**Identity and Access Management (IAM):**

Identity and Access Management (IAM) in AWS is crucial for controlling access to resources securely. Here are some key concepts:

1. Users: Individual identities that can access AWS services.
2. Groups: Collections of users that can share permissions.
3. Roles: AWS identities with specific permissions, often used for applications or services that need access to AWS resources.
4. Policies: JSON documents that define permissions. They can be attached to users, groups, or roles.
5. Multi-Factor Authentication (MFA): Adds an extra layer of security by requiring a second form of verification.

**Creating an IAM user:**

 **Sign in to the AWS Management Console:**

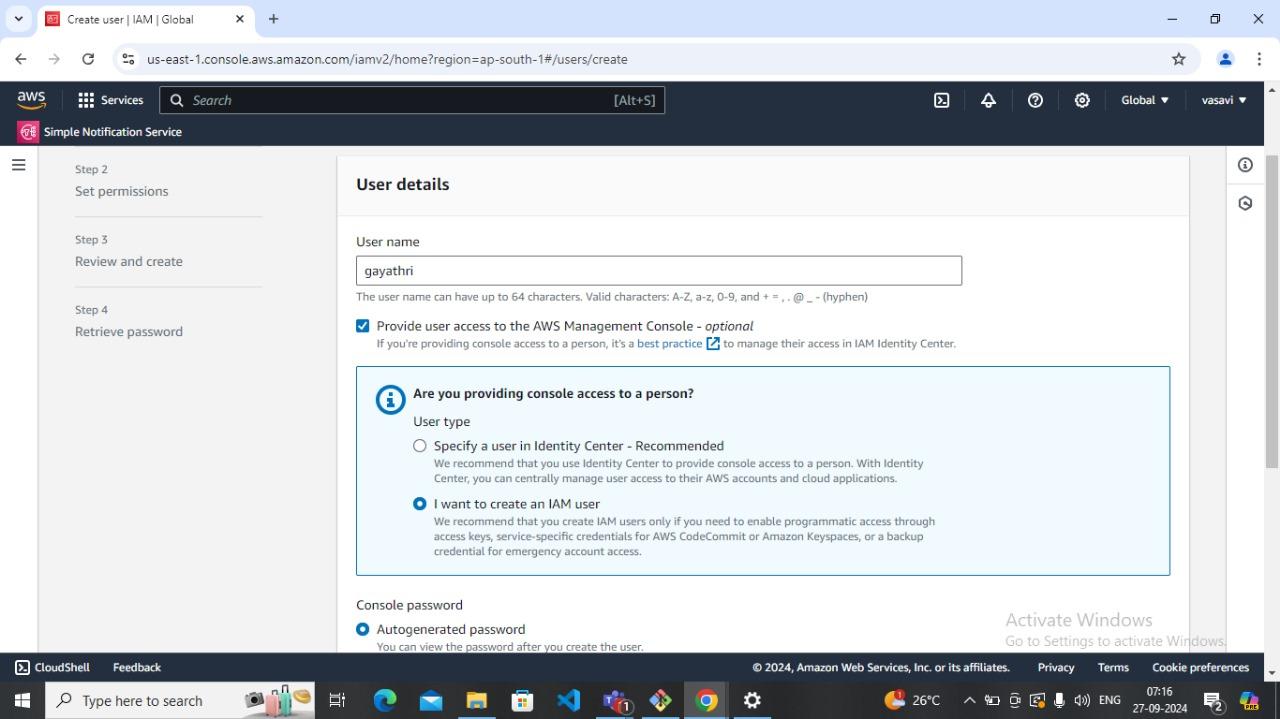
* Go to the AWS Management Console and sign in with your root account or IAM user credentials.

 **Open the IAM Console:**

* In the AWS Management Console, search for "IAM" in the services search bar and select "IAM."

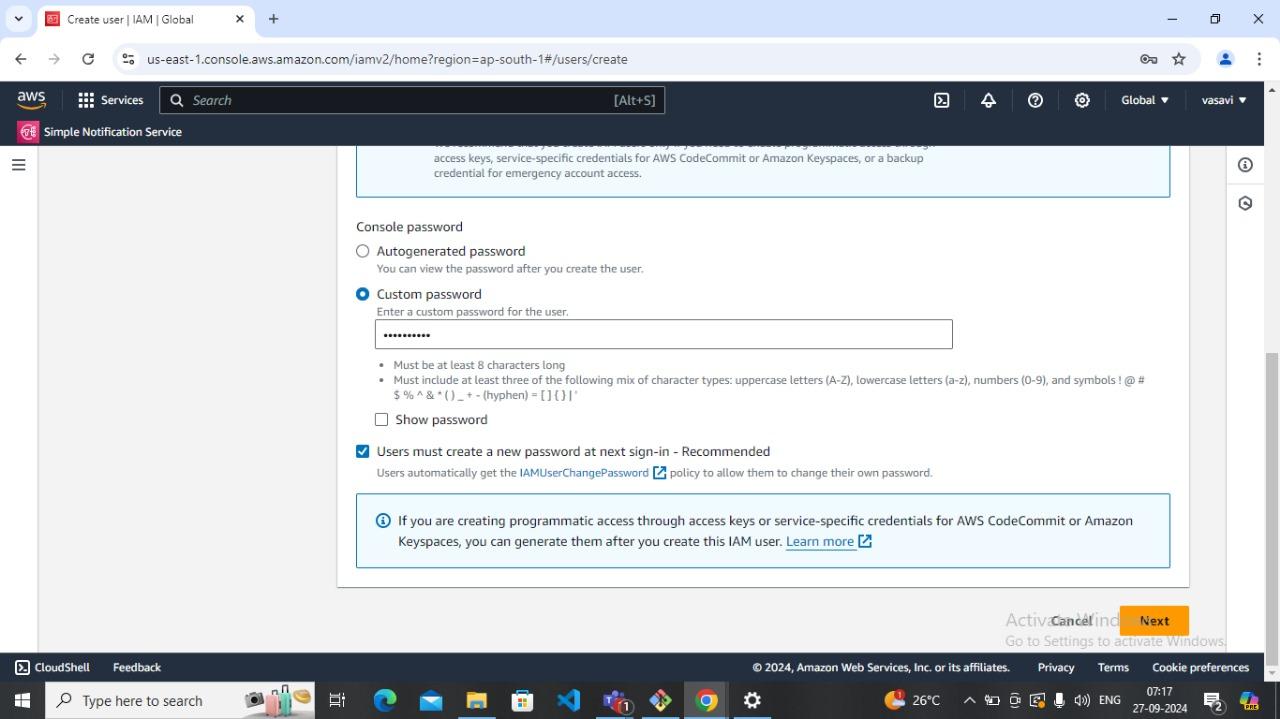
 **Add User:**

* In the IAM dashboard, click on "Users" in the sidebar.
* Click the "Add user" button.



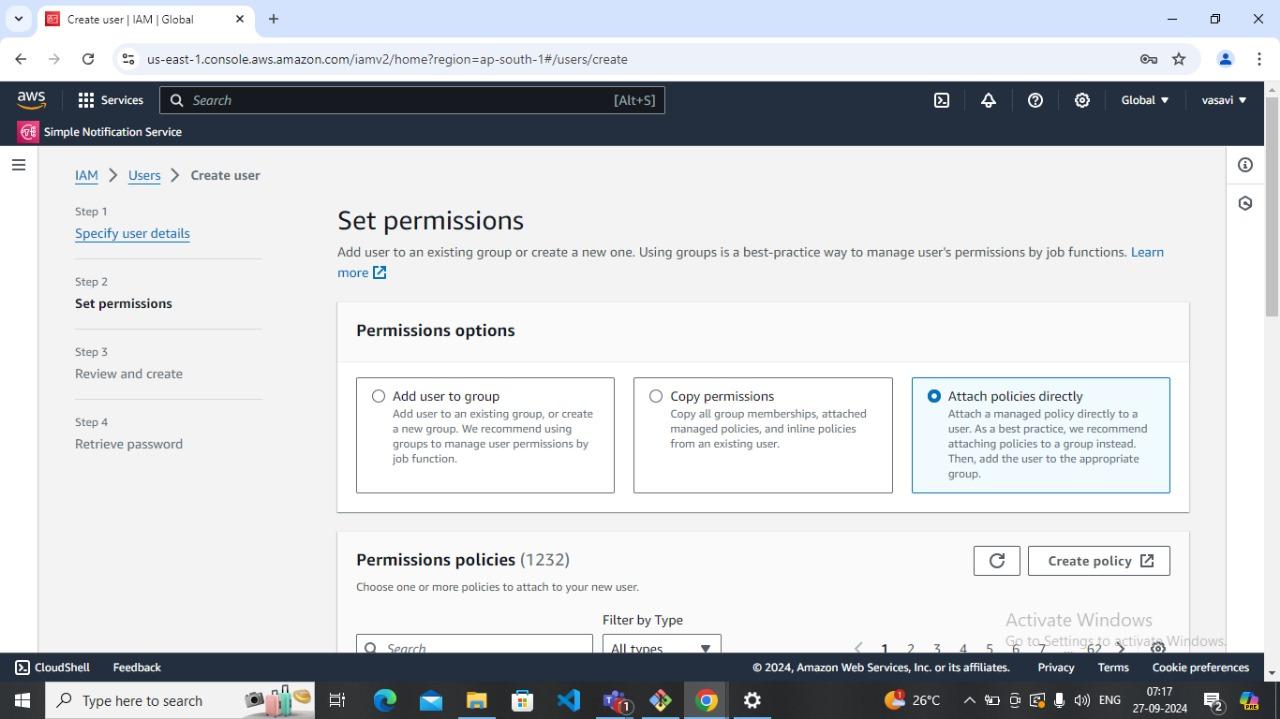
 **Set User Details:**

* Enter a username for the new user.

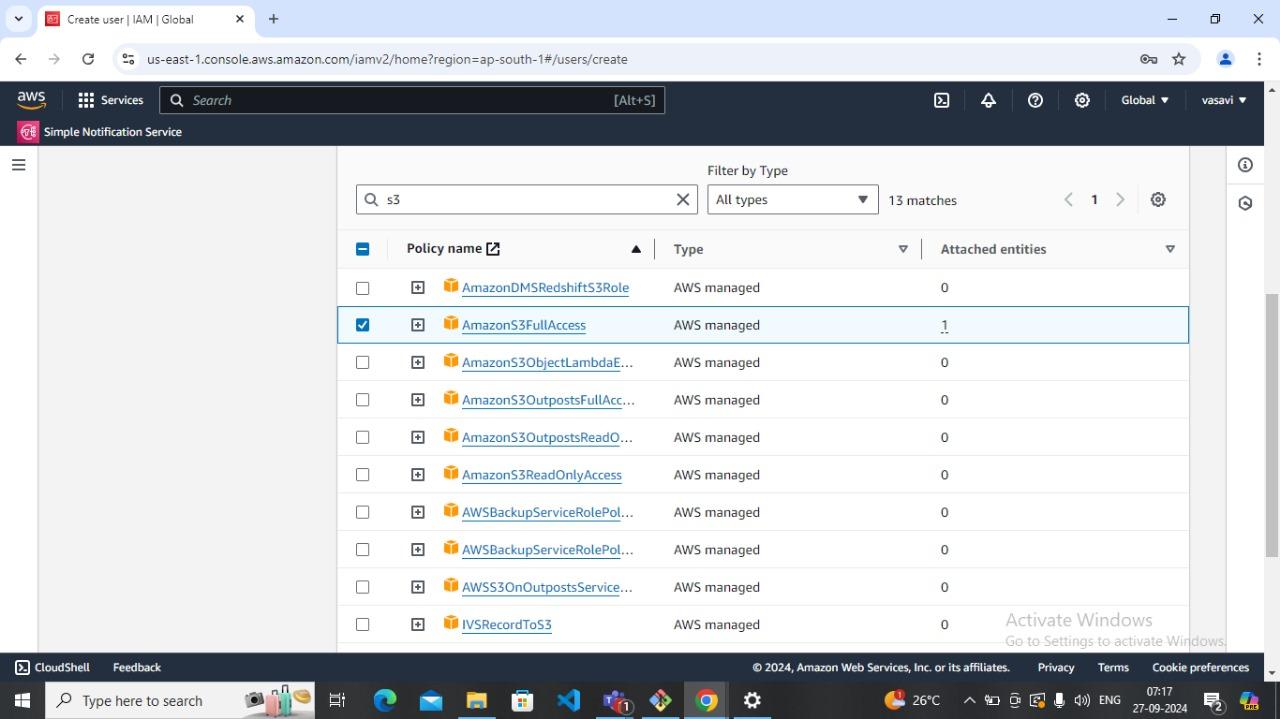


 **Set Permissions:**

* Choose how to assign permissions:
  + Attach custom policy: Create and attach a custom policy.



**Choose required policy:**

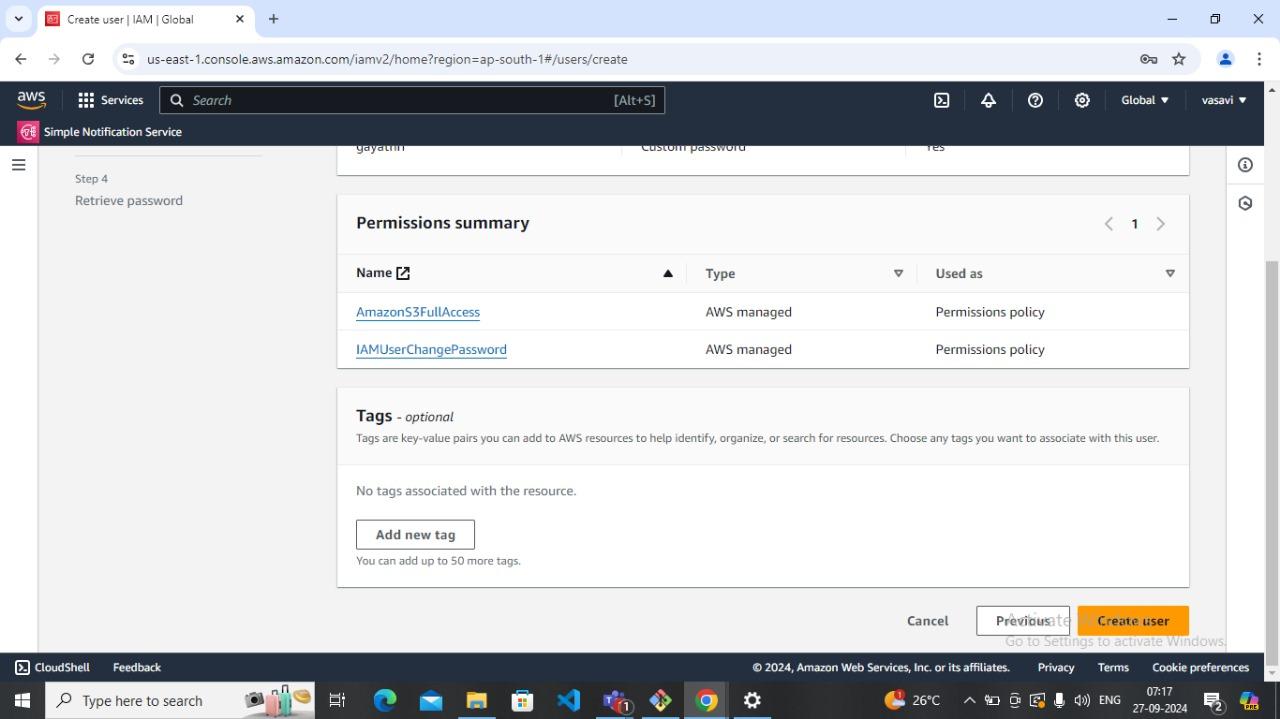


 **Review:**

* Review the settings for the new user. Ensure everything is correct.

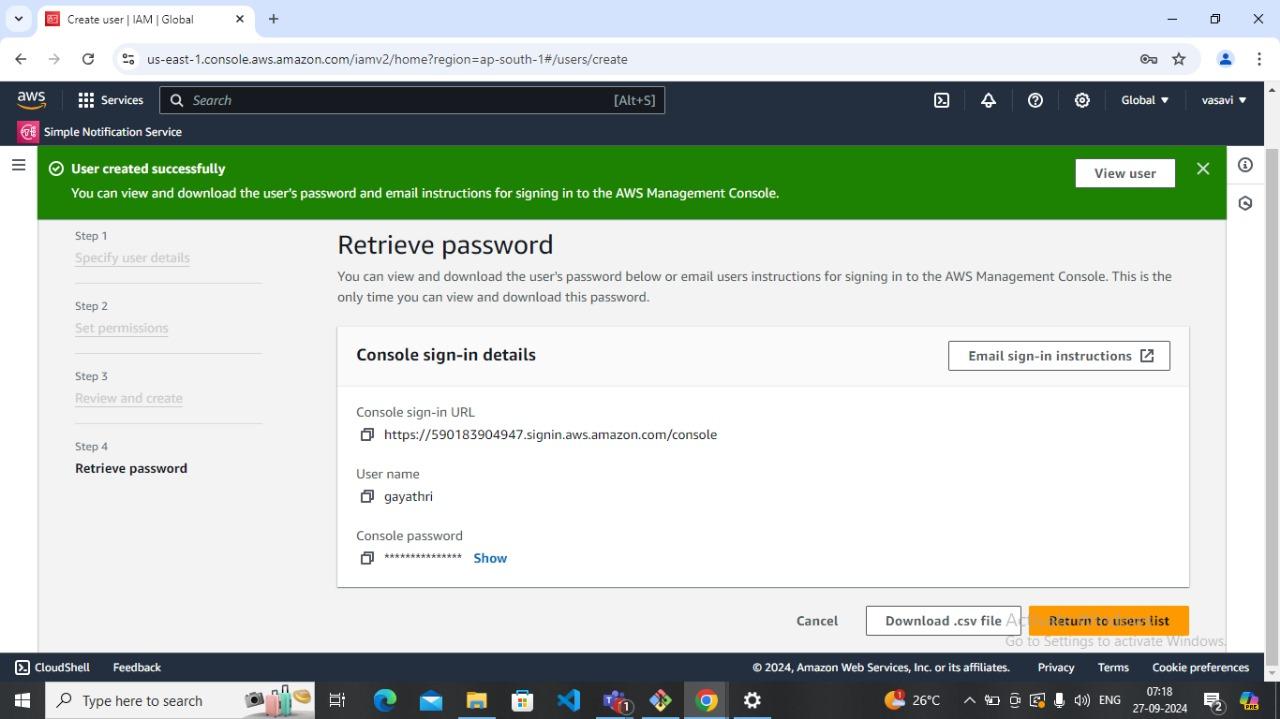
 **Create User:**

* Click the "Create user" button.



** Save Credentials:**

* After creation, you'll see a success message. Download or copy the access key ID and secret access key (if you enabled programmatic access). Note that you won’t be able to view the secret access key again.



* **Sign into IAM user using credential:**

. Provide Credentials to the Other Person

For AWS Management Console Access:

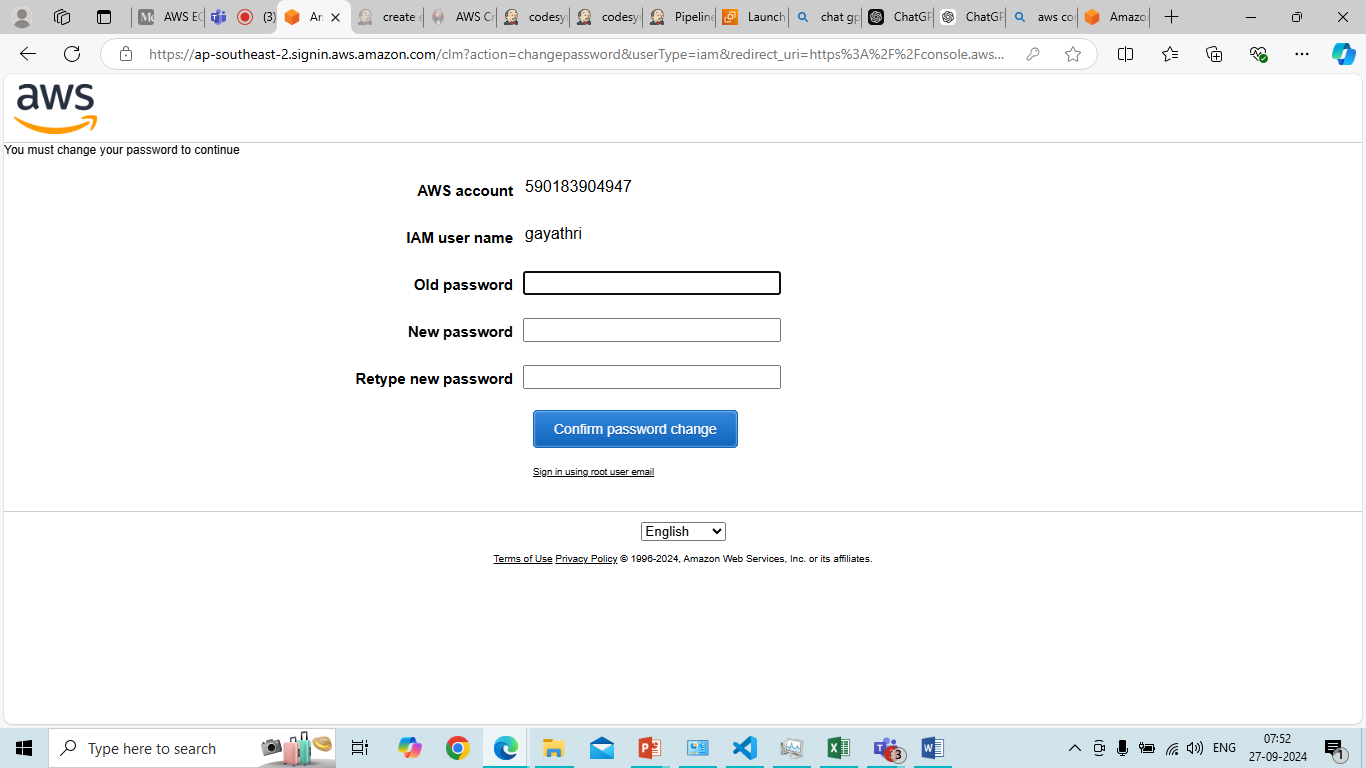
Provide the user with the username, password, and the AWS Console URL (https://.[signin.aws.amazon.com/console](http://signin.aws.amazon.com/console)).

Instruct them to reset the password if necessary on the first login

Verify Access

After the user logs in, they can access the S3 bucket based on the permissions you assigned.

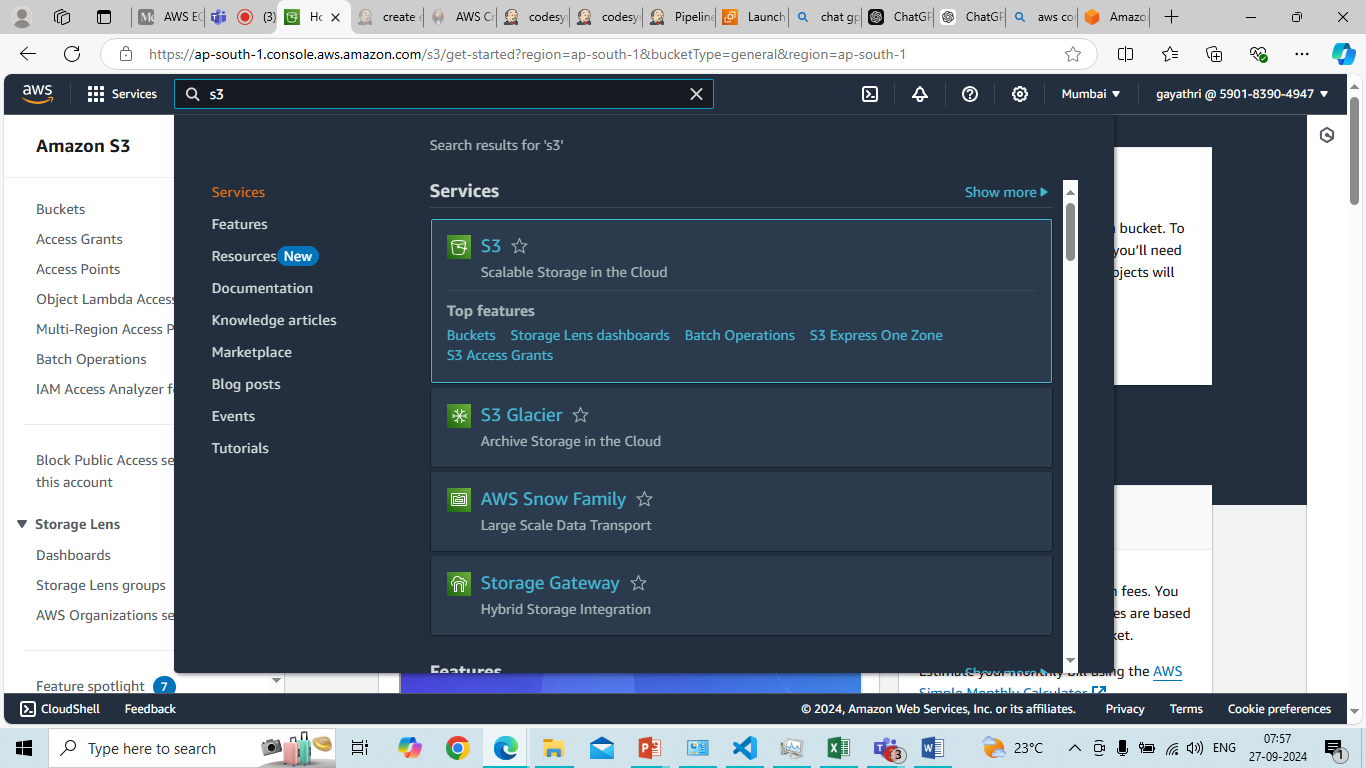
For Console Access, they should be able to navigate to the S3 service and see or manage the buckets and objects as per their policy.



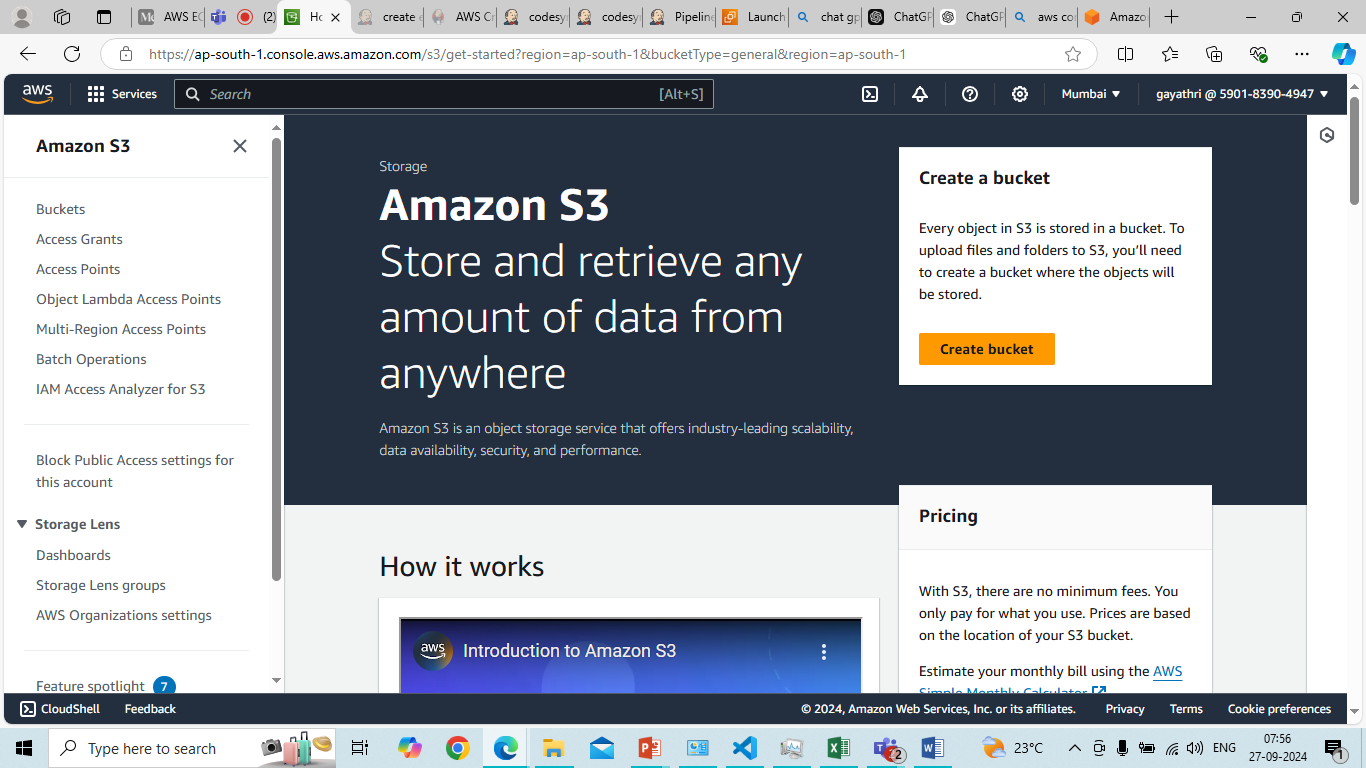
**Steps to Create an S3 Bucket:**

**Using AWS Management Console**

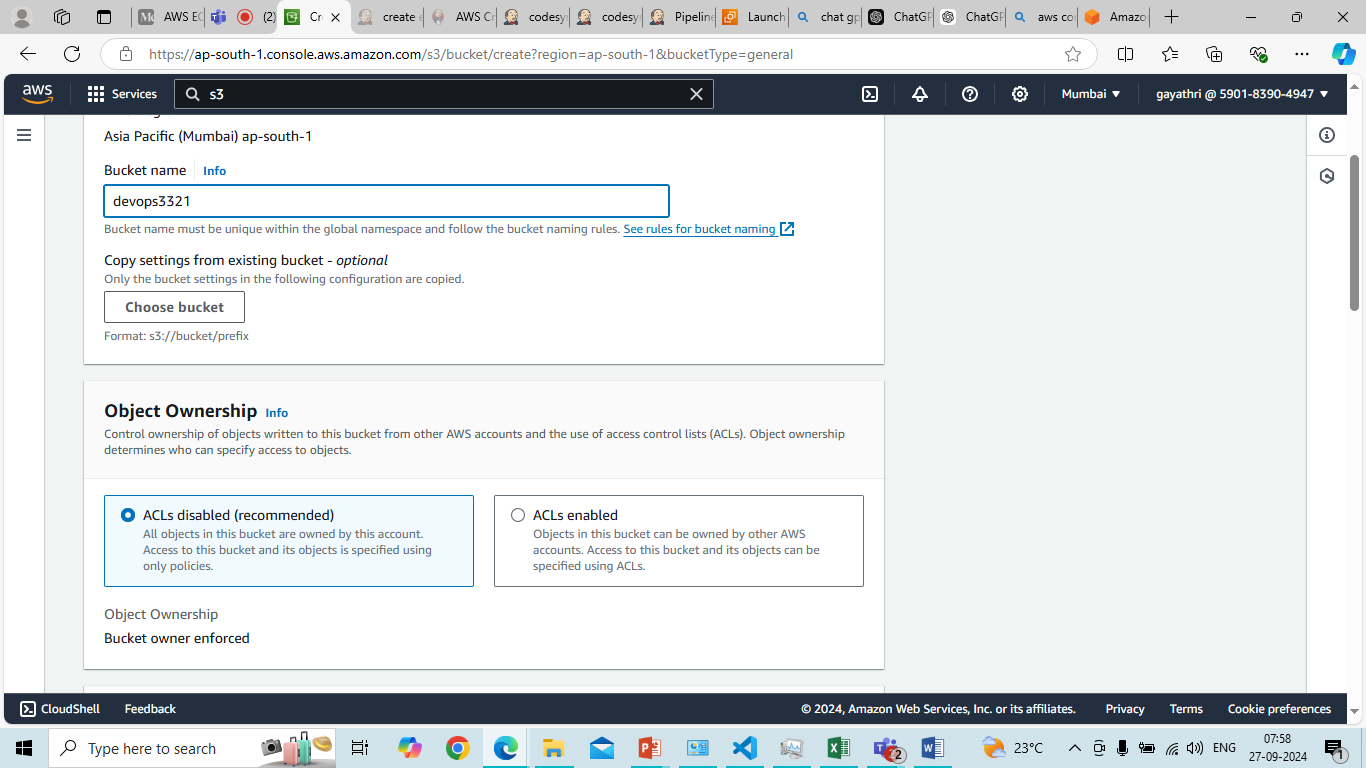
1. **Sign In:**
   * Go to the [AWS Management Console](https://aws.amazon.com/console/) and sign in with your IAM user credentials.
2. **Open S3:**
   * In the console, search for and select S3.



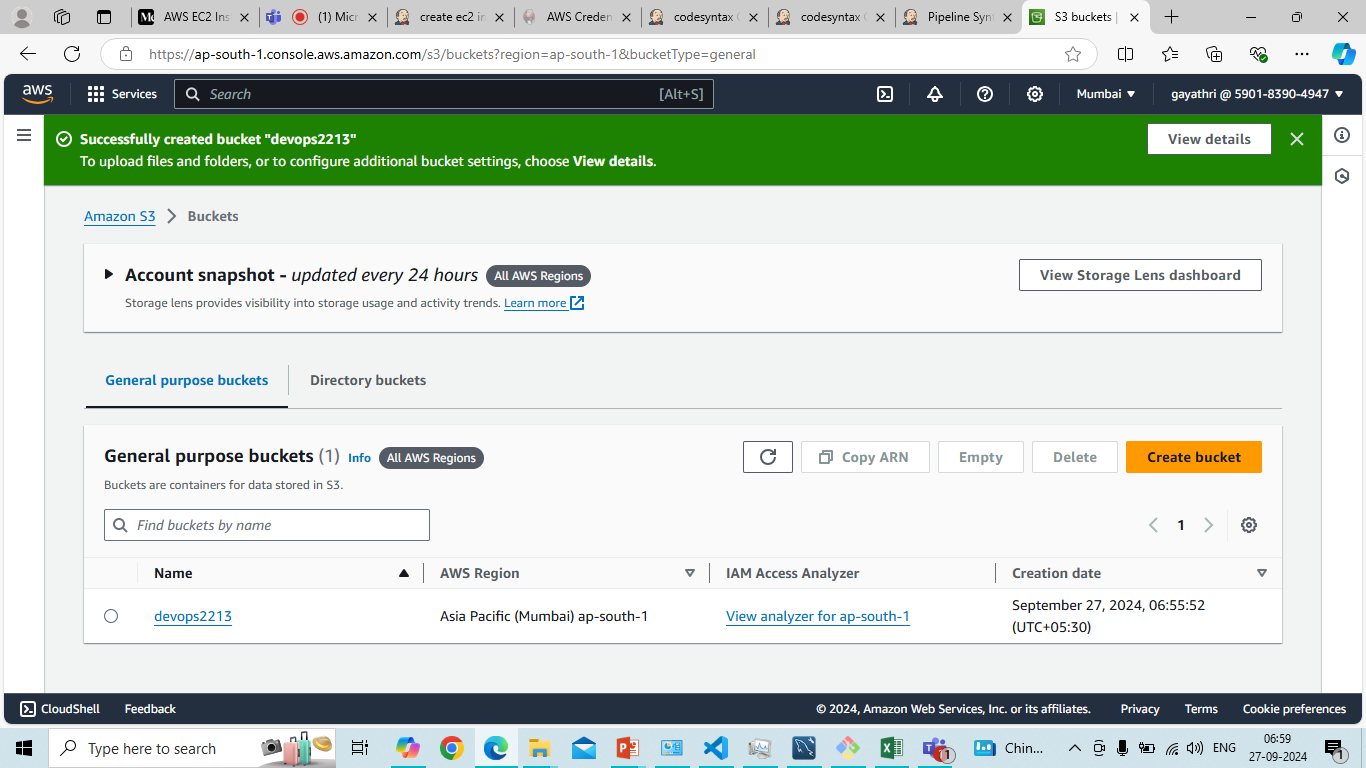
1. **Create Bucket:**
   * Click on the Create bucket button.



1. **Configure Bucket:**
   * Bucket name: Enter a unique name (must be globally unique).
   * Region: Choose the AWS region for your bucket.
   * Options: Configure additional options like versioning, logging, and encryption as needed.
   * Permissions: Set your desired permissions and public access settings.



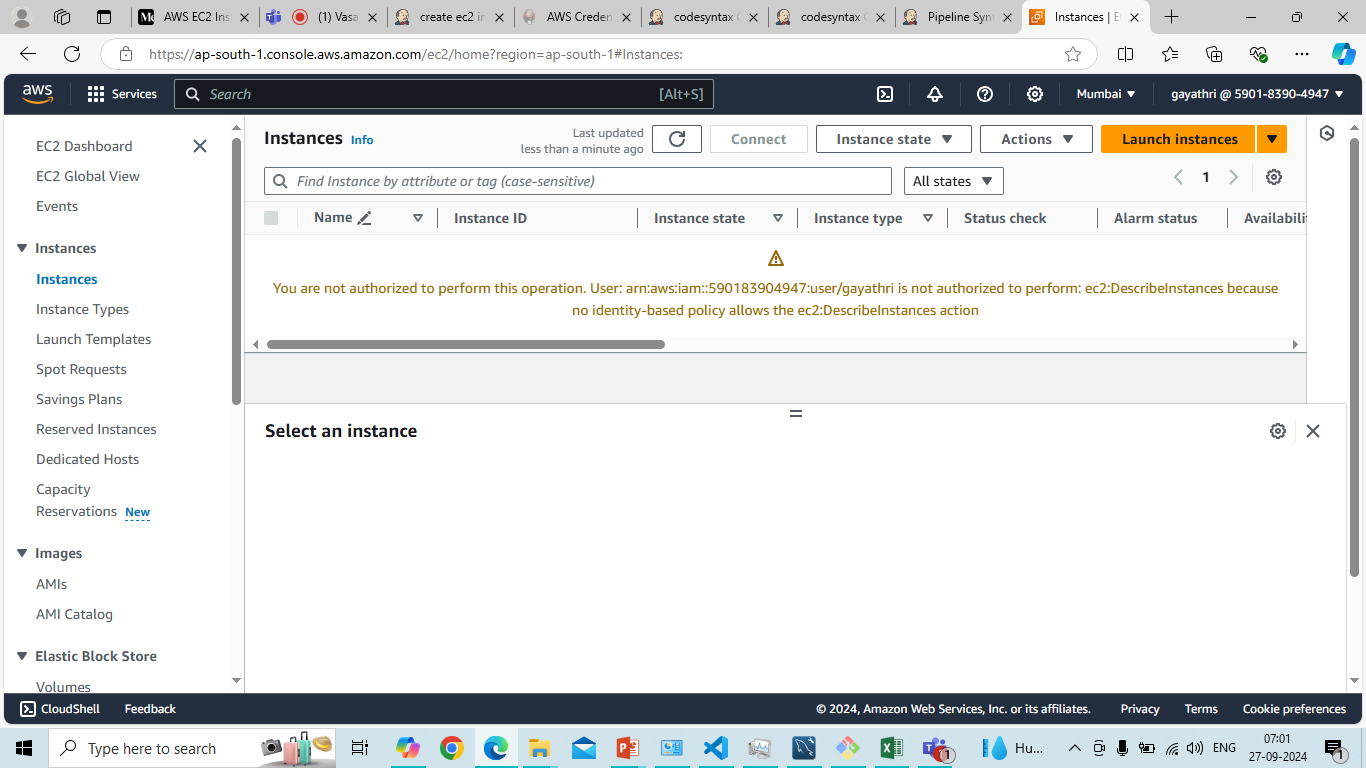
1. **Review and Create:**
   * Review your settings and click Create bucket.



**6 Complete:**

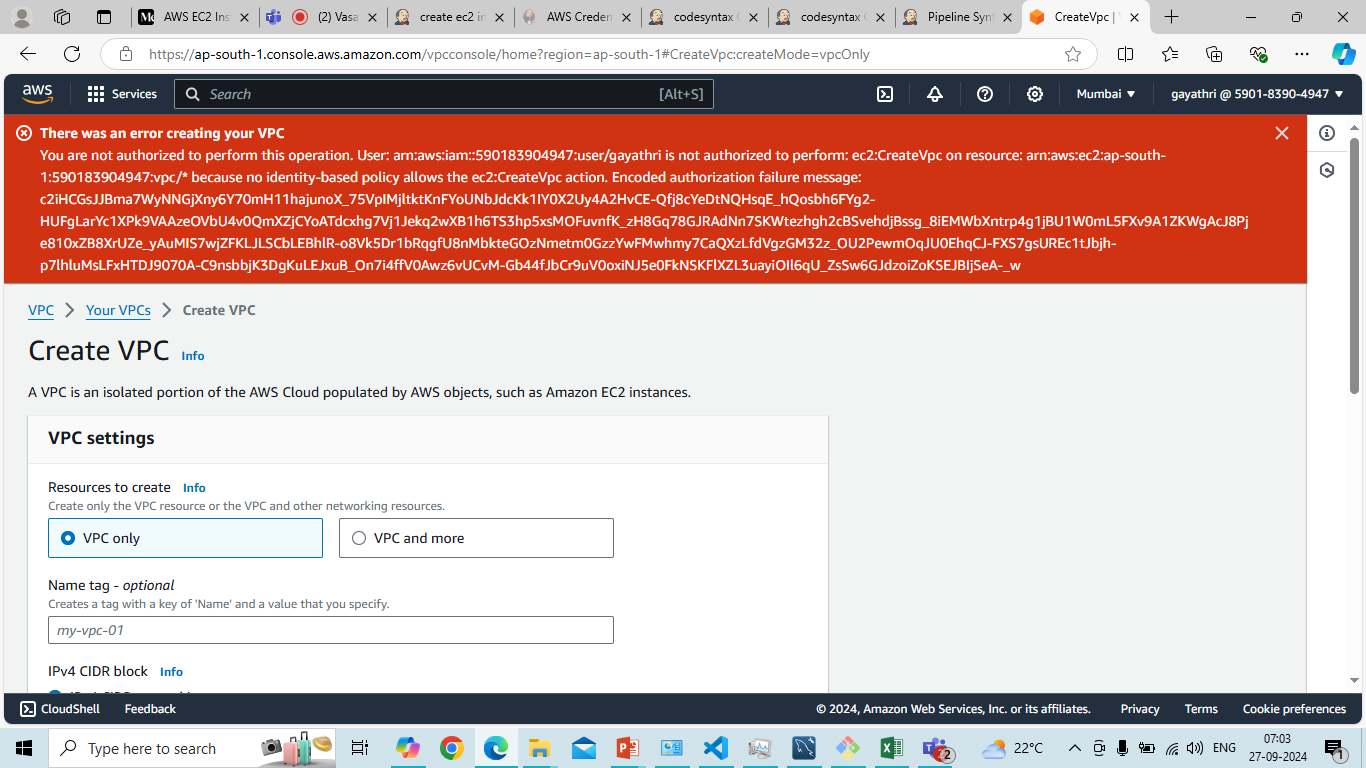
* You can now use the new IAM user according to the permissions granted.

**7 Try to access other services: example EC2**



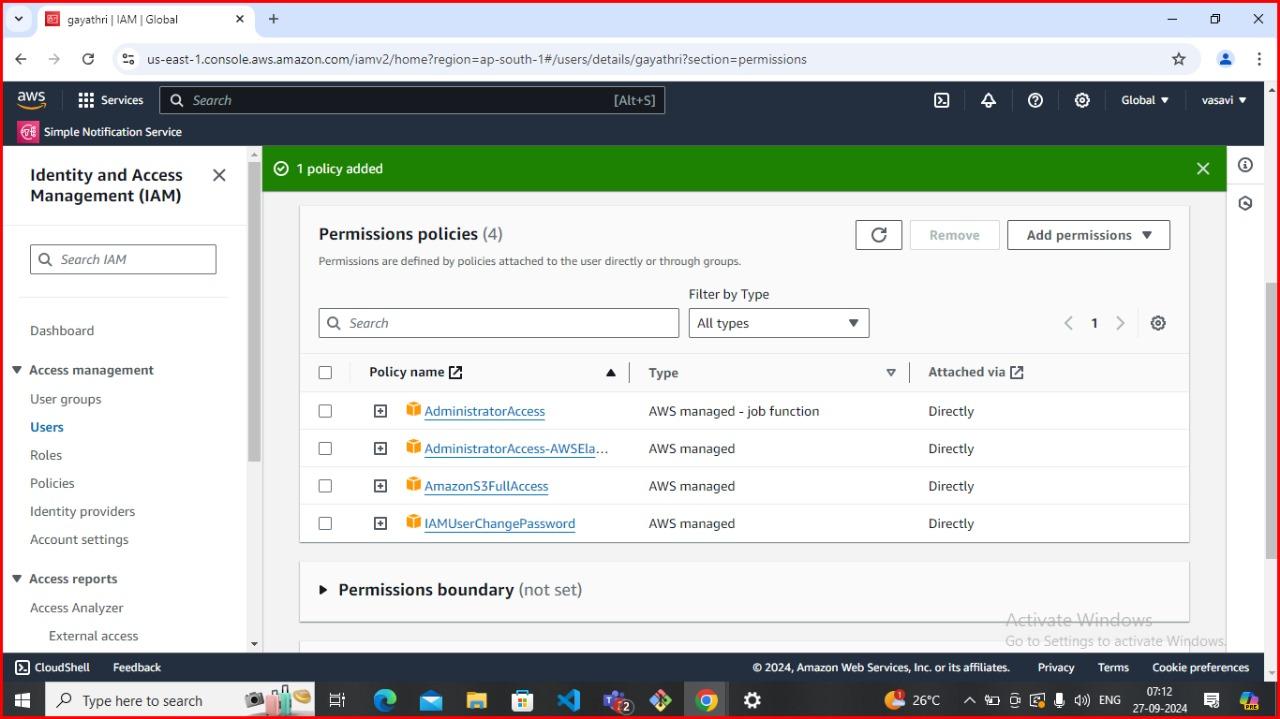
Here we can see some error because we haven’t gave any access to ec2. We have given access or permission only to S3. So if we try to access other services it will not get accessed.

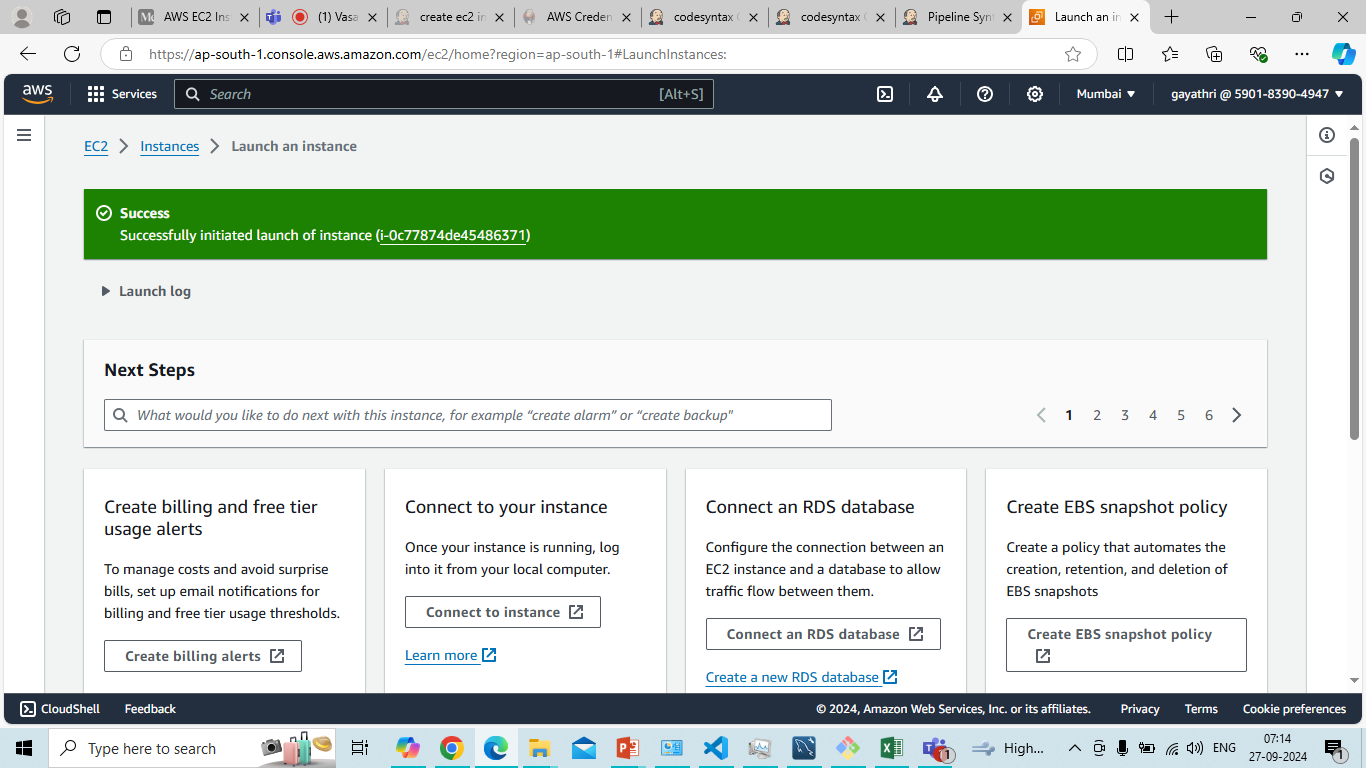
Other example: **VPC**



Here also like in the same case of Ec2 it will not get opened because we haven’t gave any access to VPC. We can only access other services if we set the permissions to the services.

8. Adding additional permission policies to IAM user:

* Select the permission policies required and click on next then the permissions will get added
* Here we gave administrator access so we can access Ec2 services.



**Recap:**

* You created a new IAM user.
* Attached an S3 access policy (either full or restricted).
* Provided credentials for either Console or CLI access to the other person

### Symmetric Encryption in IAM Context

1. **Definition**: Symmetric encryption uses the same key for both encryption and decryption. It’s faster than asymmetric encryption but requires secure key management.
2. **Use Cases**:
   * **Data Encryption**: Protecting sensitive data stored in AWS services (like S3).
   * **Secure Data Transfer**: Encrypting data in transit using symmetric keys.
3. **Key Management**: AWS offers services like AWS Key Management Service (KMS) to manage encryption keys securely, including symmetric keys.

### Asymmetric Encryption Overview

1. **Definition**:
   * Asymmetric encryption uses two keys: a public key (for encryption) and a private key (for decryption). This allows secure data exchange and digital signatures.
2. **Use Cases**:
   * **Secure Data Transmission**: Encrypting sensitive information sent over the internet.
   * **Digital Signatures**: Authenticating the identity of the sender and ensuring data integrity.

### IAM and Asymmetric Encryption

In AWS IAM, asymmetric keys can be used in various scenarios:

1. **IAM Roles and Policies**:
   * IAM policies can define permissions for users and roles, including the ability to use asymmetric keys.
2. **AWS Key Management Service (KMS)**:
   * AWS KMS allows you to create and manage asymmetric keys. You can generate a key pair (public/private) for use in cryptographic operations.
3. **Use with AWS Services**:
   * **AWS Secrets Manager**: You can encrypt secrets using asymmetric keys for added security.
   * **AWS CloudHSM**: Provides hardware security modules to manage asymmetric keys in a secure environment.

In the context of AWS Identity and Access Management (IAM), encryption typically refers to securing data at rest and in transit. Here’s how you can manage encryption with IAM:

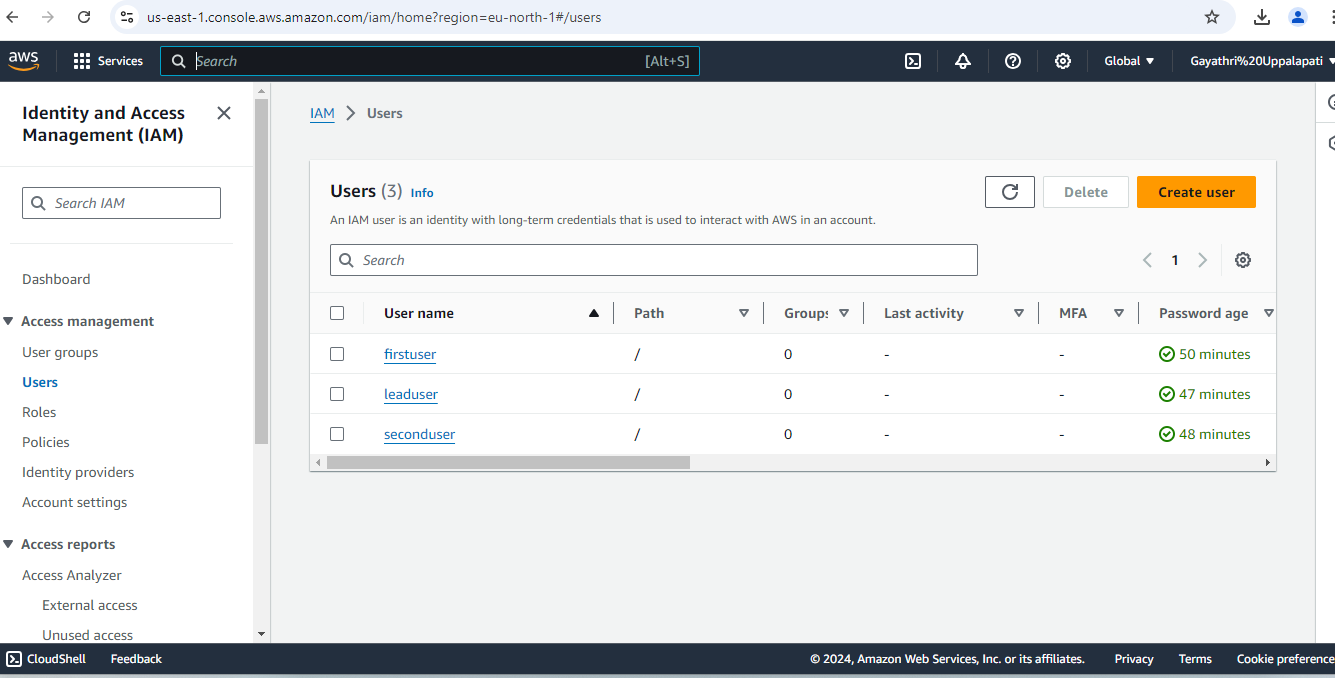
### Key Concepts

1. **Encryption Types**:
   * **At Rest**: Data stored in services like S3, RDS, or EBS is encrypted using encryption keys.
   * **In Transit**: Data transmitted over networks is protected using protocols like TLS (Transport Layer Security).
2. **AWS Key Management Service (KMS)**:
   * AWS KMS is a managed service for creating and controlling encryption keys used to encrypt your data.

In the context of AWS Identity and Access Management (IAM), decrypting data typically involves using AWS Key Management Service (KMS) along with IAM policies to control access. Here’s how to manage decryption securely:

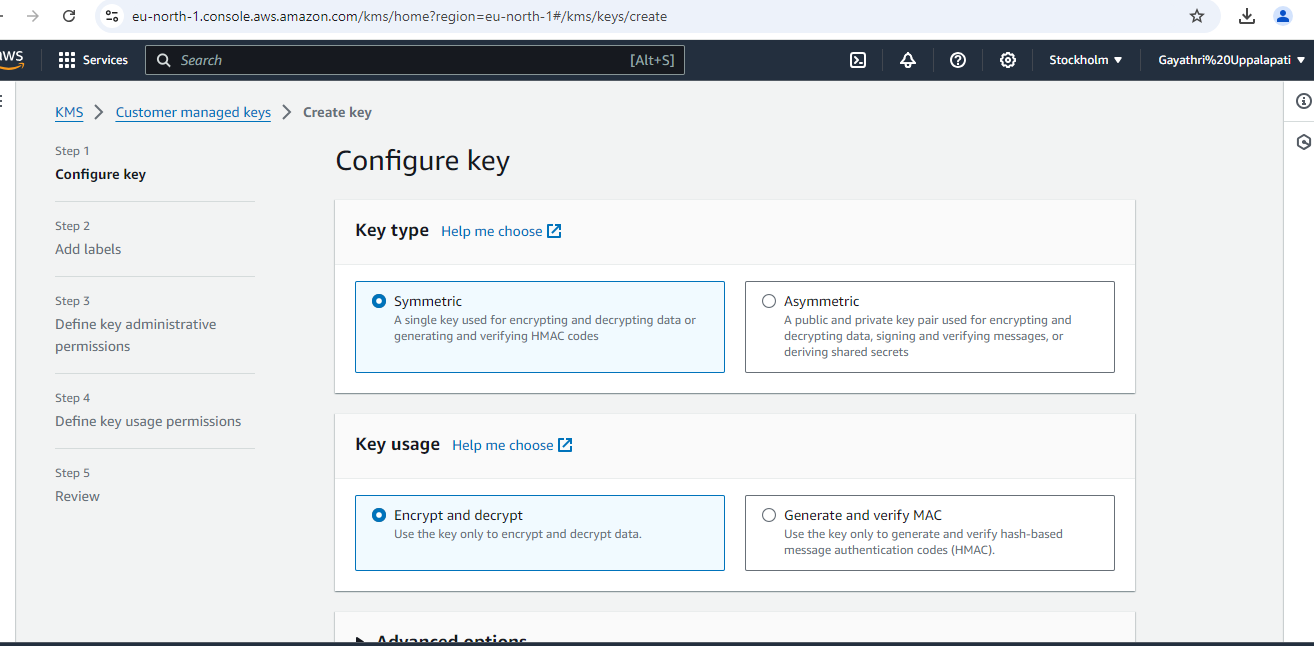
### Key Concepts

1. **Asymmetric vs. Symmetric Encryption**:
   * **Symmetric Encryption**: Uses the same key for encryption and decryption (commonly used in KMS).
   * **Asymmetric Encryption**: Uses a public key for encryption and a private key for decryption.
2. **AWS KMS**:
   * KMS is a managed service that enables you to create, manage, and control encryption keys. You can use KMS keys to encrypt and decrypt data.

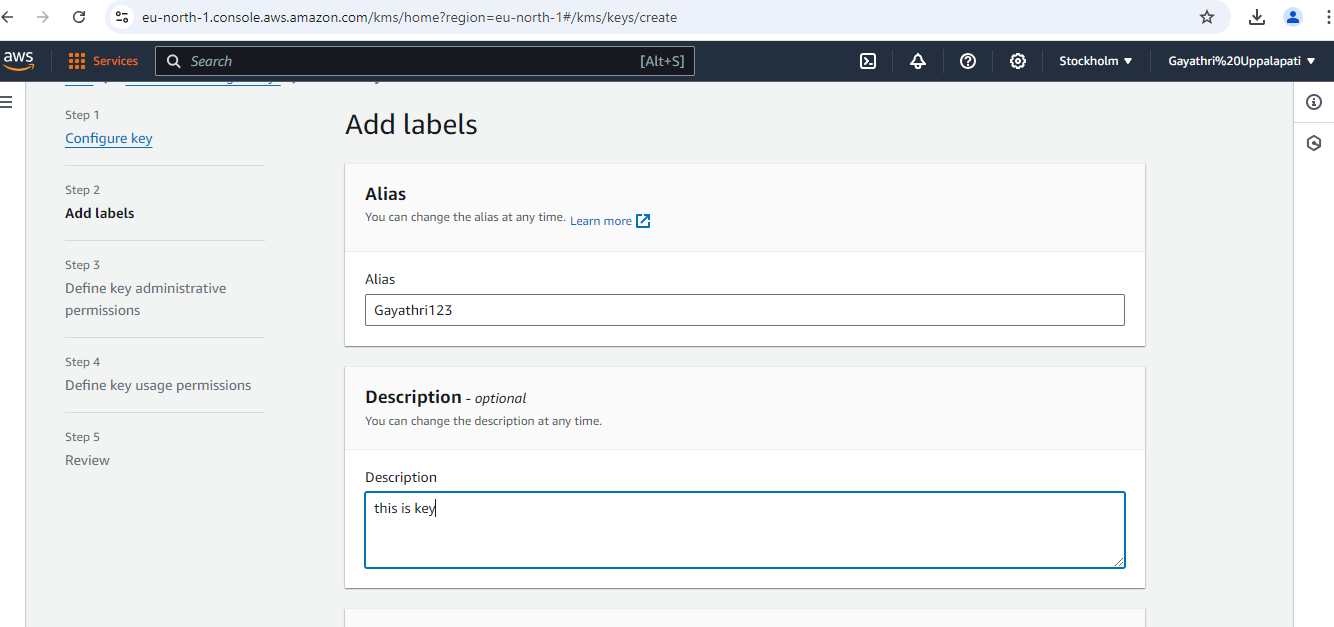
Step 1: Create 3 IAM users with different names and give S3 full access to those users:

here we have created firstuser, leaduser, seconduser.

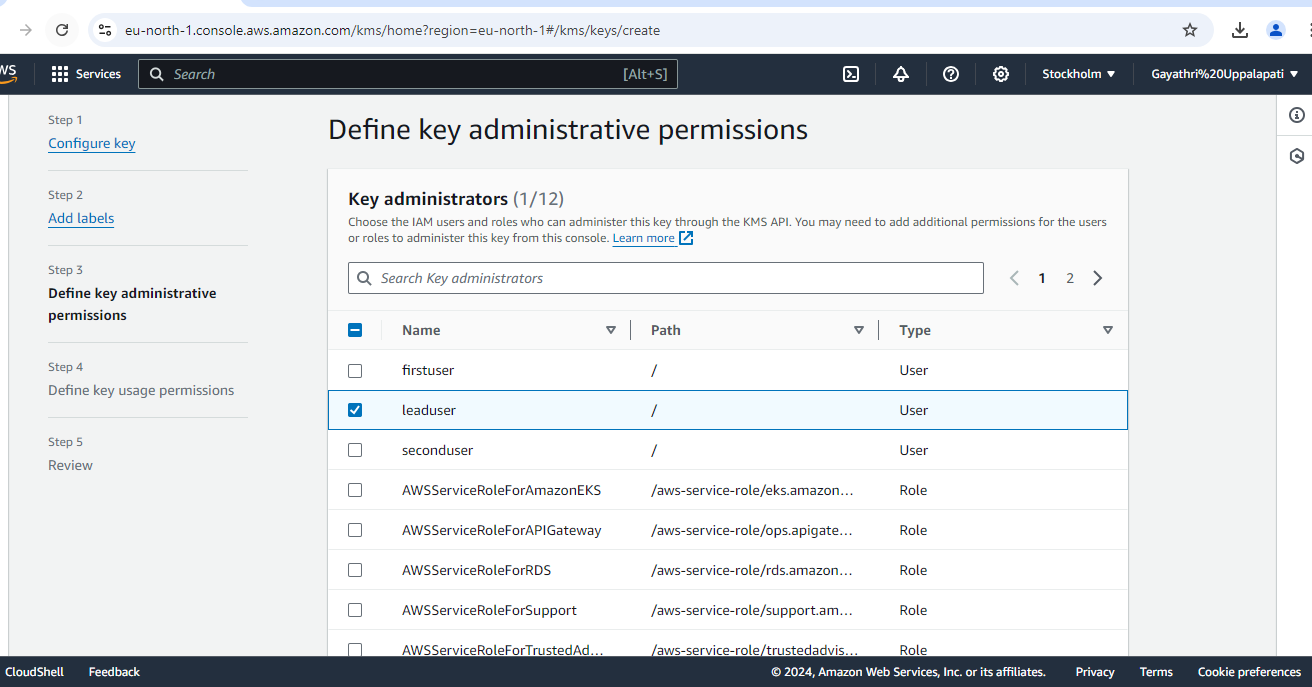
Step 2: Creating KMS key: Here we have created a KMS key only for one particular user. In this case it is leaduser



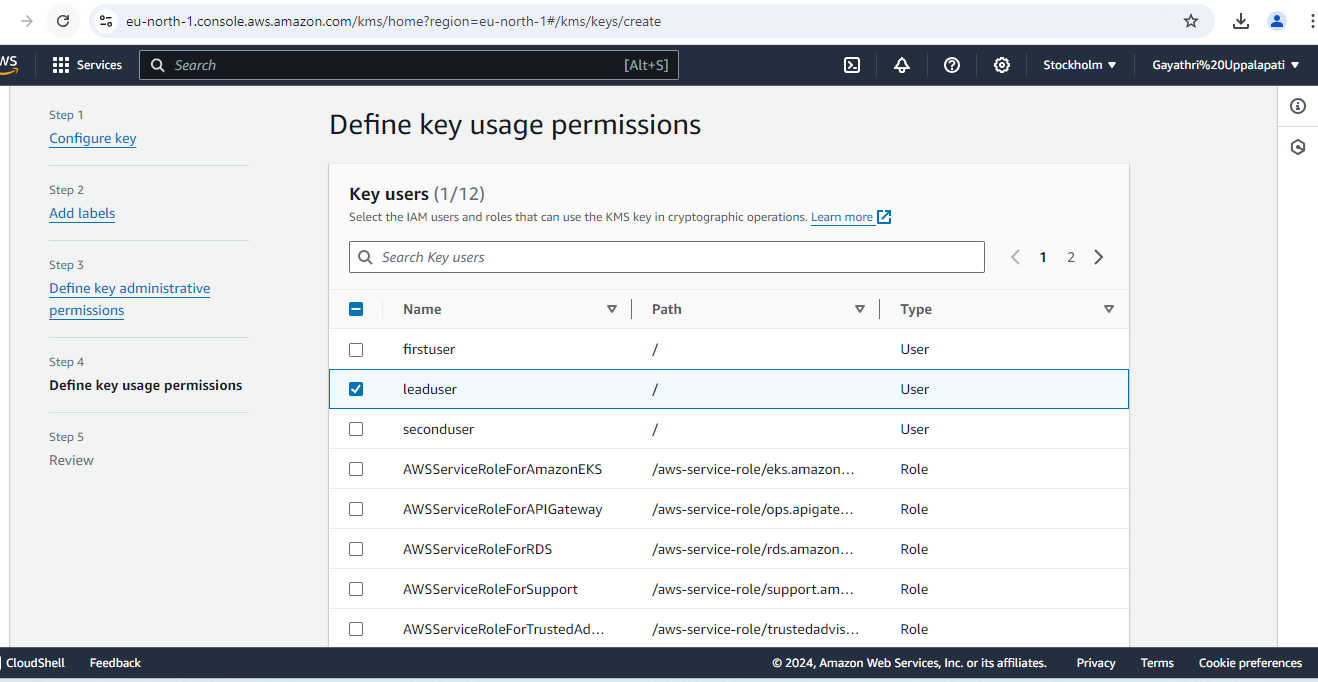
Add lable:



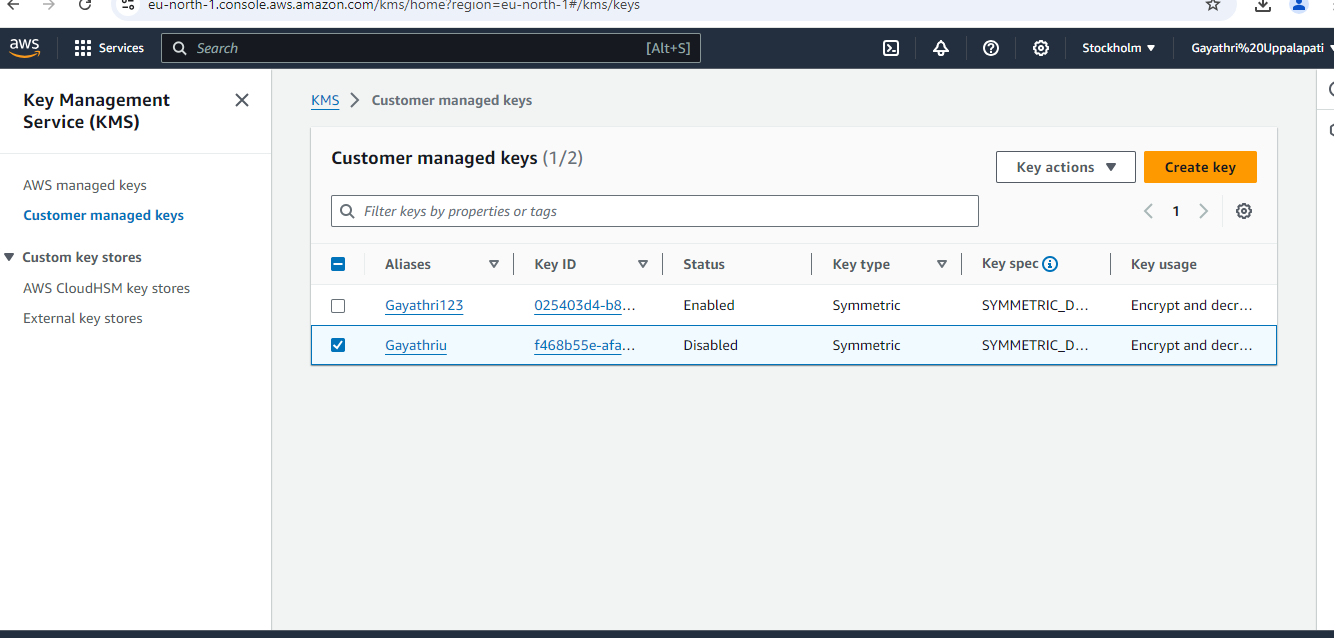
Key administrative permission:



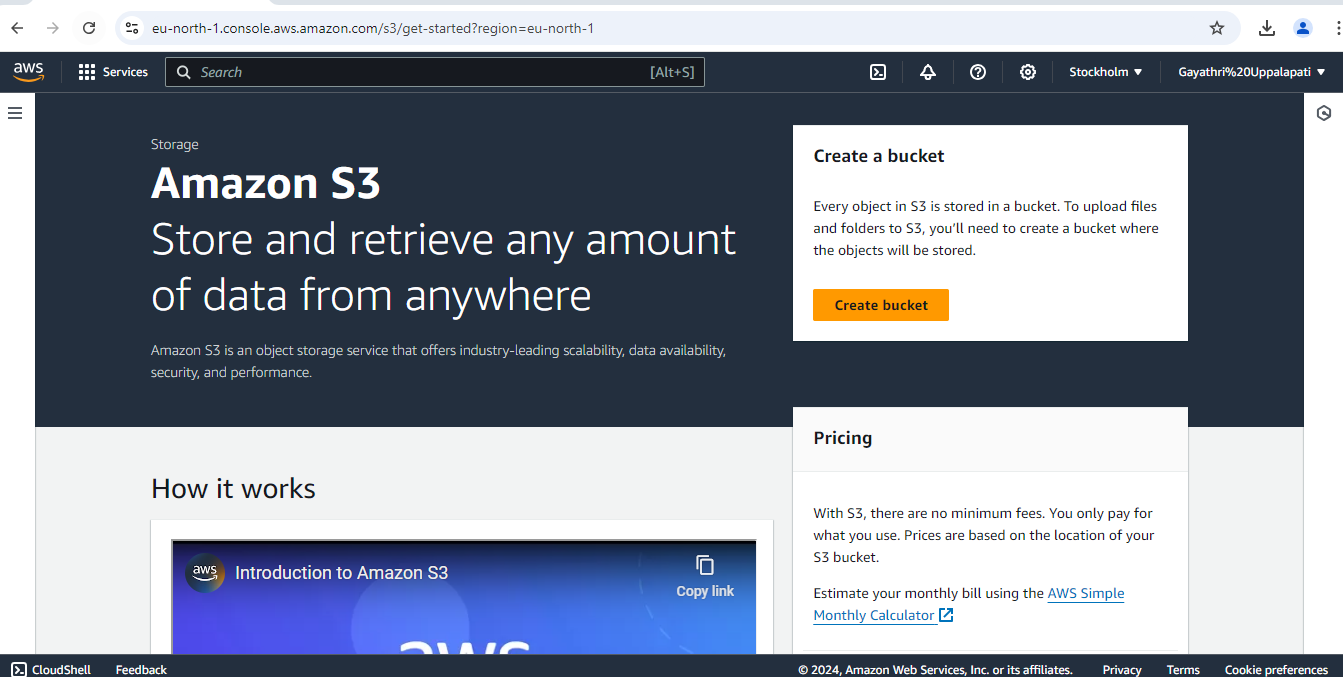
Define key usage permission:



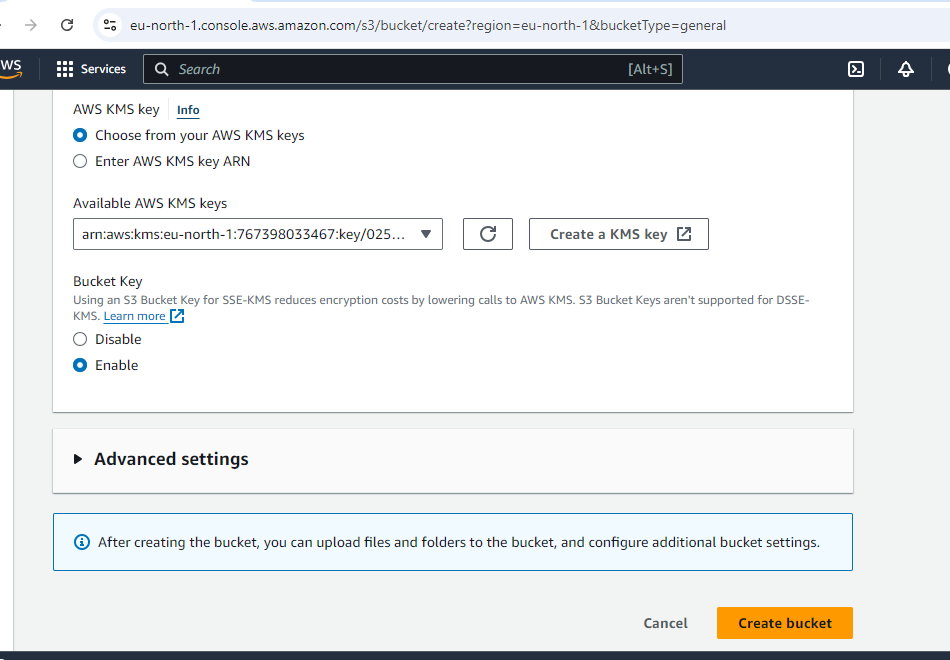
Step 3: Customer managed keys: Here we have successfully added KMS key to the lead user



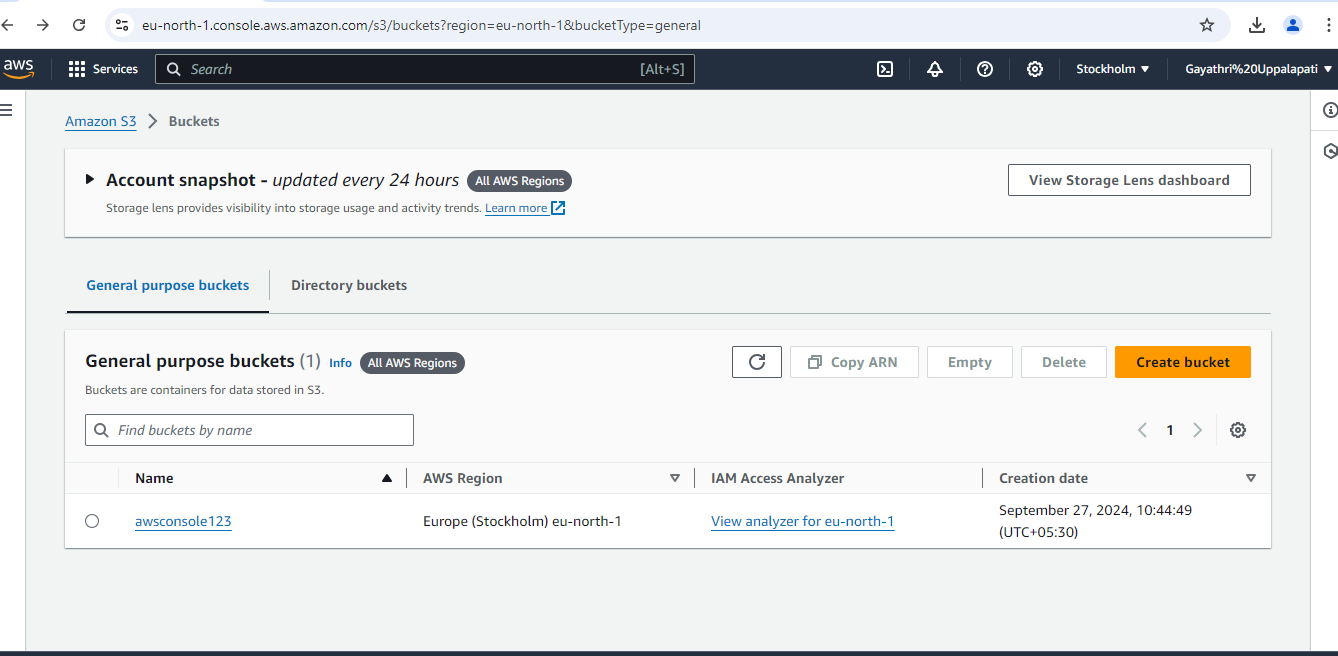
Step 4: Creating S3 bucket: here we are going to create S3 bucket in the leaduser.



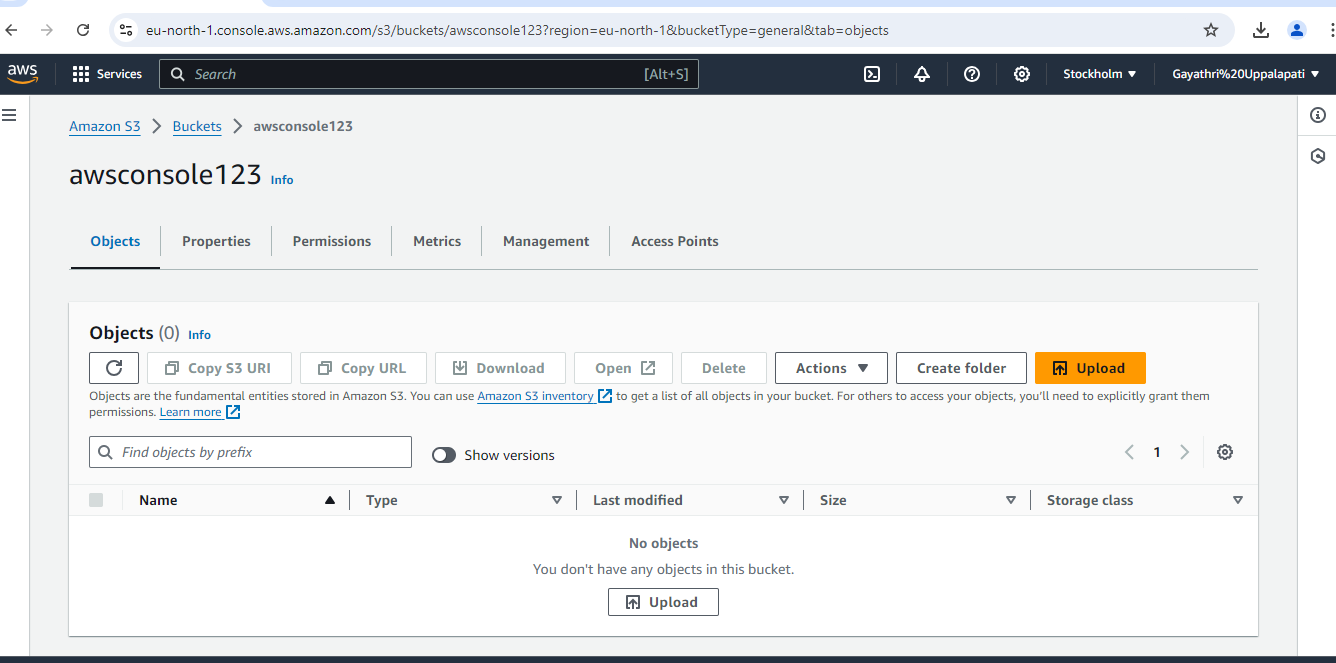
Step 5: Adding KMS key into S3 bucket: we have created a bucket and added already created KMS key to the particular bucket in the leaduser.



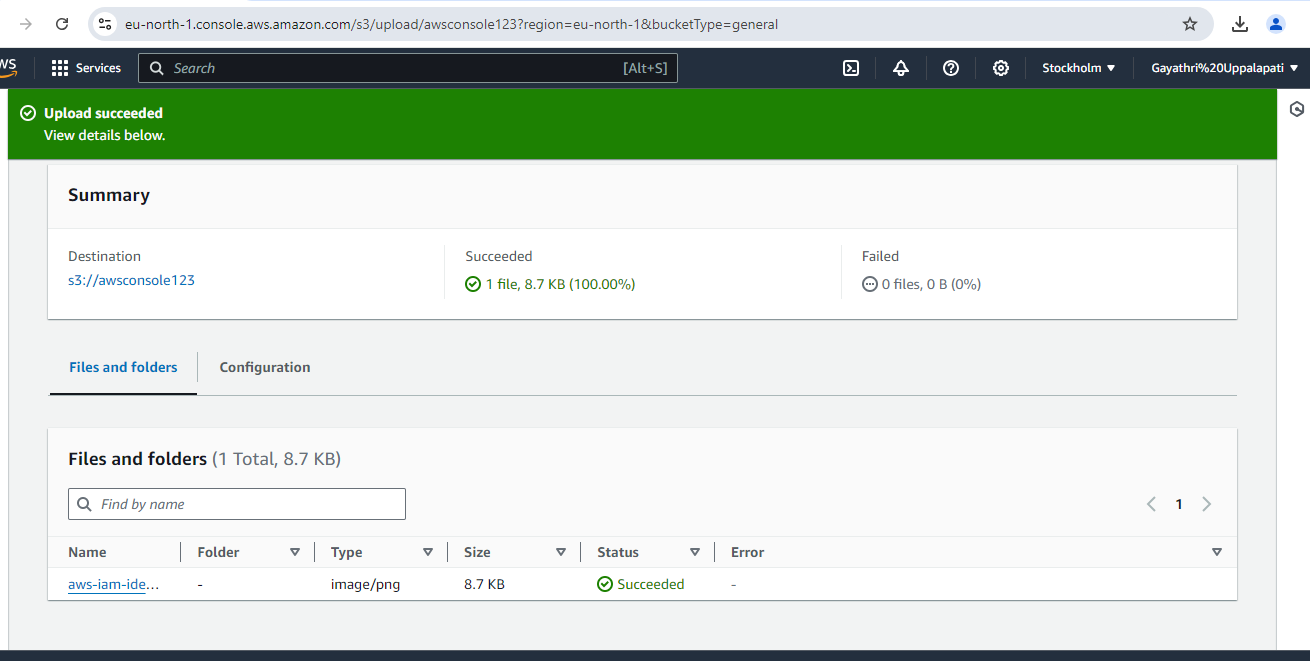
Created bucket list:



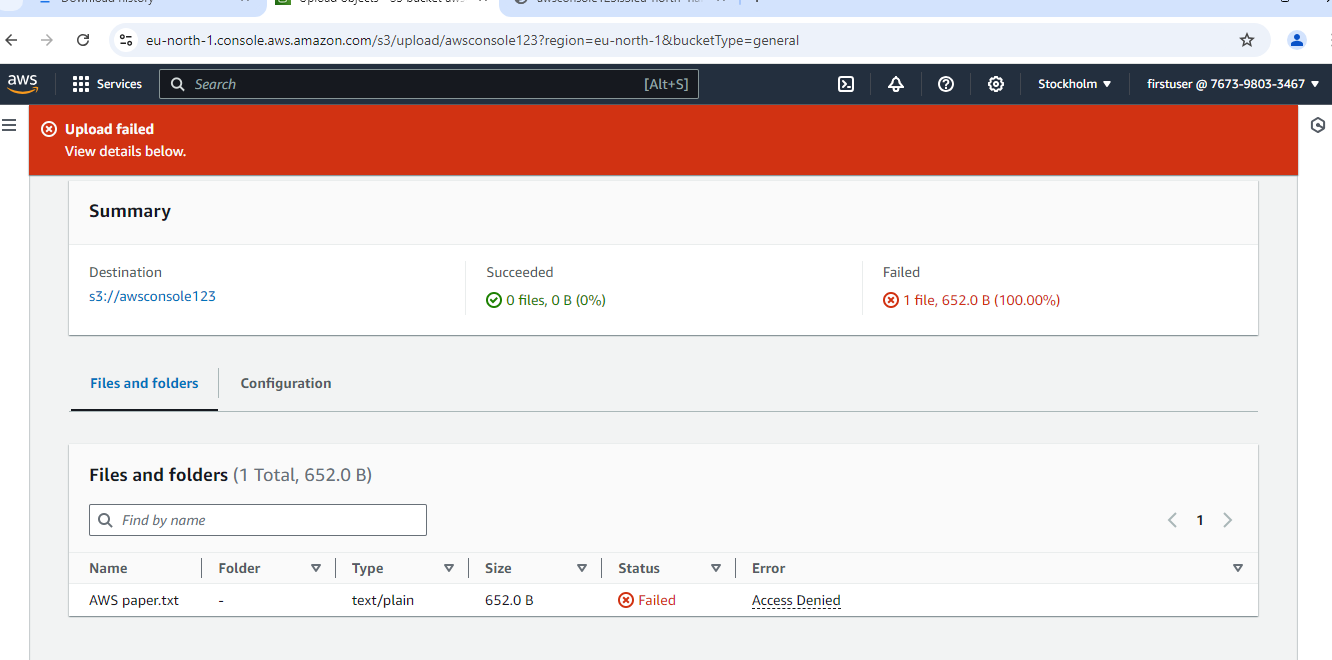
Step 6: Uploading files into S3 bucket:



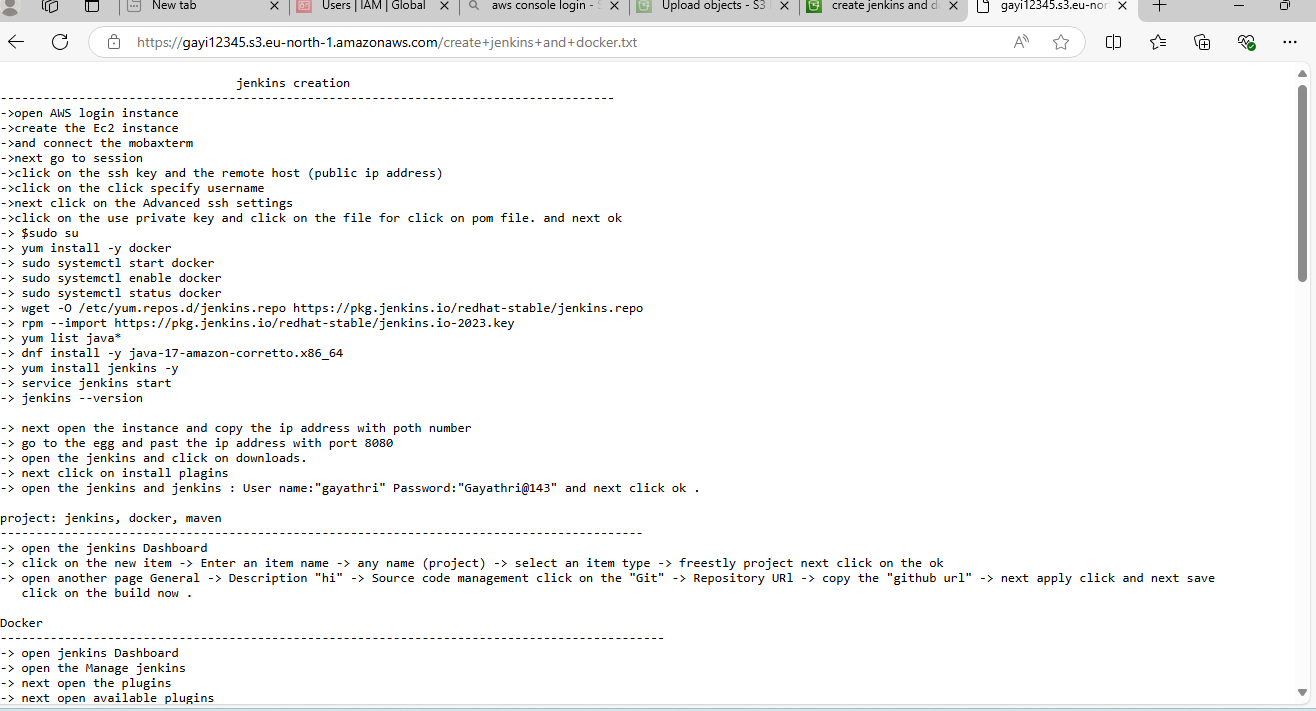
Step 7: Check the uploaded file: The file has been uploaded successfully in the bucket of the leaduser.



Step 8: uploading file in the other created users: here we have again uploaded some file into other user that is firstuser. But it has been failed to upload because we haven’t created KMS key to the firstuser of that particular bucket.



Step 9: Created a bucket without KMS key: here we have created other new bucket and uploaded some particular file in that bucket. And we can access that bucket file in public because we haven’t gave any KMS key to the particular bucket.



Conclusion: This process is referred to as key policy management or granting KMS permission.

**Summary:**

The process of **creating a KMS key and giving access to a specific user** involves:

* Creating the KMS key.
* Defining a **KMS Key Policy** to specify which IAM users or roles can use or manage the key.
* Optionally attaching an **IAM policy** to the user to define what actions they can perform on the KMS key.
* Optionally creating a **grant** to give temporary access to the key.