Download Single and Multiples Files in CMD:

- To download single file from cmd, enter command "aws s3 cp s3://bucketname/filename folder path ".
- To download multiple files through cmd, enter command "aws s3 cp s3://bucketname/foldername folder path –recursive".

```
C:\Users\penta>aws s3 cp s3://pentapalli2/div.txt "C:\Users\penta\Downloads\divya"
download: s3://pentapalli2/div.txt to Downloads\divya\div.txt

C:\Users\penta>aws s3 cp s3://pentapalli2/ "C:\Users\penta\Downloads\divya" --recursive
download: s3://pentapalli2/TABLERDS.txt to Downloads\divya\TABLERDS.txt
download: s3://pentapalli2/div.txt to Downloads\divya\div.txt
download: s3://pentapalli2/flower.webp to Downloads\divya\flower.webp
download: s3://pentapalli2/P3.pdf to Downloads\divya\P3.pdf

C:\Users\penta>
```

Fig 10.7 Downloading Single and Multiple Files

10.5 To Copy and Move Files From One Bucket to Another:

- To copy files from one bucket to another bucket through cmd, enter command "aws s3 cp s3://bucketname s3://2nd bucket --recursive"
- To moves files from one bucket to another bucket through cmd, enter command "aws s3 mv s3://1st bucket s3://2nd bucket –recursive".

```
C:\Users\penta>aws s3 cp s3://pentapalli2 s3://pentapalli3 --recursive
copy: s3://pentapalli2/div.txt to s3://pentapalli3/div.txt
copy: s3://pentapalli2/TABLERDS.txt to s3://pentapalli3/TABLERDS.txt
copy: s3://pentapalli2/P3.pdf to s3://pentapalli3/P3.pdf
copy: s3://pentapalli2/flower.webp to s3://pentapalli3/flower.webp

C:\Users\penta>aws s3 mv s3://pentapalli2 s3://pentapalli4 --recursive
move: s3://pentapalli2/flower.webp to s3://pentapalli4/flower.webp
move: s3://pentapalli2/P3.pdf to s3://pentapalli4/P3.pdf
move: s3://pentapalli2/TABLERDS.txt to s3://pentapalli4/TABLERDS.txt
move: s3://pentapalli2/div.txt to s3://pentapalli4/div.txt

C:\Users\penta>
```

Fig 10.8 Copy and Move Files

10.6 To Delete Objects and Bucket:

- To delete single file from bucket, enter command "aws s3 rm s3://bucketname/filename".
- To make a bucket empty by deleting all the files, enter command "aws s3 rm s3://bucketname/ --recursive".
- To delete bucket with files in it, enter command "aws s3 rb s3://bucketname –forcestop"
- To delete empty bucket, enter command "aws s3 rb s3://bucketname".

```
C:\Users\penta>aws s3 rm s3://pentapalli3 --recursive
delete: s3://pentapalli3/TABLERDS.txt
delete: s3://pentapalli3/flower.webp
delete: s3://pentapalli3/div.txt
delete: s3://pentapalli3/P3.pdf

C:\Users\penta>aws s3 rb s3://pentapalli3
remove_bucket: pentapalli3

C:\Users\penta>aws s3 ls
2023-06-29 08:29:33 pentapalli1
2023-06-29 08:47:12 pentapalli2
2023-06-29 09:04:08 pentapalli4

C:\Users\penta>
```

Fig 10.9 Deleting Objects and Buckets

10.7 To Create a Pre-Sign URL to Open File:

- To create pre-sign URL, enter command
  - "aws s3 presign s3://bucketname/filename with extension --expires-in time in sec ".
- Copy paste the url in web browser with in allocate time to view the file.

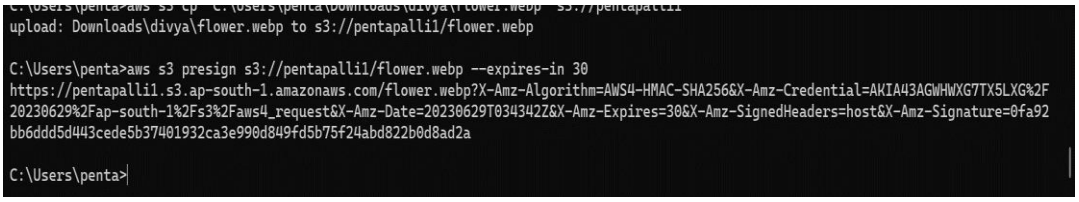

Fig 10.10 Pre-sign URL

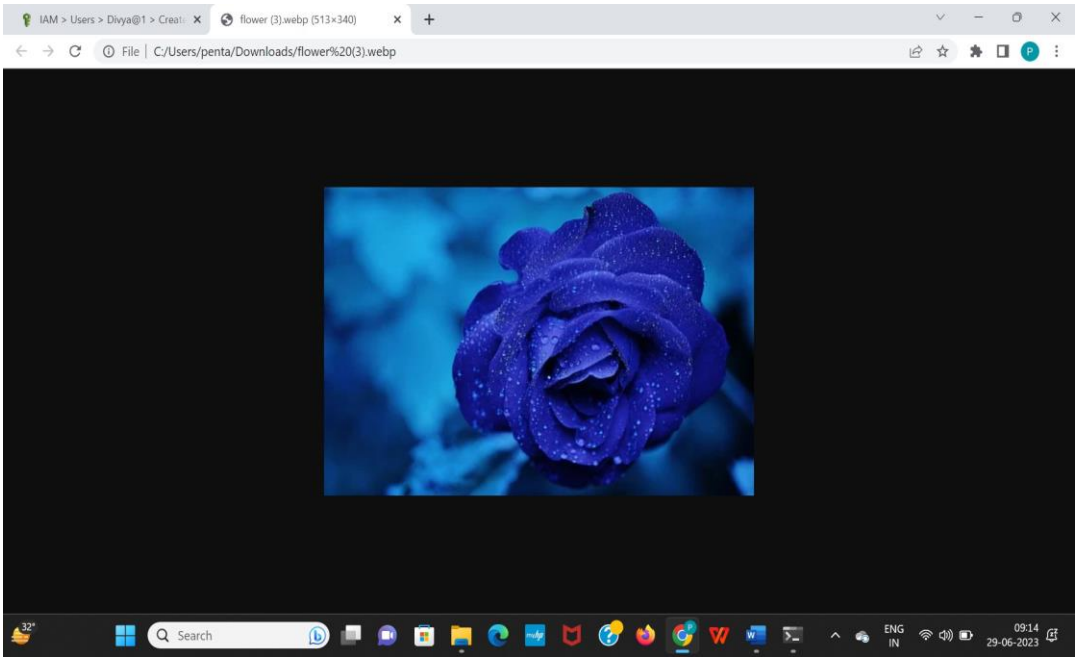

Fig 10.11 Opened Pre-sign URL

11.0 S3 Versioning:

Versioning in Amazon S3 is a means of keeping multiple variants of an object in the same bucket. You can use the S3 Versioning feature to preserve, retrieve, and restore every version of every object stored in your buckets. With versioning you can recover more easily from both unintended user actions and application failures. After versioning is enabled for a bucket, if Amazon S3 receives multiple write requests for the same object simultaneously, it stores all of those objects.

Versioning-enabled buckets can help you recover objects from accidental deletion or overwrite. For example, if you delete an object, Amazon S3 inserts a delete marker instead of removing the object permanently. The delete marker becomes the current object version. If you overwrite an object, it results in a new object version in the bucket. You can always restore the previous version.

11.1 Steps to Enable Bucket Versioning:

- Sign in to the AWS Management Console and open the Amazon S3 console at https://console.aws.amazon.com/s3/.
- In the Buckets list, choose the name of the bucket that you want to enable versioning for.
- Choose Properties.
- Under Bucket Versioning, choose Edit.
- Choose Suspend or Enable, and then choose Save changes.

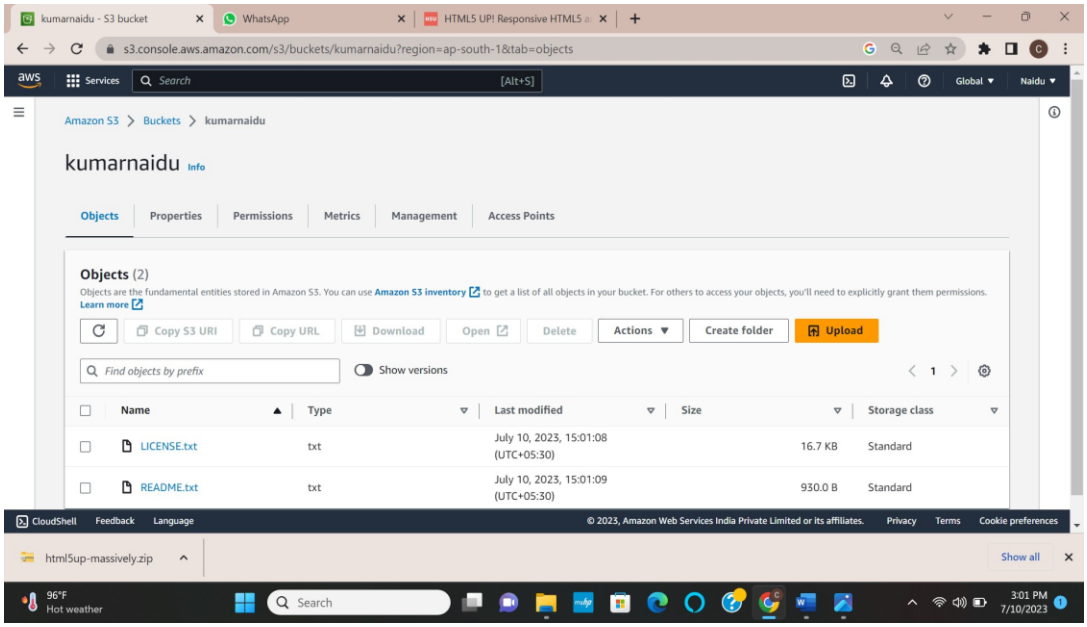

Fig 11.1 Bucket Versioning Enabled

12.0 Key Management Service (KMS):

AWS Key Management Service (KMS) gives you centralized control over the cryptographic keys used to protect your data. The service is integrated with other AWS services making it easier to encrypt data you store in these services and control access to the keys that decrypt it. AWS KMS is also integrated with AWS CloudTrail, which helps you audit who used which keys, on which resources, and when. AWS KMS helps developers to more easily add encryption or digital signature functionality to their application code either directly or by using the AWS SDK. The AWS Encryption SDK supports AWS KMS as a key provider for developers who need to encrypt/decrypt data locally within their applications.

AWS KMS provides you with centralized control over the lifecycle and permissions of your keys. You can create new keys whenever you want, and you can control who can manage keys separately from who can use them. As an alternative to using keys generated by AWS KMS, you can import keys from your own key management infrastructure, use keys stored in your AWS Cloud HSM cluster or keys kept outside AWS in your external key manager. You can choose automatic rotation of root keys generated in AWS KMS once per year without the need to re-encrypt previously encrypted data. The service automatically keeps older versions of the root key available to decrypt previously encrypted data. You can manage your root keys and audit their usage from the AWS Management Console or by using the AWS SDK or AWS Command Line Interface (CLI).

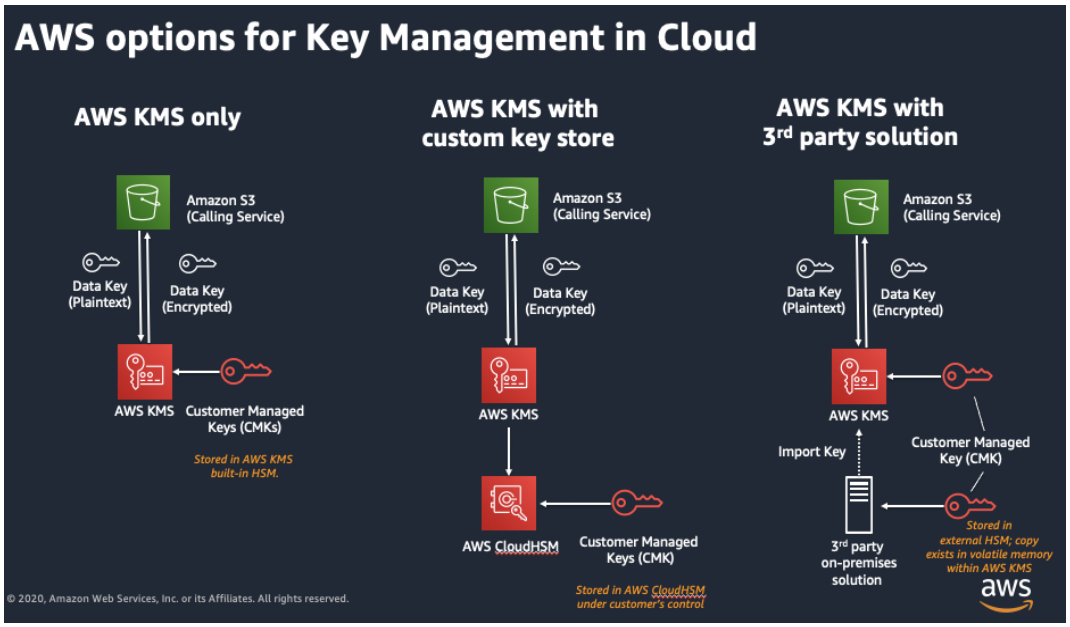

Fig 12.1 Key Management Service (KMS)

12.1 Creating a KMS:

- Sign in to the AWS Management Console and open the AWS Key Management Service (AWS KMS) console at https://console.aws.amazon.com/kms.
- To change the AWS Region, use the Region selector in the upper-right corner of the page (select ap-south-1).
- In the navigation pane, choose Customer managed keys.
- Choose Create key.
- To create a symmetric encryption KMS key, for Key type choose Symmetric.
- For information about how to create an asymmetric KMS key in the AWS KMS console, see Creating asymmetric KMS keys (console).
- In Key usage, the Encrypt and decrypt option is selected for you.
- Choose Next.

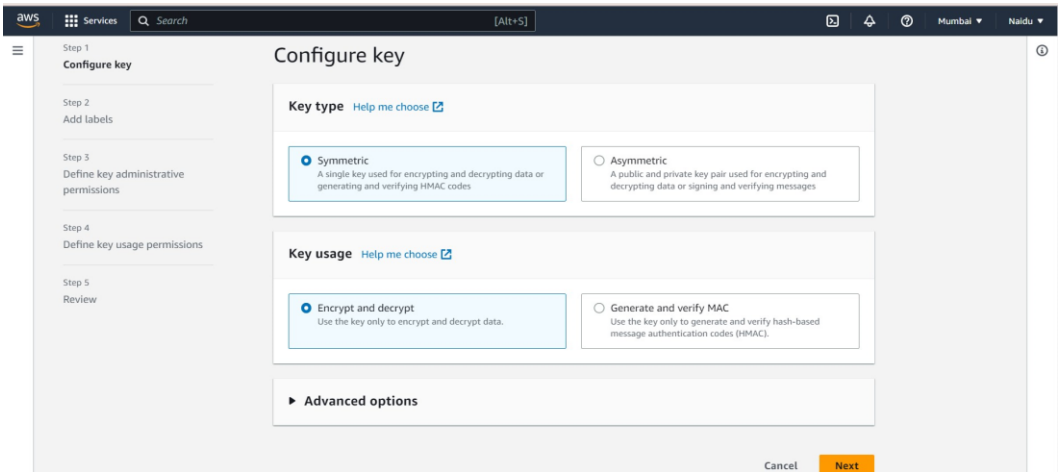

Fig 12.2 Step1: KMS

- Type an alias for the KMS key. The alias name cannot begin with aws/. The aws/ prefix is reserved by Amazon Web Services to represent AWS managed keys in your account.
- An alias is a display name that you can use to identify the KMS key.
- (Optional) Type a description for the KMS key.
- You can add a description now or update it any time unless the key state is Pending Deletion or Pending Replica Deletion. To add, change, or delete the description of an existing customer managed key, edit the description in the AWS Management Console or use the Update Key Description operation.
- (Optional) Type a tag key and an optional tag value. To add more than one tag to the KMS key, choose Add tag.
- Choose Next.

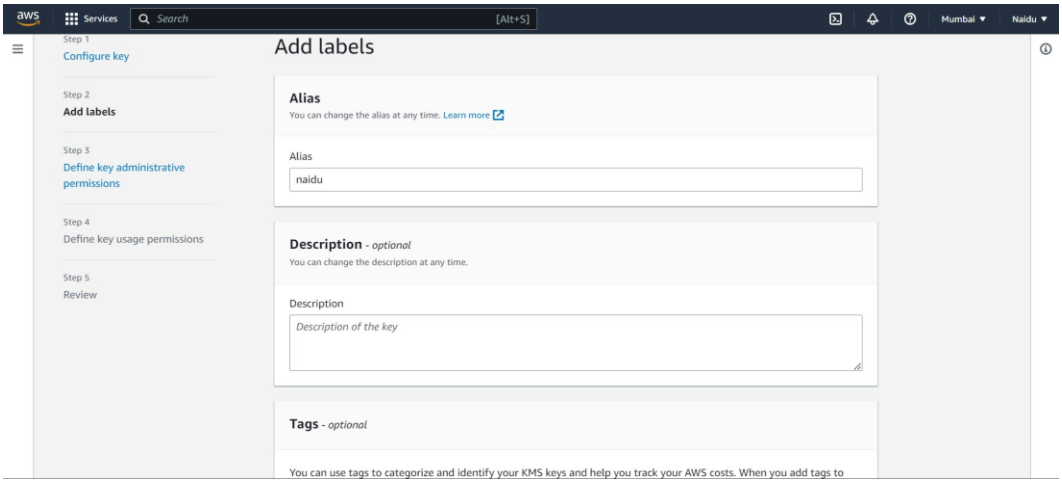

Fig 12.3 Step 2: KMS

- Select the IAM users and roles that can administer the KMS key.
- (Optional) To prevent the selected IAM users and roles from deleting this KMS key, in the Key deletion section at the bottom of the page, clear the Allow key administrators to delete this key check box.
- Choose Next.

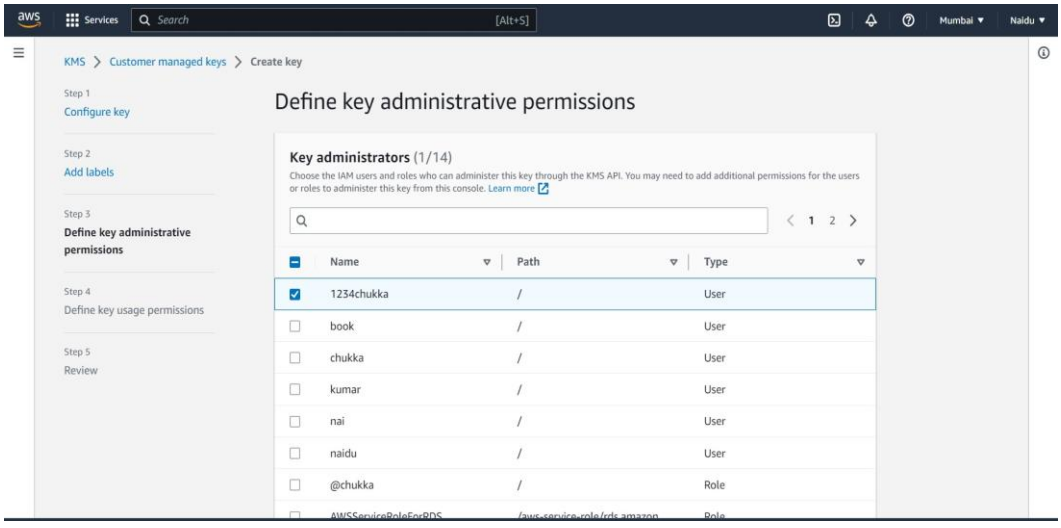

Fig 12.4 Step 3: KMS

- Select the IAM users and roles that can use the key in cryptographic operations
- (Optional) You can allow other AWS accounts to use this KMS key for cryptographic operations. To do so, in the Other AWS accounts section at the bottom of the page, choose Add another AWS account and enter the AWS

account identification number of an external account. To add multiple external accounts, repeat this step.

- Choose Next.

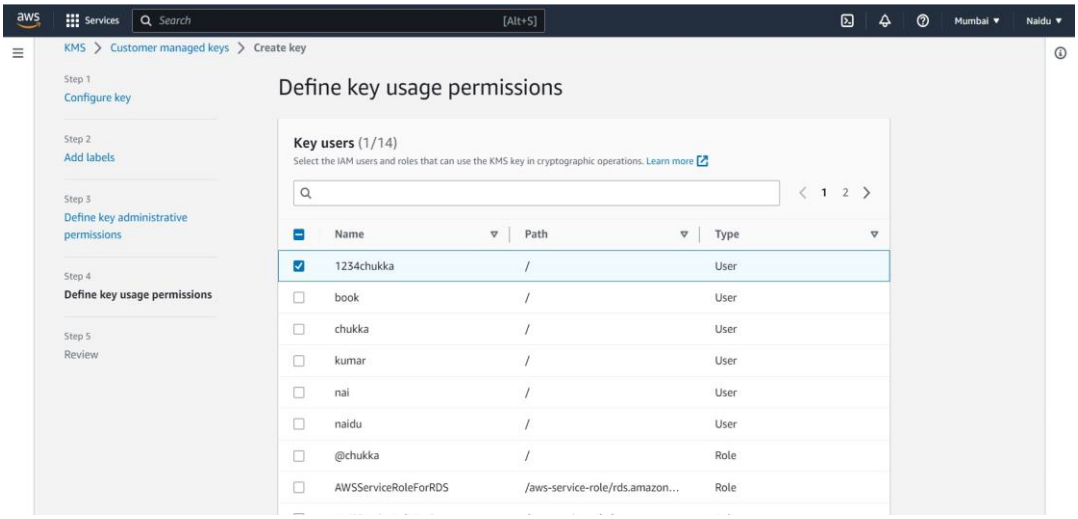

Fig 12.5 Step 4: KMS

- Review the key settings that you chose. You can still go back and change all settings.
- Choose Finish to create the KMS key.

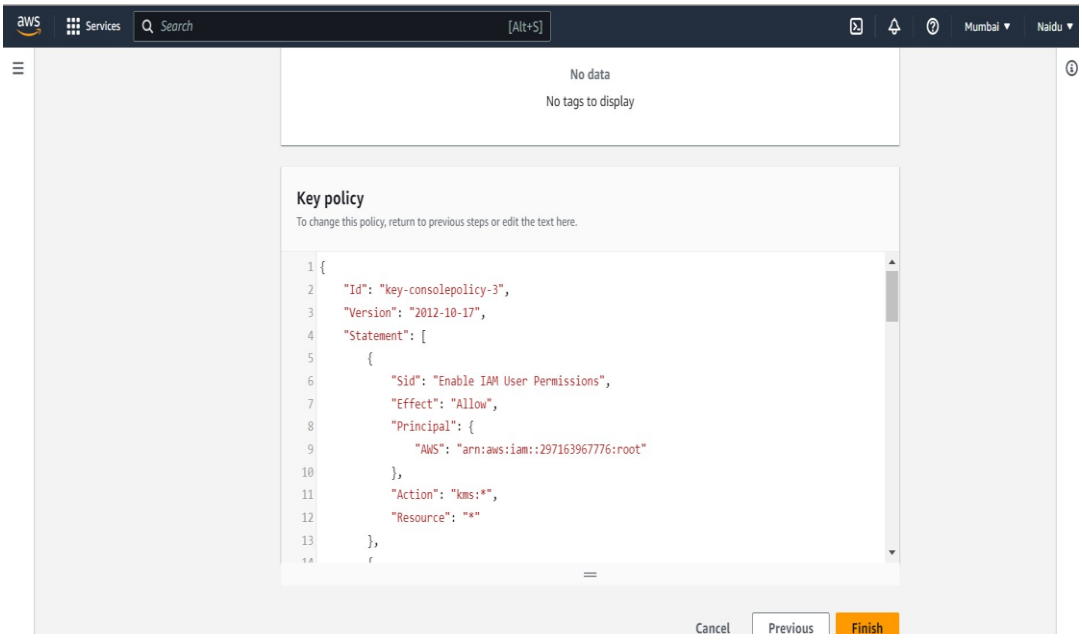

Fig 12.6 Step 5: KMS

12.2 S3 with KMS:

Amazon S3 integrates with AWS Key Management Service (AWS KMS) to provide server-side encryption of Amazon S3 objects. Amazon S3 uses AWS KMS keys to encrypt your Amazon S3 objects. The encryption keys that protect your objects never leave AWS KMS unencrypted. This integration also enables you to set permissions on the AWS KMS key and audit the operations that generate, encrypt, and decrypt the data keys that protect your secrets.

To reduce the volume of Amazon S3 calls to AWS KMS, use Amazon S3 bucket keys, which are KMS key-protected key-encryption-keys that are reused for a limited time within Amazon S3. Bucket keys can reduce costs for AWS KMS requests by up to 99 percent. You can configure a bucket key for all objects in an Amazon S3 bucket, or for a particular object in an Amazon S3 bucket. The bucket that is created with kms key will be only accessed by the administrator and user in kms.

To create a bucket with kms key enable bucket key while creating bucket and then choose kms key from your account and then click on create bucket.

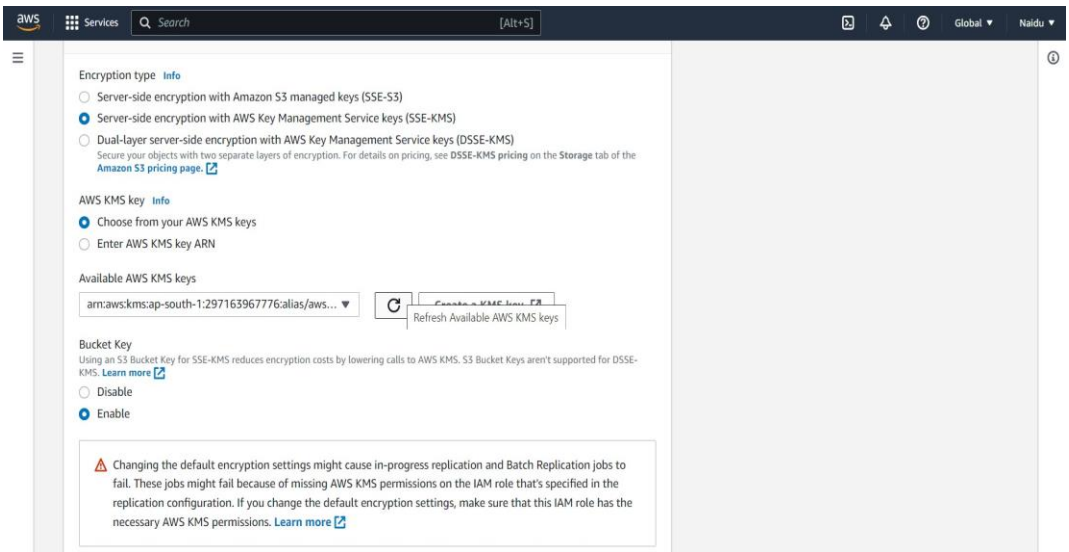

Fig 12.7 S3 with KMS Key

13.0 Replication Rule:

S3 Replication – Replicate objects and their respective metadata and object tags to one or more destination buckets in the same or different AWS Regions for reduced latency, compliance, security, and other use cases. Replication enables automatic, asynchronous copying of objects across Amazon S3 buckets. Buckets that are configured for object replication can be owned by the same AWS account or by different accounts. You can replicate objects to a single destination bucket or to multiple destination buckets. The destination buckets can be in different AWS Regions or within the same Region as the source bucket. You can view replication rules from the source bucket in your replication configuration.

to view important details about your replication rule, including the following:

- The replication rule's status (enabled or disabled)
- The rule's priority
- The scope of objects to replicate
- The destination bucket.
- The storage class of your replicated objects
- The object ownership of your replicated objects
- Whether objects encrypted by AWS Key Management Service (AWS KMS) keys are replicated
- The KMS key that's used for replica encryption
- Whether additional replication options are enabled, including S3 Replication Time Control (S3 RTC), replication metrics and notifications, delete marker replication, and replica modification sync

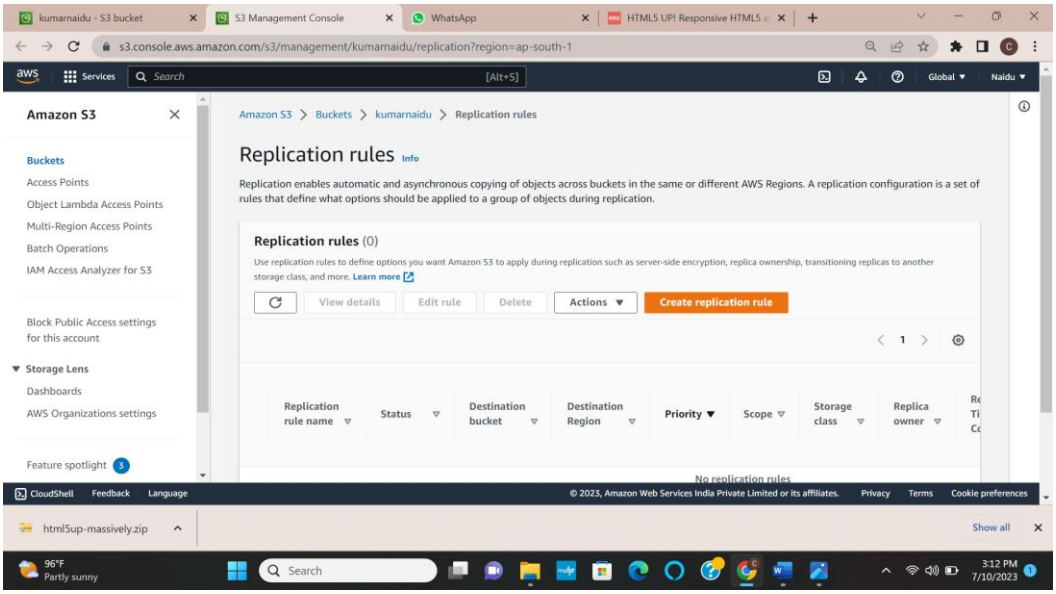

Fig13.1 Replication Rule

14.0 Inventory Configuration:

Inventory configuration in AWS refers to the process of collecting and storing metadata about your AWS resources. It helps you maintain an inventory of resources, track changes, and monitor compliance. AWS provides a service called AWS Config that allows you to define and manage inventory configurations.

- Creating inventory configuration.
- Open a web browser and sign in to the AWS Management Console.
- Click Services → Storage → S3 to open the Amazon S3 console. Click Bucket on the left and click a bucket on the list of buckets on the right.
- Click on the Management tab, scroll down, and click Create inventory configuration under Inventory configurations to create a new inventory configuration for the bucket. This action opens the Create inventory configuration page.
- Object versions: You can include object versions in the inventory report. Helpful for tracking different versions of the same object. For this tutorial, you'll only include the current version of each object.
- Destination bucket: Select This account since you will be the one accessing the report.
- Destination: Browse to the destination S3 bucket where you want to save the report and select it
- Destination bucket permission: This JSON editor sets the permissions on the destination bucket. You can leave the default permission since the S3 Management Console provides a sufficient default policy for this tutorial.
- Output format: Select CSV since most spreadsheet applications can read this format.
- Status: Select Enabled since you want the inventory to start generating reports right away.
- Server-side encryption: Select Disable. You will not use server-side encryption since it is not required.

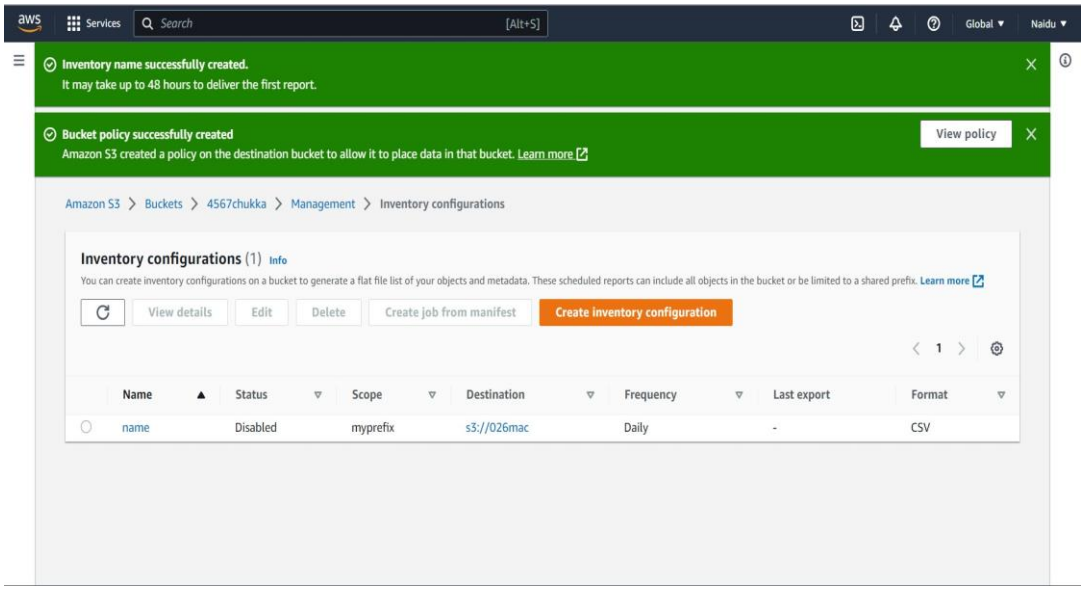

Fig 14.1 Inventory Configuration

15.0 Server Access Logs:

Server access logging provides detailed records for the requests that are made to an Amazon S3 bucket. Server access logs are useful for many applications. For example, access log information can be useful in security and access audits. It can also help you learn about your customer base and understand your Amazon S3 bill.

By default, Amazon S3 doesn't collect server access logs. When you enable logging, Amazon S3 delivers access logs for a source bucket to a target bucket that you choose. The target bucket must be in the same AWS Region and AWS account as the source bucket, and must not have a default retention period configuration. The target bucket must also not have Requester Pays enabled.

An access log record contains details about the requests that are made to a bucket. This information can include the request type, the resources that are specified in the request, and the time and date that the request was processed.

You can enable or disable server access logging by using the Amazon S3 console, Amazon S3 API, the AWS Command Line Interface (AWS CLI), or AWS SDKs.

- In the Buckets list, choose the name of the bucket that you want to enable server access logging for
- Choose Properties.
- In the Server access logging section, choose Edit.
- Under Server access logging, select Enable.
- For Target bucket, enter the name of the bucket that you want to receive the log record objects.
- The target bucket must be in the same Region as the source bucket, must be owned by the same AWS account as source bucket, and must not have a default retention period configuration. The target bucket must also not have Requester Pays enabled.
- Choose Save changes.
- When you enable server access logging on a bucket, the console both enables logging on the source bucket and updates the bucket policy for the target bucket to grant s3: Put Object permissions to the logging service principal.
- You can view the logs in the target bucket. After you enable server access logging, it might take a few hours before the logs are delivered to the target bucket.

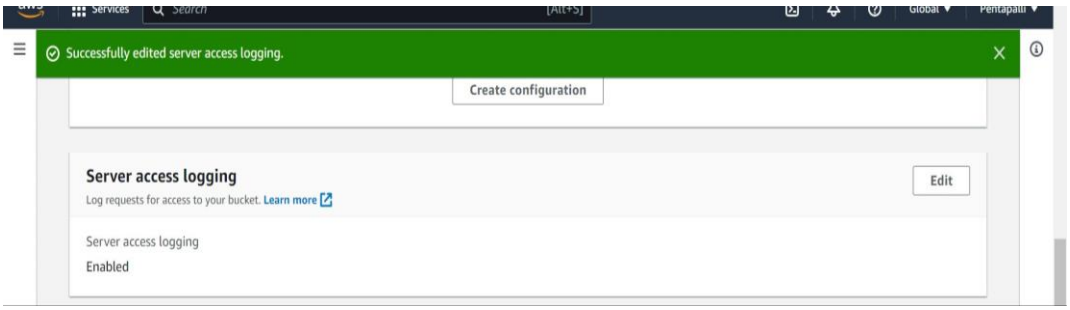

Fig 15.1 Server Access Logs

16.0 Life Cycle Rules:

This section explains how you can set a S3 Lifecycle configuration on a bucket using AWS SDKs, the AWS CLI, or the Amazon S3 console. For information about S3 Lifecycle configuration, see Managing your storage lifecycle.

You can use lifecycle rules to define actions that you want Amazon S3 to take during an object's lifetime (for example, transition objects to another storage class, archive them, or delete them after a specified period of time).

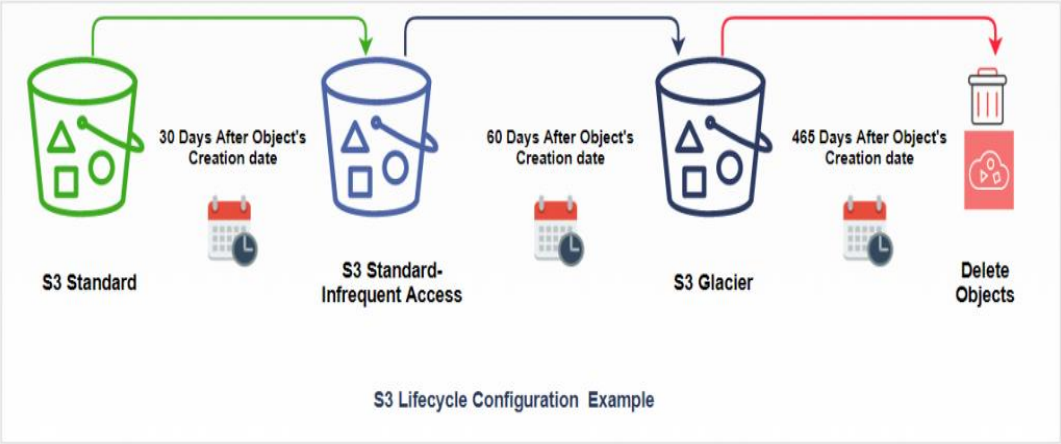

Fig 16.1 Life Cycles Rules

- Sign in to the AWS Management Console and open the Amazon S3 console at https://console.aws.amazon.com/s3/.
- In the **Buckets** list, choose the name of the bucket that you want to create a lifecycle rule.
- Choose the **Management** tab, and choose **Create lifecycle rule**.
- In **Lifecycle rule name**, enter a name for your rule.
  - The name must be unique within the bucket.
- Choose the scope of the lifecycle rule:
  - To apply this lifecycle rule to *all objects with a specific prefix or tag*, choose **Limit the scope to specific prefixes or tags**.
  - To limit the scope by prefix, in **Prefix**, enter the prefix.
  - To limit the scope by tag, choose **Add tag**, and enter the tag key and value.

- For more information about object name prefixes, see Creating object key names. For more information about object tags, see Categorizing your storage using tags.
- To apply this lifecycle rule to *all objects in the bucket*, choose **This rule applies to *all* objects in the bucket**, and choose **I acknowledge that this rule applies to all objects in the bucket**.

- To filter a rule by object size, you can check **Specify minimum object size**, **Specify maximum object size**, or both options.
  - When you're specifying a **minimum object size** or **maximum object size**, the value must be larger than 0 bytes and up to 5TB. You can specify this value in bytes, KB, MB, or GB.
  - When you're specifying both, the maximum object size must be larger than the minimum object size.
- Under **Lifecycle rule actions**, choose the actions that you want your lifecycle rule to perform:
  - Transition *current* versions of objects between storage classes
  - Transition *previous* versions of objects between storage classes
  - Expire *current* versions of objects
  - Permanently delete *previous* versions of objects
  - Delete expired delete markers or incomplete multipart uploads
  - Depending on the actions that you choose, different options appear.
- To transition *current* versions of objects between storage classes, under **Transition current versions of objects between storage classes**:

  In **Storage class transitions**, choose the storage class to transition to:
  - Standard-IA
  - Intelligent-Tiering
  - One Zone-IA
  - S3 Glacier Flexible Retrieval
  - Glacier Deep Archive
  - In **Days after object creation**, enter the number of days after creation to transition the object.
  - For more information about storage classes, see Using Amazon S3 storage classes. You can define transitions for current or previous object versions or for both current and previous versions. Versioning enables you to keep multiple versions of an object in one bucket. For more information about versioning, see Using the S3 console.

- To transition *non-current* versions of objects between storage classes, under **Transition non-current versions of objects between storage classes**:
  In **Storage class transitions**, choose the storage class to transition to:
  - Standard-IA
  - Intelligent-Tiering
  - One Zone-IA
  - S3 Glacier Flexible Retrieval
  - Glacier Deep Archive
  - In **Days after object becomes non-current**, enter the number of days after creation to transition the object.
- To expire *current* versions of objects, under **Expire current versions of objects**, in **Number of days after object creation**, enter the number of days.
- To permanently delete previous versions of objects, under **Permanently delete noncurrent versions of objects**, in **Days after objects become noncurrent**, enter the number of days. You can optionally specify the number of newer versions to retain by entering a value under **Number of newer versions to retain**.
- Under **Delete expired delete markers or incomplete multipart uploads**, choose **Delete expired object delete markers** and **delete incomplete multipart uploads**. Then, enter the number of days after the multipart upload initiation that you want to end and clean up incomplete multipart uploads.
  - For more information about multipart uploads, see Uploading and copying objects using multipart upload.
- Choose **Create rule**.

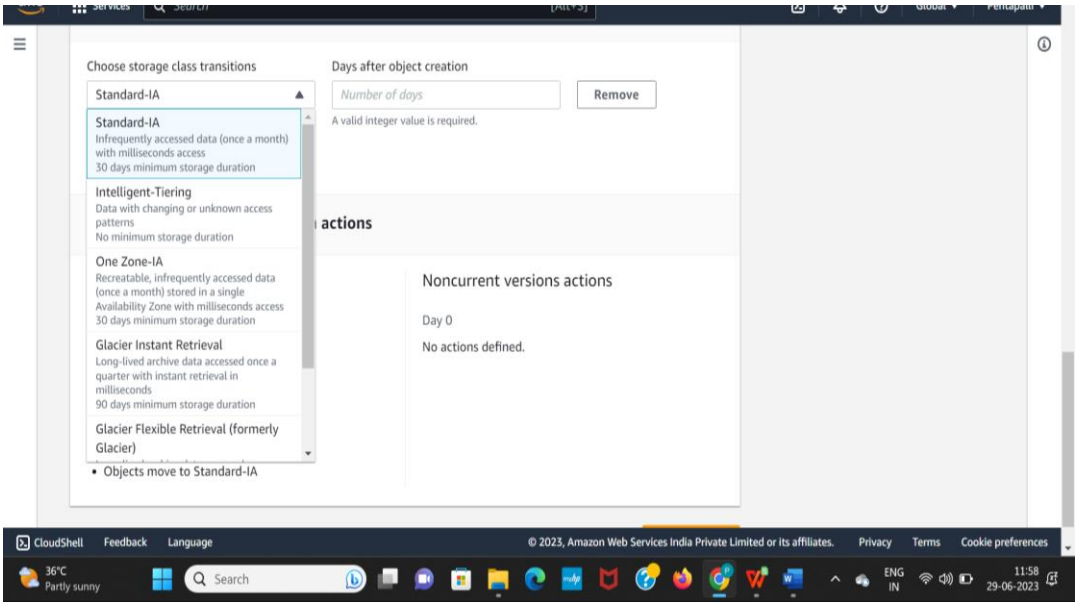

Fig 16.2 Creating Life Cycle Rule

17.0 Elastic Compute Cloud (EC2):

Amazon Elastic Compute Cloud (Amazon EC2) provides on-demand, scalable computing capacity in the Amazon Web Services (AWS) Cloud. Using Amazon EC2 reduces hardware costs so you can develop and deploy applications faster. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage. You can add capacity (scale up) to handle compute-heavy tasks, such as monthly or yearly processes, or spikes in website traffic. When usage decreases, you can reduce capacity (scale down) again.

The following diagram shows a basic architecture of an Amazon EC2 instance deployed within an Amazon Virtual Private Cloud (VPC). In this example, the EC2 instance is within an Availability Zone in the Region. The EC2 instance is secured with a security group, which is a virtual firewall that controls incoming and outgoing traffic. A private key is stored on the local computer and a public key is stored on the instance. Both keys are specified as a key pair to prove the identity of the user. In this scenario, the instance is backed by an Amazon EBS volume.

Fig 17.1 EC2

17.1 Features of Amazon EC2:

Amazon EC2 provides the following high-level features:

- Instance types
- Keypairs
- Instance store volumes
- Amazon EBS volumes
- Security groups
- Elastic IP address
- Tags
- VPC'S

17.1.1 Instance Types:

Various configurations of CPU, memory, storage, networking capacity, and graphics hardware for your instances.

17.1.2 Key Pairs:

Secure login information for your instances. AWS stores the public key and you store the private key in a secure place.

17.1.3 Instance Store Volumes:

Storage volumes for temporary data that is deleted when you stop, hibernate, or terminate your instance.

17.1.4 Amazon EBS Volumes:

Persistent storage volumes for your data using Amazon Elastic Block Store (Amazon EBS). Regions, Availability Zones, Local Zones, AWS Outposts, and Wavelength Zones, Multiple physical locations for your resources, such as instances and Amazon EBS volumes.

17.1.5 Security Groups:

A virtual firewall that allows you to specify the protocols, ports, and source IP ranges that can reach your instances, and the destination IP ranges to which your instances can connect.

17.1.6 Elastic IP Addresses:

Static IPv4 addresses for dynamic cloud computing.

17.1.7 Tags:

Metadata that you can create and assign to your Amazon EC2 resources.

17.1.8 Virtual Private Cloud (VPC):

Virtual networks you can create that are logically isolated from the rest of the AWS Cloud. You can optionally connect these virtual networks to your own network.

17.2 To Launch EC2 Instance in Windows:

- Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
- From the EC2 console dashboard, in the Launch instance box, choose Launch instance, and then choose Launch instance from the options that appear.
- Under Name and tags, for Name, enter a descriptive name for your instance.
- Under Application and OS Images (Amazon Machine Image), do the following:
- Choose Quick Start, and then choose **Windows.** This is the operating system (OS) for your instance.
- From Amazon Machine Image (AMI), select the AMI for Windows Server 2016 Base or later. Notice that these AMIs are marked Free tier eligible. An Amazon Machine Image (AMI) is a basic configuration that serves as a template for your instance.
- Under Instance type, from the Instance type list, you can select the hardware configuration for your instance. Choose **the t2.micro** instance type, which is selected by default. The t2.micro instance type is eligible for the free tier. In Regions where t2.micro is unavailable, you can use a t3.micro instance under the free tier. For more information, see AWS Free Tier.
- Under Key pair (login), for Key pair name, choose the key pair that you created when getting set up. Note that you must select an **RSA key**. ED25519 keys are not supported for Windows instances.
- Warning
- Do not choose Proceed without a key pair (Not recommended). If you launch your instance without a key pair, then you can't connect to it.
- Next to Network settings, choose Edit. For Security group name, you'll see that the wizard created and selected a security group for you. You can use this security group, or alternatively you can select the security group that you created when getting set up using the following steps:
- Choose Select existing security group.
- From Common security groups, choose your security group from the list of existing security groups.
- Keep the default selections for the other configuration settings for your instance.
- Review a summary of your instance configuration in the Summary panel, and when you're ready, choose Launch instance.

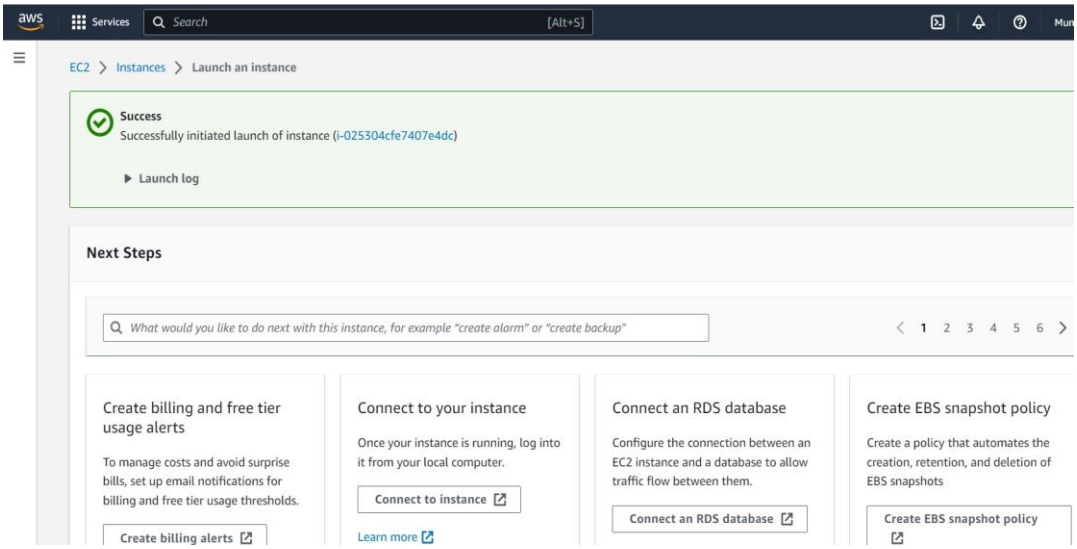

Fig 17.2 EC2 Windows

A confirmation page lets you know that your instance is launching. Choose View all instances to close the confirmation page and return to the console.

On the Instances screen, you can view the status of the launch. It takes a short time for an instance to launch. When you launch an instance, its initial state is pending. After the instance starts, its state changes to running and it receives a public DNS name. If the Public IPv4 DNS column is hidden, choose the settings icon (Settings icon.) in the top-right corner, toggle on Public IPv4 DNS, and choose Confirm.

It can take a few minutes for the instance to be ready for you to connect to it. Check that your instance has passed its status checks; you can view this information in the Status check column.

17.2.1 Connect to Your Instance:

To connect to a Windows instance, you must retrieve the initial administrator password and then enter this password when you connect to your instance using Remote Desktop. It takes a few minutes after instance launch before this password is available.

17.2.2 Connect to Your Windows Instance Using an RDP Client:

- Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
- In the navigation pane, select Instances. Select the instance and then choose Connect.
- On the Connect to instance page, choose the RDP client tab, and then choose Get password.

- Choose Browse and navigate to the private key (.pem) file you created when you launched the instance. Select the file and choose Open to copy the entire contents of the file to this window.
- Choose Decrypt Password. The console displays the default administrator password for the instance under Password, replacing the Get password link shown previously. Save the password in a safe place. This password is required to connect to the instance.
- Choose Download remote desktop file. Your browser prompts you to either open or save the RDP shortcut file. When you have finished downloading the file, choose Cancel to return to the Instances page.
- If you opened the RDP file, you'll see the Remote Desktop Connection dialog box.
- If you saved the RDP file, navigate to your download's directory, and open the RDP file to display the dialog box.
- You may get a warning that the publisher of the remote connection is unknown. Choose Connect to continue to connect to your instance.
- The administrator account is chosen by default. Copy and paste the password that you saved previously.
- Due to the nature of self-signed certificates, you may get a warning that the security certificate could not be authenticated. Use the following steps to verify the identity of the remote computer, or simply choose Yes (Windows) or Continue (Mac OS X) if you trust the certificate.
    - If you are using Remote Desktop Connection on a Windows computer, choose View certificate. If you are using Microsoft Remote Desktop on a Mac, choose Show Certificate.
    - Choose the Details tab, and scroll down to Thumbprint (Windows) or SHA1 Fingerprints (Mac OS X). This is the unique identifier for the remote computer's security certificate.
    - In the Amazon EC2 console, select the instance, choose Actions, Monitor and troubleshoot, Get system log.
    - In the system log output, look for RDPCERTIFICATE-THUMBPRINT. If this value matches the thumbprint or fingerprint of the certificate, you have verified the identity of the remote computer.
    - If you are using Remote Desktop Connection on a Windows computer, return to the Certificate dialog box and choose OK. If you are using Microsoft Remote Desktop on a Mac, return to the Verify Certificate and choose Continue.
    - [Windows] Choose Yes in the Remote Desktop Connection window to connect to your instance.

- [Mac OS X] Log in as prompted, using the default administrator account and the default administrator password that you recorded or copied previously. Note that you might need to switch spaces to see the login screen. For more information, see Add spaces and switch between them.

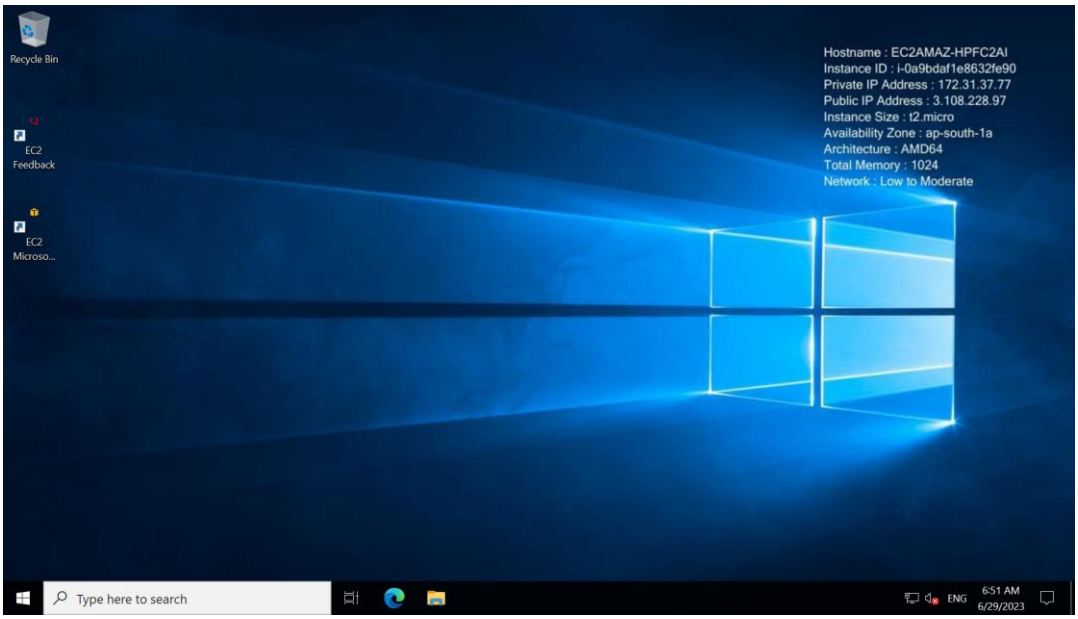

Fig 17.3 Remote Desktop for Windows Opened

17.2.3 To Terminate Instance:

- In the navigation pane, choose **Instances**. In the list of instances, select the instance.
- Choose **Instance state**, **Terminate instance**.
- Choose **Terminate** when prompted for confirmation.
- Amazon EC2 shuts down and terminates your instance. After your instance is terminated, it remains visible on the console for a short while, and then the entry is automatically deleted. You cannot remove the terminated instance from the console display yourself.

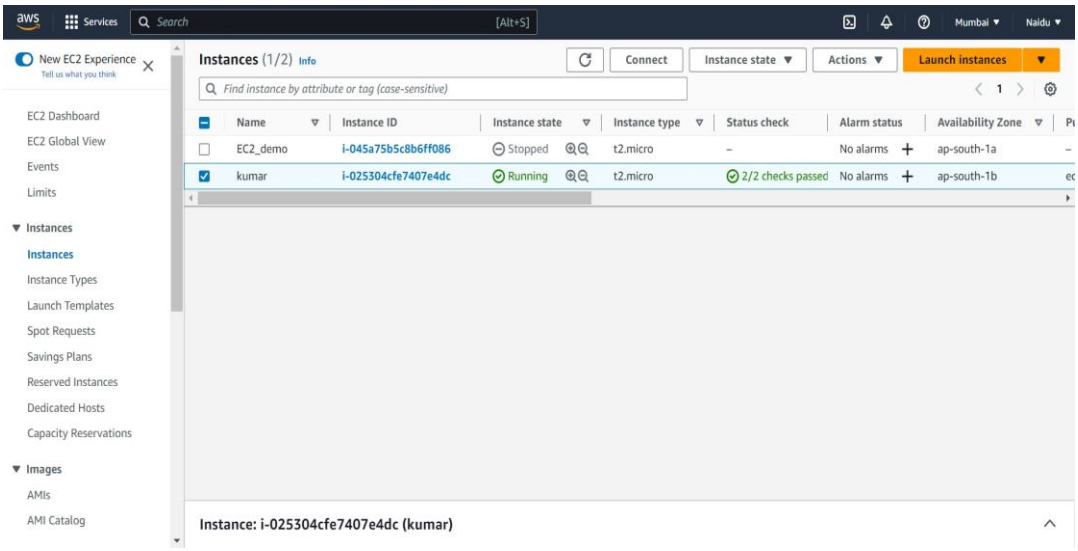

Fig 17.4 EC2 Terminated

17.3 To Launch EC2 Instance Through Linux:

- Launch Instance: On the EC2 Dashboard, click on the "Launch Instance" button to start the Linux instance creation process.
- Choose an Amazon Machine Image (AMI): Select the AMI that matches your requirements. An AMI is a pre-configured template that contains the operating system and other software for your instance.
- Choose an Instance Type: Select the instance type based on your needs. The instance type determines the hardware of the host computer used for the instance.
- Configure Instance Details: Configure the instance details, such as the number of instances you want to launch, network settings, security groups, and more. You can customize these settings according to your requirements.
- Add Storage: Specify the amount of storage you need for your instance. You can add additional volumes or modify the default settings as needed.
- Configure Security Group: Configure the security group to control inbound and outbound traffic to your instance. You can specify the protocols, ports, and IP ranges that are allowed to access your instance.
- Review Instance Launch: Review all the settings you've configured for your instance. If everything looks correct, click on "Launch" to proceed.
- Create a Key Pair: Select an existing key pair or create a new one. A key pair allows you to securely connect to your instance using SSH. If you create a new key pair, make sure to download and save the private key file (.pem) in a secure location, as you won't be able to download it again.
- Launch Instances: Once you've selected or created a key pair, click on "Launch Instances" to launch your EC2 instances.

- Access Your Instance: Once the instance is launched, you can access it using SSH or other remote access methods. Use the private key file (.pem) associated with your key pair to establish a secure connection to your instance.

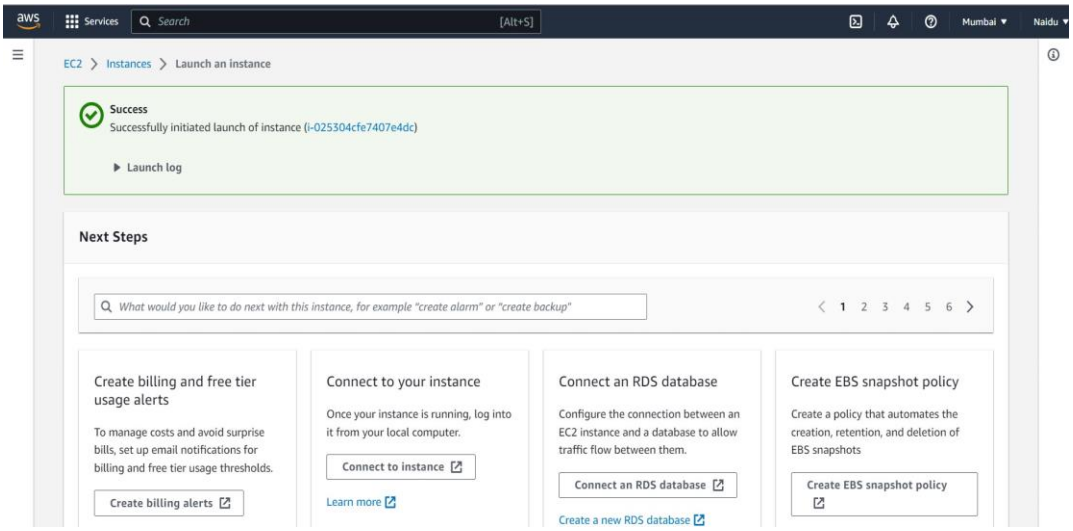

Fig 17.5 EC2 Linux

17.3.1 How to Install MobaXterm:

- **Search for MobaXterm download on Google.** Search MobaXterm download and click on the first link.
- **Select Edition.** We'll select the Home Edition as it has limited features free to use. Click on Download now button.
- **Select the version for the home edition.** MobaXterm home edition is available in two versions i.e., Portable and Installer. We'll choose Portable Edition. Click on the Portable Edition button.
- **Extract the downloaded file contents.**
- **Initialize the software.** After extracting, an executable file will be displayed as **MobaXterm_Personal_22.2.exe**. Click on that file.
- **MobaXterm will be started.** Click on **Start local terminal.**

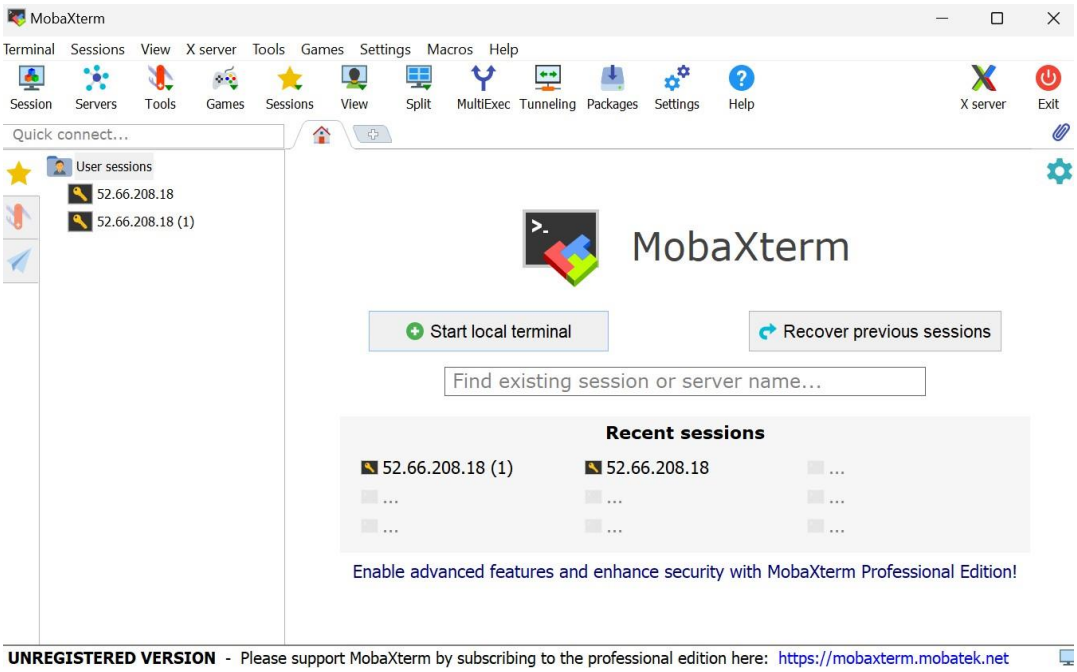

Fig 17.6 MobaXterm

17.3.2 To Connect EC2 Instance to MobaXterm:

- Amazon EC2 Instance Connect provides a simple and secure way to connect to your Linux instances using Secure Shell (SSH). With EC2 Instance Connect
- You can use EC2 Instance Connect to connect to your instances using the Amazon EC2 console or an SSH client of your choice.
- When you connect to an instance using EC2 Instance Connect, the Instance Connect API pushes an SSH public key
- Specify SSH username: In the "Specify username" field, enter the username for your Linux instance. The default username for most Linux AMIs is "ec2-user". However, depending on the AMI you are using, it may be different (e.g., "ubuntu" for Ubuntu-based AMIs).
- Specify the private key: In the "Advanced SSH settings" section, under the "Use private key" option, select the checkbox and browse to the location where you have saved your private key (.pem) file that corresponds to the key pair you used when launching the EC2 instance.
- Connect to the EC2 instance: In the Session Manager, select the session you just created and click on the "Start" button to establish the SSH connection to your EC2 instance.
- Authenticate with the private key: If the private key and settings are correct, MobaXterm will establish the SSH connection to your EC2 instance.

- Once the connection is established, you will have a terminal window within MobaXterm through which you can interact with your EC2 Linux instance. You can run commands, transfer files, and perform other tasks as needed.

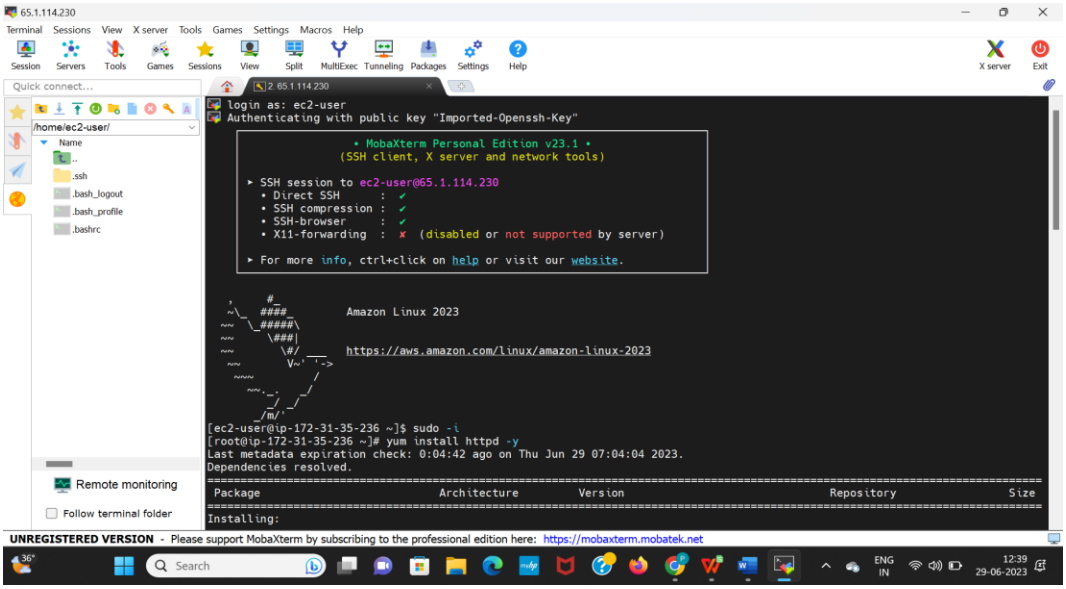

Fig 17.7 Login to User

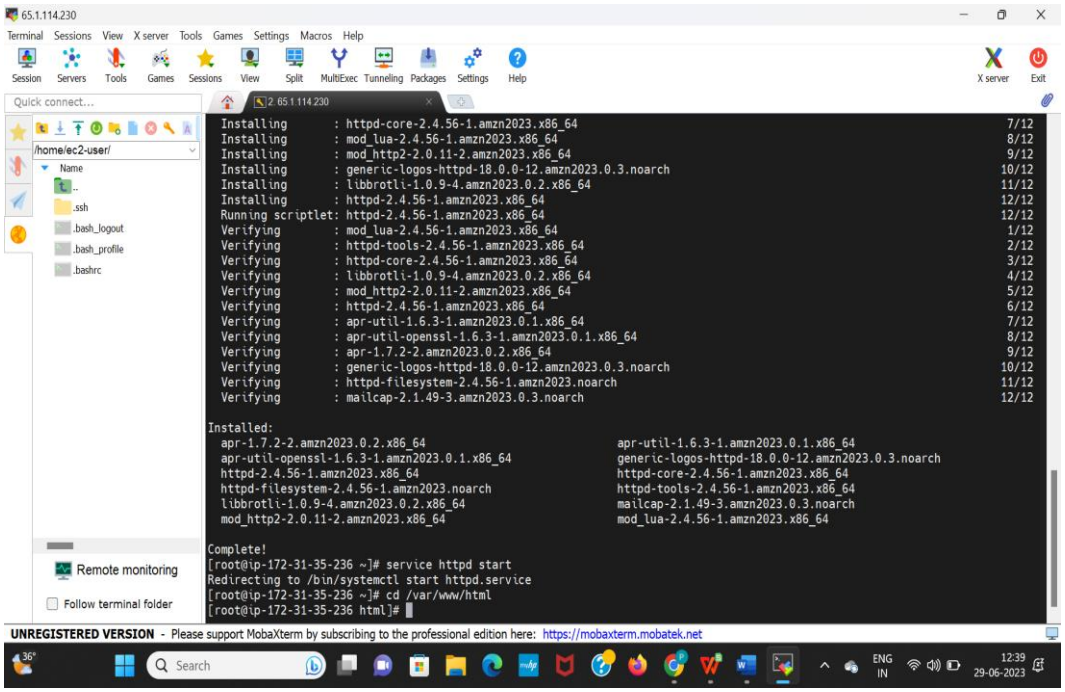

Fig 17.8 Start Server and Create a HTML File

18.0 Elastic Block Storage (EBS):

Amazon Elastic Block Store (Amazon EBS) provides block level storage volumes for use with EC2 instances. EBS volumes behave like raw, unformatted block devices. You can mount these volumes as devices on your instances. EBS volumes that are attached to an instance are exposed as storage volumes that persist independently from the life of the instance. You can create a file system on top of these volumes, or use them in any way you would use a block device (such as a hard drive). You can dynamically change the configuration of a volume attached to an instance.

We recommend Amazon EBS for data that must be quickly accessible and requires long-term persistence. EBS volumes are particularly well-suited for use as the primary storage for file systems, databases, or for any applications that require fine granular updates and access to raw, unformatted, block-level storage. Amazon EBS is well suited to both database-style applications that rely on random reads and writes, and to throughput-intensive applications that perform long, continuous reads and writes.

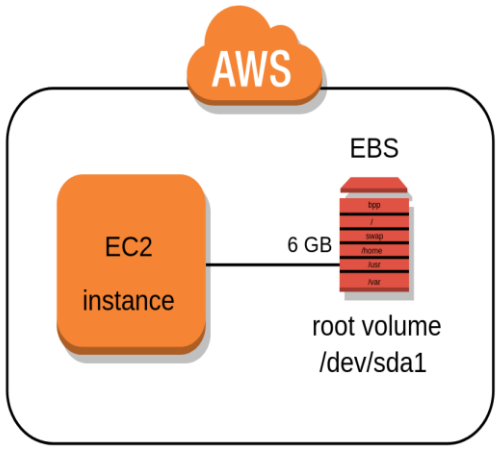

Fig 18.1 Elastic Block Storage (EBS)

18.1 Features of EBS:

- You create an EBS volume in a specific Availability Zone, and then attach it to an instance in that same Availability Zone. To make a volume available outside of the Availability Zone, you can create a snapshot and restore that snapshot to a new volume anywhere in that Region. You can copy snapshots to other Regions and then restore them to new volumes there, making it easier to

leverage multiple AWS Regions for geographical expansion, data center migration, and disaster recovery.

- Amazon EBS provides the following volume types: General Purpose SSD, Provisioned IOPS SSD, Throughput Optimized HDD, and Cold HDD.
    - The following is a summary of performance and use cases for each volume type.
    - General Purpose SSD volumes (gp2 and gp3) balance price and performance for a wide variety of transactional workloads. These volumes are ideal for use cases such as boot volumes, medium-size single instance databases, and development and test environments.
    - Provisioned IOPS SSD volumes (io1 and io2) are designed to meet the needs of I/O-intensive workloads that are sensitive to storage performance and consistency. They provide a consistent IOPS rate that you specify when you create the volume. This enables you to predictably scale to tens of thousands of IOPS per instance. Additionally, io2 volumes provide the highest levels of volume durability.
    - Throughput Optimized HDD volumes (st1) provide low-cost magnetic storage that defines performance in terms of throughput rather than IOPS. These volumes are ideal for large, sequential workloads such as Amazon EMR, ETL, data warehouses, and log processing.
    - Cold HDD volumes (sc1) provide low-cost magnetic storage that defines performance in terms of throughput rather than IOPS. These volumes are ideal for large, sequential, cold-data workloads. If you require infrequent access to your data and are looking to save costs, these volumes provide inexpensive block storage.
- You can create your EBS volumes as encrypted volumes, in order to meet a wide range of data-at-rest encryption requirements for regulated/audited data and applications. When you create an encrypted EBS volume and attach it to a supported instance type, data stored at rest on the volume, disk I/O, and snapshots created from the volume are all encrypted. The encryption occurs on the servers that host EC2 instances, providing encryption of data-in-transit from EC2 instances to EBS storage.
- You can create point-in-time snapshots of EBS volumes, which are persisted to Amazon S3. Snapshots protect data for long-term durability, and they can be used as the starting point for new EBS volumes. The same snapshot can be used

to create as many volumes as needed. These snapshots can be copied across AWS Regions.

- Performance metrics, such as bandwidth, throughput, latency, and average queue length, are available through the AWS Management Console. These metrics, provided by Amazon CloudWatch, allow you to monitor the performance of your volumes to make sure that you are providing enough performance for your applications without paying for resources you don't need.

19.0 VPC (Virtual Private Cloud):

With Amazon Virtual Private Cloud (Amazon VPC), you can launch AWS resources in a logically isolated virtual network that you've defined. This virtual network closely resembles a traditional network that you'd operate in your own data center, with the benefits of using the scalable infrastructure of AWS.

The following diagram shows an example VPC. The VPC has one subnet in each of the Availability Zones in the Region, EC2 instances in each subnet, and an internet gateway to allow communication between the resources in your VPC and the internet.

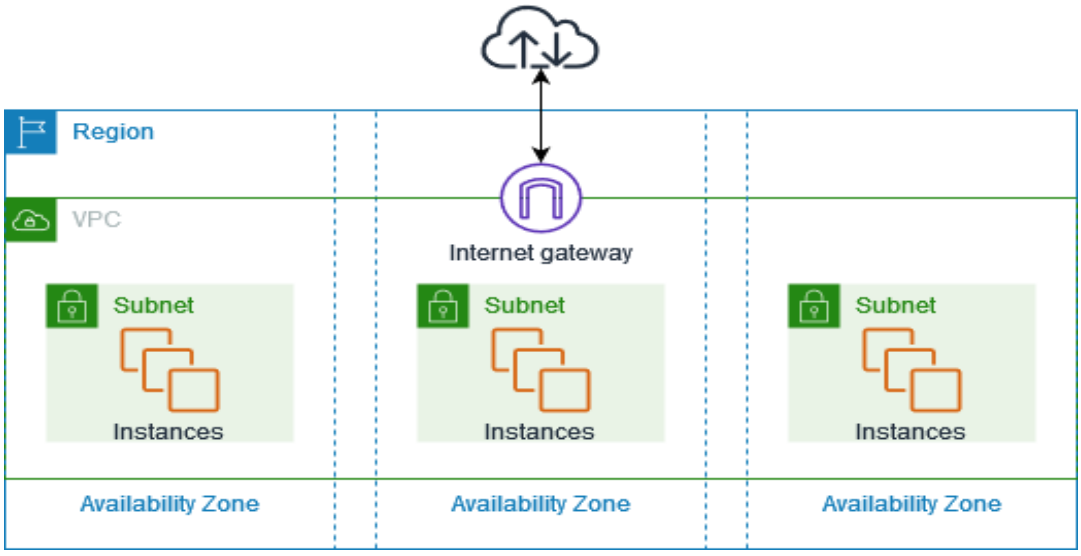

Fig 19.1 VPC

19.1 Features:

The following features help you configure a VPC to provide the connectivity that your applications need:

- Virtual private cloud
- Subnets
- Ip addressing
- Routing
- Gateway and endpoints
- Peering connections
- VPC flow logs
- VPC connections

19.1.1 Virtual Private Cloud (VPC):

A VPC is a virtual network that closely resembles a traditional network that you'd operate in your own data center. After you create a VPC, you can add subnets.

19.1.2 Subnets:

A subnet is a range of IP addresses in your VPC. A subnet must reside in a single Availability Zone. After you add subnets, you can deploy AWS resources in your VPC.

19.1.3 IP Addressing:

You can assign IP addresses, both IPv4 and IPv6, to your VPCs and subnets. You can also bring your public IPv4 and IPv6 GUA addresses to AWS and allocate them to resources in your VPC, such as EC2 instances, NAT gateways, and Network Load Balancers.

19.1.4 Routing:

Use route tables to determine where network traffic from your subnet or gateway is directed.

19.1.5 Gateways and Endpoints:

A gateway connects your VPC to another network. For example, use an internet gateway to connect your VPC to the internet. Use a VPC endpoint to connect to AWS services privately, without the use of an internet gateway or NAT device.

19.1.6 Peering Connections:

Use a VPC peering connection to route traffic between the resources in two VPCs.

19.1.7 VPC Flow Logs:

A flow log captures information about the IP traffic going to and from network interfaces in your VPC.

19.1.8 VPN Connections:

Connect your VPCs to your on-premises networks using AWS Virtual Private Network (AWS VPN).

19.2 Creating a VPC:

- Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
- On the VPC dashboard, choose **Create VPC**.
- For Resources to create, choose **VPC only**.
- (Optional) For **Name tag**, enter a name for your VPC. Doing so creates a tag with a key of Name and the value that you specify.
- For **IPv4 CIDR block**, do one of the following:
  - Choose **IPv4 CIDR manual input** and enter an IPv4 address range for your VPC.
  - Choose **IPAM-allocated IPv4 CIDR block**, select your IPAM IPv4 address pool and a netmask. The size of the CIDR block is limited by the allocation rules on the IPAM pool.
  - If you are using IPAM to manage your IP addresses, we recommend that you choose this option.
- (Optional) To create a dual stack VPC, specify an IPv6 address range for your VPC. For **IPv6 CIDR block**, do one of the following:
  - Choose **IPAM-allocated IPv6 CIDR block** and select your IPAM IPv6 address pool. The size of the CIDR block is limited by the allocation rules on the IPAM pool.
  - Choose **Amazon-provided IPv6 CIDR block** to request an IPv6 CIDR block from an Amazon pool of IPv6 addresses. Amazon provides a fixed IPv6 CIDR block size of /56.
  - Choose **IPv6 CIDR owned by me** to use an IPv6 CIDR block that you brought to AWS using . For **Pool**, choose the IPv6 address pool from which to allocate the IPv6 CIDR block.
- (Optional) Choose a **Tenancy** option. you must use Default tenancy only. If dedicated it will charge you according to your requirements.
- (Optional) To add a tag to your VPC, choose **Add new tag** and enter a tag key and a tag value.
- Choose **Create VPC**.

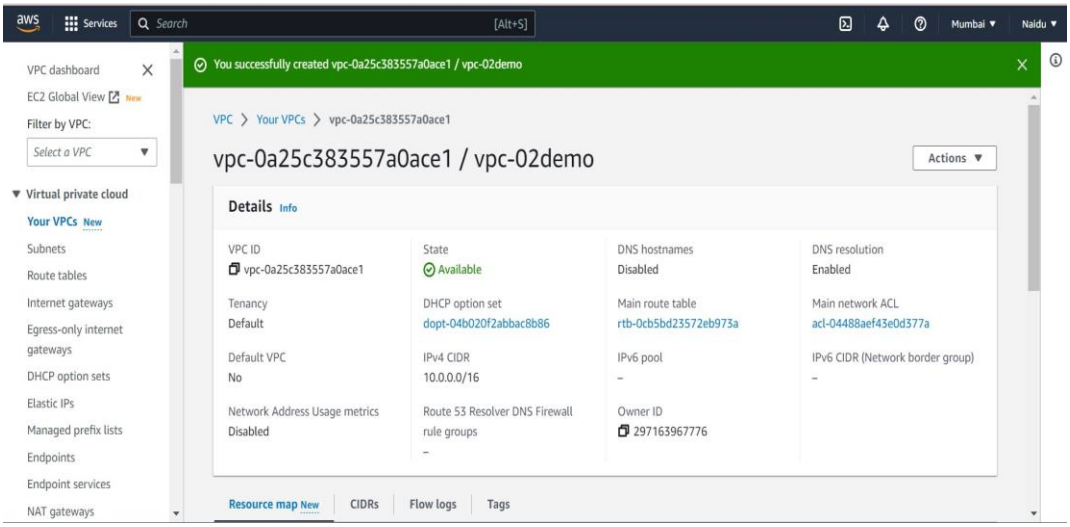

Fig 19.2 VPC Created Successfully

19.2.1 Creating Subnets:

- Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
- In the navigation pane, choose **Subnets**.
- Choose **Create subnet**.
- For **VPC ID**: Choose the VPC for the subnet.
- (Optional) For **Subnet name**, enter a name for your subnet. Doing so creates a tag with a key of Name and the value that you specify.
- For **Availability Zone**, you can choose a Zone for your subnet, or leave the default **No Preference** to let AWS choose one for you.
- If the subnet should be an IPv6-only subnet, choose **IPv6-only**. This option is only available if the VPC has an associated IPv6 CIDR block. If you choose this option, you can't associate an IPv4 CIDR block with the subnet.
- For **IPv4 CIDR block**, enter an IPv4 CIDR block for your subnet. For example, 10.0.0.0/24 for first subnet, 10.0.1.0/24 for second subnet. If you chose **IPv6-only**, this option is unavailable.
- For **IPv6 CIDR block**, choose **Custom IPv6 CIDR** and specify the hexadecimal pair value (for example, **00**). This option is available only if the VPC has an associated IPv6 CIDR block.

- To add another subnet for same VPC click on add subnet. Or repeat the same process again.
- Choose **Create subnet**.

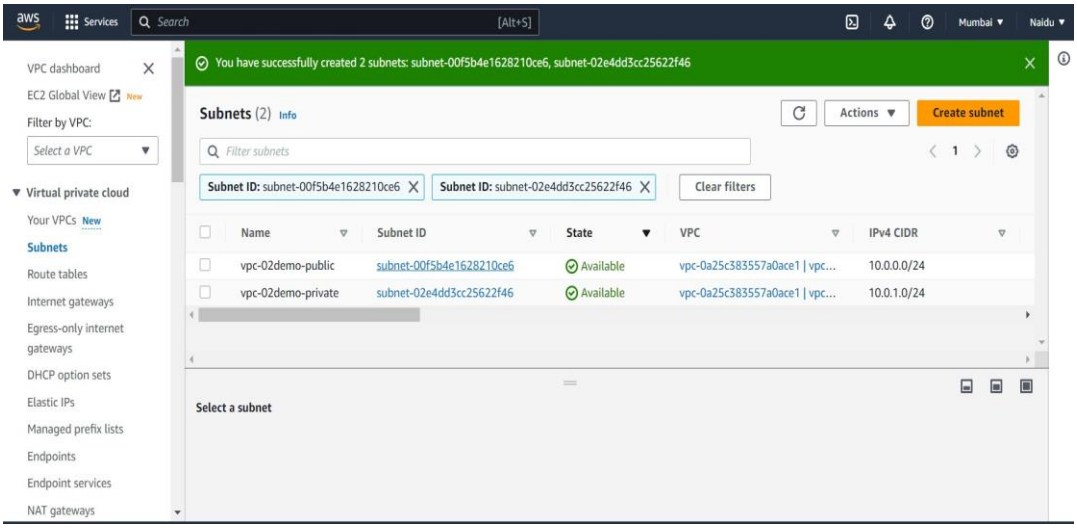

Fig 19.3 Subnets Created Successfully

19.2.2 Creating Internet Gateway:

- Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
- In the navigation pane, choose **Internet gateways**, and then choose **Create internet gateway**.
- Optionally name your internet gateway.
- Optionally add or remove a tag.
  - [Add a tag] Choose **Add tag** and do the following:
  - For **Key**, enter the key name.
  - For **Value**, enter the key value.
  - [Remove a tag] Choose **Remove** to the right of the tag's Key and Value.
- Choose **Create internet gateway**.
- Select the internet gateway that you just created, and then choose **Actions, Attach to VPC**.
- Select your VPC from the list, and then choose **Attach internet gateway**.

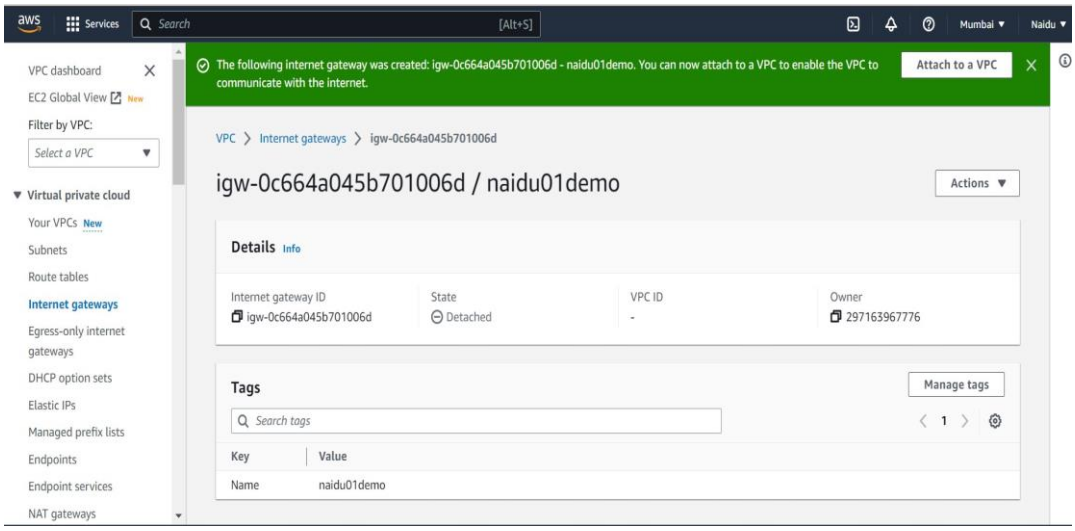

Fig 19.4 Internet Gateway

19.2.3 Creating Route Tables:

- Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
- In the navigation pane, choose **Route Tables**, and then choose **Create route table**.
- In the **Create route table** dialog box, optionally name your route table, then select your VPC, and then choose **Create route table**.
- Create one route table i.e, private, another public route table is given as default while creating your VPC.

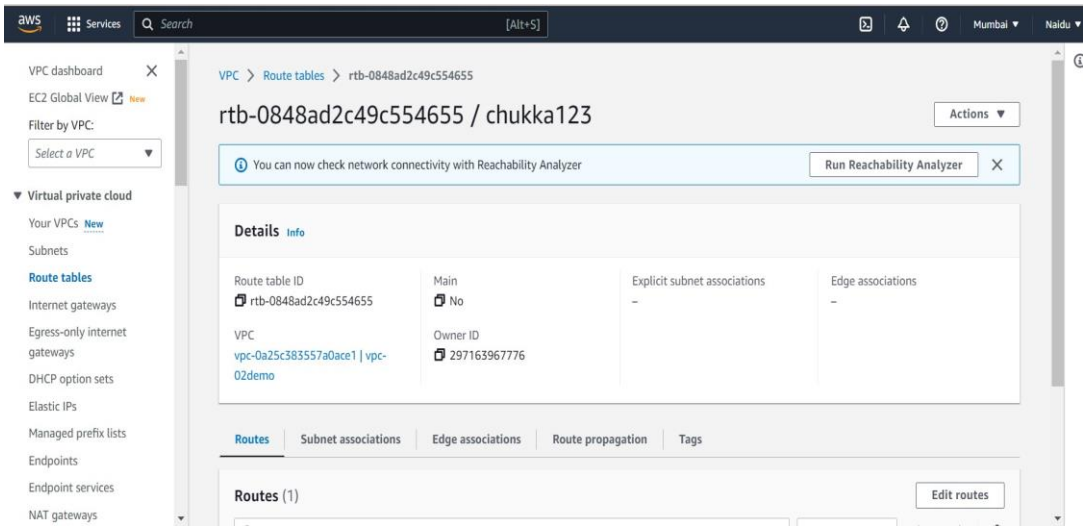

Fig 19.5 Route Table

19.2.4 Creating Connections for VPC:

- open the Amazon VPC console.
- In the navigation pane, choose **route tables**, and then choose **edit routes.**
- In IPv4 select 0.0.0.0/0, attach your internet gateway and then click on **save changes** (attach internet gateway for public route table)**.**
- Go to **subnet associations,** click on **edit subnet associations,** attach public subnet to public route table and private subnet to private route table.
- Now all the connections are made successfully.

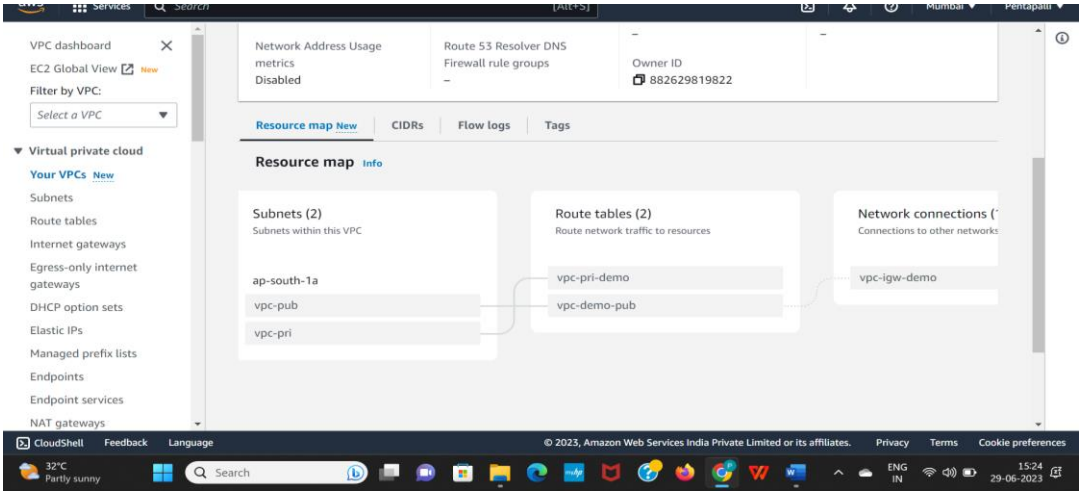

Fig 19.6 VPC Connections

19.3 Creating EC2 with VPC Connection:

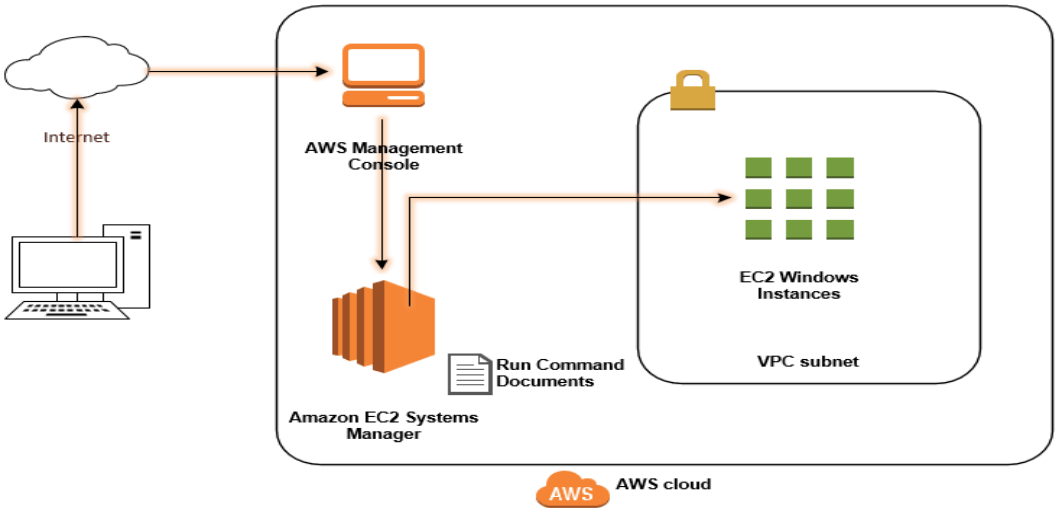

Fig 19.7 EC2 with VPC

- While creating ec2 in network settings attach vpc and select subnet (private subnet for private VPC and public subnet for public VPC).
- Click on add a security group rule.
- Attach IPv4 0.0.0.0/0, and **all TCP** as a source with port range **80.**
- click on **launch instance.**

- Go to mobaxtrem, connect to private ec2 using public IPv4, enter command "ec2-user" to check login is successful or not.

- Login to private ec2 using ssh command, go to private ec2 click on connect go to ssh client copy the commands and login to private ec2 in mobaxtrem.

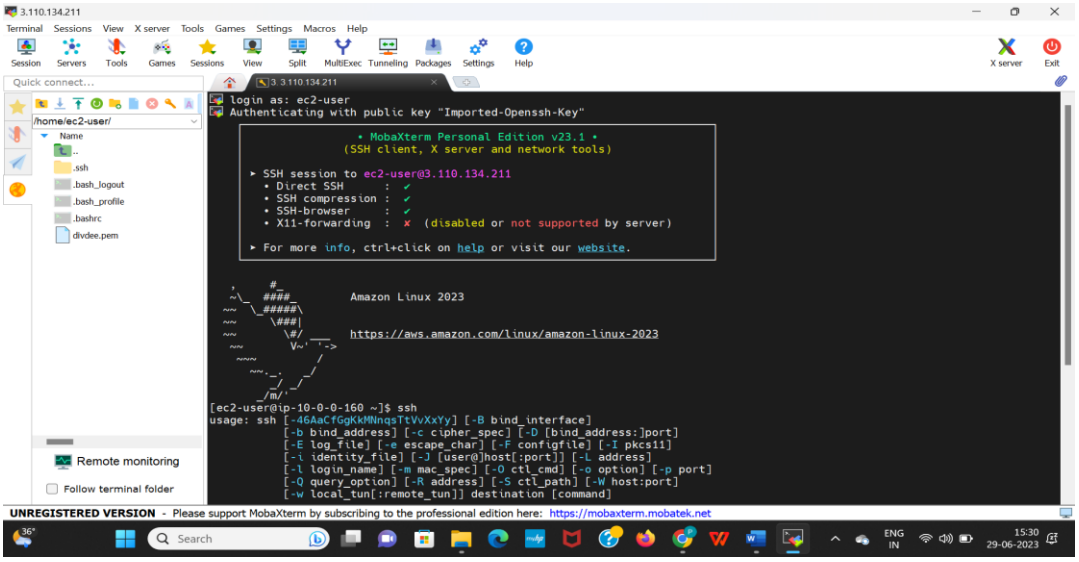

Fig 19.8 VPC Login with MobaXtrem

19.4 Creating Nat-Gateway:

A NAT gateway is a Network Address Translation (NAT) service. You can use a NAT gateway so that instances in a private subnet can connect to services outside your VPC but external services cannot initiate a connection with those instances.

Each NAT gateway is created in a specific Availability Zone and implemented with redundancy in that zone. There is a quota on the number of NAT gateways that you can create in each Availability Zone. For more information, see Amazon VPC quotas.

If you have resources in multiple Availability Zones and they share one NAT gateway, and if the NAT gateway's Availability Zone is down, resources in the other Availability Zones lose internet access. To improve resiliency, create a NAT gateway in each Availability Zone, and configure your routing to ensure that resources use the NAT gateway in the same Availability Zone.

- Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
- In the navigation pane, choose **NAT gateways**.

- Choose **Create NAT gateway**.

- (Optional) Specify a name for the NAT gateway. This creates a tag where the key is **Name** and the value is the name that you specify.
- Select the subnet in which to create the NAT gateway.
- For **Connectivity type**, leave the default **Public** selection to create a public NAT gateway or choose **Private** to create a private NAT gateway. For more information about the difference between a public and private NAT gateway, see NAT gateways.
- If you chose **Public**,
    - Choose an **Elastic IP allocation ID** to assign an EIP to the NAT gateway or choose **Allocate Elastic IP** to automatically allocate an EIP for the public NAT gateway. You are limited to associating 2 Elastic IP addresses to your public NAT gateway by default. You can increase this limit by requesting a quota adjustment.
- Choose **Create a NAT gateway**

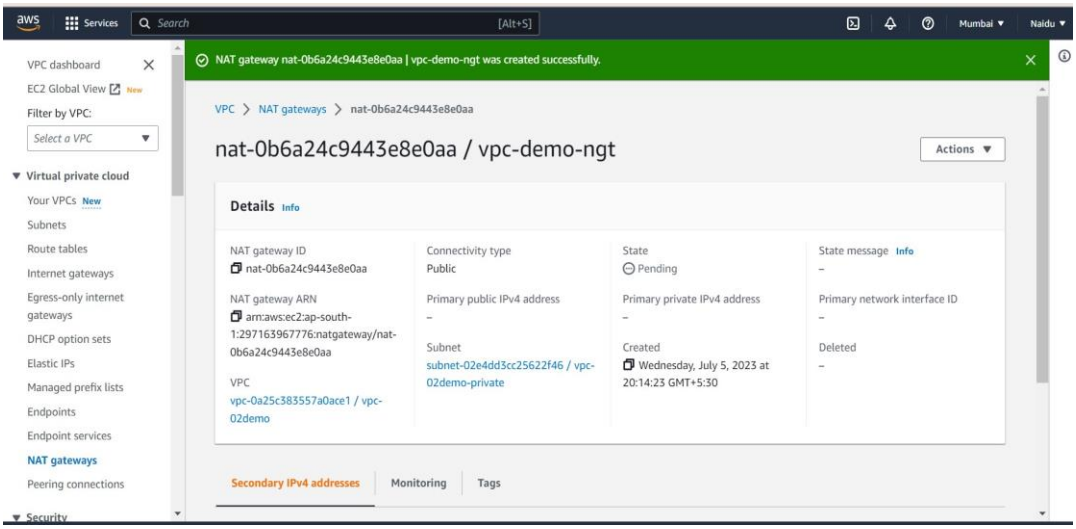

Fig 19.9 Nat-Gateway

19.4.1 Connecting Nat-gateway with VPC:

- Go to Amazon VPC console page and then go to route tables.
- Go to private route table and then click on **edit routes.**
- Select IPv4 as 0.0.0.0/0, attach your **Nat-gateway** and then click on **save changes.**

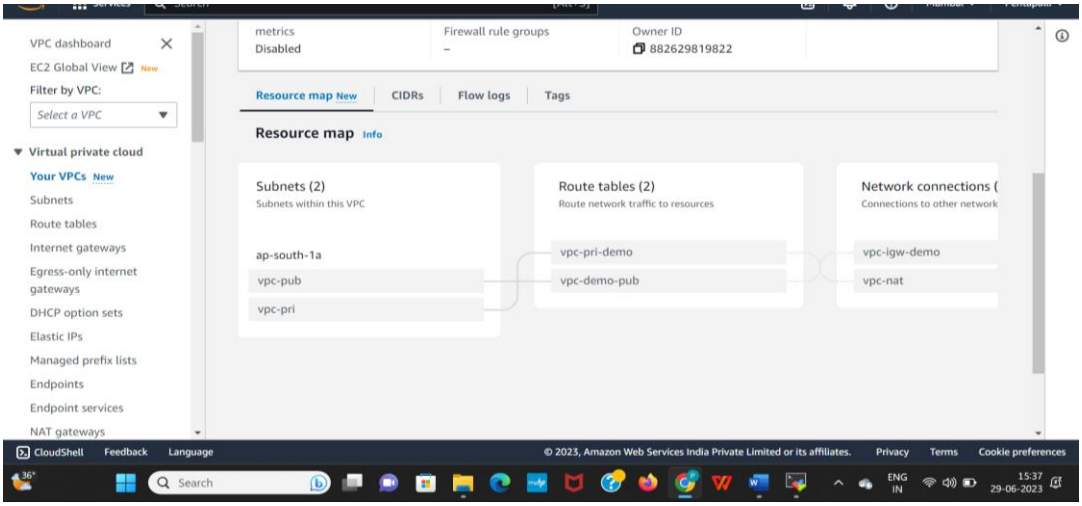

Fig 19.10 Connect to VPC

19.5 Peering Connections:

A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them privately. Resources in peered VPCs can communicate with each other as if they are within the same network. You can create a VPC peering connection between your own VPCs, with a VPC in another AWS account, or with a VPC in a different AWS Region. Traffic between peered VPCs never traverses the public internet.

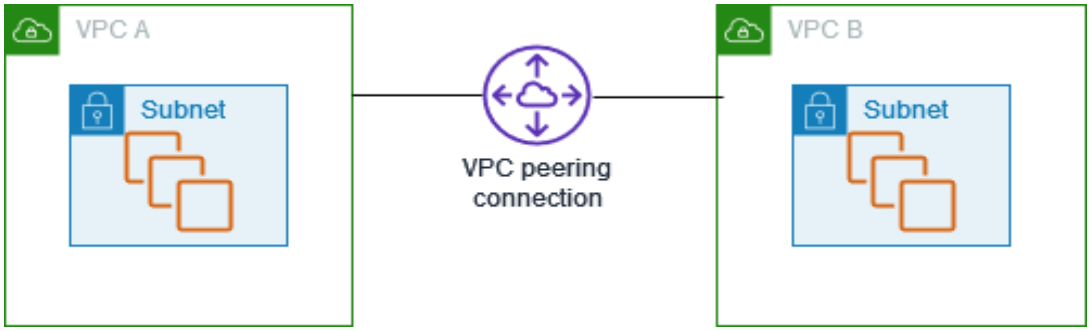

Fig 19.11 VPC Peering Connection

- Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
- In the navigation pane, choose **Peering connections**.
- Choose **Create peering connection**.

- Configure the following information, and choose **Create peering connection** when you are done:

- **Name**: You can optionally name your VPC peering connection.
- **VPC ID (Requester)**: Select the VPC in your account with which you want to create the VPC peering connection.
- For **Select another VPC to peer with**, choose **My account** and select another of your VPCs.
- (Optional) To add a tag, choose **Add new tag** and enter the tag key and value.

- Choose **Actions**, **Accept request**.
- When prompted for confirmation, choose **Accept request**.
- Choose **Modify my route tables now** to add a route to the VPC route table so that you can send and receive traffic across the peering connection.

19.5.1 Connecting Peering with Route Tables:

- Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
- In the navigation pane, choose **Route tables**.
- Select the check box next to the route table that's associated with the subnet in which your instance resides.
  - If you do not have a route table explicitly associated with that subnet, the main route table for the VPC is implicitly associated with the subnet.
- Choose **Actions**, **Edit routes**.
- Choose **Add route**.
- For **Destination**, enter the IPv4 address range to which the network traffic in the VPC peering connection must be directed. You can specify the entire IPv4 CIDR block of the peer VPC, a specific range, or an individual IPv4 address, such as the IP address of the instance with which to communicate. For example, if the CIDR block of the peer VPC is 10.0.0.0/16, you can specify a portion 10.0.0.0/24, or a specific IP address 10.0.0.7/32.
- For **Target**, select the VPC peering connection.
- Choose **Save changes**.

19.6 VPC Endpoints:

A VPC endpoint enables customers to privately connect to supported AWS services and VPC endpoint services powered by AWS Private Link. Amazon VPC instances do not require public IP addresses to communicate with resources of the

service. Traffic between an Amazon VPC and a service does not leave the Amazon network.

VPC endpoints are virtual devices. They are horizontally scaled, redundant, and highly available Amazon VPC components that allow communication between instances in an

Amazon VPC and services without imposing availability risks or bandwidth constraints on network traffic.

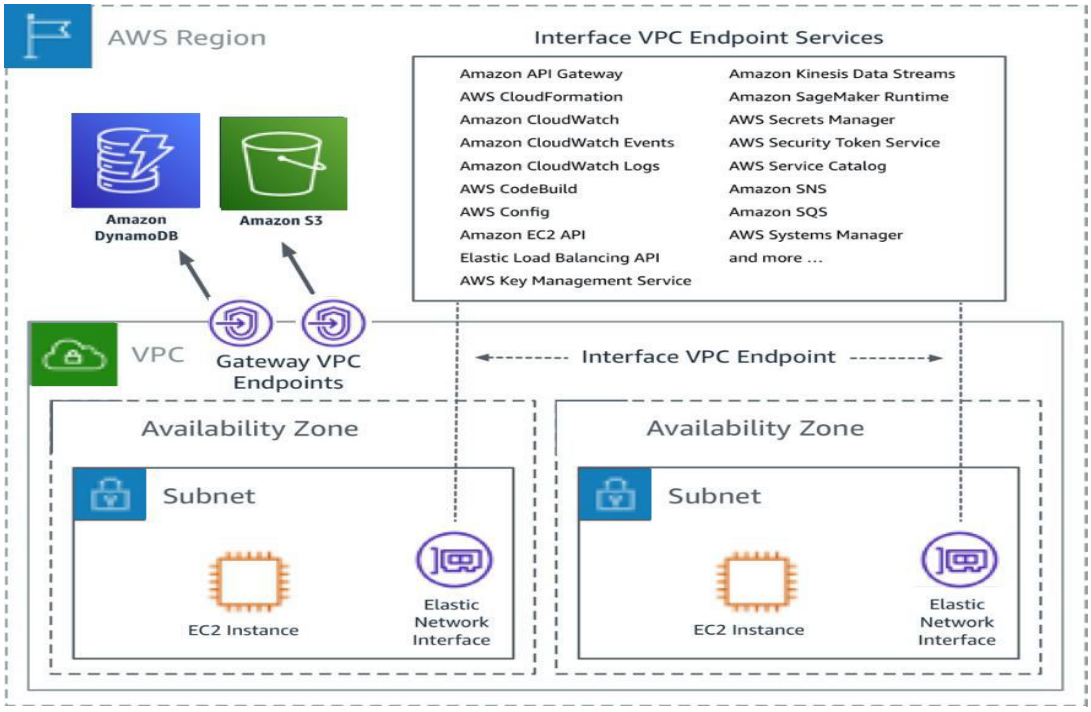

Fig 19.12 VPC Endpoints

- Open the Amazon VPC console.
- Select "Endpoints" from the sidebar.
- Click on "Create Endpoint."
- Choose the following configuration options:
- Select the VPC in which you want to create the endpoint.
- Choose the service category as "AWS services."
- Select "aws amazon.com. <region>. s3" as the service name. Replace "<region>" with the appropriate region identifier, such as "us-east-1" for the US East (N. Virginia) region.
- Ensure that the "Enable Private DNS Name" option is checked.
- Specify the security options:
- Choose the subnets in which you want to create the endpoint.

- Configure the security group settings according to your requirements.
- Review the configuration and click "Create Endpoint."

- Once the VPC endpoint is created and associated with the specified subnets, you can now interact with S3 buckets securely within your VPC without requiring internet access.
- To list the buckets using the VPC endpoint, you can use the AWS Command Line Interface (CLI) or SDKs with appropriate credentials and the following command
    - “AWS s3api list-buckets --endpoint-url <VPC endpoint URL>”
- Replace "<VPC endpoint URL>" with the URL of the VPC endpoint you created, which typically follows the format "vpce-xxxx.s3.<region> .amazonaws.com".
- Make sure that the IAM user or role used to execute the command has the necessary permissions to list S3 buckets.
- By creating a VPC endpoint for S3 and associating it with your VPC, you can securely access S3 resources without the need for internet connectivity or traversing public networks.
- To attach an IAM role to instances in AWS and then log in to MobaXterm using the endpoints, you can follow these steps:

19.6.1 Create an IAM Role:

- Open the IAM console in AWS.
- Navigate to "Roles" in the sidebar.
- Click on **"Create role."**
- Select the service that will use this role (EC2 in this case).
- Choose the use case that best fits your requirements.
- Attach the necessary policies to the role.
- Provide a unique name for the role and optionally add tags.
- Review and create the role.

19.6.2 Launch an EC2 Instance:

- Open the EC2 console.
- Click on "Launch Instances."
- Select an Amazon Machine Image (AMI) and instance type.
- Configure the instance details, such as subnet, security groups, and storage.
- In the "Configure Instance Details" section, select the IAM role you created in last.
- Launch the instance.
- Connect to the EC2 instance using MobaXterm:

- Open MobaXterm on your local machine.
- Click on "Session" in the top menu bar.

- In the "Remote host" field, enter the public IP address or DNS name of the EC2 instance.
- Specify the username for the instance (e.g., "ec2-user" for Amazon Linux or "ubuntu" for Ubuntu instances).
- In the "Advanced SSH settings" section, click on the "Use private key" checkbox.
- Provide the private key associated with the key pair used when launching the instance.
- Click on "OK" to initiate the SSH connection.

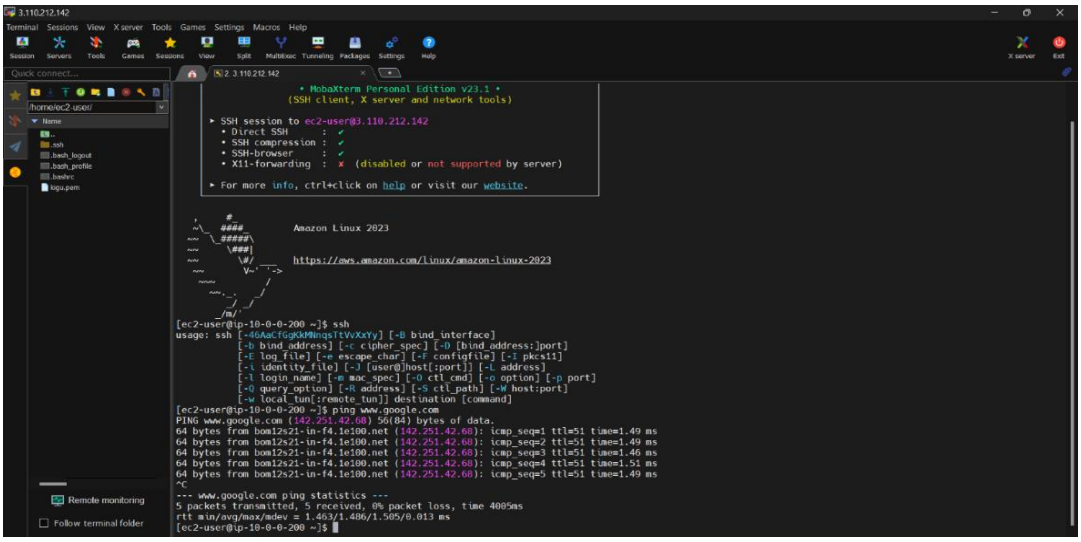

Fig 19.13 VPC Endpoints

- Once connected to the EC2 instance using MobaXterm, you can interact with the endpoint:
- Run the necessary commands or access the required resources using the provided endpoint URLs.
- “AWS s3 ls –endpoint-url https://bucketname.DNS name link"

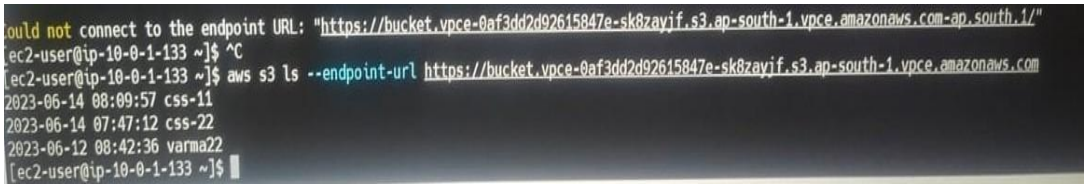

Fig 19.14 Access S3

- By attaching an IAM role to EC2 instances, you grant them specific permissions to access AWS resources without having to manage individual

access keys or credentials. This ensures secure and controlled access to AWS services.

- Ensure that the IAM role has the necessary permissions to access the endpoints and perform the required actions. Additionally, make sure that the security groups associated with the EC2 instance allow inbound SSH (port 22) access.
- Now we can access the s3 without remote network.

20.0 RDS (Relational Database Service):

Amazon Relational Database Service (Amazon RDS) makes it easy to set up, operate and scale relational databases in the cloud. It provides cost-efficient and resizable capabilities while automating time-consuming administration tasks such as hardware provisioning, database setup, patching, and backup. It frees you up to focus on your applications so that you can provide them with the fast performance, high availability, security and compatibility they need.

Amazon RDS is available on multiple database instance types - optimized for memory, performance or I/O - and gives you six familiar database engines to choose from, including Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle Database and SQL Server Huh. You can use the AWS Database Migration Service to migrate easily or replicate your existing databases to Amazon RDS.

Databases store large amounts of data that applications can draw upon to help them perform various tasks. A relational database uses tables to store data and is called relational because it organizes data points with defined relationships.

Administrators control Amazon RDS with the AWS Management Console, Amazon RDS API calls, or the AWS command-line interface. They use these interfaces to deploy database instances to which users can apply specific settings.

Amazon provides several instance types with different resources, such as CPU, memory, storage options, and networking capability. Each type comes in a variety of sizes to suit the needs of different workloads.

RDS users can use AWS Identity and Access Management to define and set permissions to access RDS databases.

20.1 To Create a DB Instance:

- Sign in to the AWS Management Console and open the Amazon RDS console at https://console.aws.amazon.com/rds/.
- In the upper-right corner of the Amazon RDS console, choose the AWS Region in which you want to create the DB instance.
- In the navigation pane, choose **Databases**.
- Choose **Create database**, then choose **Standard create**.
- For **Engine type**, choose MariaDB, Microsoft SQL Server, MySQL, Oracle, or PostgreSQL.
- **Microsoft SQL Server** is shown here.

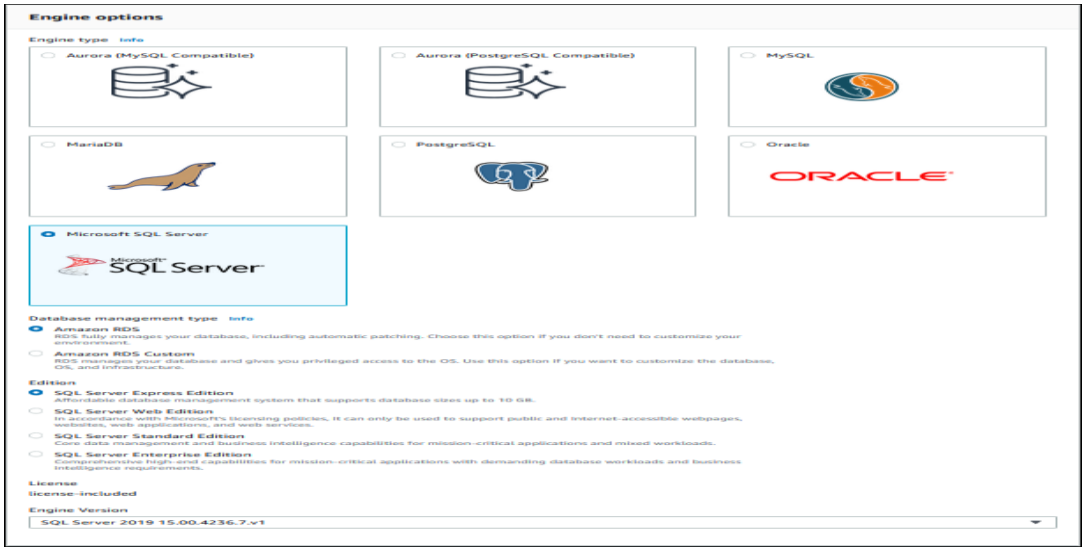

Fig 20.1 Data Base Engine Selection

- For **Database management type**, if you're using Oracle or SQL Server choose **Amazon RDS** or **Amazon RDS Custom**.
- **Amazon RDS** is shown here. For more information on RDS Custom, see Working with Amazon RDS Custom.
- For **Edition**, if you're using Oracle or SQL Server choose the DB engine edition that you want to use.
- MySQL has only one option for the edition, and MariaDB and PostgreSQL have none.
- For **Version**, choose the engine version.

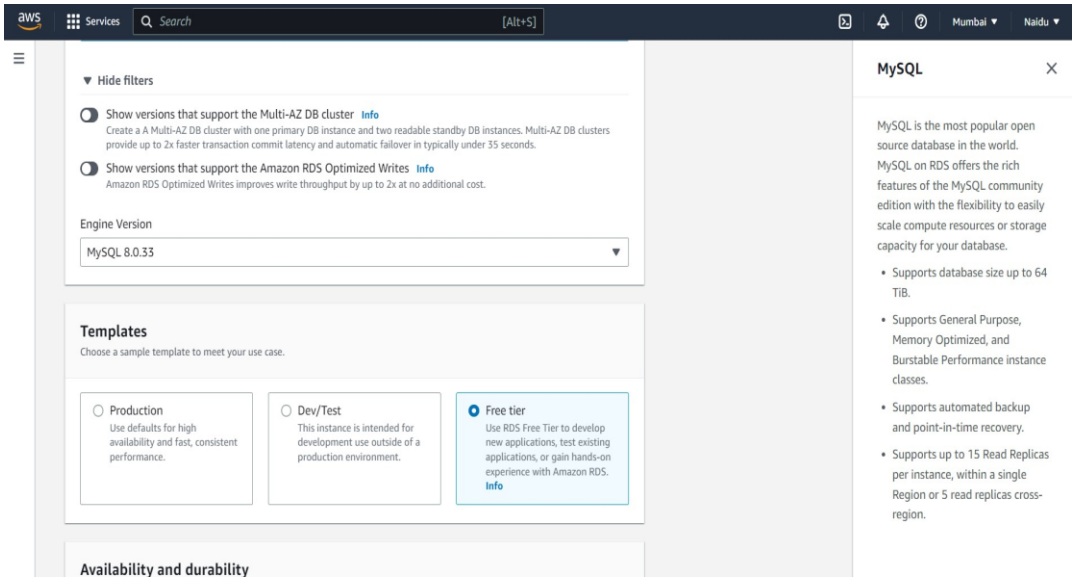

Fig 20.2 Selecting Templates

- In **Templates**, choose the template that matches your use case. If you choose **Production**, the following are preselected in a later step:
  - **Multi-AZ** failover option
  - **Provisioned IOPS SSD (io1)** storage option
  - **Enable deletion protection** option
- We recommend these features for any production environment.
- Template choices vary by edition.
- To enter your master password, do the following:
  - In the **Settings** section, open **Credential Settings**.
  - If you want to specify a password, clear the **Auto generate a password** check box if it is selected.
  - (Optional) Change the **Master username** value.
  - Enter the same password in **Master password** and **Confirm password**.
- (Optional) Set up a connection to a compute resource for this DB instance.
- You can configure connectivity between an Amazon EC2 instance and the new DB instance during DB instance creation. For more information, see Configure automatic network connectivity with an EC2 instance.

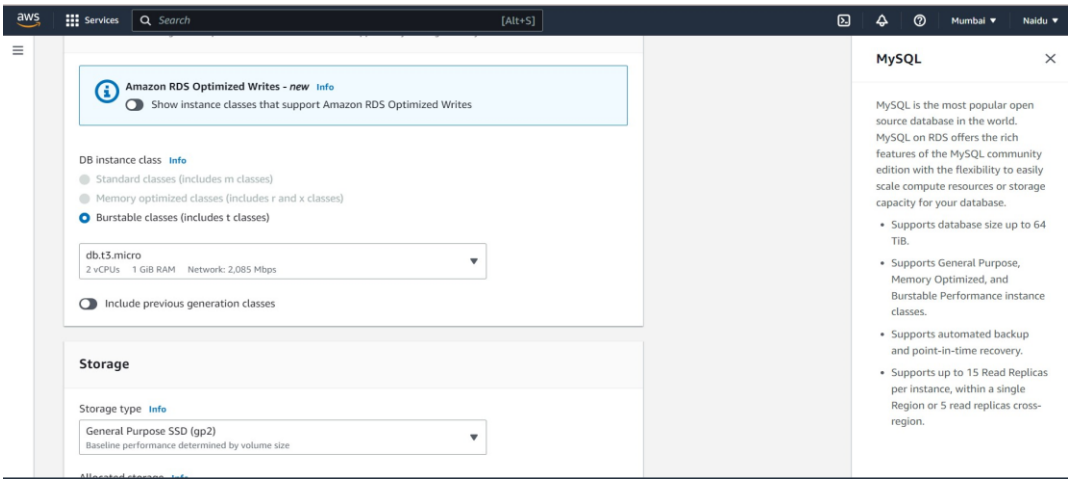

Fig 20.3 Allocating Instance Configuration

- In the **Connectivity** section under **VPC security group (firewall)**, if you select **Create new**, a VPC security group is created is added to the database with an inbound rule that allows your local computer's IP address to access the database.
- For the remaining sections, specify your DB instance settings. For information about each setting, see Settings for DB instances.
- Choose **Create database**.

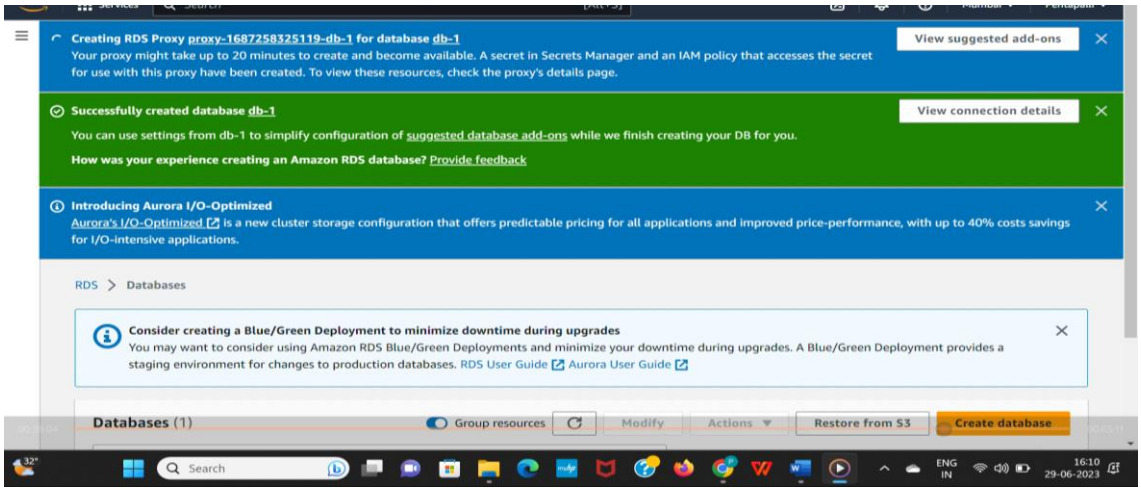

Fig 20.4 RDS Creation

- If you chose to use an automatically generated password, the **View credential details** button appears on the **Databases** page.
- To view the master user name and password for the DB instance, choose **View credential details**.
- To connect to the DB instance as the master user, use the user name and password that appear.
- You can't view the master user password again. If you don't record it, you might have to change it. If you need to change the master user password after the DB instance is available, modify the DB instance to do so. For more information about modifying a DB instance, see Modifying an Amazon RDS DB instance.
- For **Databases**, choose the name of the new DB instance.
- On the RDS console, the details for the new DB instance appear. The DB instance has a status of **Creating** until the DB instance is created and ready for use. When the state changes to **Available**, you can connect to the DB instance. Depending on the DB instance class and storage allocated, it can take several minutes for the new instance to be available.

20.2 Installing Workbench:

- From an account with Administrator or Power User privileges, right-click the MSI file and select the **Install** item from the pop-up menu, or double-click the file.
- In the Setup Type page, select either a Complete or Custom installation. To use all features of MySQL Work bench choose the Complete setup type.
- Unless you choose otherwise, MySQL Workbench is installed in C:\%PROGRAMFILES%\MySQL\MySQL Workbench 8.0 edition_type\, where ***%PROGRAMFILES%*** is the default directory for programs for your locale. The %PROGRAMFILES% directory is defined as C:\Program Files\ on most systems.

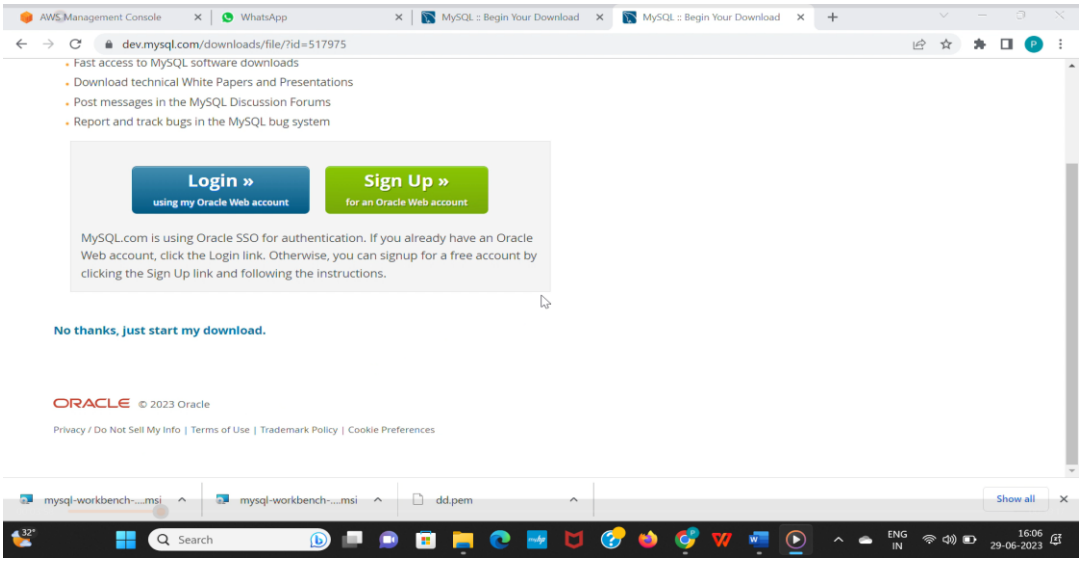

Fig 20.5 Work Bench Installation

20.3 Constraints in MySQL:

- Primary Key
- Foreign Key
- Unique Key
- Default Key
- Check Key
- Not Null

20.3.1 Primary Key:

- This constraint is used to identify each record in a table uniquely.
- if the column contains primary key constraints, then it cannot be null or empty
- A table can contain only one primary key. it always contains unique value into a column.

20.3.2 Foreign Key:

- This constraint is used to link two tables together.
- It is also known as referencing key.
- A foreign key column matches the primary key field of another table.
- It means a foreign key field in one table refers to the primary key field of another table.

20.3.3 Unique Key:

- This constraint ensures that all values inserted into the column will be unique
- It means a column cannot stores duplicate values.
- MySQL allows us to use more than one column with UNIQUE constraint data

20.3.4 Default Key:

- This constraint is used to set the default value for the particular column where we have not specified the value
- It means the column must contain value, including NULL

20.3.5 Check Key:

- It controls the value in a particular column.
- It ensures that the inserted value in a column must be satisfied with the given condition
- In other words, it determines whether the value associated with the column is valid or not with the given condition.

20.3.6 Not Null Key:

- This constraint specifies that the column cannot have NULL or empty values.
- The below statement creates a table with NOT NULL constraint.

20.4 Working on Workbench:

- Download and install MySQL Workbench.
- Open MySQL Workbench and choose the ⊕ sign beside MySQL Connections to set up a new connection.
- In the Setup New Connection dialog box, enter a name for your connection.
- In the Parameters section, enter these details:
    - Host name: Enter the RDS endpoint.

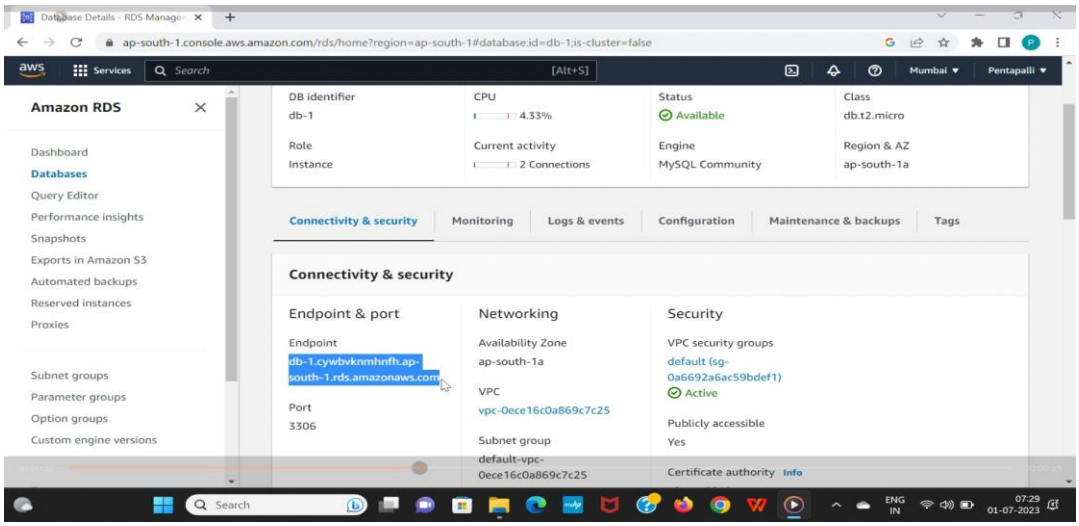

Fig 20.6 DNS Endpoint Link

- Port: Enter the Port number.
- Username: Enter the primary user.
- Password: Enter the password for the primary user.

o Choose Test Connection.

20.4.1 Create Your First MySQL Database:

Execute the command to create your database in workbench.

CREATE database Data base name;

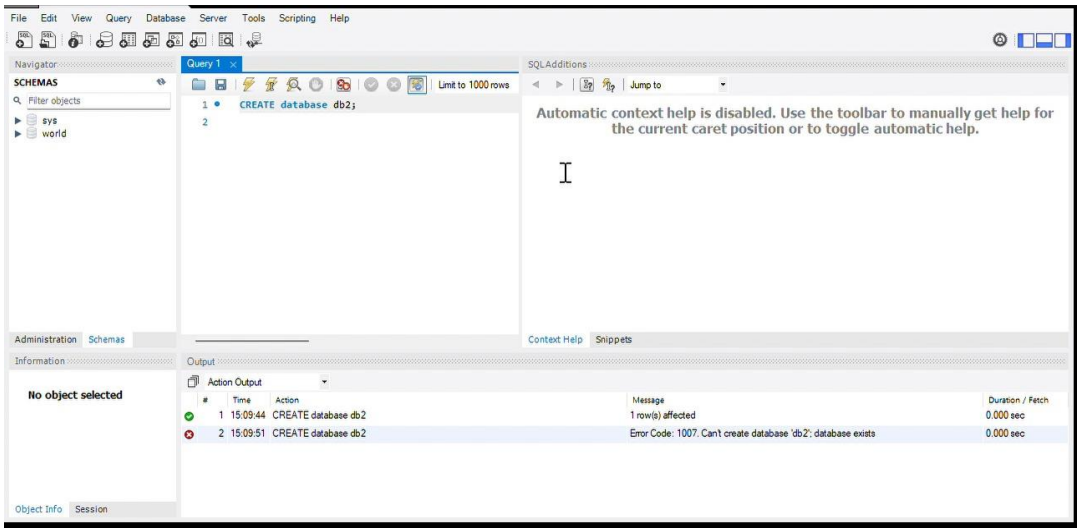

Fig 20.7 Database Created Successfully

20.4.3 Create Your Table:

```
CREATE TABLE Data base name.emp (
Id INT NOT NULL,
EmpName   VARCHAR (100),
EmpDept   VARCHAR (45),
);
```

(or)

```
USE Database name;
CREATE TABLE emp (
Id INT NOT NULL,
EmpName VARCHAR (100),
EmpDept VARCHAR (45),
);
```

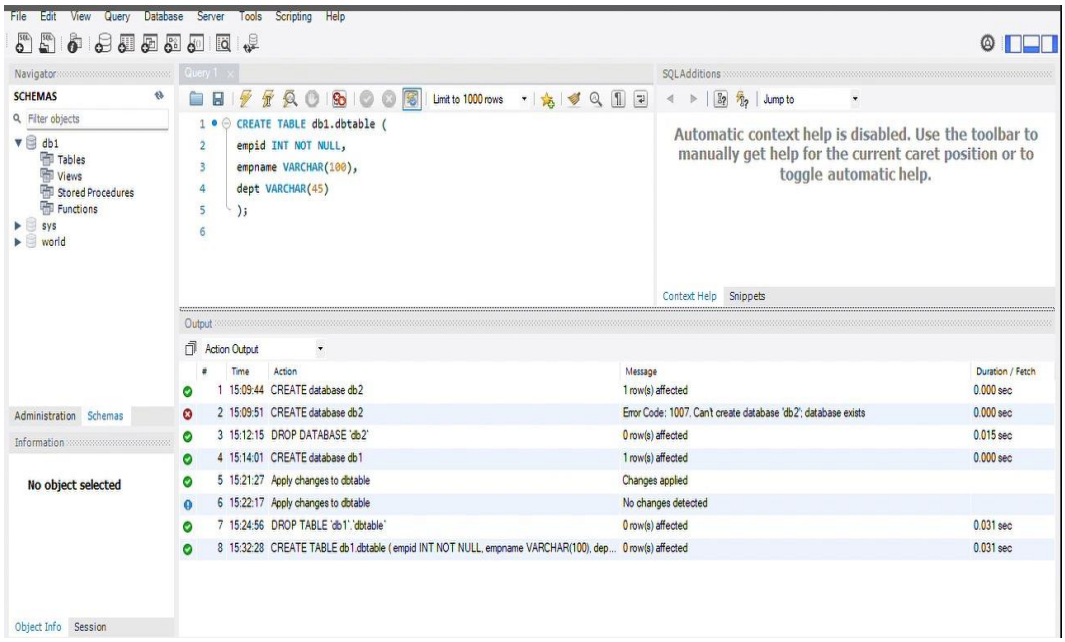

Fig 20.8 Table Created Successfully

20.4.4 Insert Data into Tables:

```
INSERT INTO databasename.tablename(Id,EmpName,EmpDept)
Values(1,'vasu','IT');
SELECT * FROM databasename. tablename;
```

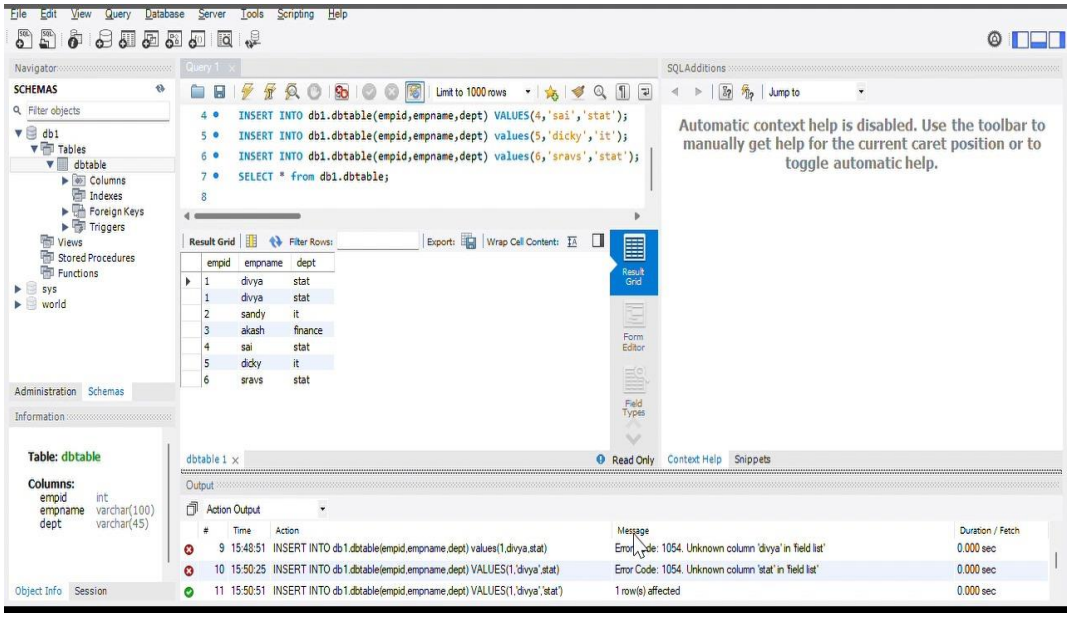

Fig 20.9 Inserting Values

20.4.5 Update Table:

SET SQL_SAFE_UPDATES = 0;
UPDATE databasename.tablename Set EmpDept='ITES' WHERE EmpDept='IT';
SELECT * FROM databasename.tablename;

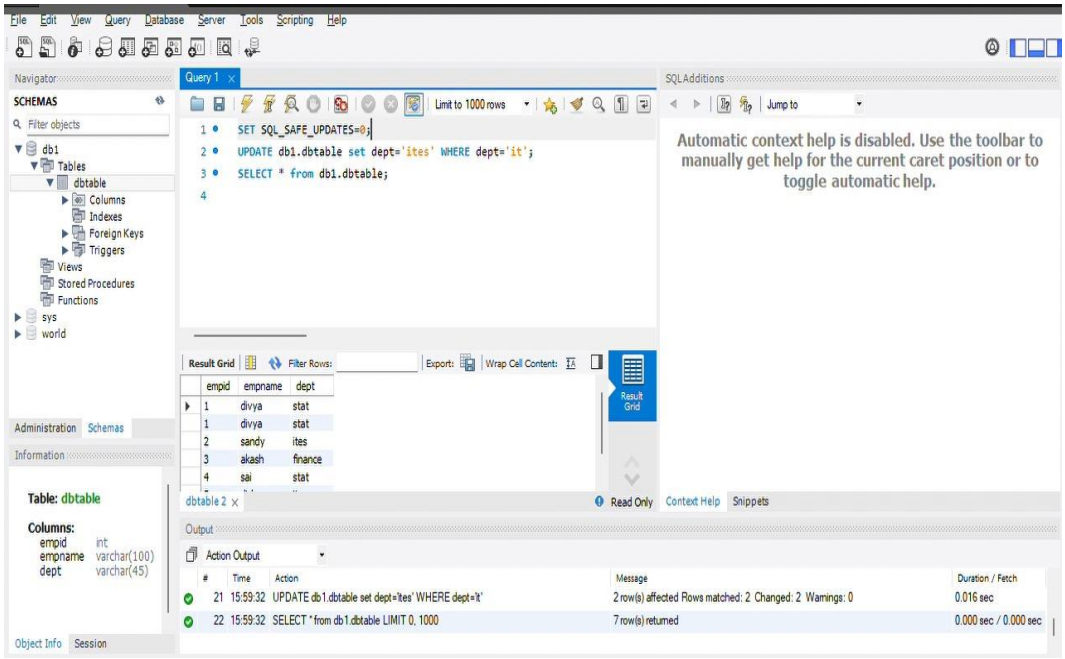

Fig 20.10 Update Table

20.4.6 Delete Data From The Table:

DELETE FROM databasename.tablename
DELETE FROM databasename.tablename Where EmpDept='ITES';
SELECT * FROM databasename.tablename;

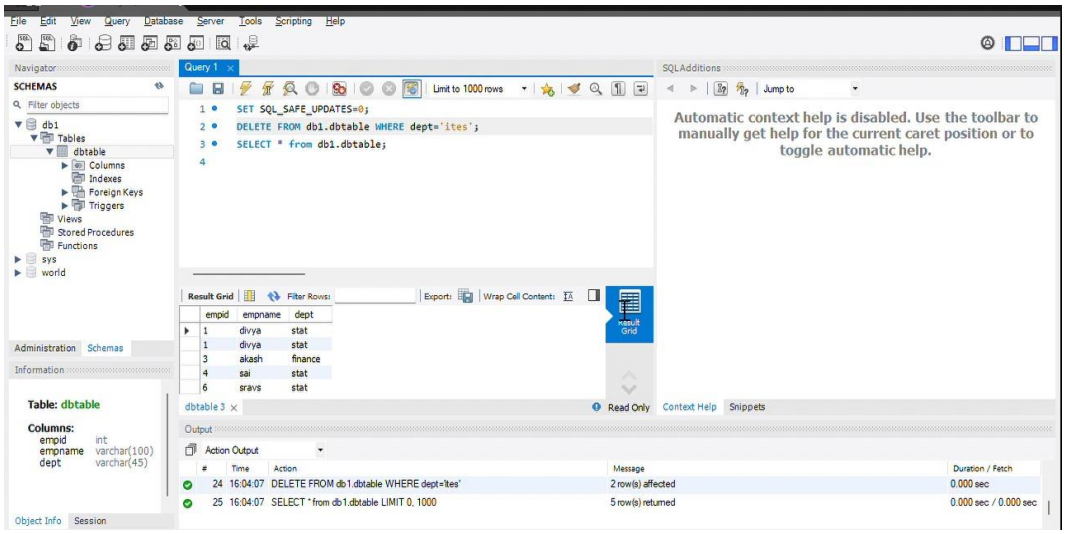

Fig 20.11 Deleting Data from Table

20.4.7 Primary Key:

To add and drop primary key we can use the following commands
ALTER TABLE databasename. tablename ;
DROP primary key;

ALTER TABLE databasename.tablename ;
ADD primary key (column name);

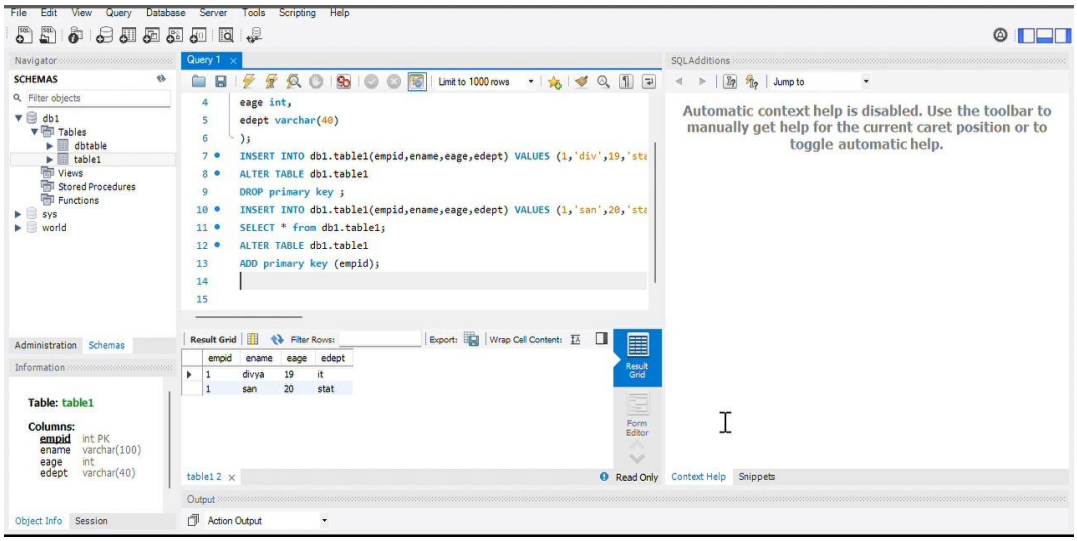

Fig 20.12 Primary Key

21.0 Lambda:

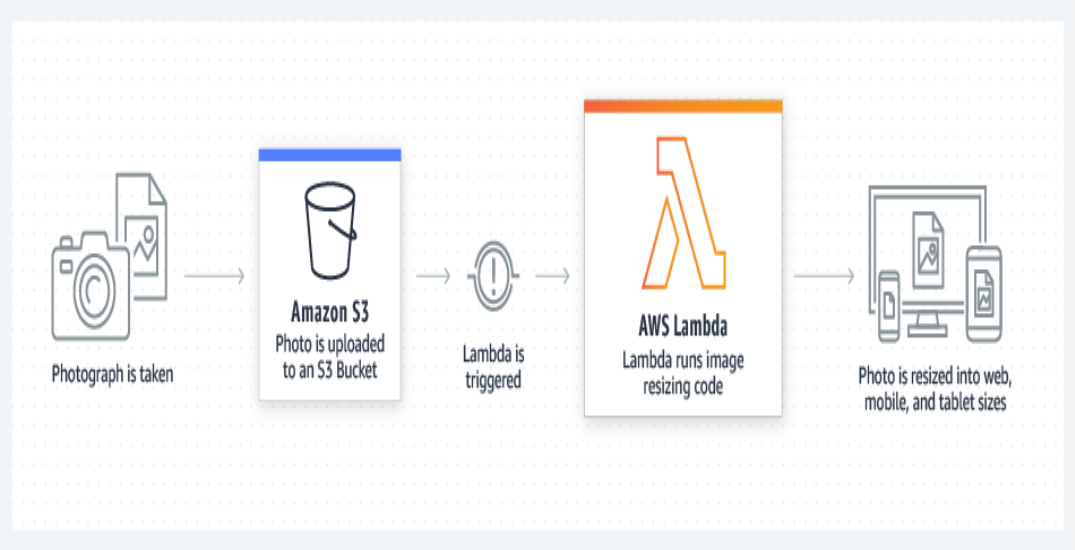

Fig 21.1 AWS Lambda

AWS Lambda is a compute service that lets you run code without provisioning or managing servers.

Lambda runs your code on a high-availability compute infrastructure and performs all of the administration of the compute resources, including server and operating system maintenance, capacity provisioning and automatic scaling, and logging. With Lambda, all you need to do is supply your code in one of the language runtimes that Lambda supports.

You organize your code into Lambda functions. The Lambda service runs your function only when needed and scales automatically. You only pay for the compute time that you consume—there is no charge when your code is not running.

21.1 Key Features

The following key features help you develop Lambda applications that are scalable, secure, and easily extensible:

- Configuring function options
- Environment Variables
- Versions
- Container Images

- Layers
- Lambda Extensions

- Function URLs
- Response streaming
- Code signing
- Private networking
- Data Base access proxy

21.1.1 Configuring Function Options:

Configure your Lambda function using the console or AWS CLI.

21.1.2 Environment Variables:

Use environment variables to adjust your function's behavior without updating code.

21.1.3 Versions:

Manage the deployment of your functions with versions, so that, for example, a new function can be used for beta testing without affecting users of the stable production version.

21.1.4 Container Images:

Create a container image for a Lambda function by using an AWS provided base image or an alternative base image so that you can reuse your existing container tooling or deploy larger workloads that rely on sizable dependencies, such as machine learning.

21.1.5 Layers:

Package libraries and other dependencies to reduce the size of deployment archives and makes it faster to deploy your code.

21.1.6 Lambda Extensions:

Augment your Lambda functions with tools for monitoring, observability, security, and governance.

21.1.7 Function URLs:

Add a dedicated HTTP(S) endpoint to your Lambda function.

21.1.8 Response Streaming:

Configure your Lambda function URLs to stream response payloads back to clients from Node.js functions, to improve time to first byte (TTFB) performance or to return larger payloads.

21.1.9 Code Signing:

Verify that only approved developers publish unaltered, trusted code in your Lambda functions

21.1.10 Private Networking:

Create a private network for resources such as databases, cache instances, or internal services.

21.1.11 Database Access and Proxy:

Create an Amazon RDS Proxy database proxy to manage a pool of database connections and relay queries from a function. With a proxy, your function can achieve high concurrency levels without exhausting database connections.

22.0 Conclusions:

Providing highly secure and resilient infrastructure and services to our customers is a top priority for AWS. Our commitment to our customers is focused on working to continuously earn customer trust and ensure customers maintain confidence in operating their workloads securely on AWS. To achieve this, AWS has integrated risk and compliance mechanisms that include:

- The implementation of a wide array of security controls and automated tools.
- Continuous monitoring and assessment of security controls to help ensure AWS operational effectiveness and strict adherence to compliance regimes.
- Independent risk assessment by the AWS Business Risk Management program.
- Operational and business management mechanisms.

In addition, AWS regularly undergoes independent third-party audits to provide assurance that the control activities are operating as intended. These audits, along with the many certifications AWS has obtained, provide an additional level of validation of the AWS control environment that benefit customers.

Taken together with customer-managed security controls, these efforts allow AWS to securely innovate on behalf of customers and help customers improve their security posture when building on AWS.

23.0 References:

- AWS Architecture Center
- AWS Whitepapers
- AWS Architecture Monthly
- AWS Architecture Blog
- This Is My Architecture videos
- AWS Documentation