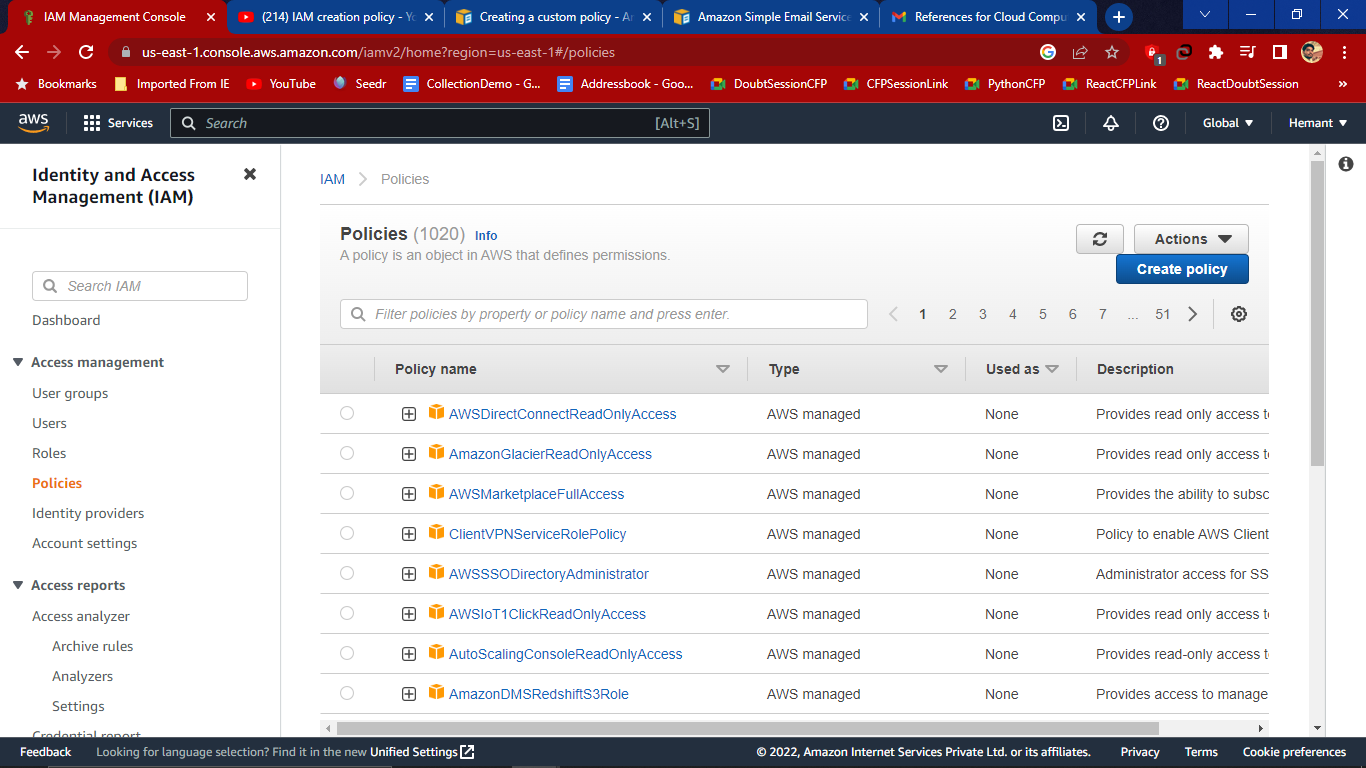
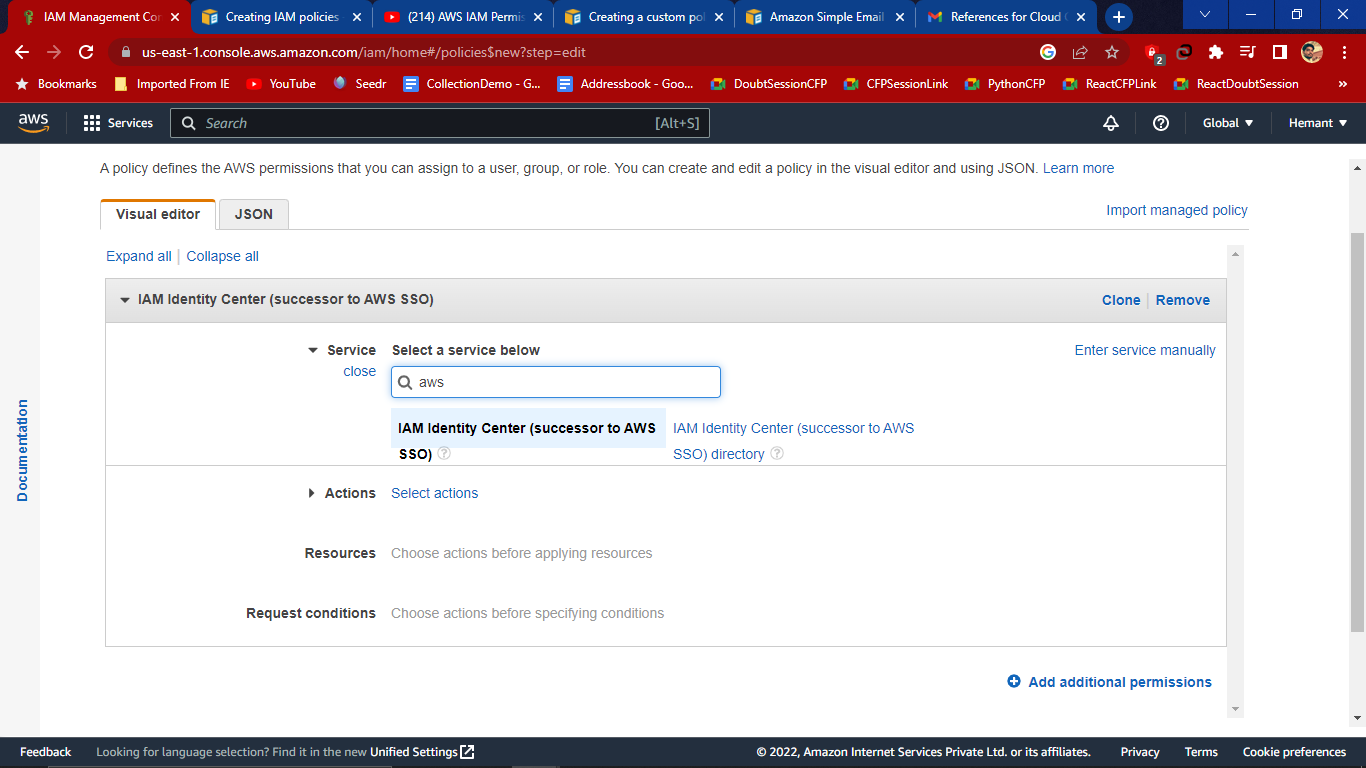
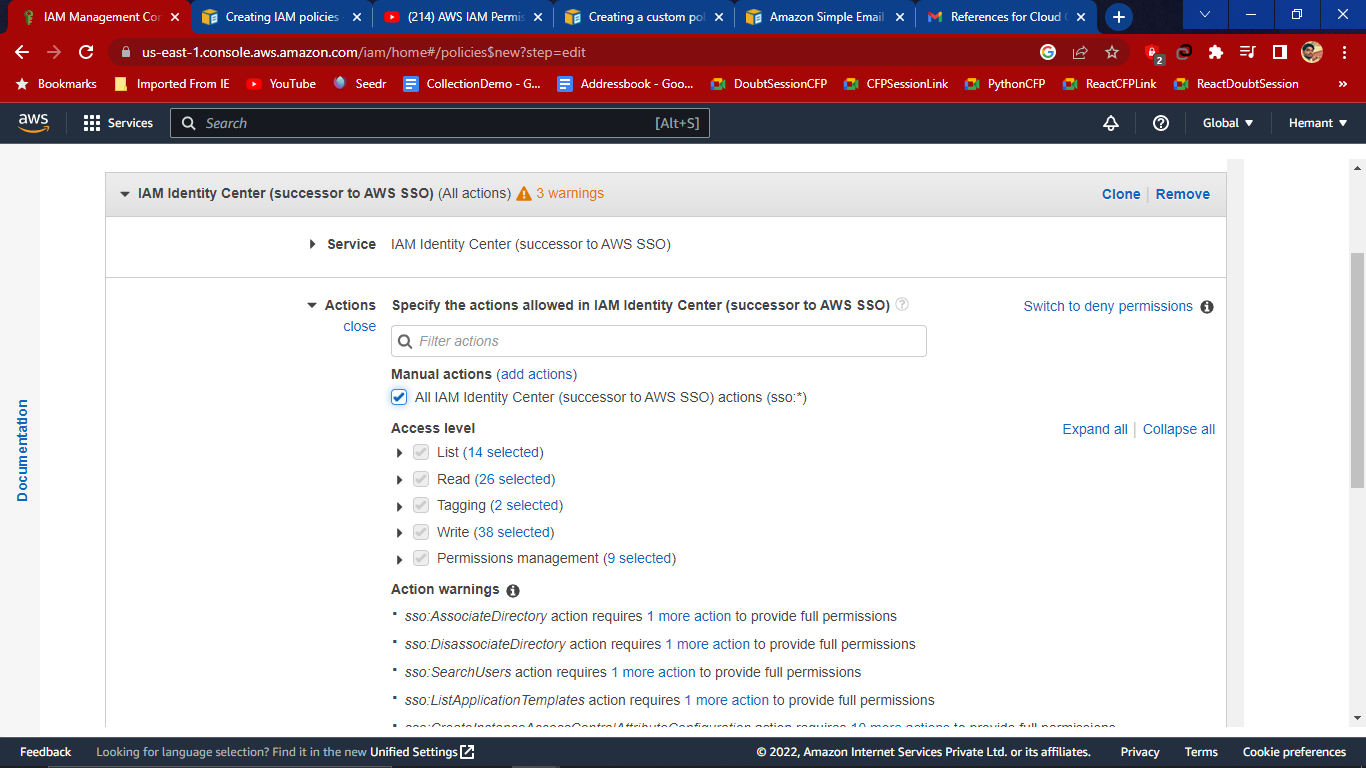
# Create a Custom Policy

**Policies:-**A policy is an object in AWS that defines permissions.

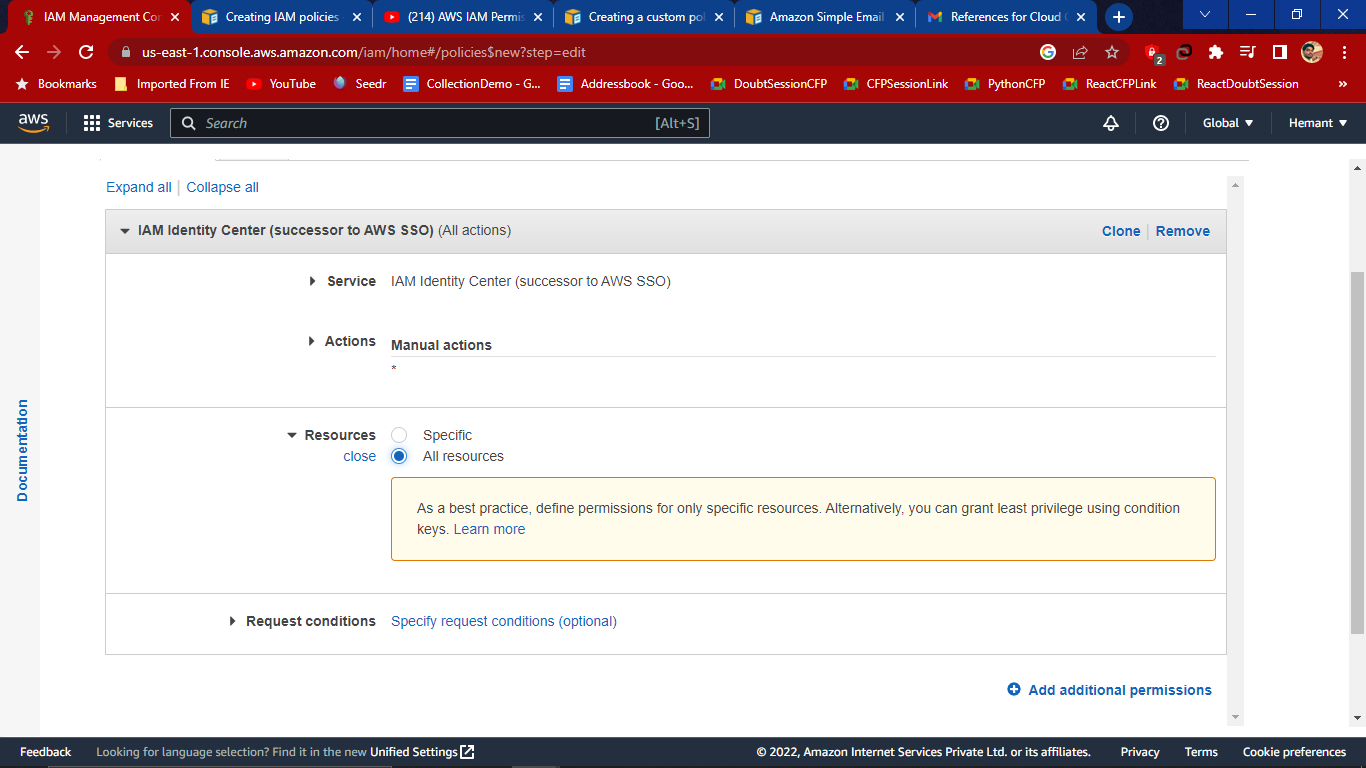
1. Create Policy by clicking on the button
2. On the Visual editor tab, Choose a service and then choose an AWS service.

You can choose only one service within a visual editor permission block. To grant access to more than one service, add multiple permission blocks by choosing Add additional permissions.



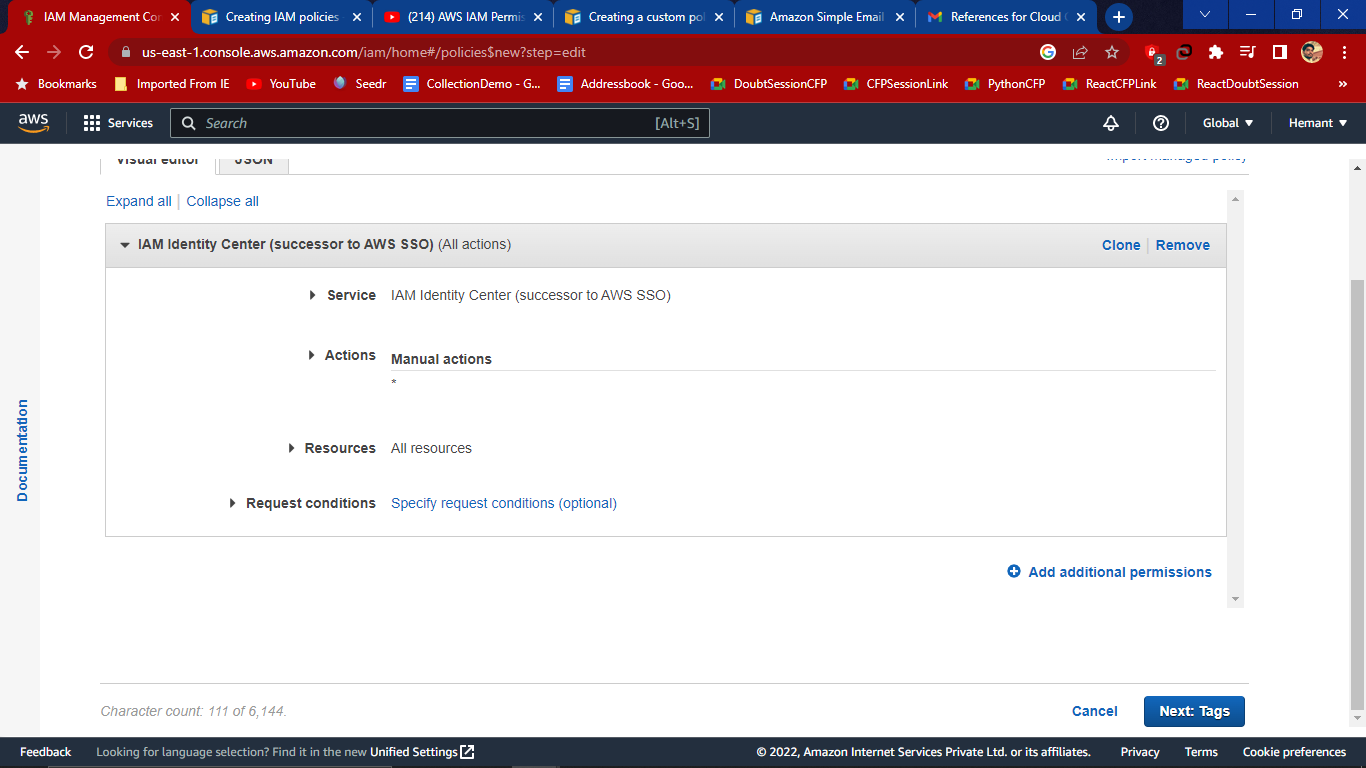
For Actions, choose the actions to add to the policy. I have chosen Manual Actions.

1. For Resources, if the service and actions that you selected in the previous steps do not support choosing specific resources, all resources are allowed and you cannot edit this section.

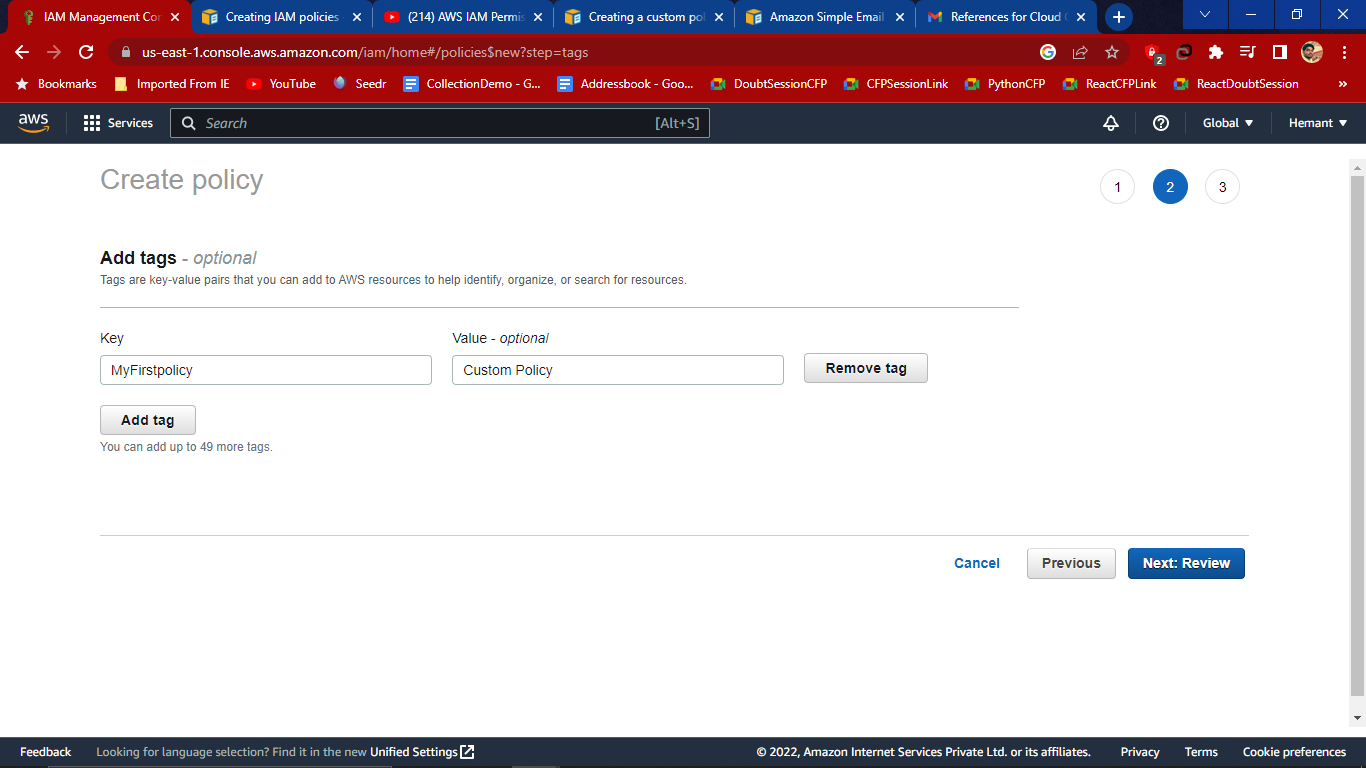


1. (Optional) Choose Specify request conditions (optional) to add conditions to the policy that you are creating.

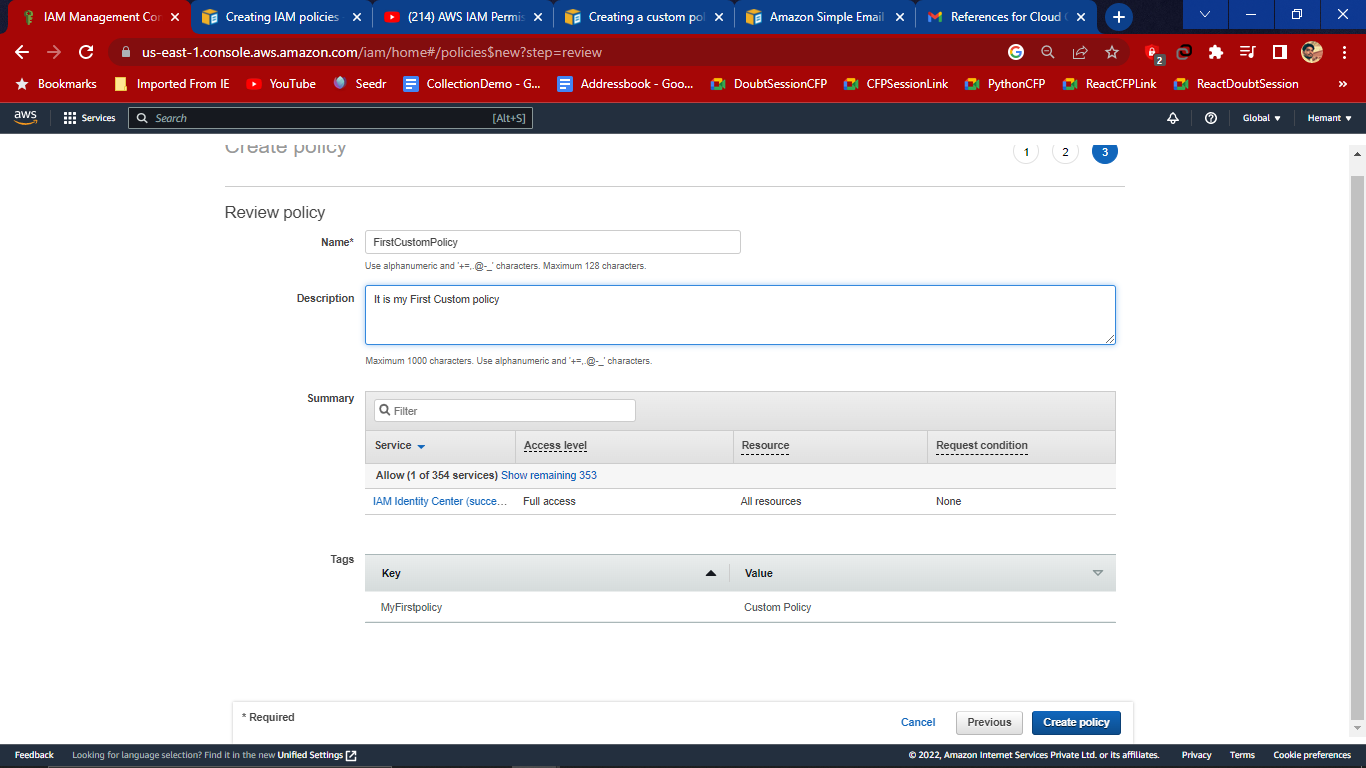
Then click on next Tags



1. Add Tags-Tags are key-value pairs that you can add to AWS resources to help identify, organise, or search for resources.

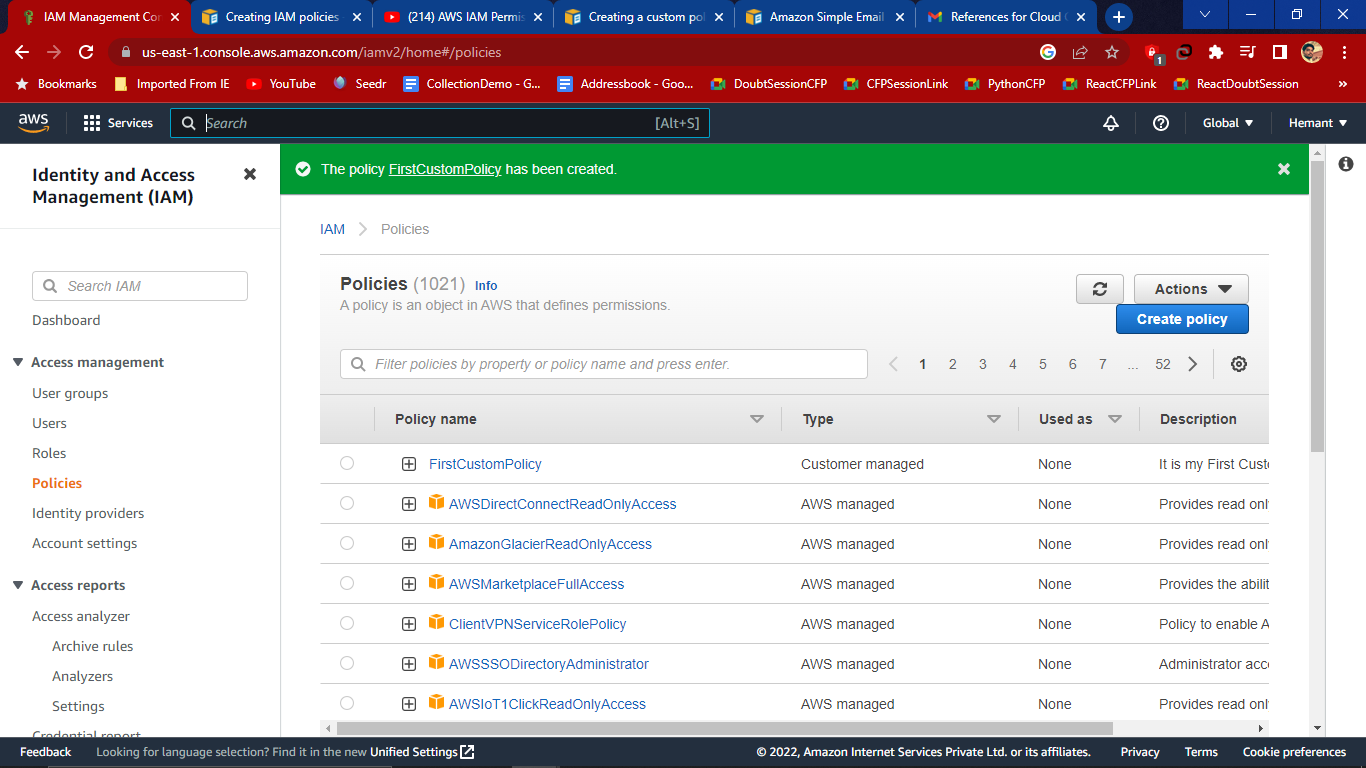


Then click on Next: Review

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

Then click on Create Policy

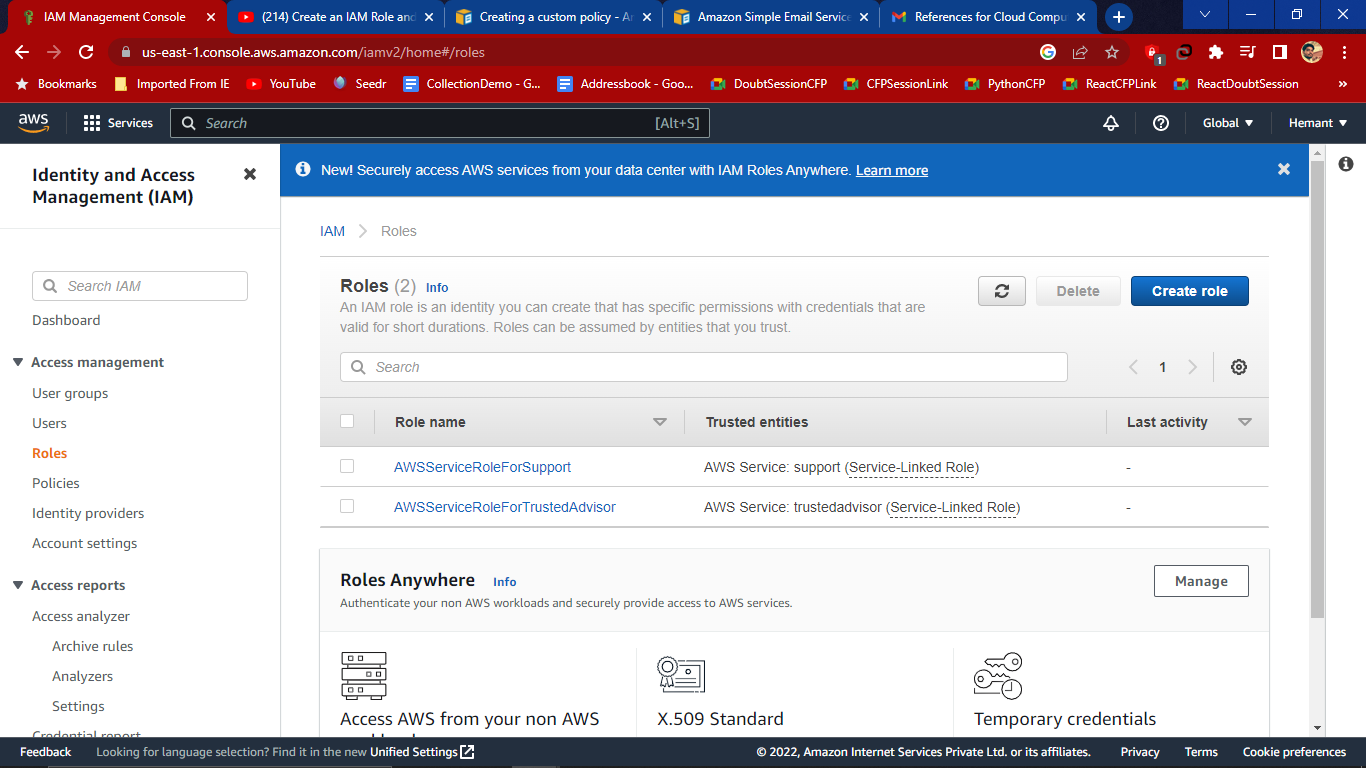
After creation of policy, you can attach it to your groups, users, or roles.



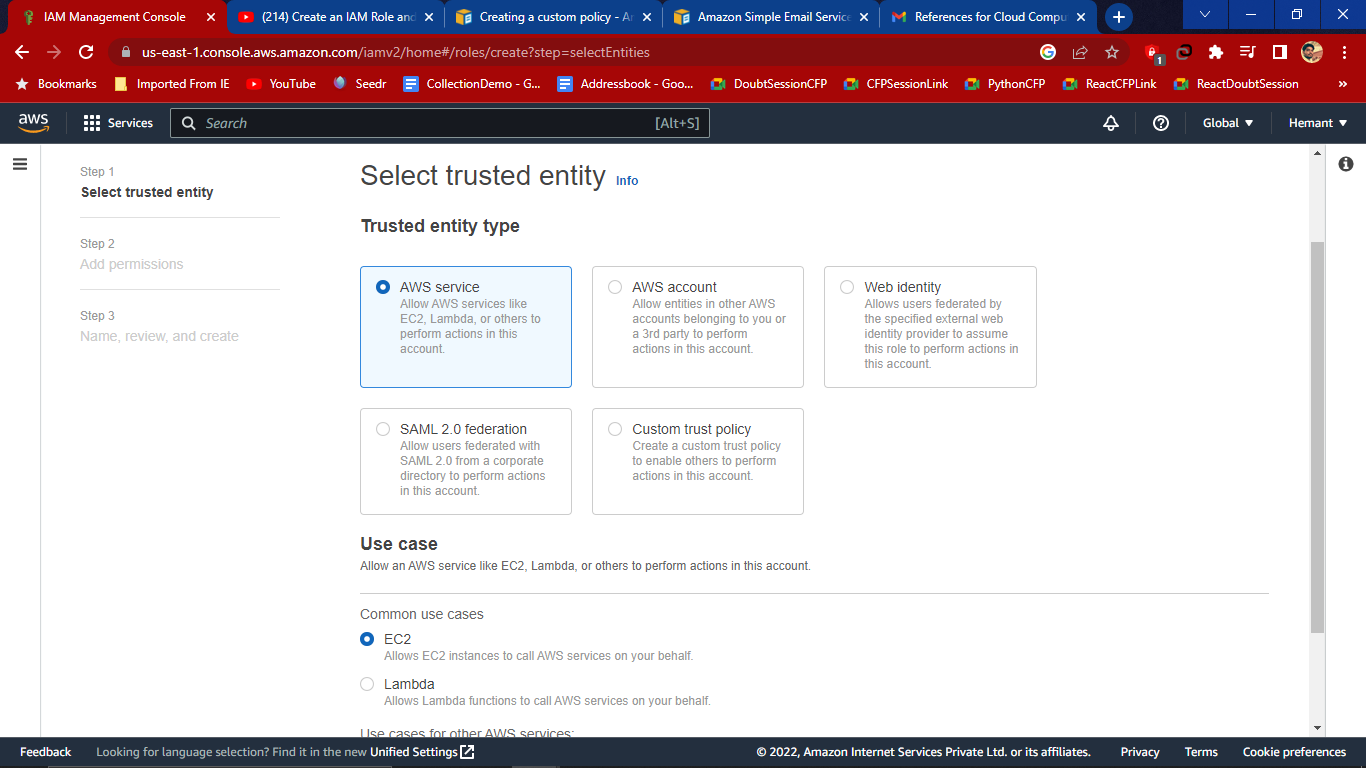
# Create a user Defined Roles

# **Roles**:-An IAM role is an identity you can create, that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

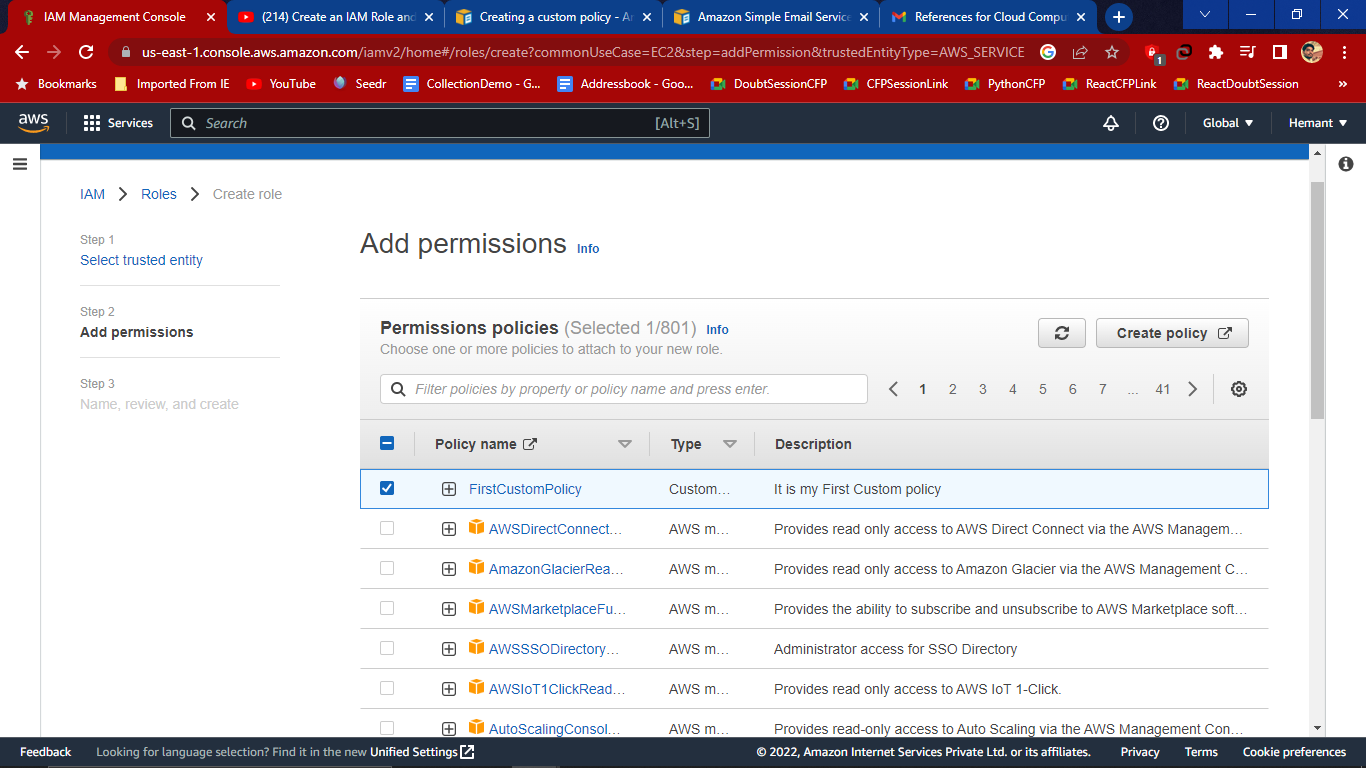
1. To create a role click on Create Role



1. Select Trusted entity and select AWS services which allows AWS services like EC2 and Lambda or other to perform actions in this account.

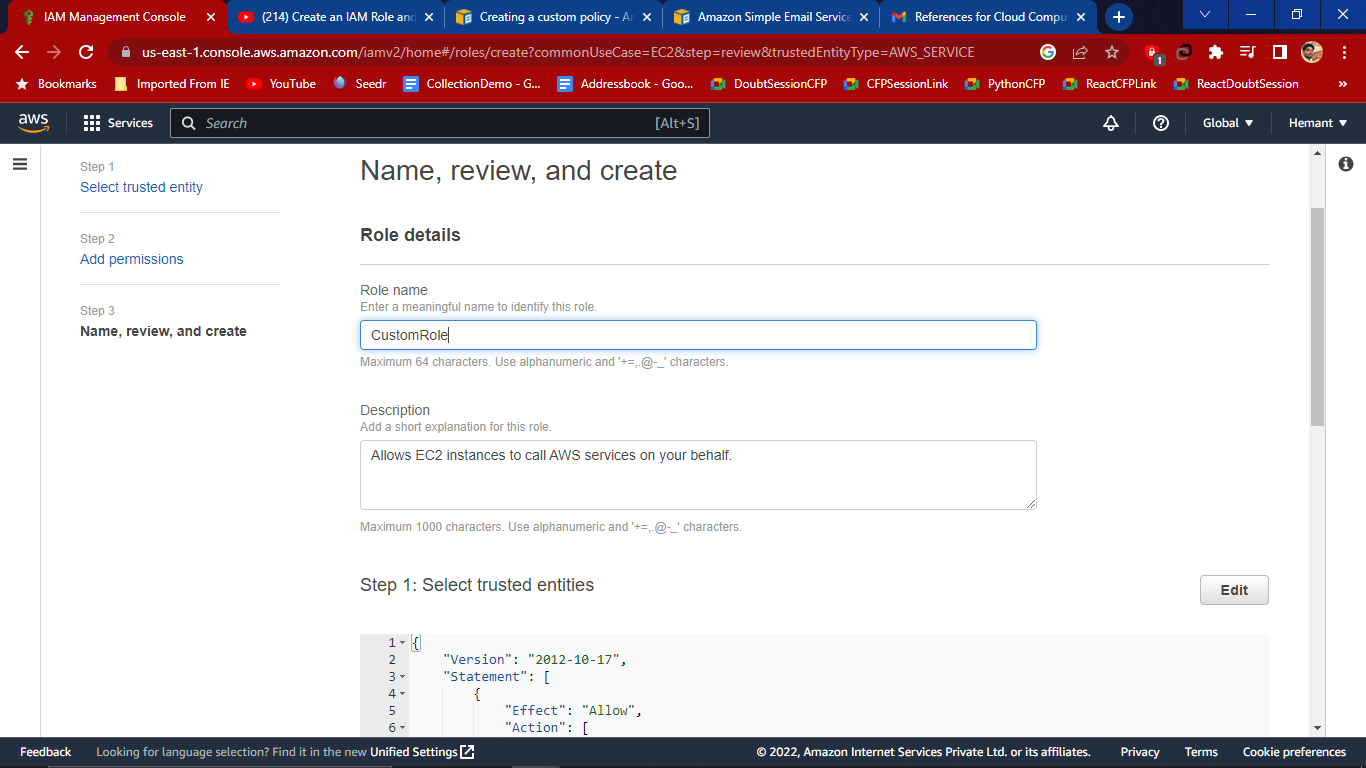
Then select Use Case as EC2.Then click on Next.

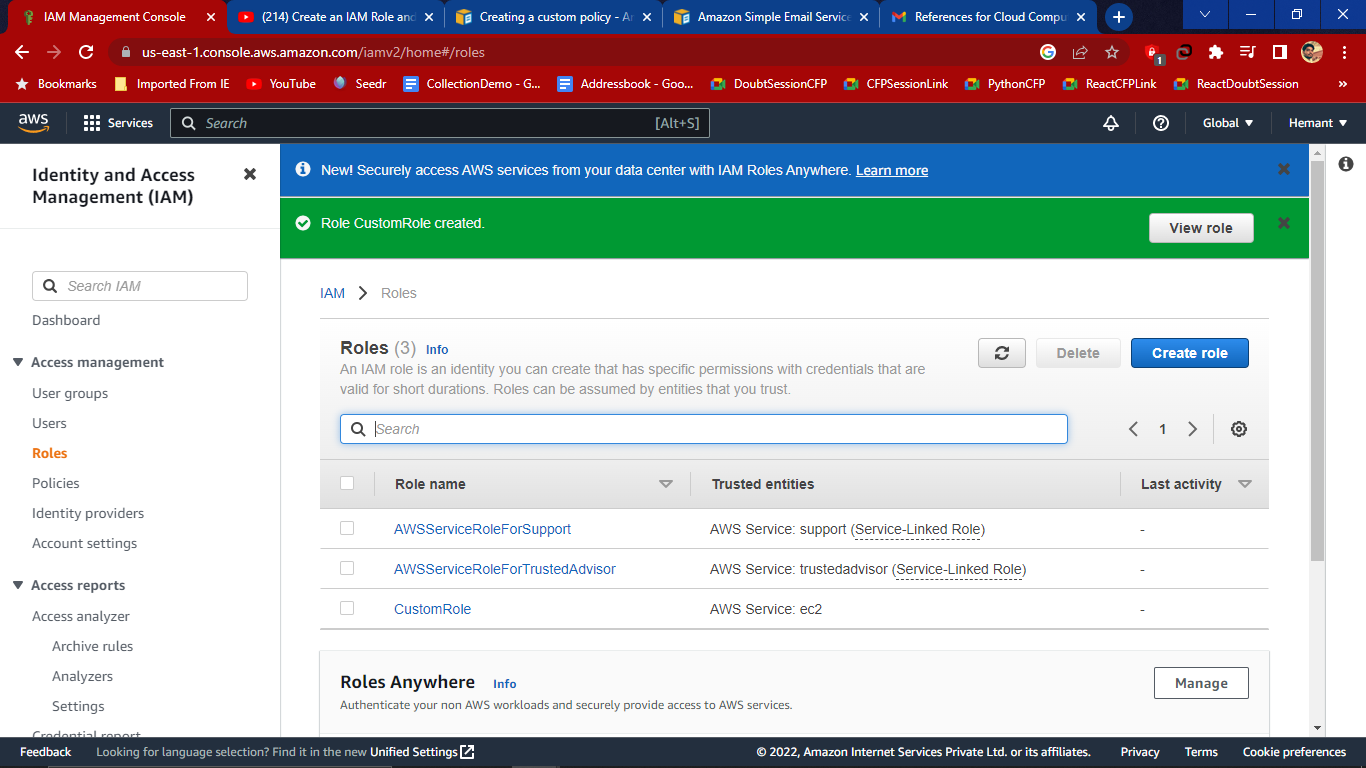
1. Add Permissions

Under the policy name select the policy that we have created (custom policy)

Then click on Next

1. Name, Review and create

Give Role name and description and the click on create role

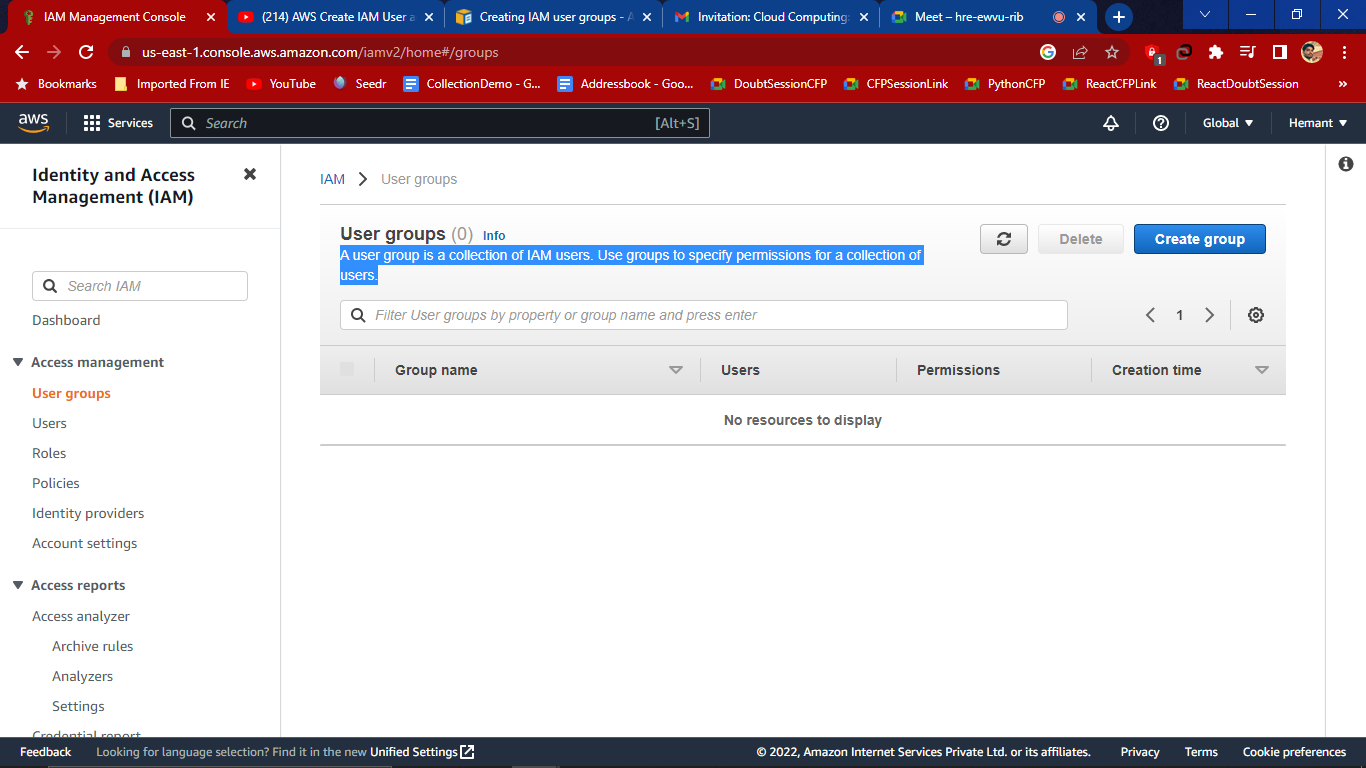
Role is created and appear in dashboard

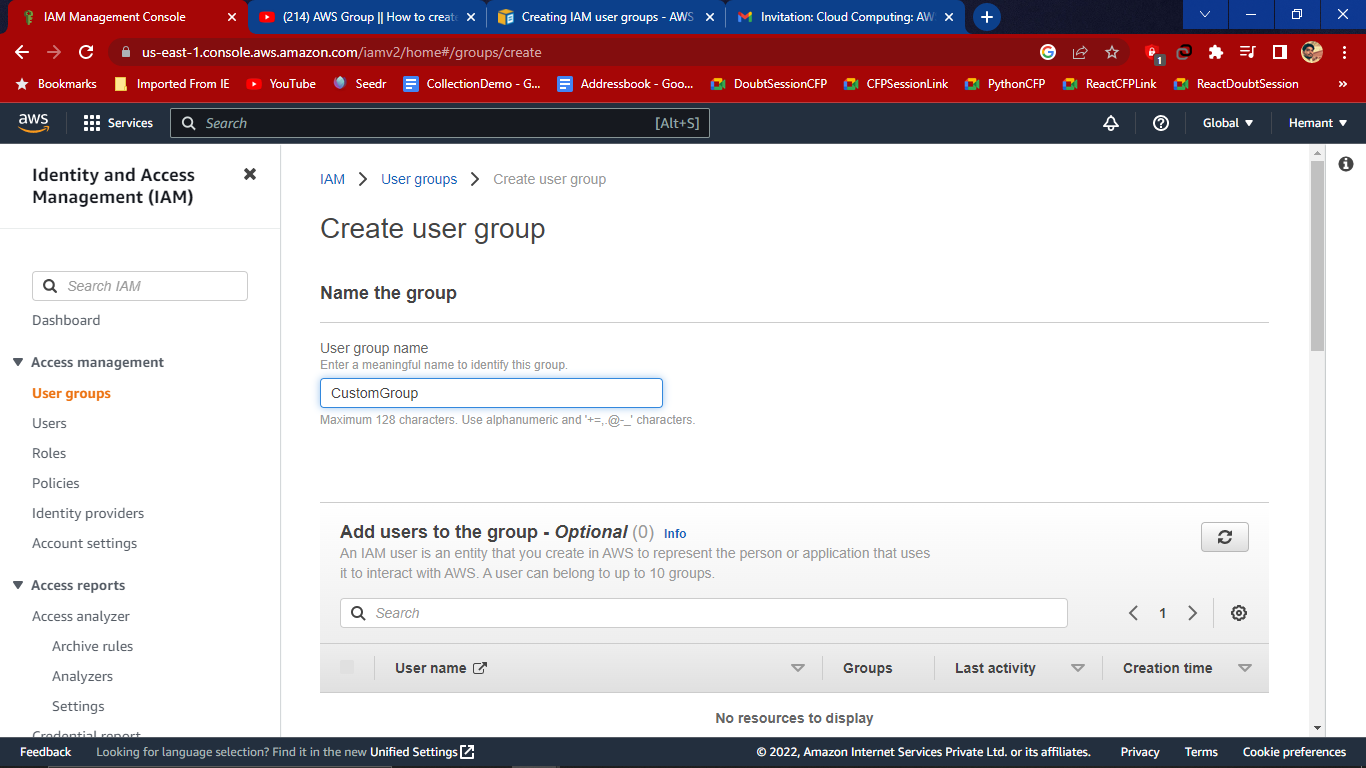
# Create a user group

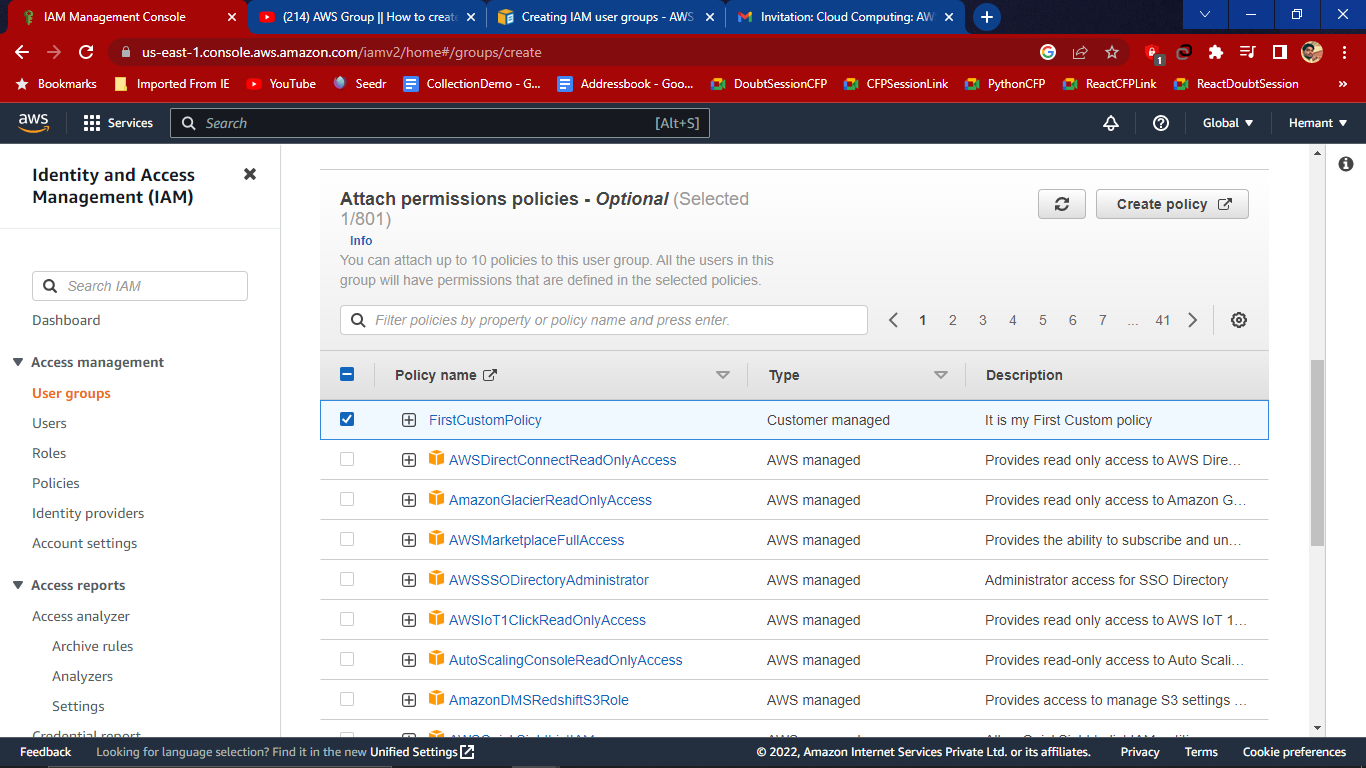
**User groups:-**A user group is a collection of IAM users. Use groups to specify permissions for a collection of users.

For that we need to login from the root user.

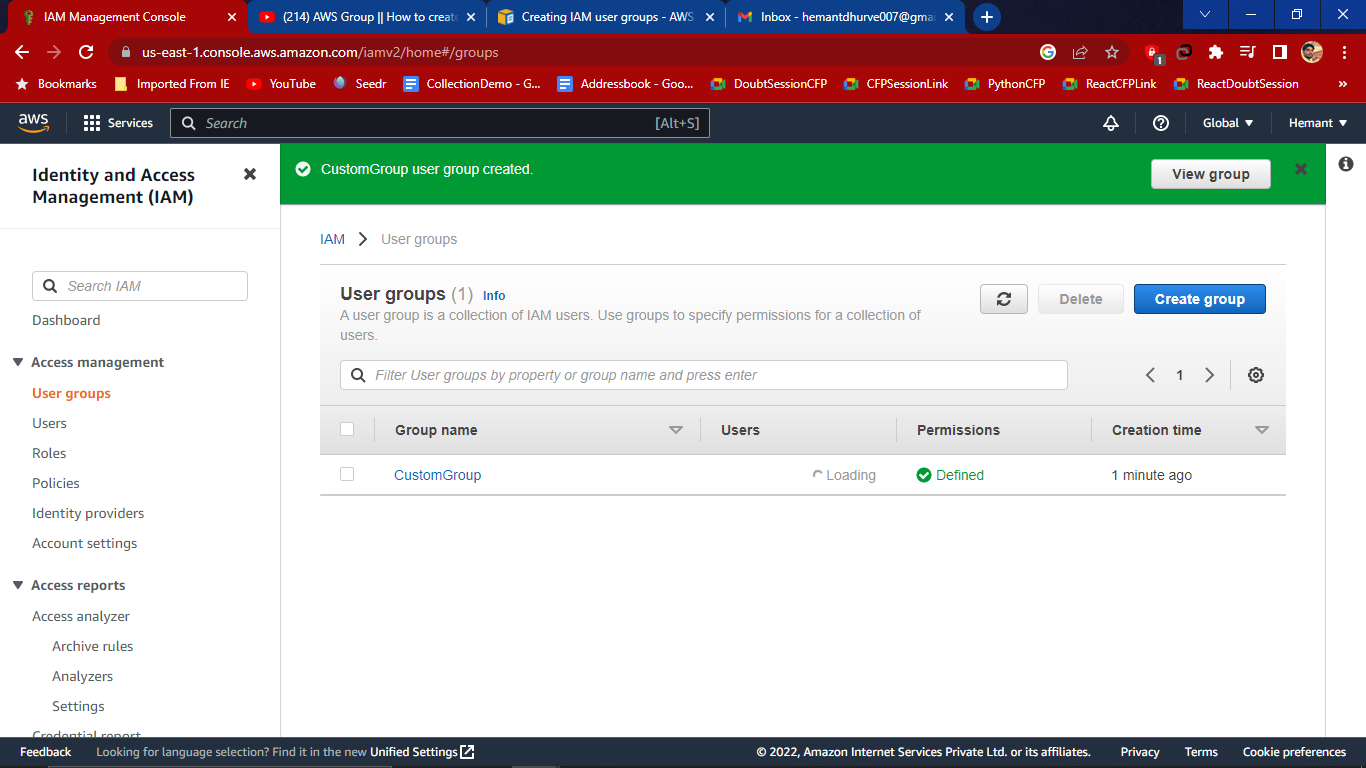
1. Create Group



1. Name the user group

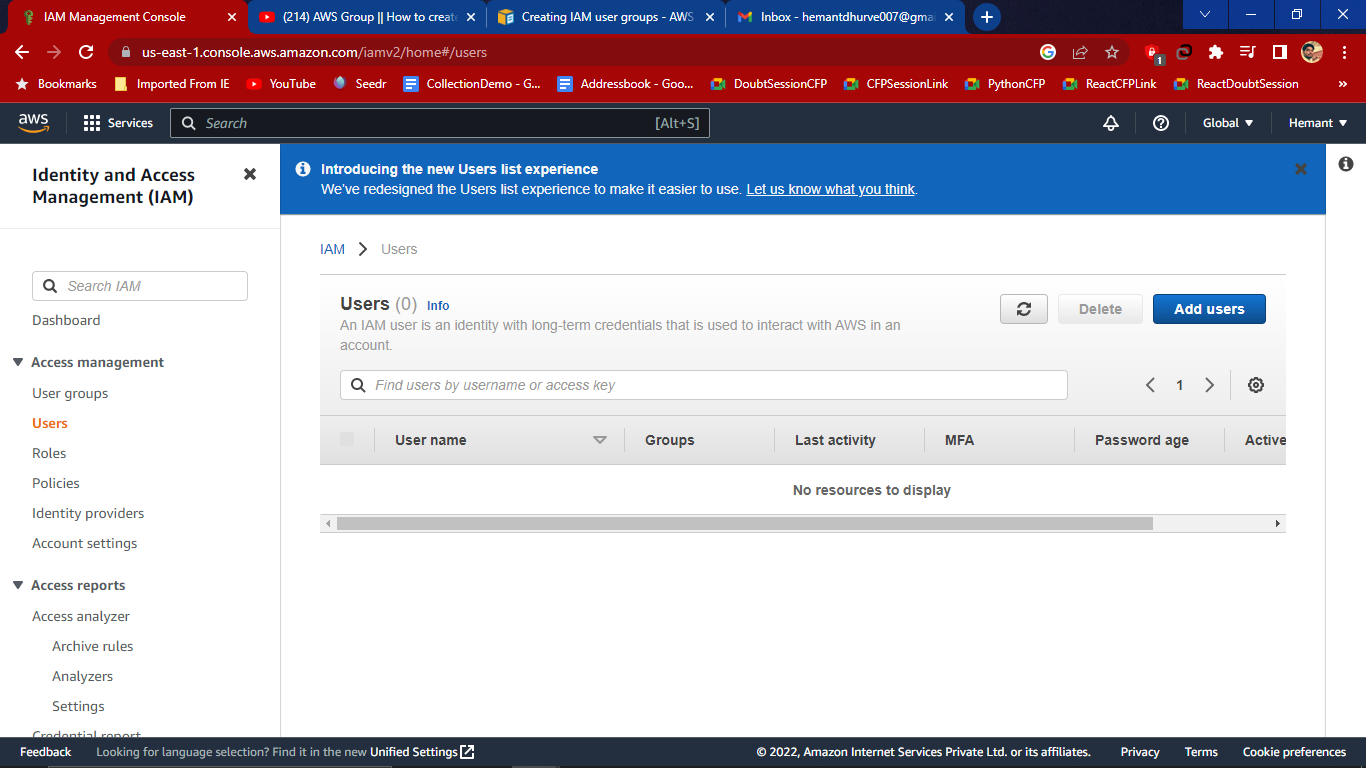
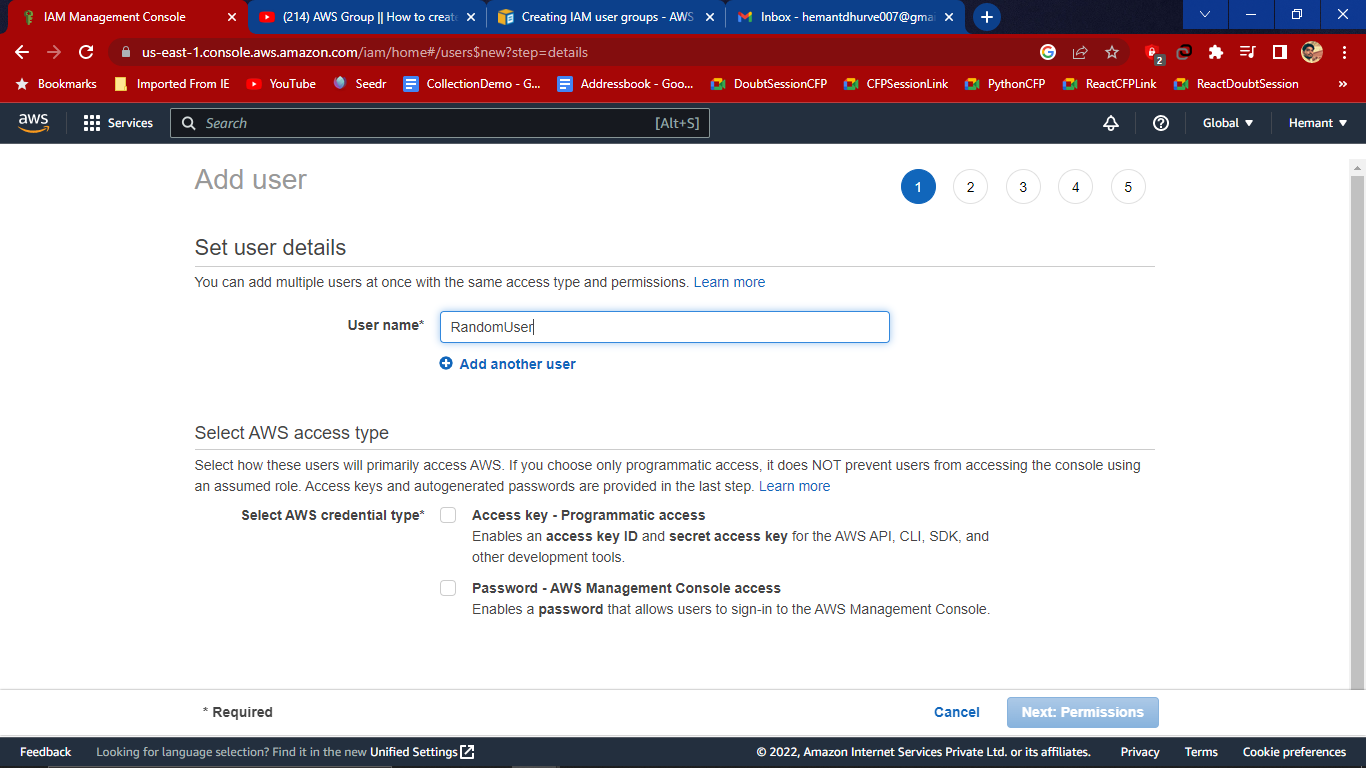
Attach Permission Policies and select custom policy that we have created and choose any if you like EC2

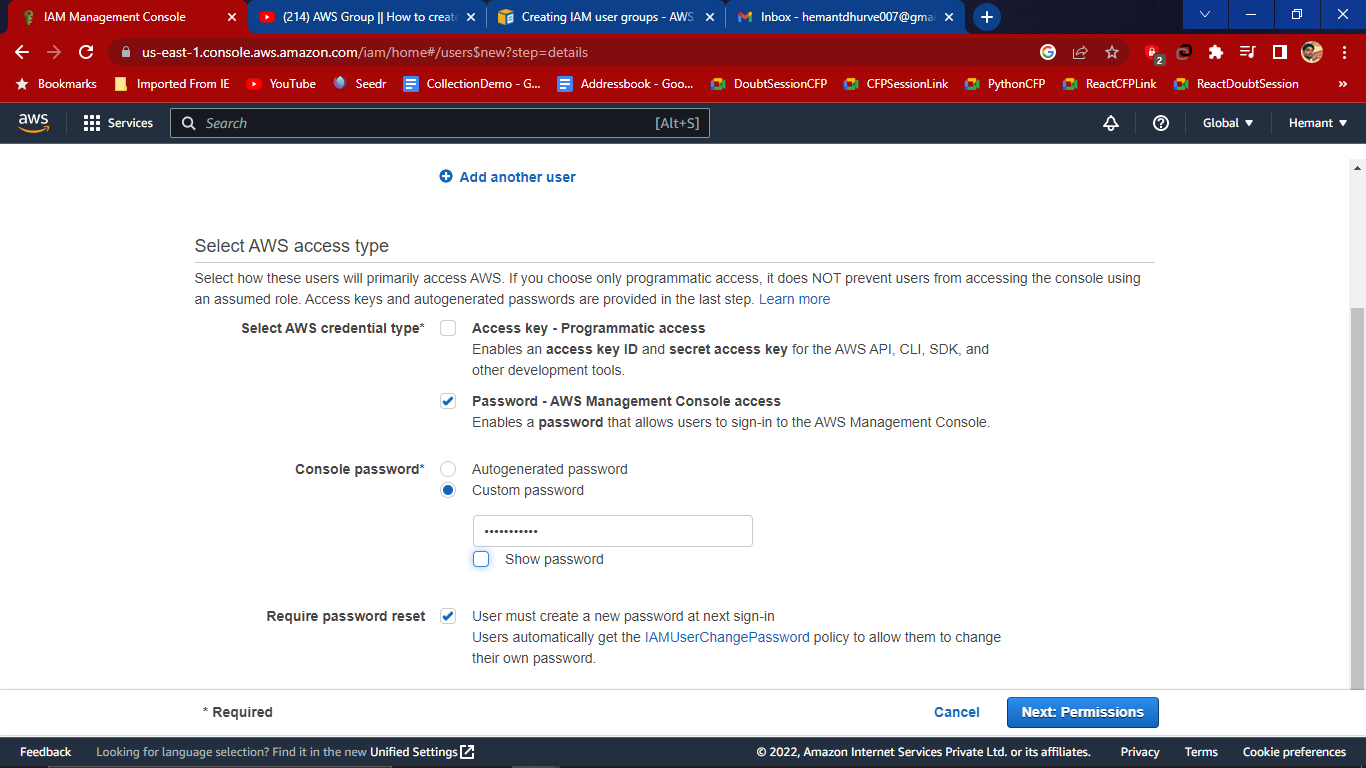
Then click on create Group

1. User group created 

# Create a user in User-Defined Group

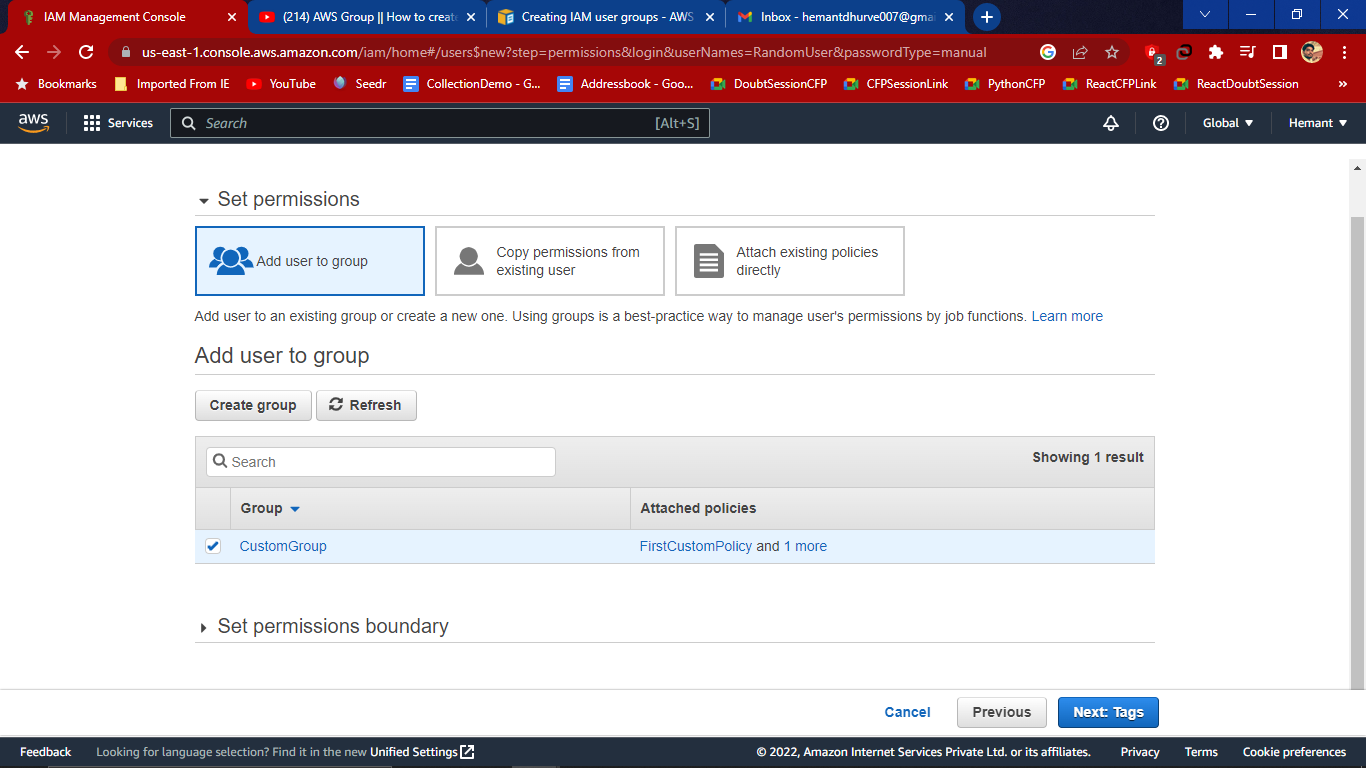
**Users:-**An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

1. We have created a group and now we want to add user in that group
2. Set user Details
3. Select AWS access type
4. Select credential type:- Password-AWS management Console Access
5. Console Password:-



1. Then click on Next Permission

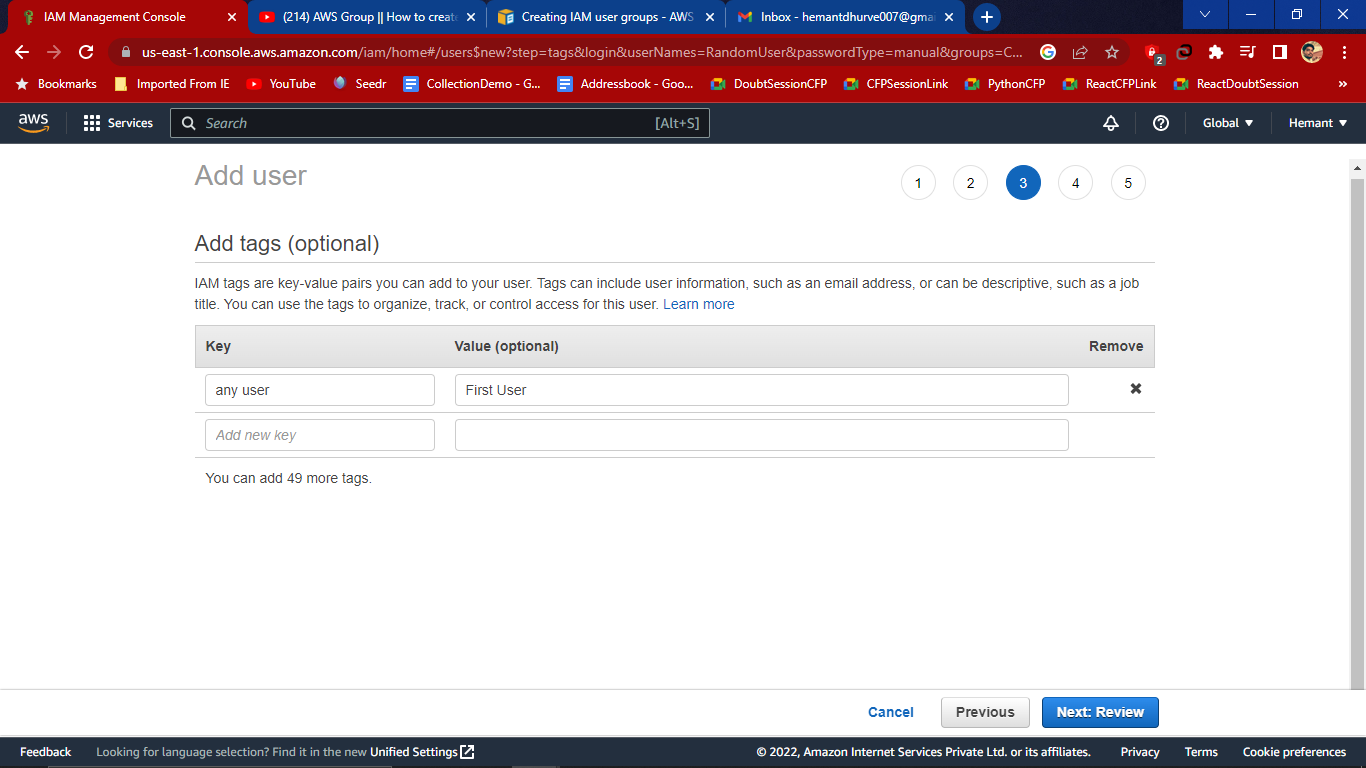
4. Set Permission

Select Add user to group and add to custom group with their policies given.

Then click on Next Tags

5. Add Tags

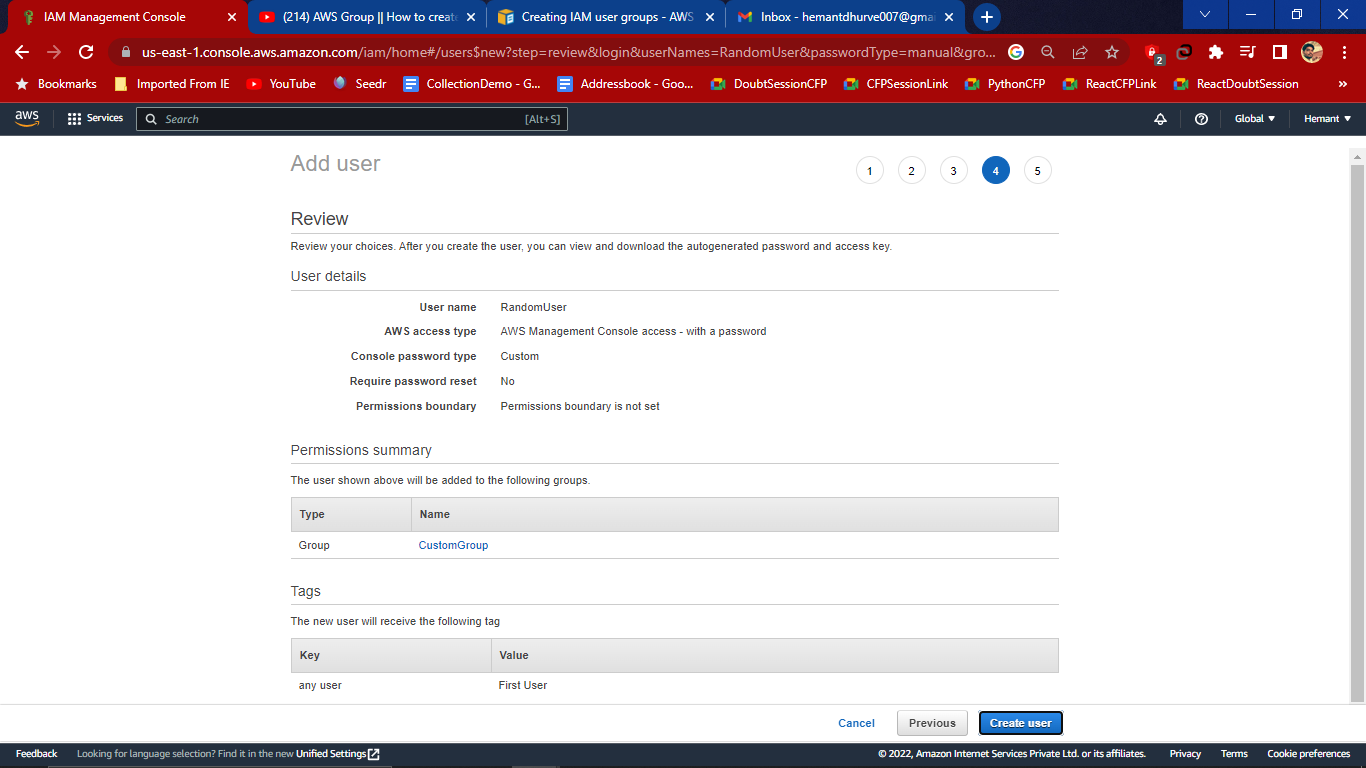
IAM tags are key-value pairs you can add to your user.



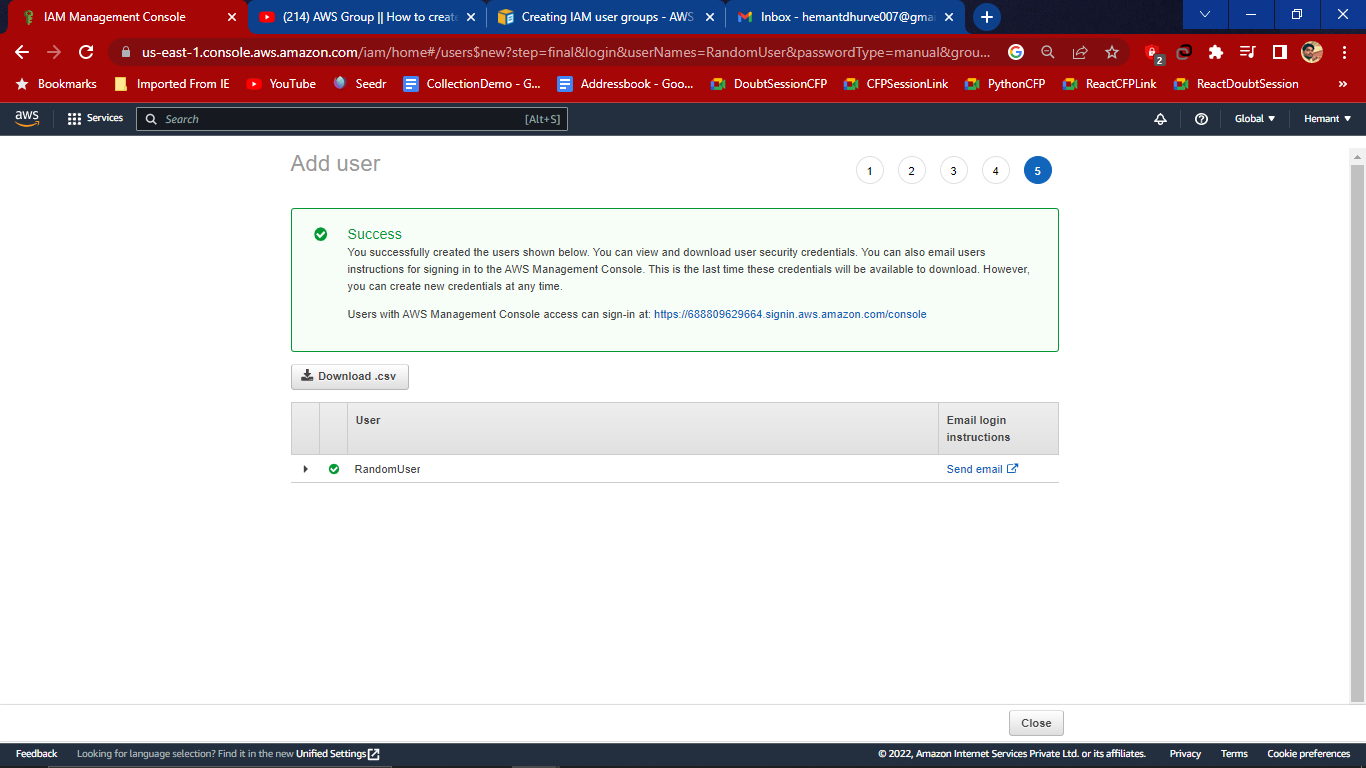
Then click on Next Review

6. Review Page

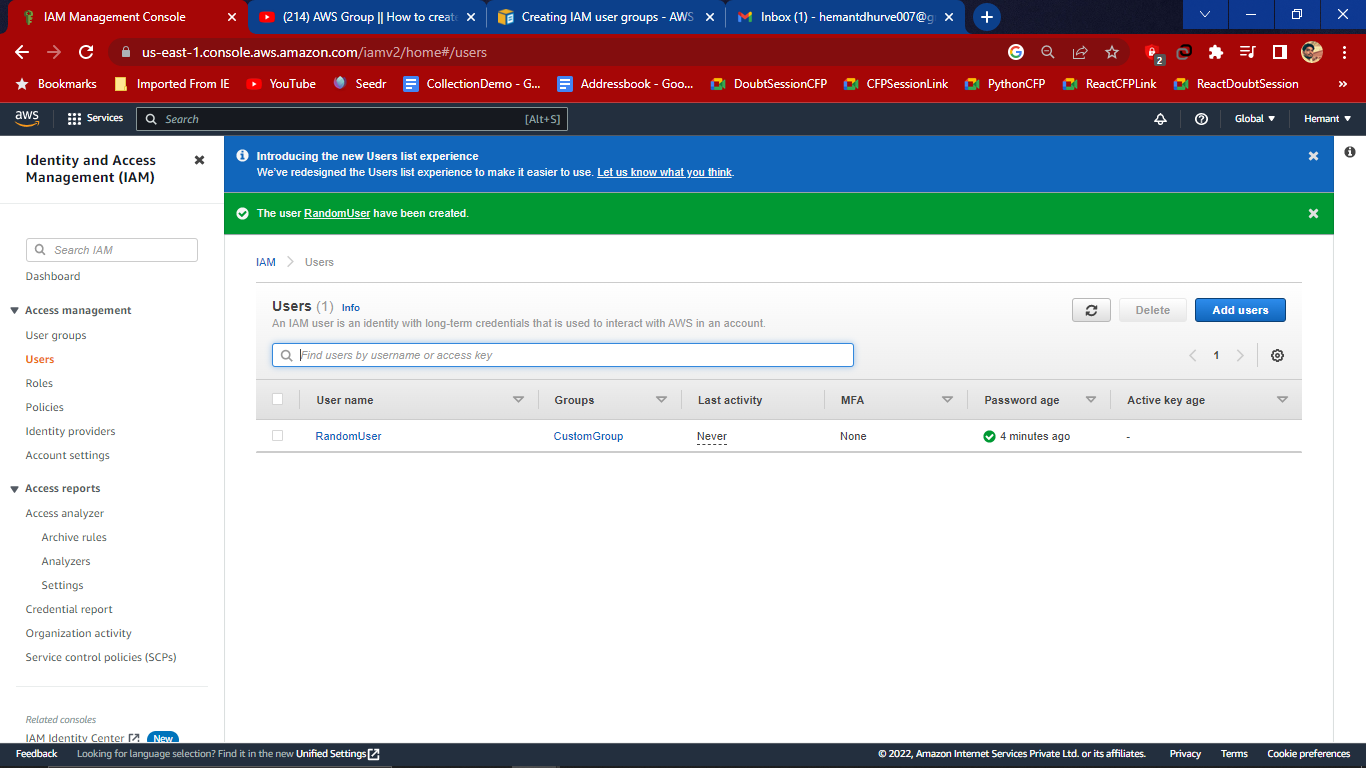
Review your choices. After you create the user, you can view and download the auto generated password and access key.

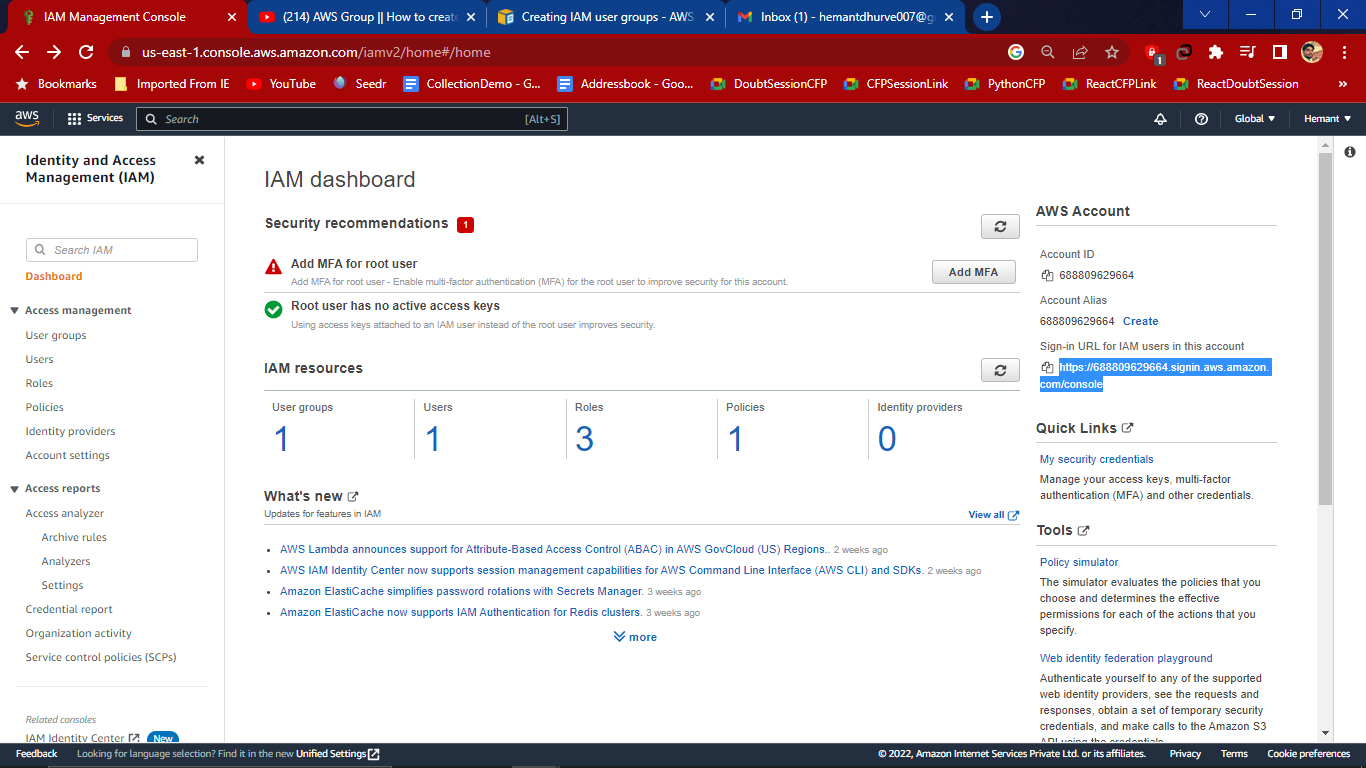


If everything is fine then click on create User

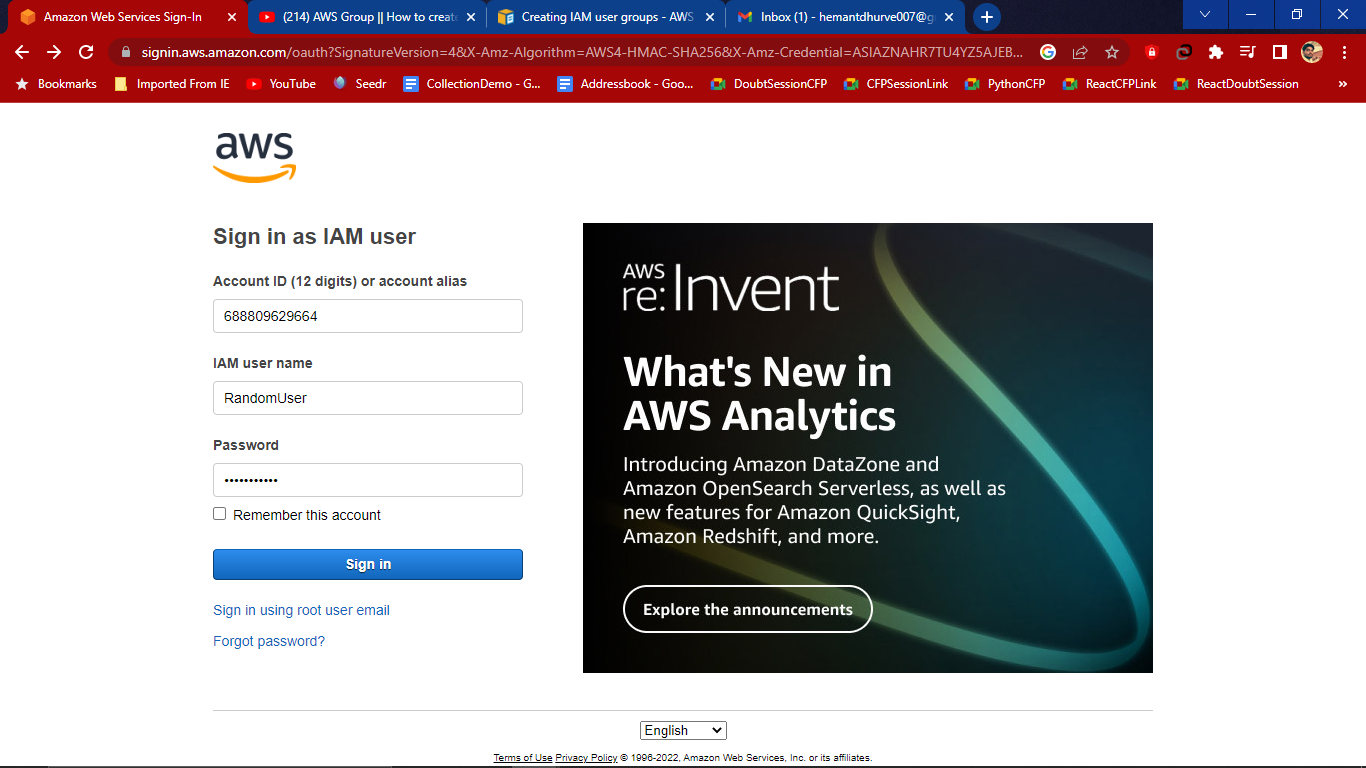


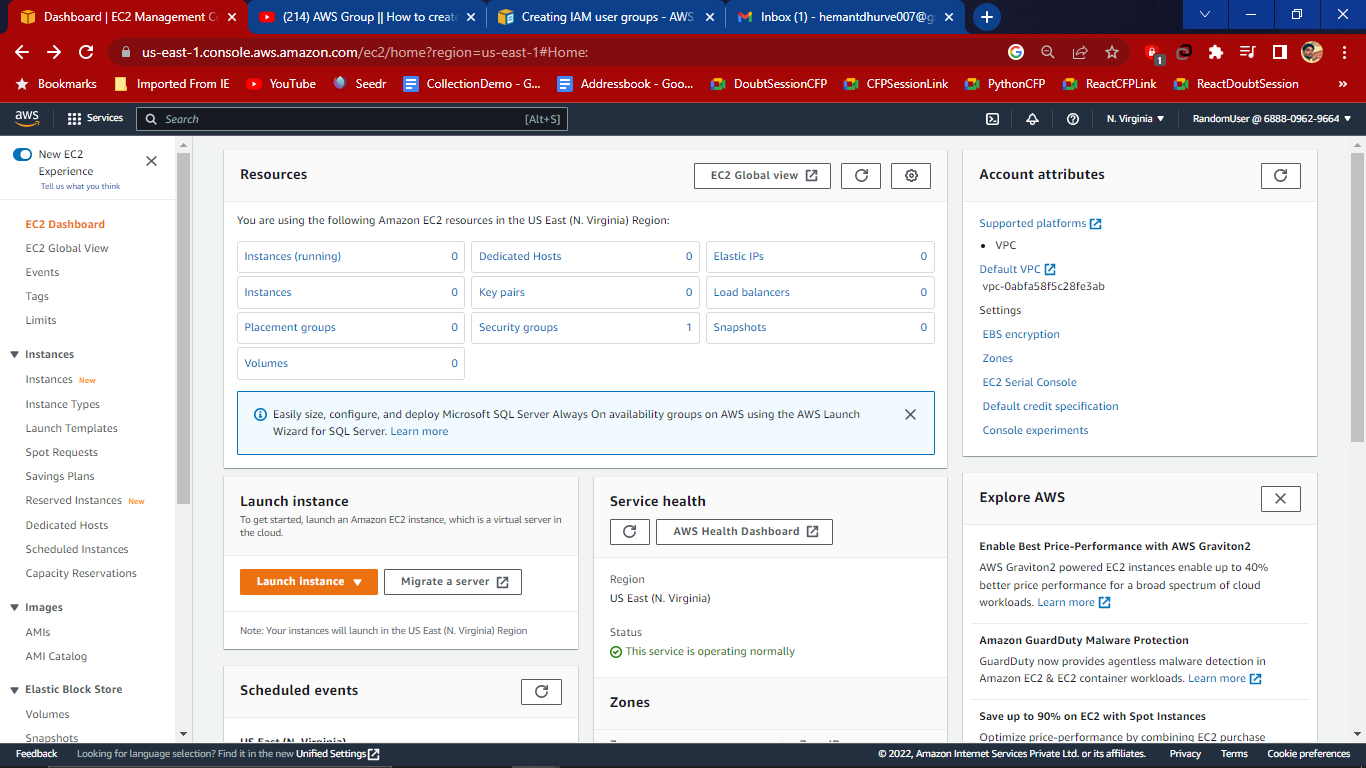
And download csv file and to login from this user it has generated console sign-in

Now the user is added to the group

7. Now we have to login from the user so goto the dashboard and copy the console link and login 

Enter credentials in the sign-in page



Select the EC2 service which we have given permissions 

# **What are 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.

The following policy types, listed in order of frequency, can affect whether a request is authorised.

1. **Identity-based policies** – You can attach managed and inline policies to IAM identities (users, groups to which users belong, and roles).
2. **Resource-based policies** – You can attach inline policies to resources in some AWS services. The most common examples of resource-based policies are Amazon S3 bucket policies and IAM role trust policies. AWS RoboMaker does not support resource-based policies.
3. **Organisations SCPs** – You can use an AWS Organizations service control policy (SCP) to apply a permissions boundary to an AWS Organizations organisation or organisational unit (OU). Those permissions are applied to all entities within the member accounts.
4. **Access control lists (ACLs)** – You can use ACLs to control what principals can access a resource. ACLs are similar to resource-based policies, although they are the only policy type that does not use the JSON policy document structure. AWS RoboMaker does not support ACLs. difference between role and policies?

# **What is the difference between role and policies?**

| **IAM Roles** | **IAM Policies** |
| --- | --- |
| An IAM role is very similar to a user, in that it is an identity with permission policies that determine what the identity can and cannot do in AWS. | IAM Policies determine what actions a user, role, or member of a user group can perform, on which AWS resources, and under what conditions. |
| An IAM Role with no IAM Policy attached to it, won’t have to access any AWS resources | An IAM Policy that is not attached to an IAM Role is of no use. |
| IAM Roles manage who has access to your AWS resources | IAM policies control their permissions. |

**How do we create a User Group?**

1. Sign in to the AWS Management Console and open the IAM console.
2. In the navigation pane, choose User groups and then choose Create group.
3. For the User group name, type the name of the group.
4. In the list of users, select the check box for each user that you want to add to the group.
5. In the list of policies, select the check box for each policy that you want to apply to all members of the group.
6. Choose Create group.

**How many policies and users can you add in 1 User Group?**

You can attach up to 10 managed policies to each group, for a maximum of 120 policies (20 managed policies attached to the IAM user, 10 IAM groups, with 10 policies each).

One IAM user can be a part of a maximum of 5 groups. Within the IAM service a **GROUP** is regarded as a: **A collection of AWS accounts**. It's the group of EC2 machines that gain the permissions specified in the GROUP.

# **What are Identity Providers?**

IAM identity providers (IdPs) **allow** you to **manage** your **identities outside of AWS**. When you use an external IdP, you must identify that IdP in IAM by creating an IAM identity provider resource. To allow users from your IdP to access AWS, create a role and then trust the IAM identity provider. Then your users can assume the role to get access to the AWS resources in your account.

# **Why are they used?**

An identity provider (IdP) is a system that **creates, stores, and manages digital identities**. The IdP can either directly **authenticate the user** or can provide **authentication services** to **third-party service providers** (apps, websites, or other digital services). Simply put, an IdP offers user **authentication as-a-service**.

# **What is the difference between access key and password while creating an IAM user?**

1. **Application users** should use only **access keys** to **programmatically access data** in AWS cloud and
2. **Administrators** who need **console access** should use **only passwords** to manage AWS resources.

**Programmatic Access**: In this, you will be given an access key ID and secret access key which will be provided when the IAM user is created by the root user. An IAM user signs in using Command Line Interface(CLI) with the provided access key ID and secret access key.

**AWS Management Console access**: It is **easy-to-access** through a **web-based portal**. Here you will be given an **account ID, name, and password** which will be provided by the root user while creating the IAM user sign-in using the name, account ID, and password to the AWS management console.

# **What is MFA?**

An **authentication system** that requires **more than one distinct authentication factor** for **successful authentication**. **Multi factor authentication** can be performed using a multi factor authentication or by a combination of authenticators that provide different factors.

Multi-factor authentication (MFA) is a **multi-step account login process** that **requires** users to enter **more information** than just a **password.**

For example, along with the password, users might be asked to enter a code sent to their email, answer a secret question, or scan a fingerprint. A second form of authentication can help prevent unauthorised account access if a system password has been compromised.

# **Benefits of Multi-Factor Authentication?**

### **Reduces security risk**

Multi-factor authentication minimises risks due to human error, misplaced passwords, and lost devices.

**Enables digital initiatives**

Organisations can undertake digital initiatives with confidence. Businesses use multi-factor authentication to help protect organisational and user data so that they can carry out online interactions and transactions securely.

**Improves security response**

Companies can configure a multi-factor authentication system to actively send an alert whenever it detects suspicious login attempts. This helps both companies and individuals to respond faster to cyberattacks, which minimises any potential damage.