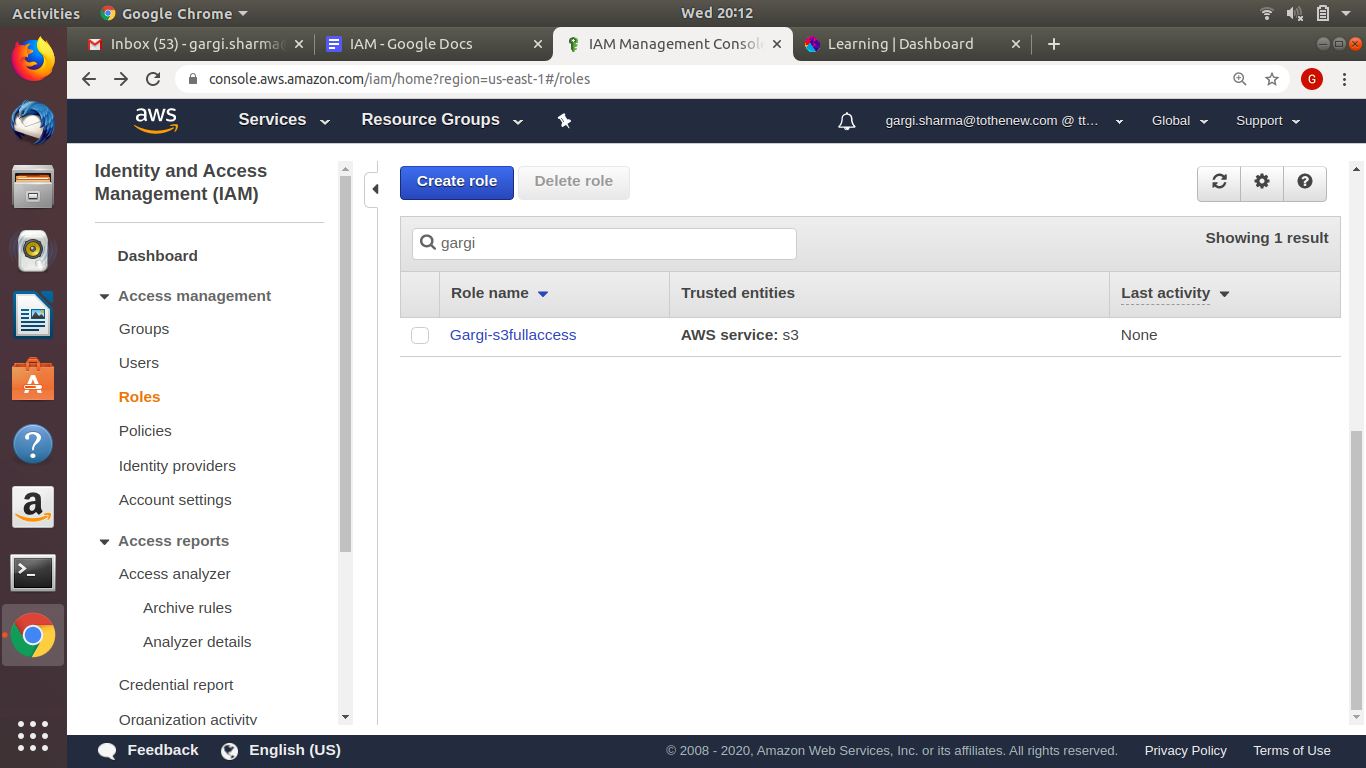
**Assessment 11 – IAM**

**Trainee Name : Gargi Sharma**

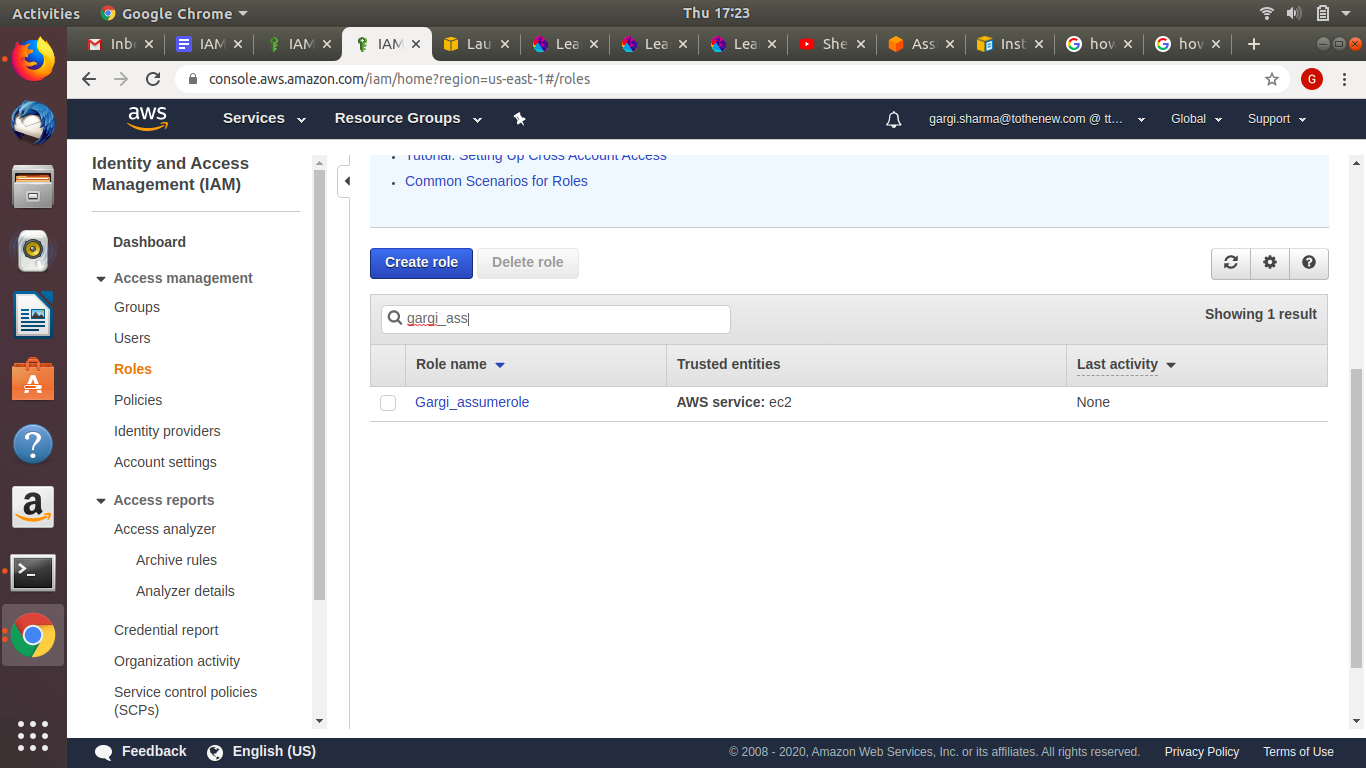
**Mentor Name : Mr. Ravi Kumar**

**College Name : UPES**

1. **Create a Role with full access to S3**



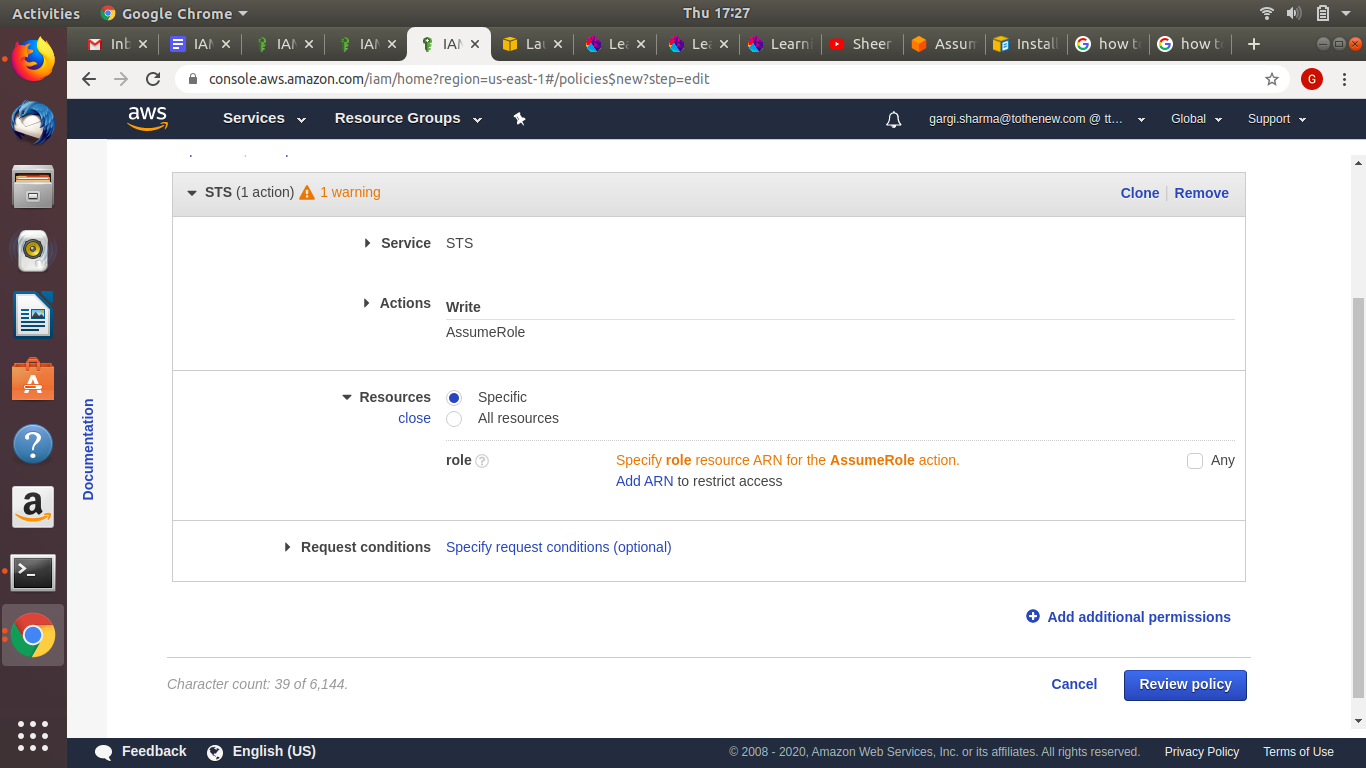
1. **Create another which has the policy to assume the previous Role.**

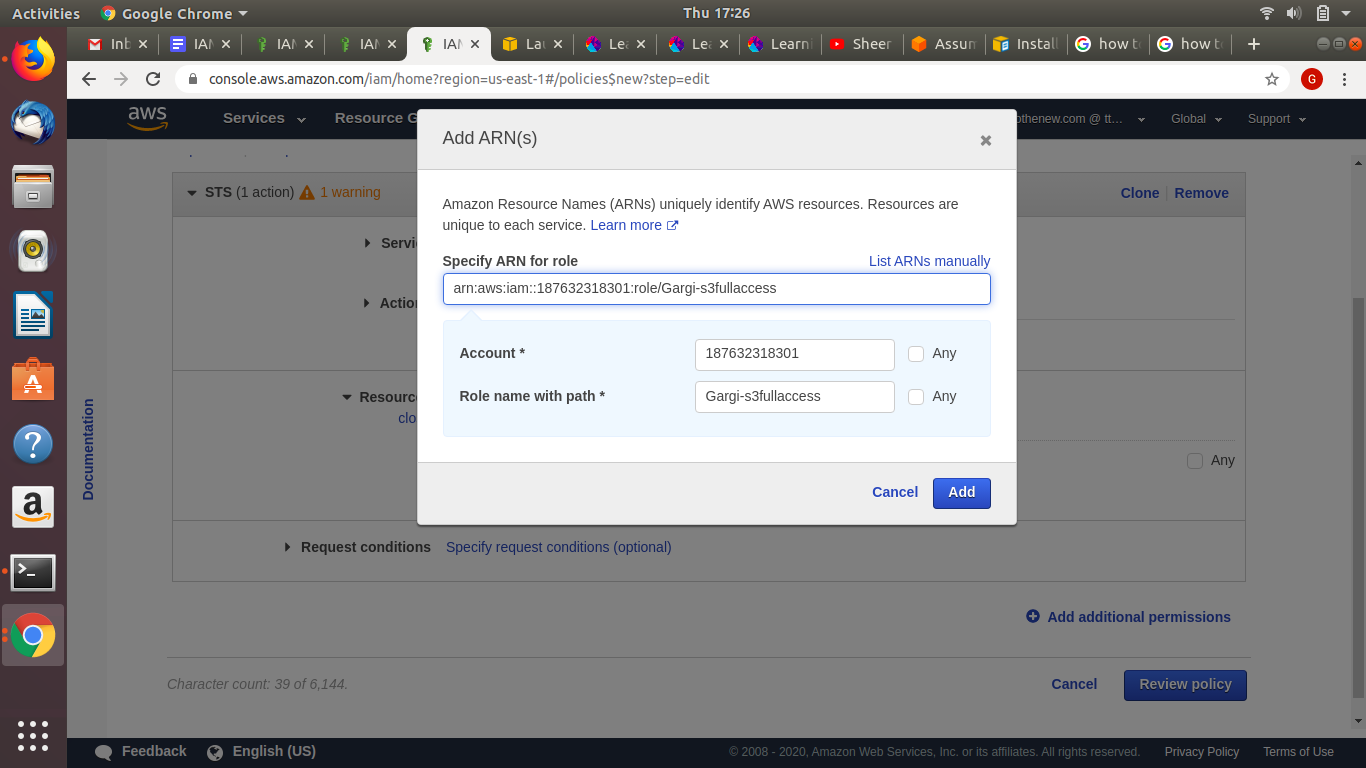
**Create a new role**

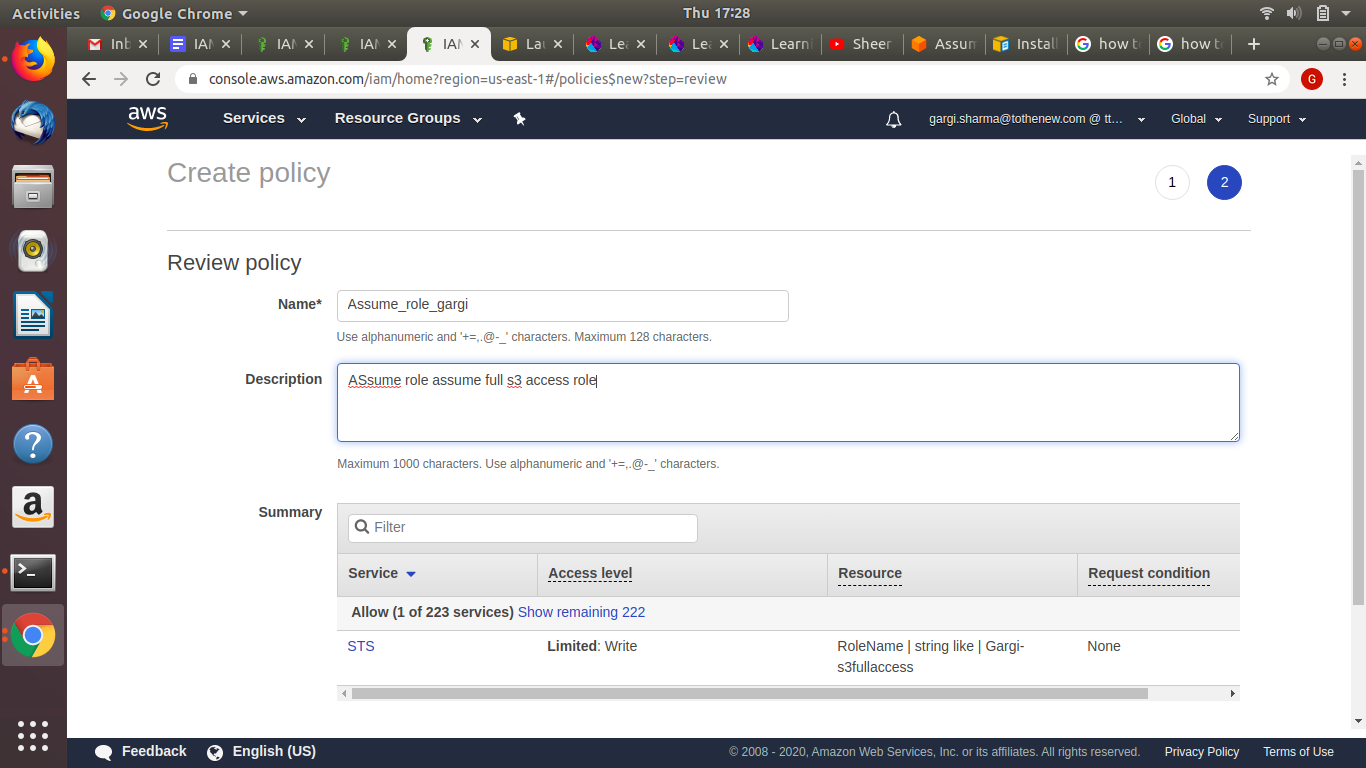
**Create a new policy**

**Select service STS and action assume role**

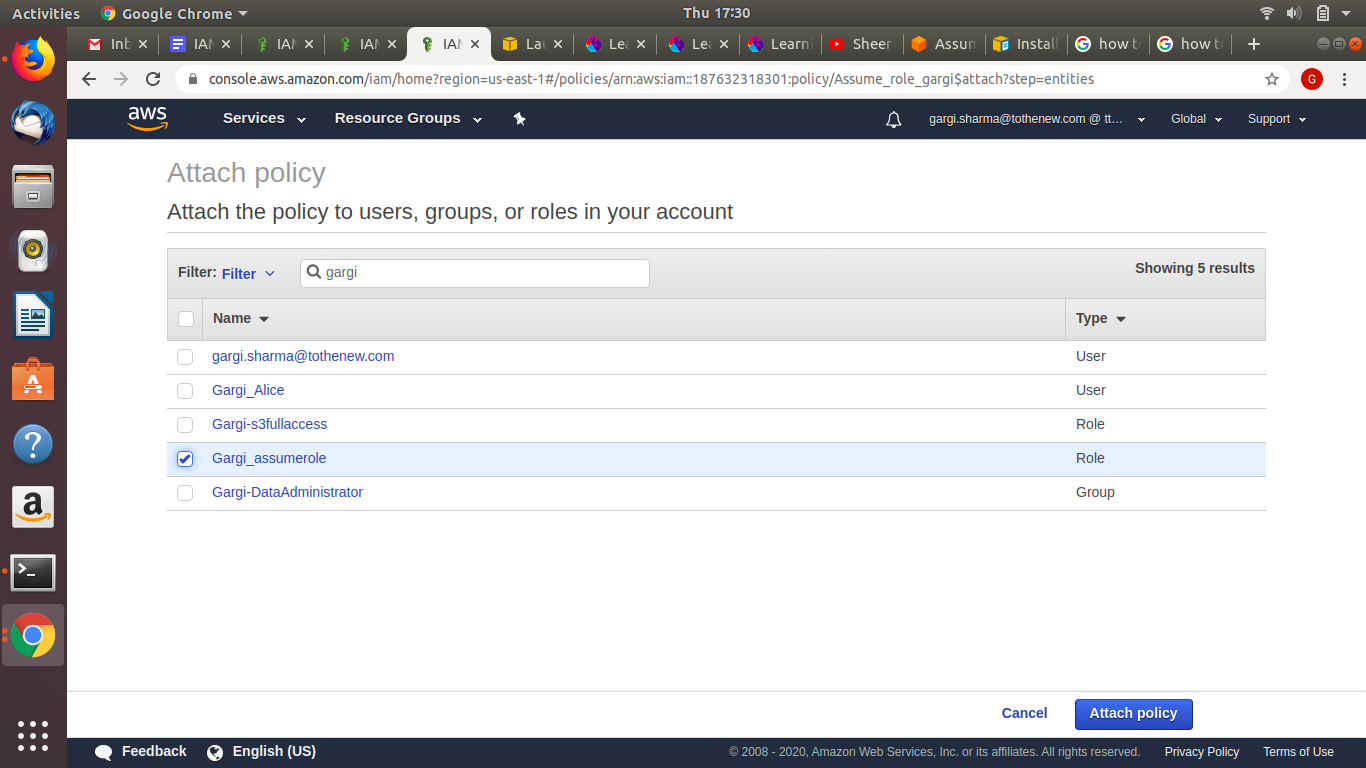
**Go to resources(specific) and Copy the ARN of s3 full access and paste**

****

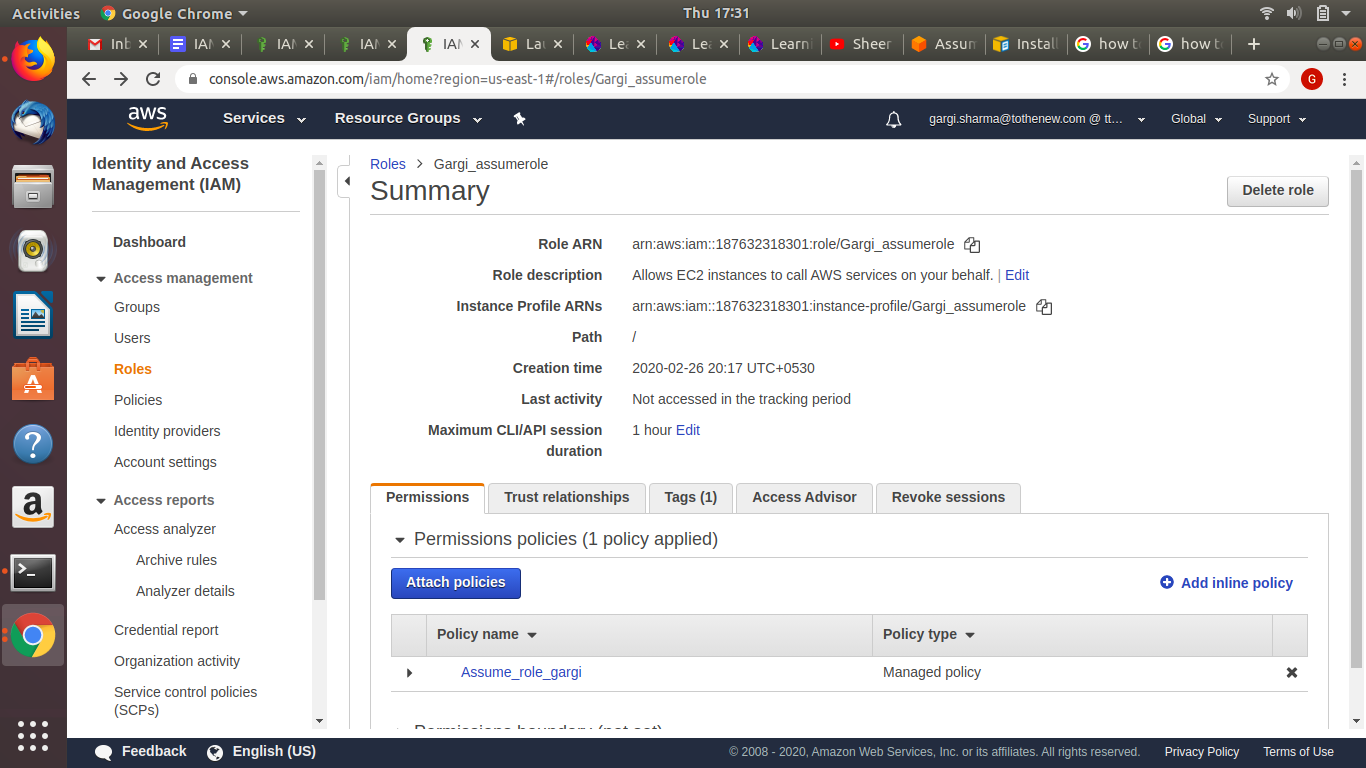
****

****

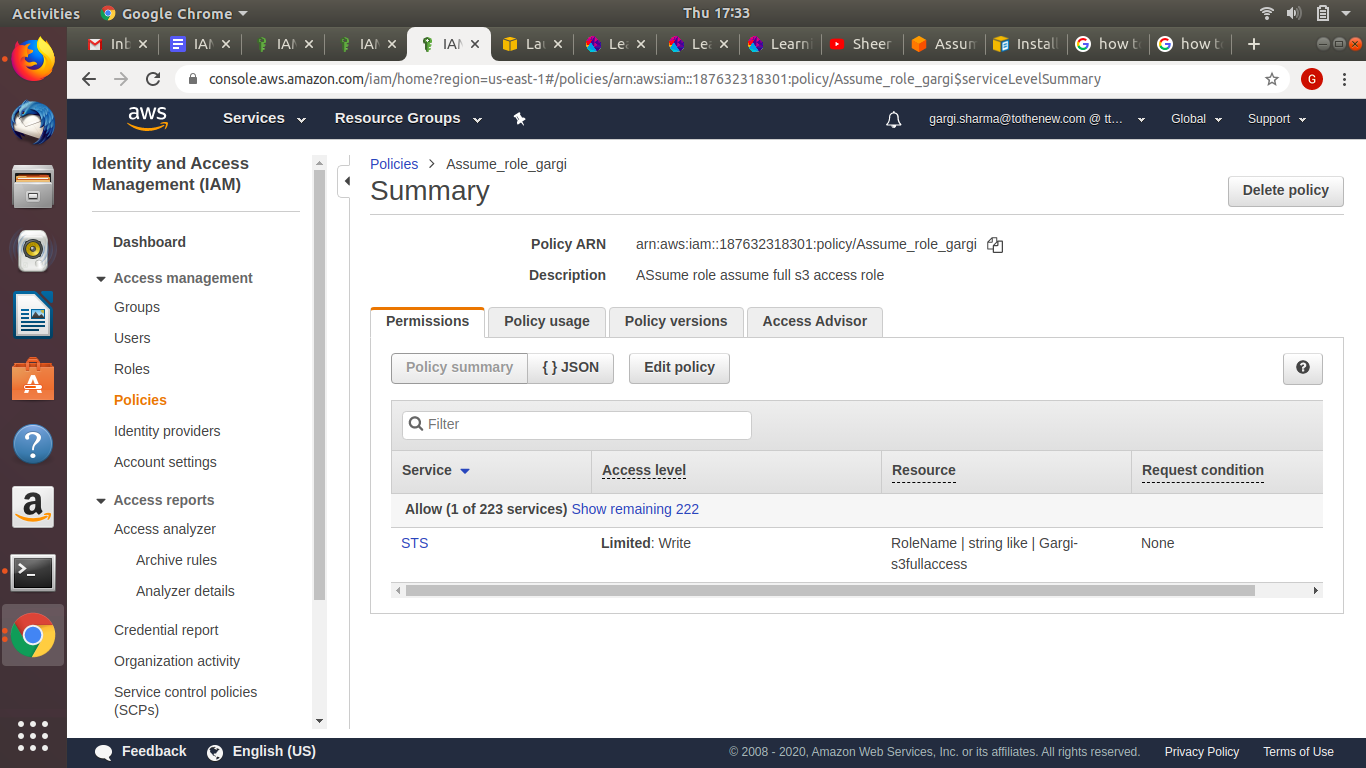
**Attach the policy to the new role created**

****

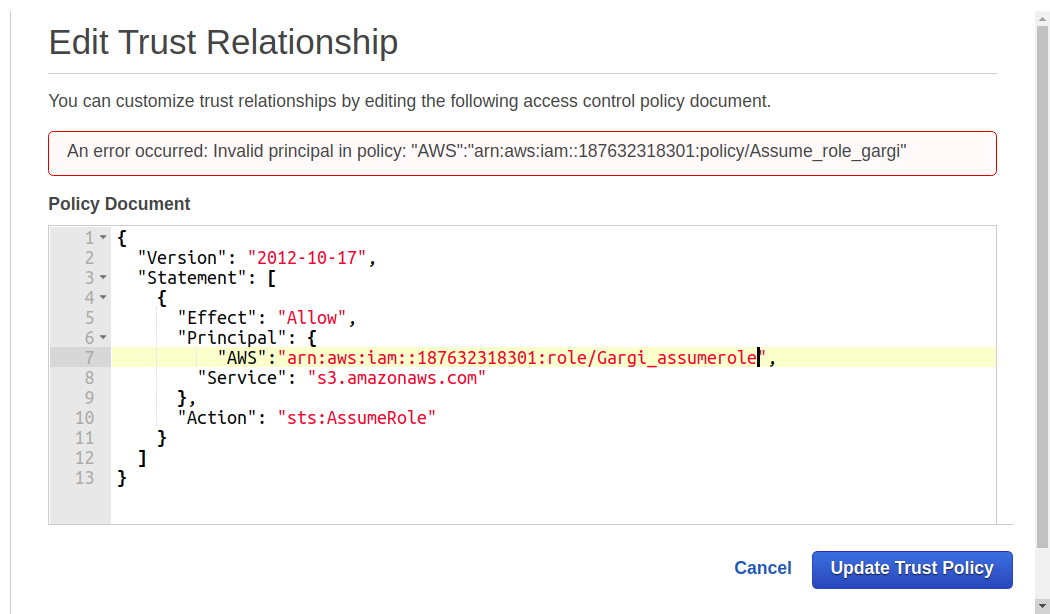
**Now open the newly created role and check for the assume role**

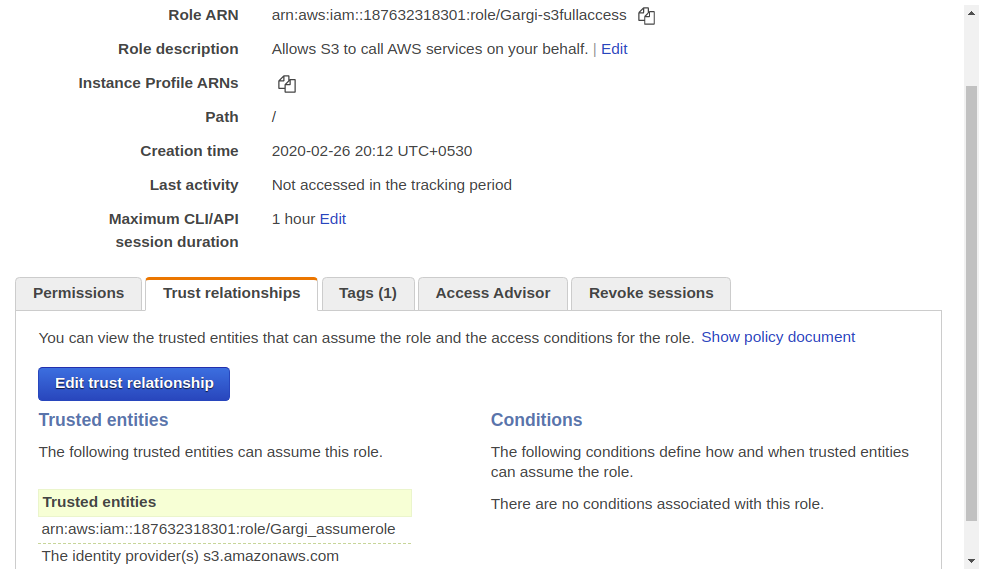
****

**Go inside Assume\_role\_gargi(policy name)**

****

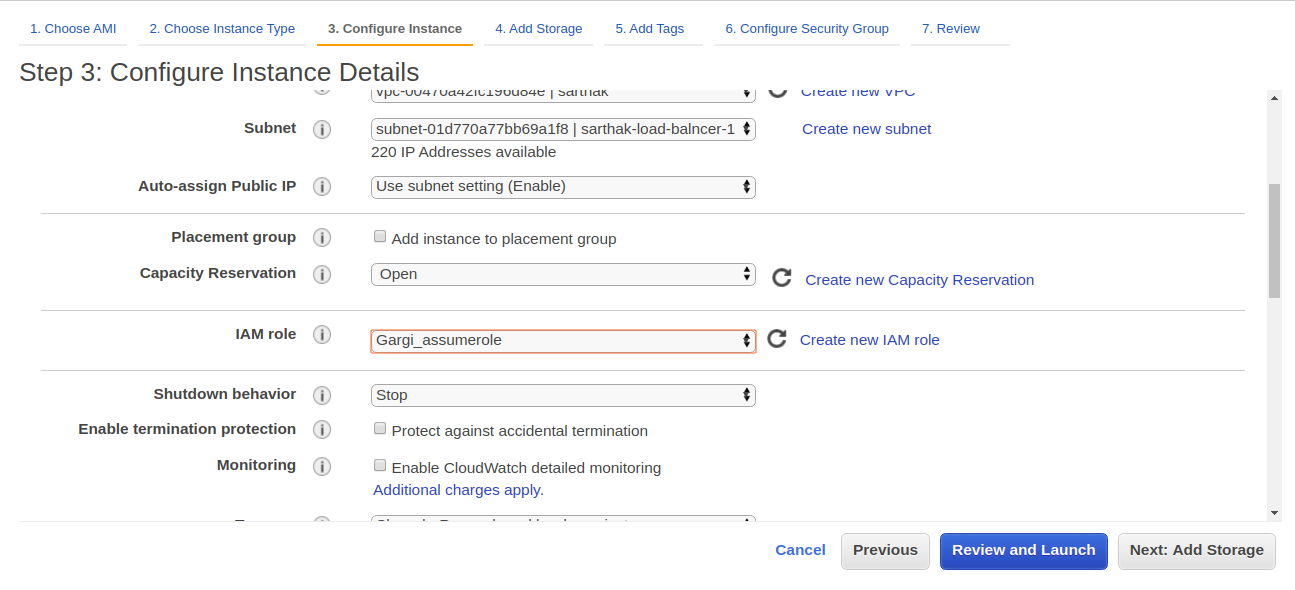
**Go to the newly created role(Gargi\_assumerole) and copy the ARN. Now go to the old role(GargiS3fullaccess) and edit trust relationships. Then paste the ARN as follows.**

****

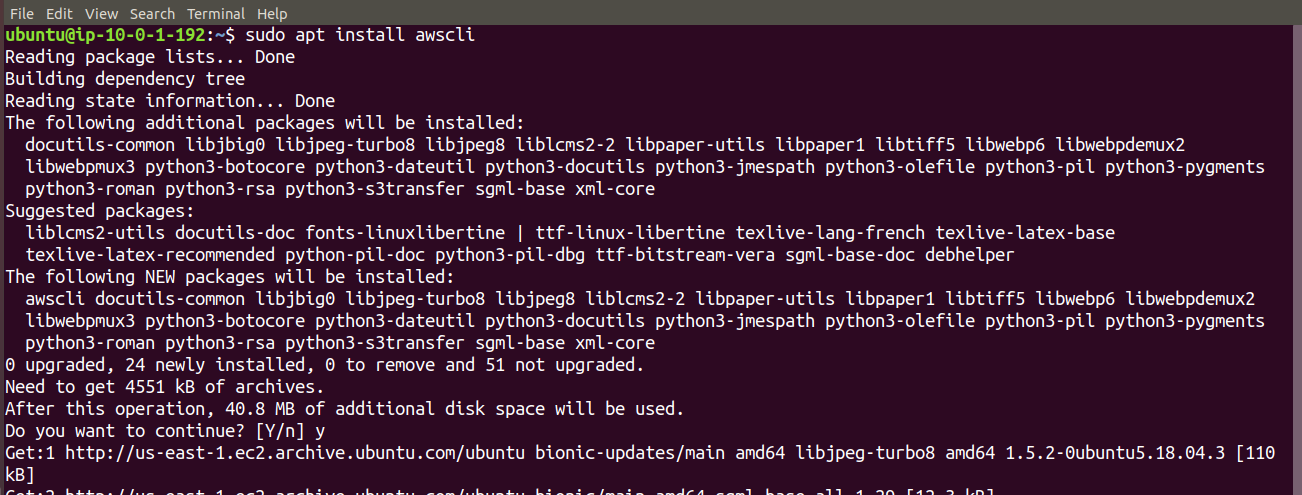
****

1. **Attach this to an instance and get an sts token.**

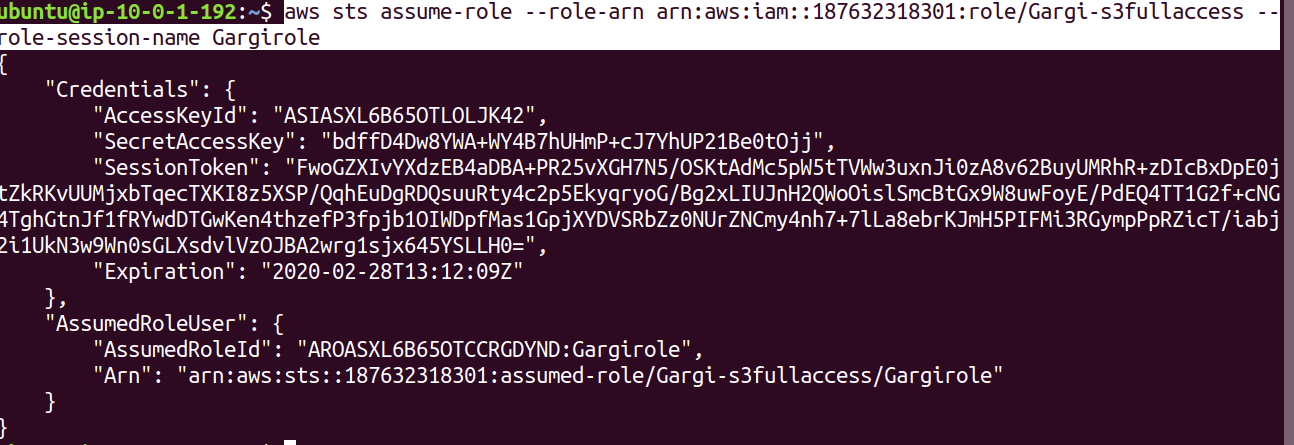
**Create a new instance and then attach the new role(Gargi\_assume\_role)**

****

**SSh into the instance and install awscli**

****

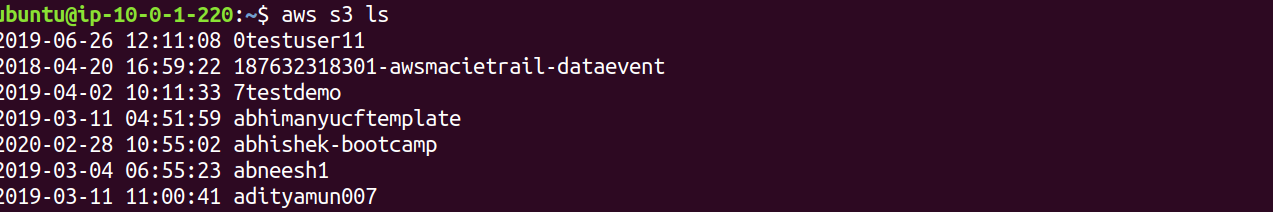
**Now execute the following command :aws sts assume-role --role-arn arn:aws:iam::187632318301:role/Gargi-s3fullaccess --role-session-name Gargirole to generate the sts token.**

****

**Once the token is generated, export variables:**

****

**Now we can execute the “aws s3 ls” command to list all buckets.**

****

1. **Create a group for "Data Administrator" where the user 'Alice' be a member of this group. This group will prepare the data for the analysis. So Provide the following access to the group.**

**Service: Amazon S3;**

**Action:**

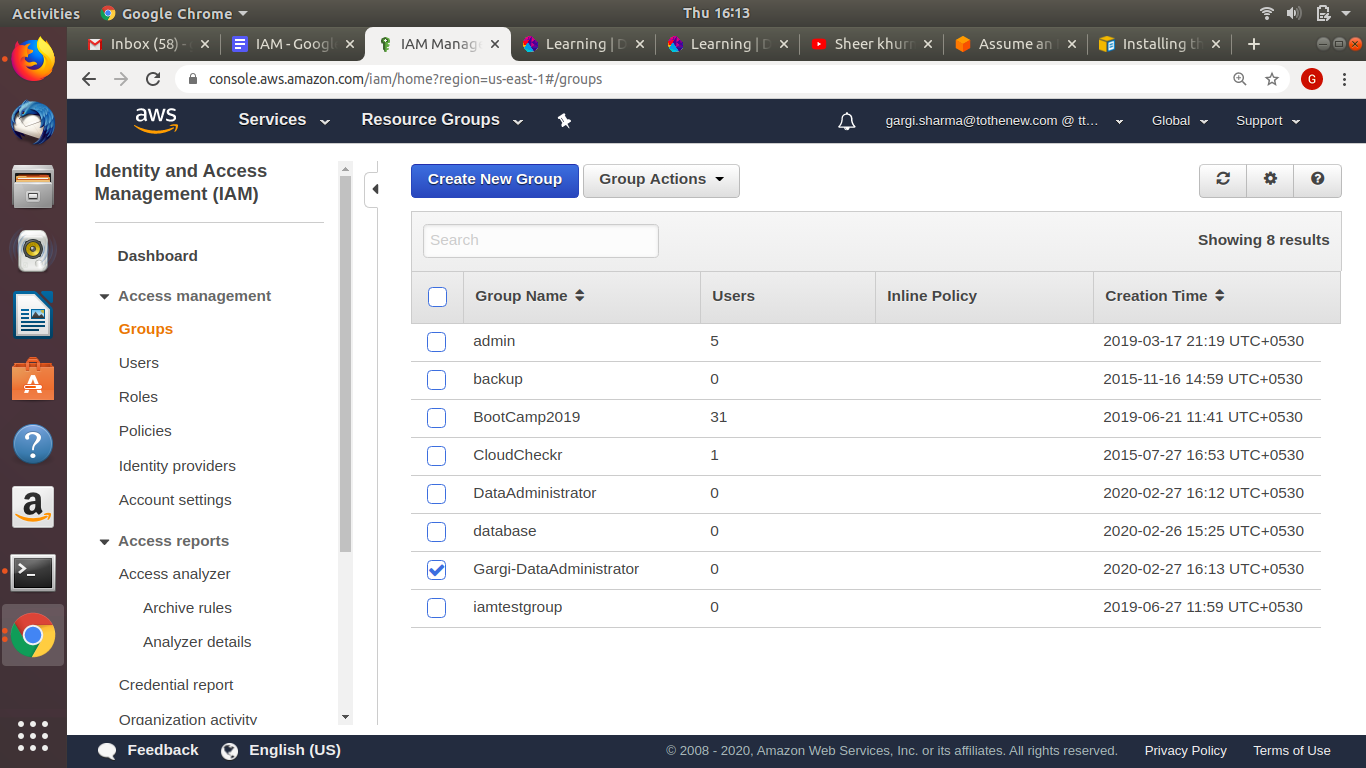
**Get\*,**

**List\*,**

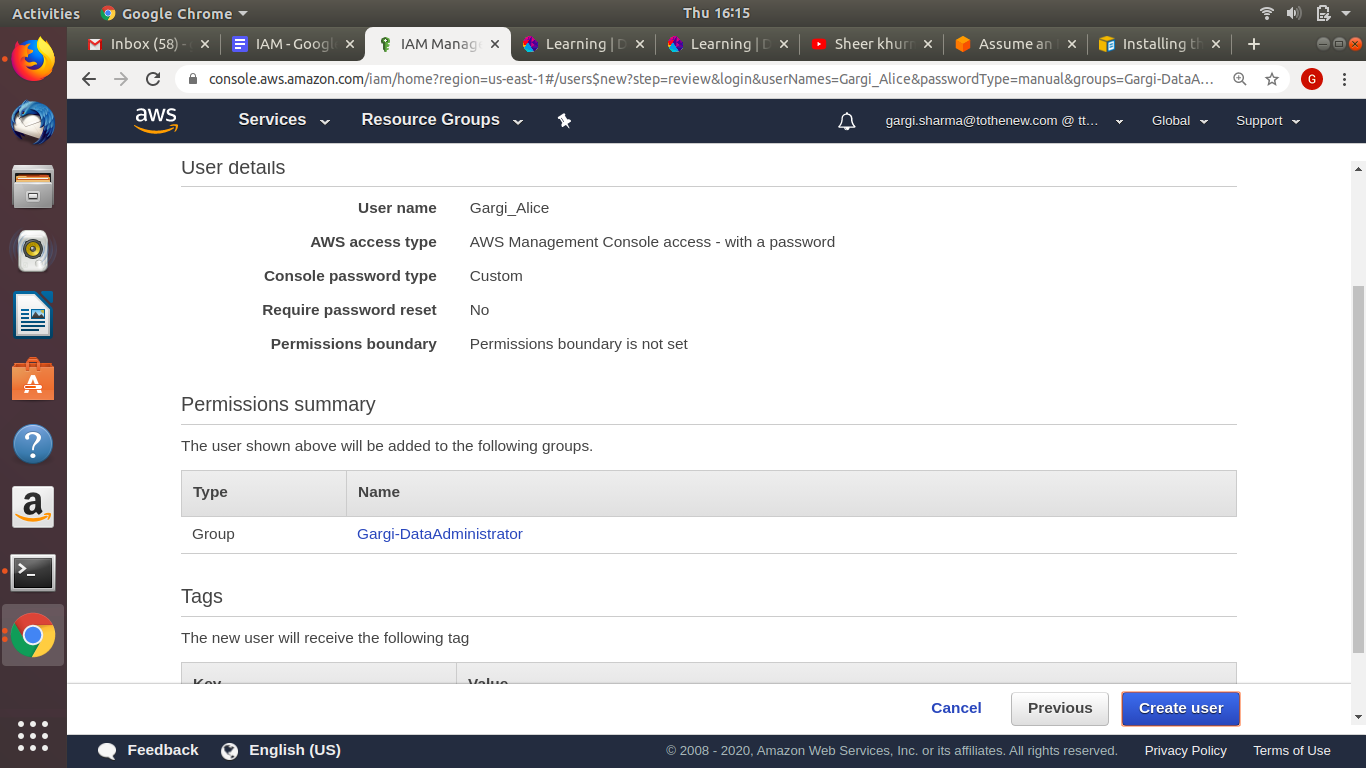
**Put\*,**

**ARN: Input and output Buckets (no conditions)**

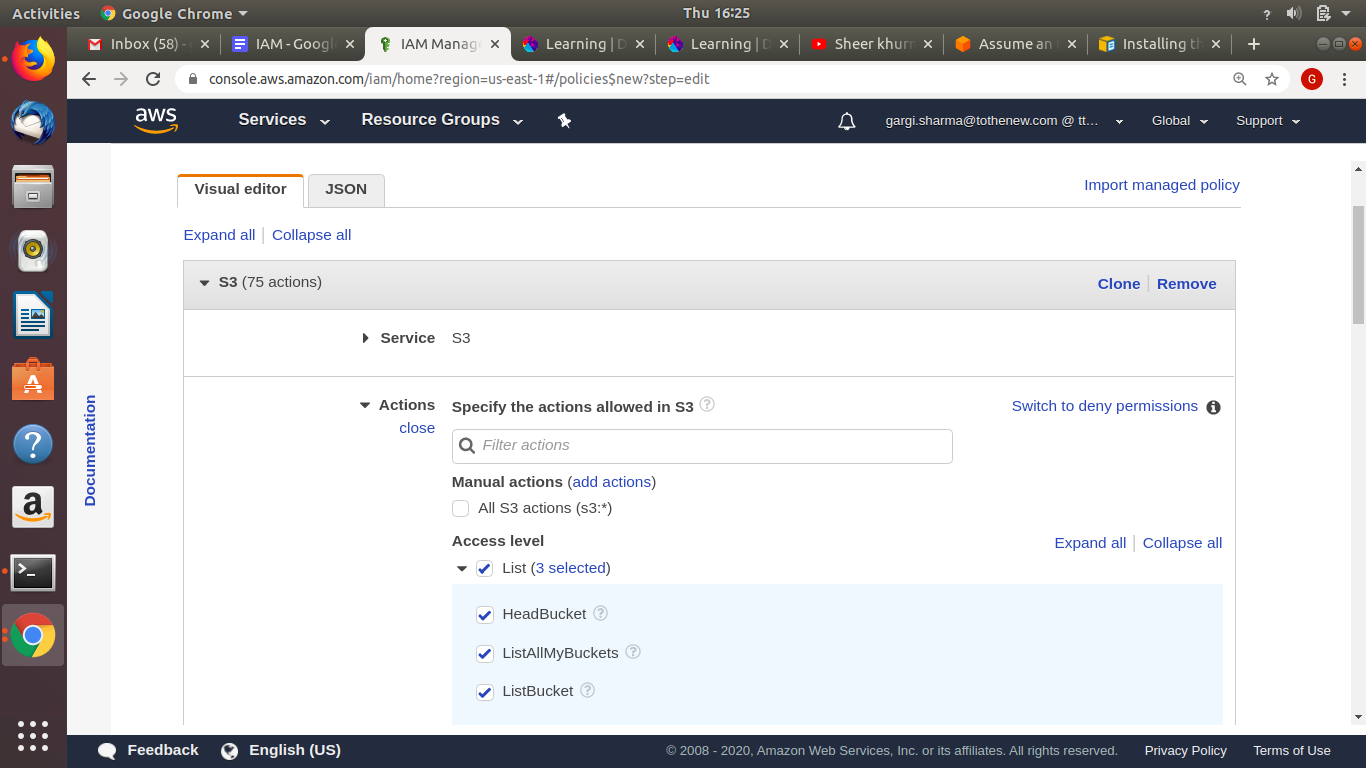
Ans. **Step1: Create user named “dataAdministrator” and attach no policies**

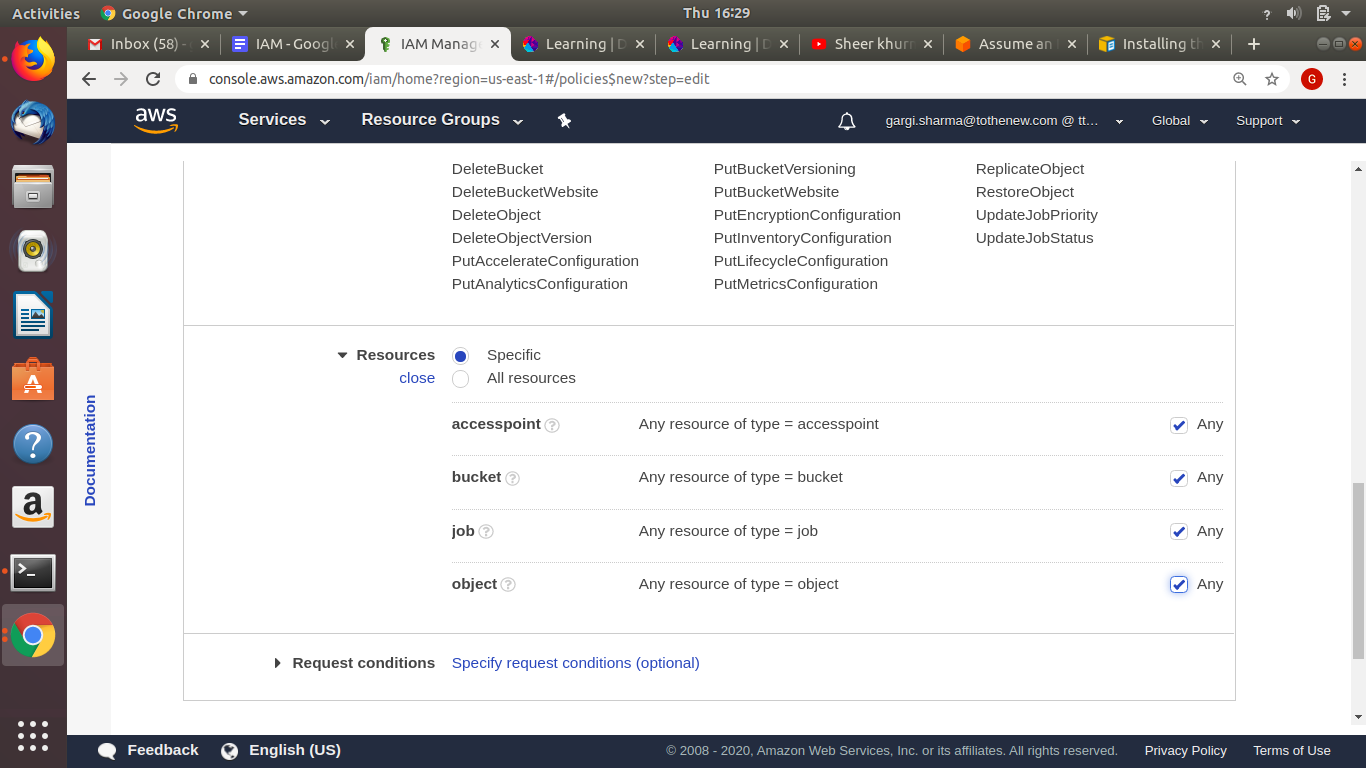


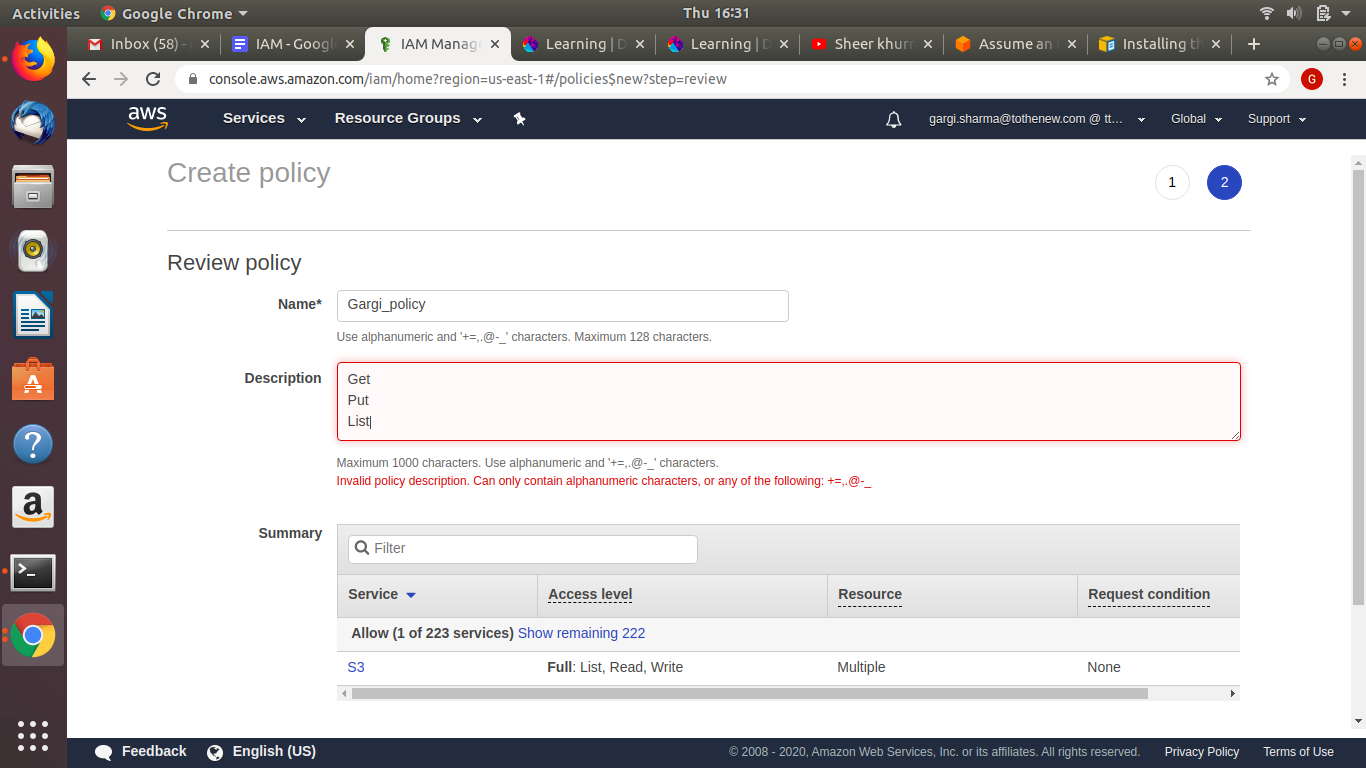
**Step2: Create user named Alice and attach it to the group**



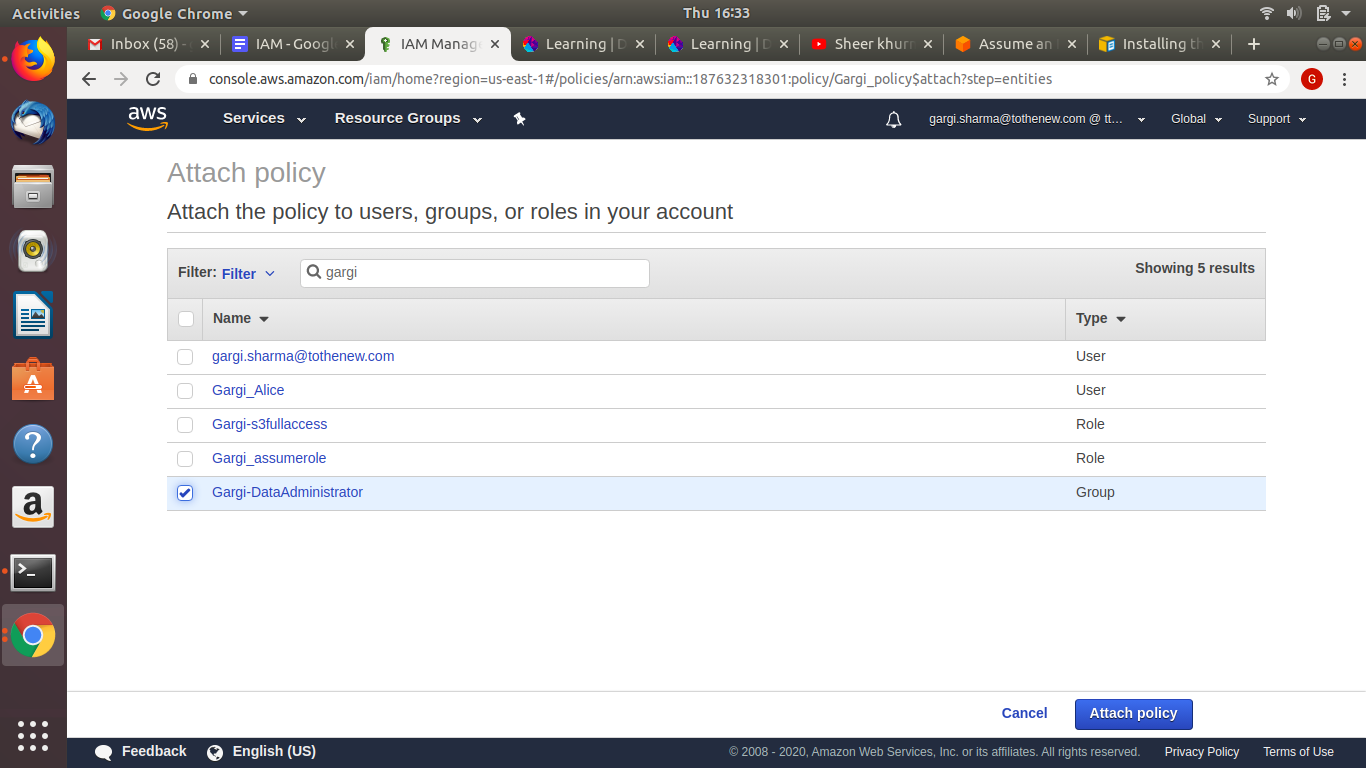
**Step3: Create policies and provide get(all), put(all) list(all) and resources(all any)**







**Step4: Attach the policy to the group**

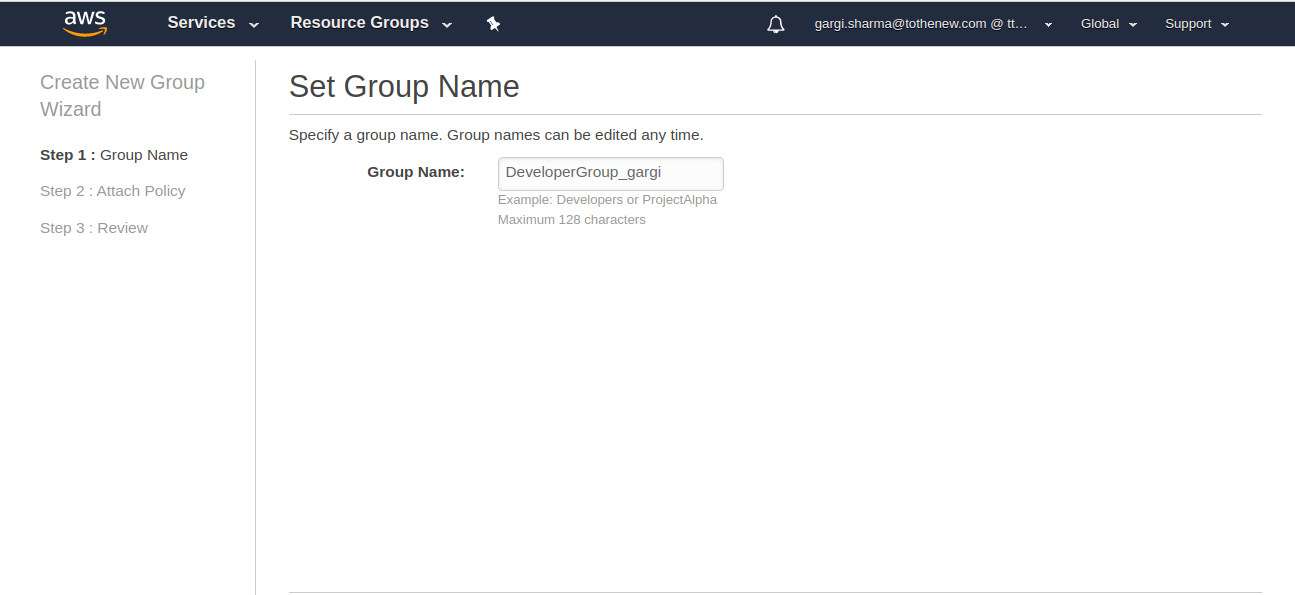


1. **Create a group for the "Developer group " where the user 'bob ' is a member of this group. This group with Test Newly Developed Features for which they require access to EC2 instances. Provide the following access to this group:**

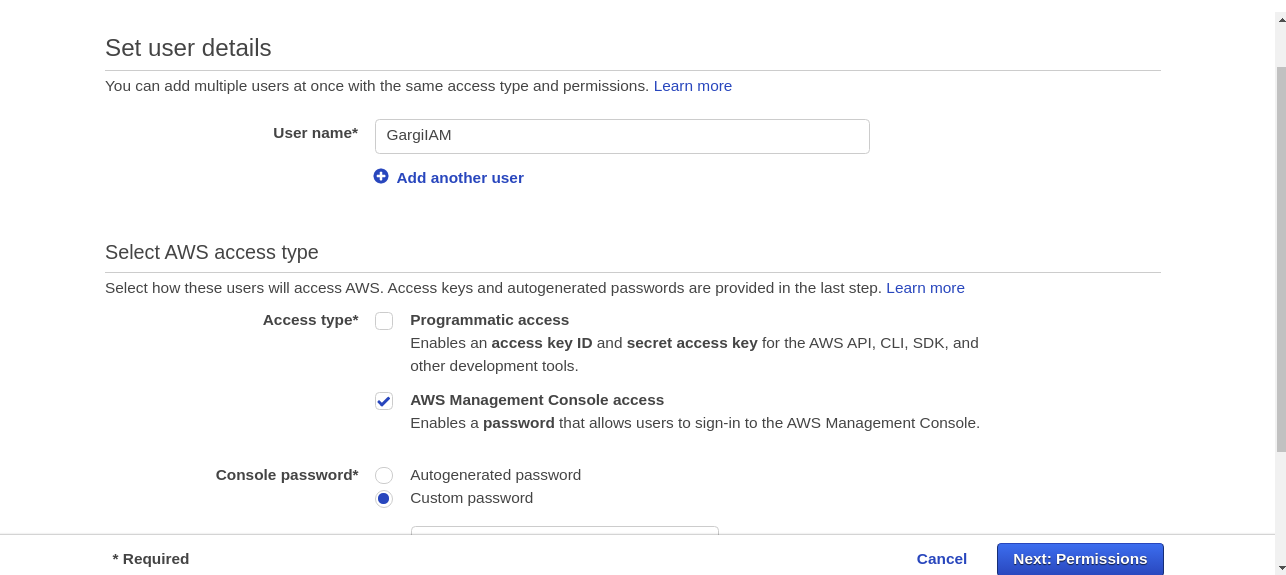
**Service: Amazon EC2**

**Action: \*Instances, \*Volume, Describe\*, CreateTags;**

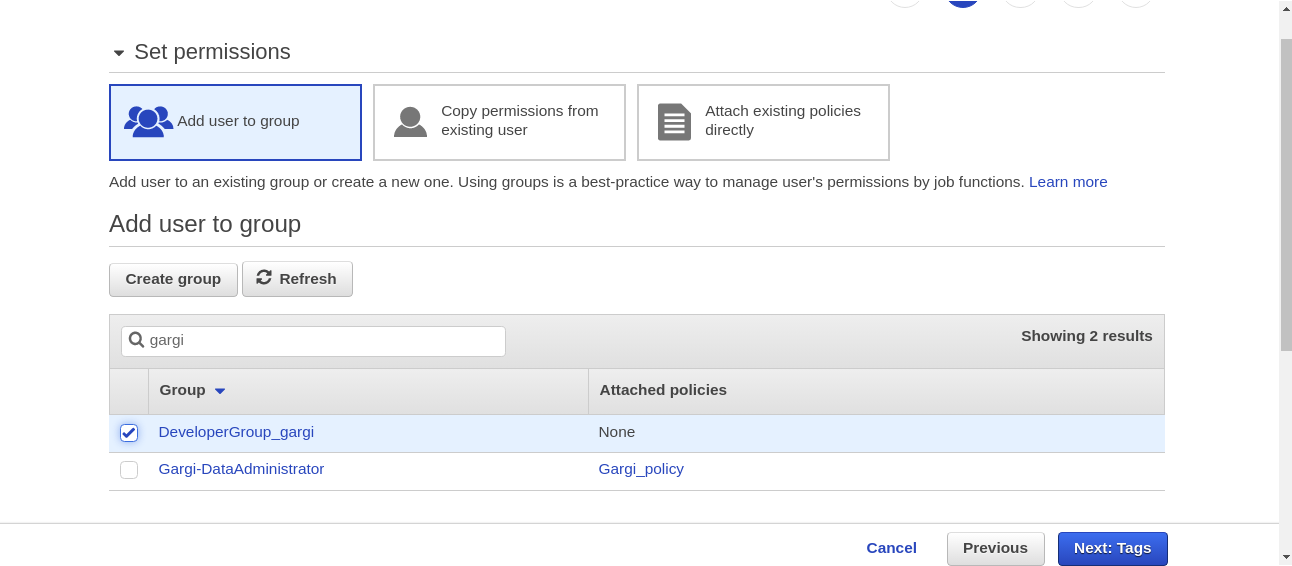
**Condition: Dev Subnets only**

Create group “Develop\_group-gargi”

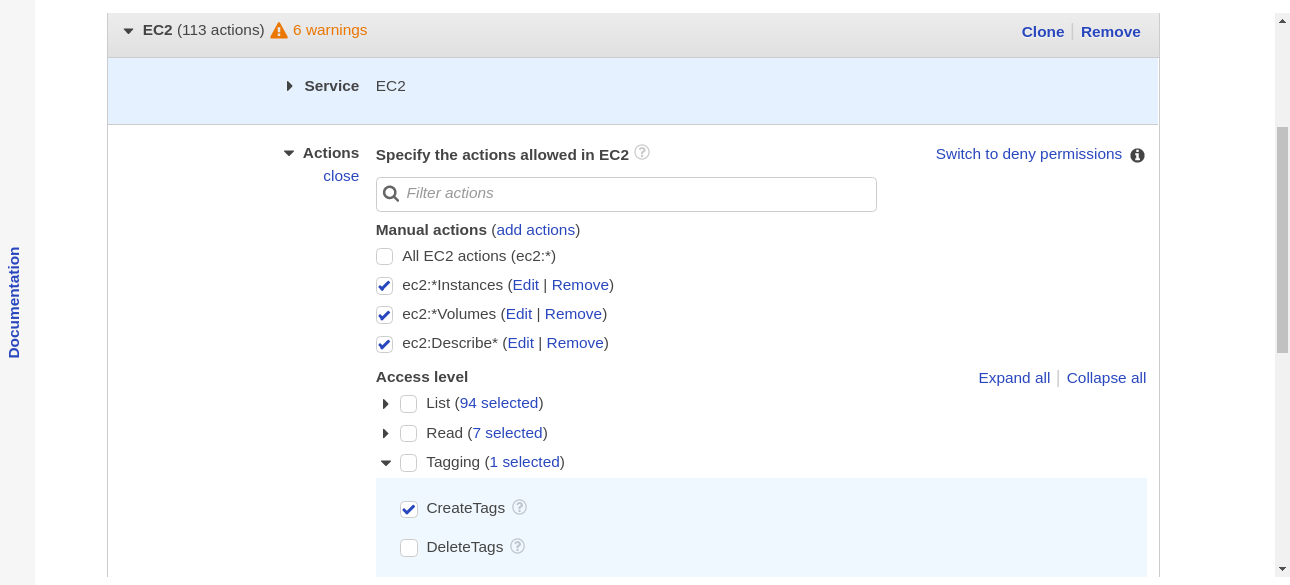
Create user “Gargi-IAM”

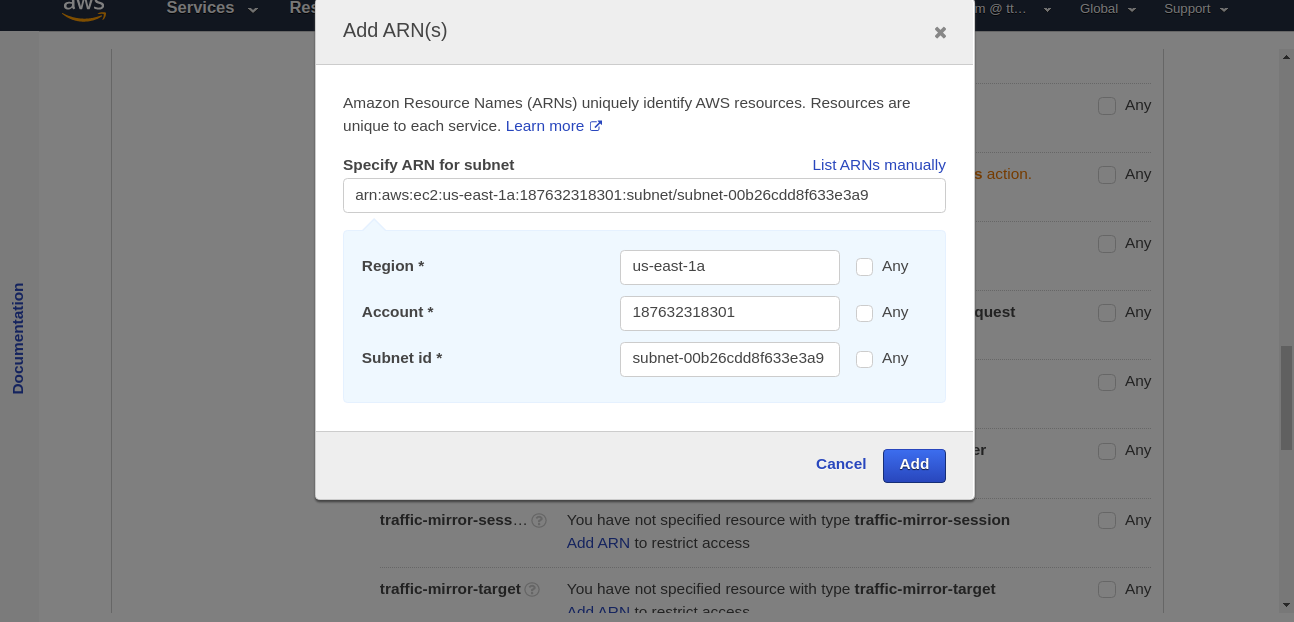


Attach it to the group created:

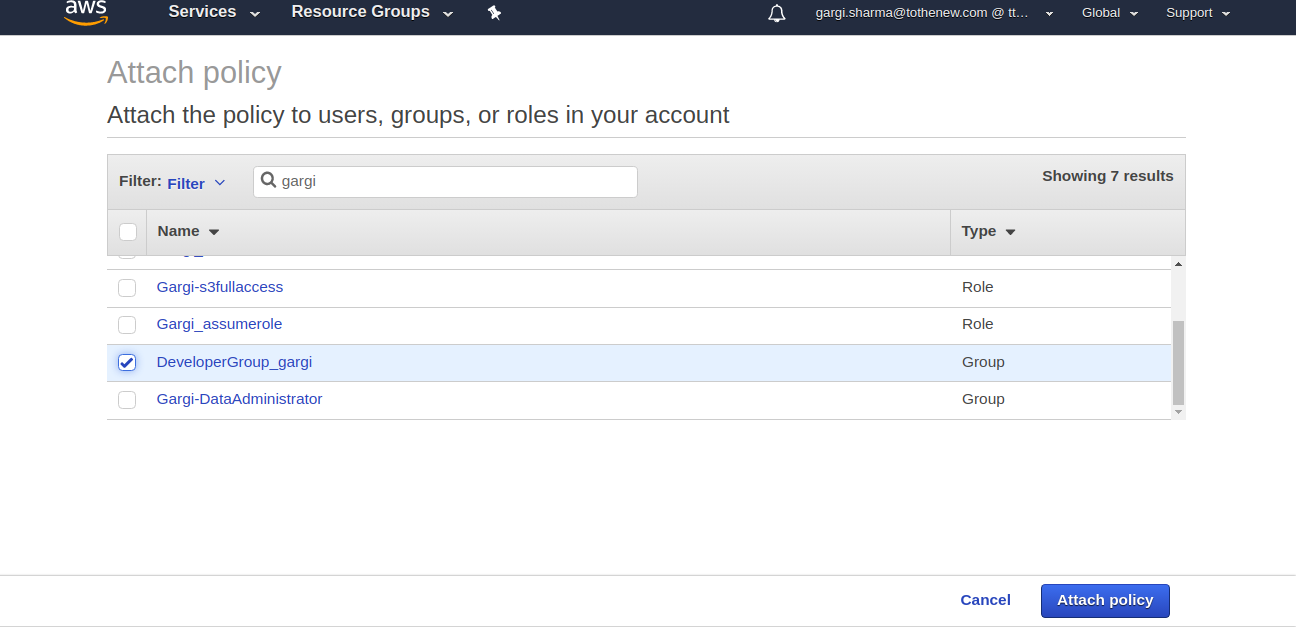


Create a policy “dev\_group\_policy” and specify the action and condition as mentioned in the question(Providing arn of Dev subnet)

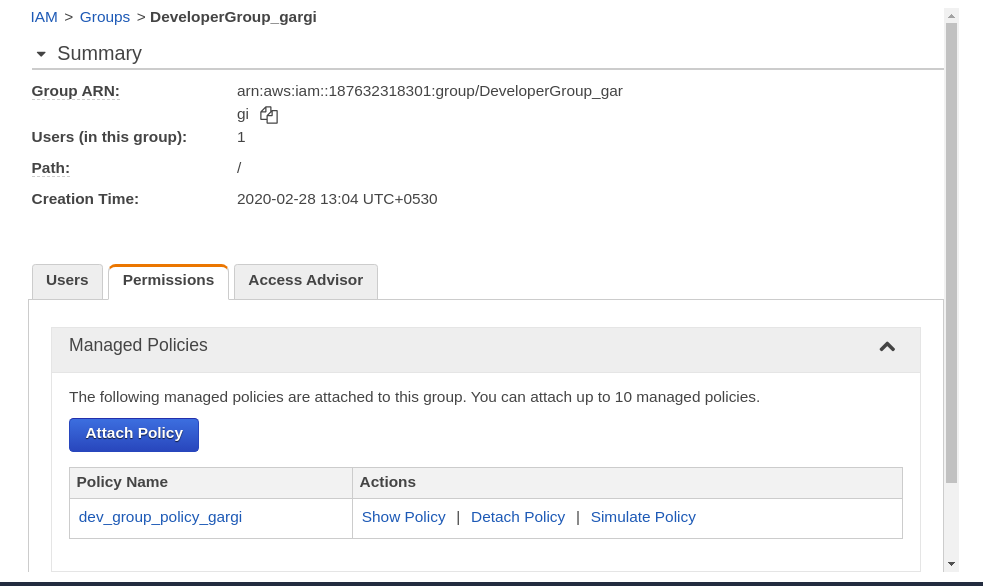


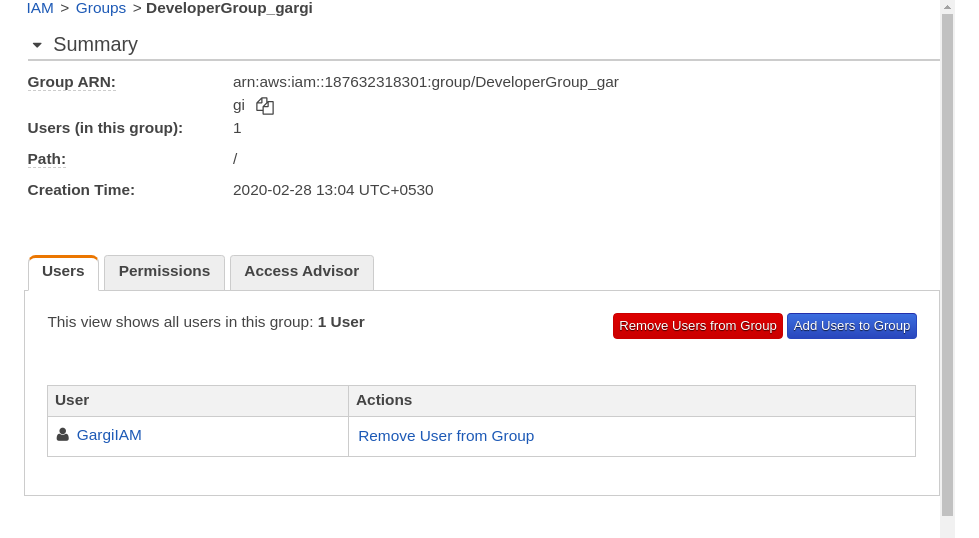
Add ARN for subnet

Attach the above policy to the “DeveloperGroup\_gargi”



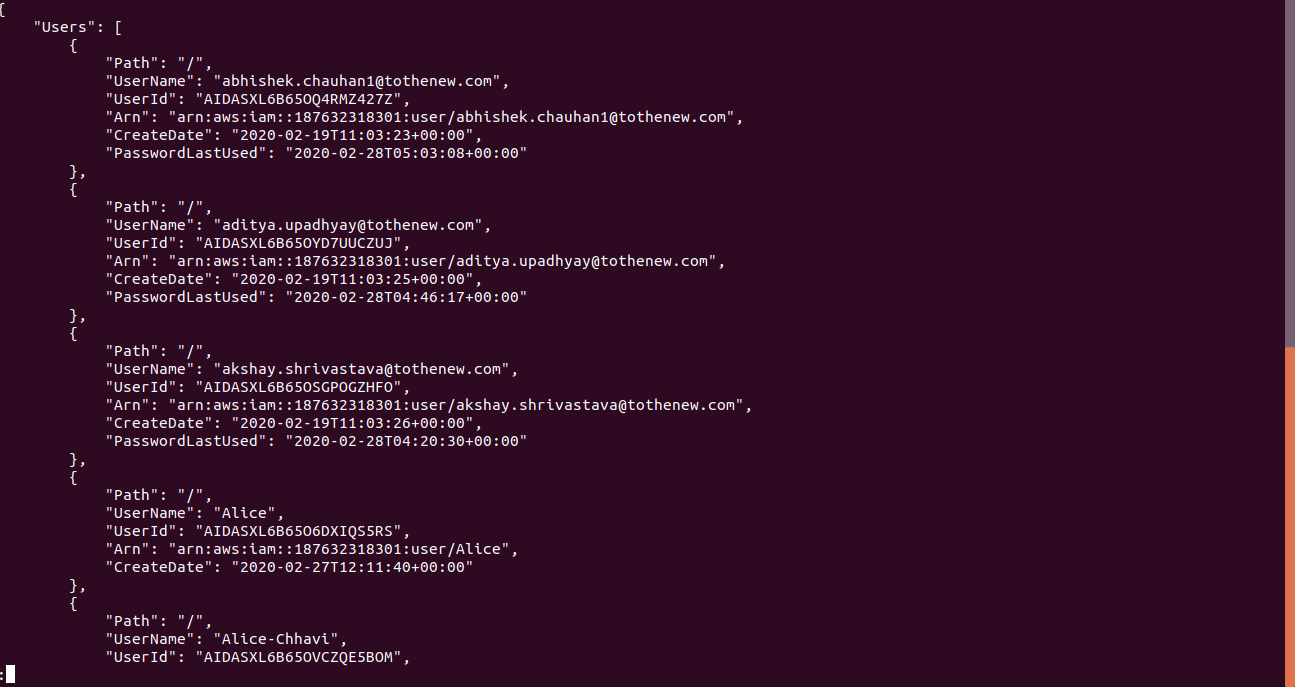
Open the group and check for the users and policies attached.



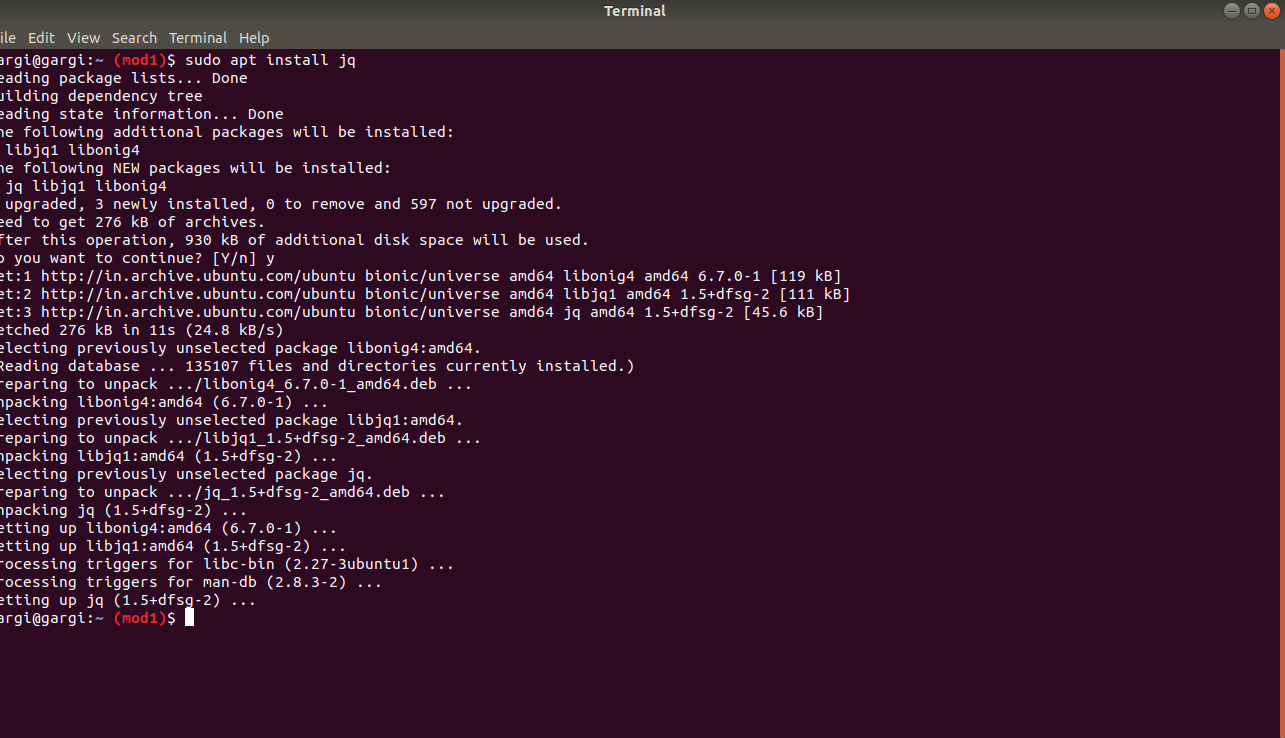


1. **Identify the unused IAM users/credentials using AWS CLI.**

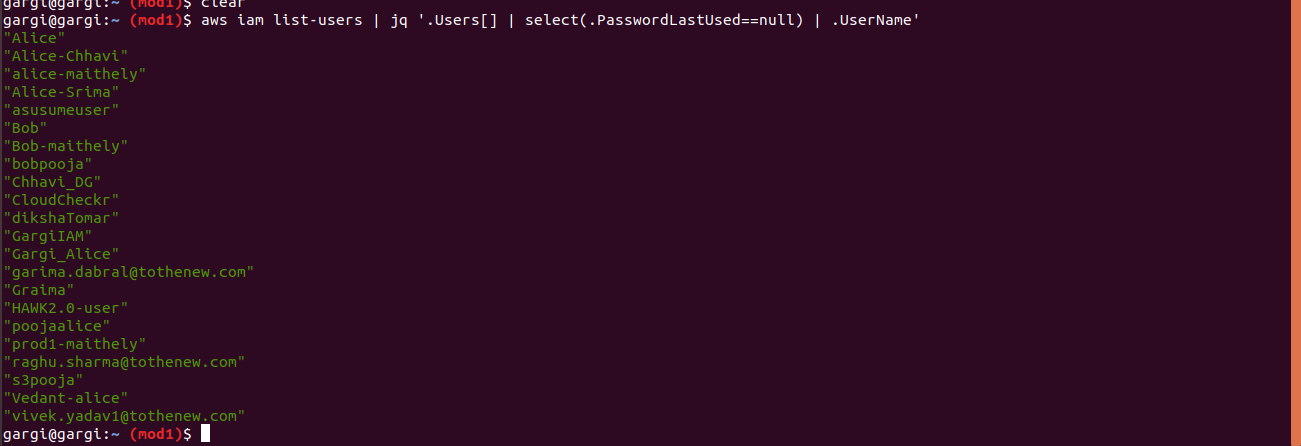
Step1: List all users and Install jq



**jq(JSON QUERY) is like sed for JSON data - you can use it to slice and filter and map and transform structured data with the same ease that sed, awk, grep and friends let you play with text.**

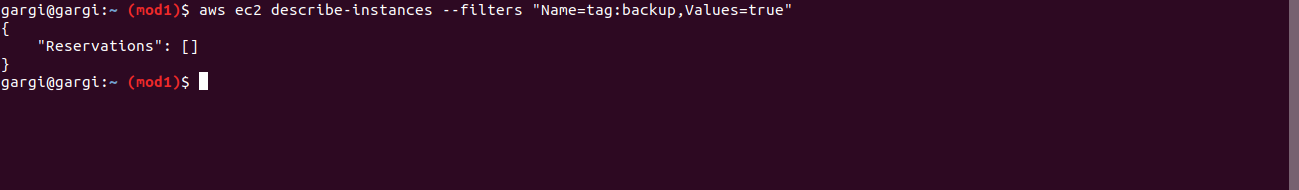
****

**JQ Query: Aws iam list-users | jq ‘.Users[ ] | select(.PasswordLastUsed==null) | .UserName’**

****

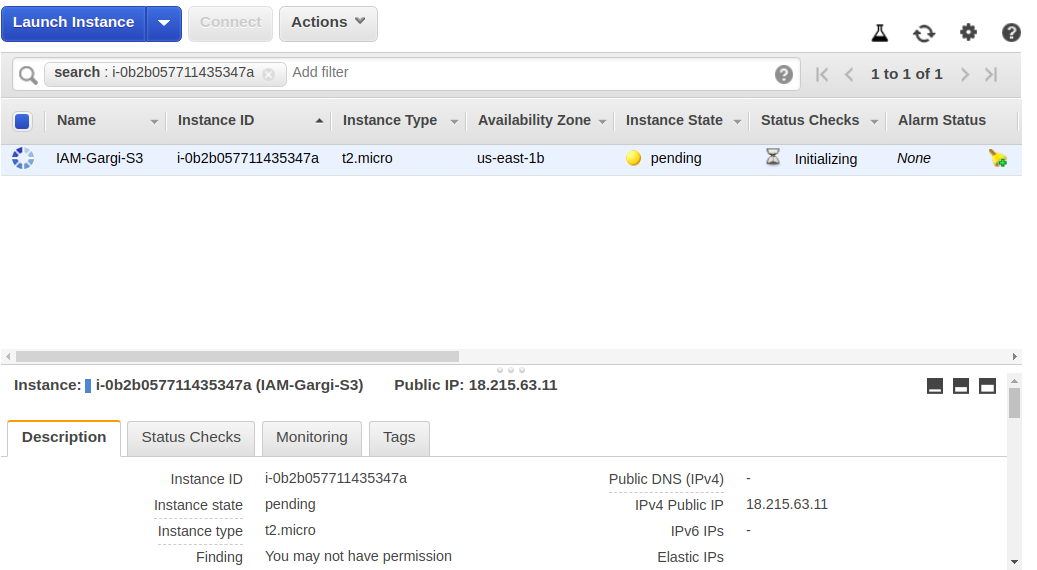
1. **Identify all the instances having the tag key-value "backup=true" using AWS CLI.**

Command: aws ec2 describe-instances --filters “Name=tag:backup,Values=true”

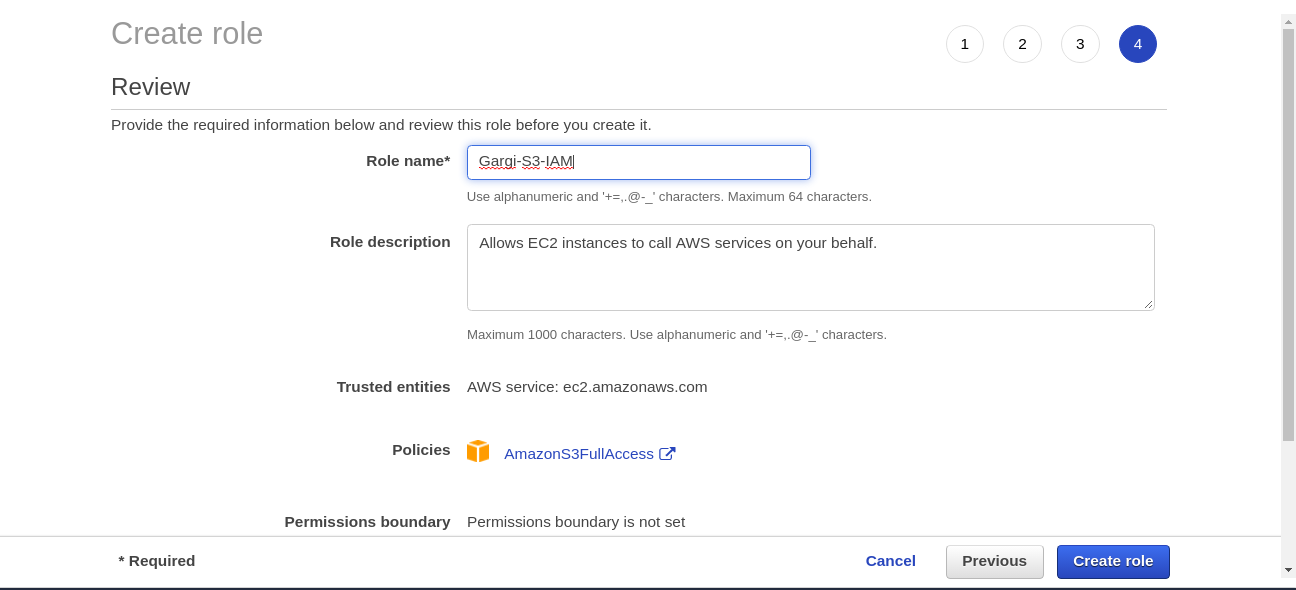
****

1. **An EC2 Instance hosts a Java-based application that accesses an s3 bucket. This EC2 Instance is currently serving production users. Create the role and assign the role to EC2 instance.**

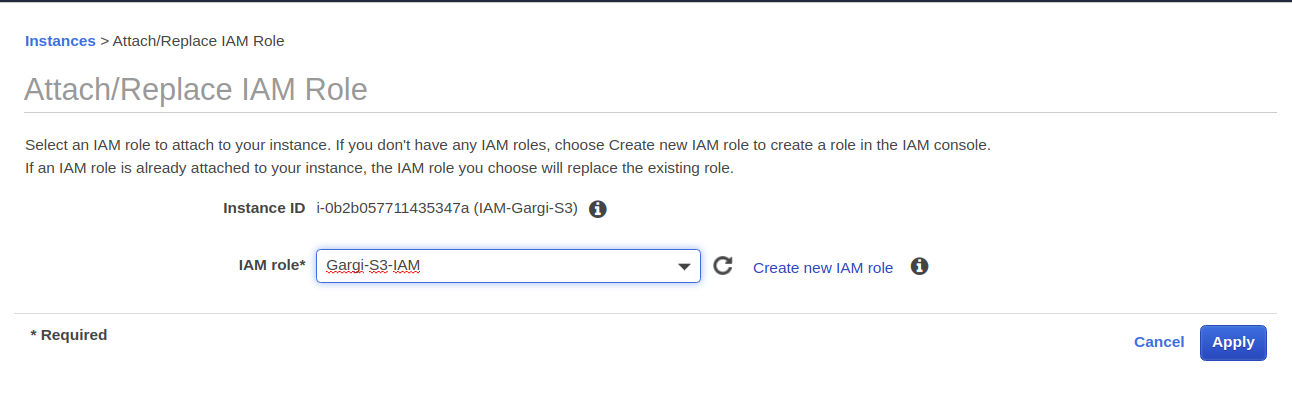
Step1:Launch an EC2 instance:



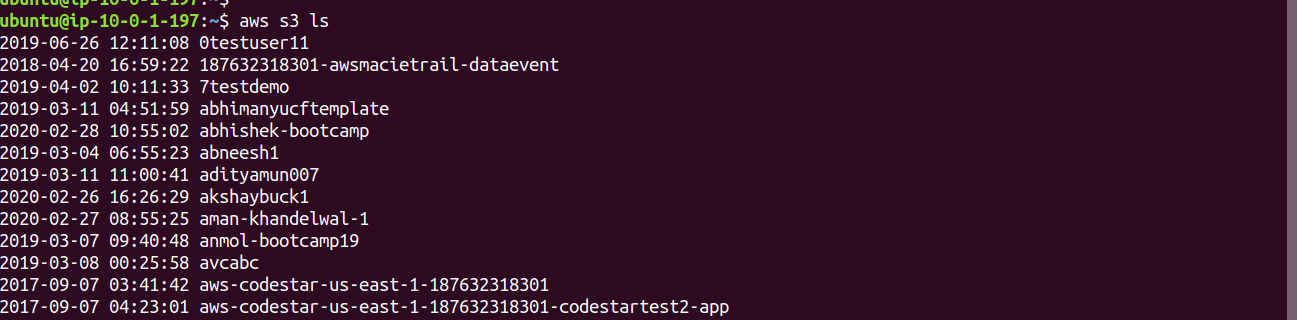
Step2:Create a role and attach S3 full access policy to it.



Step3:Attach the above role created to the EC2 instances.



Step4: ssh into instance, install awscli and run “aws s3 ls”



**9. You have both production and development based instances running on your VPC. It is required to ensure that people responsible for the development instances do not have access to work on production instances for better security. Define the tags on the test and production servers and add a condition to the IAMPolicy which allows access to specific tags.**

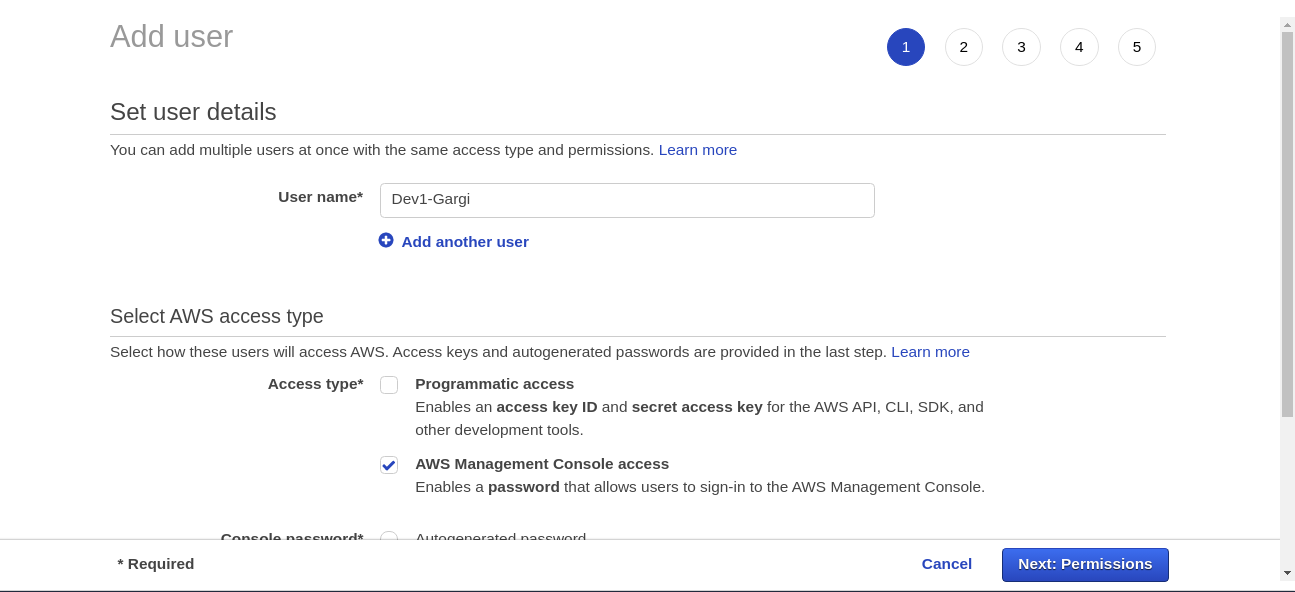
**ANS.**

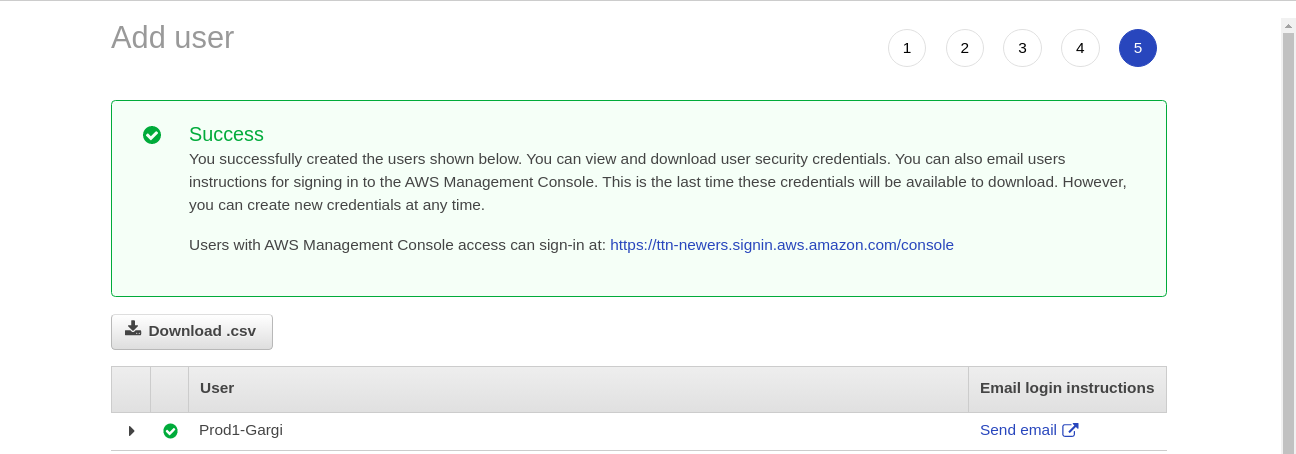
Step1: We will create two instances in the default VPC.

Gargi-Prod and Gargi-Dev



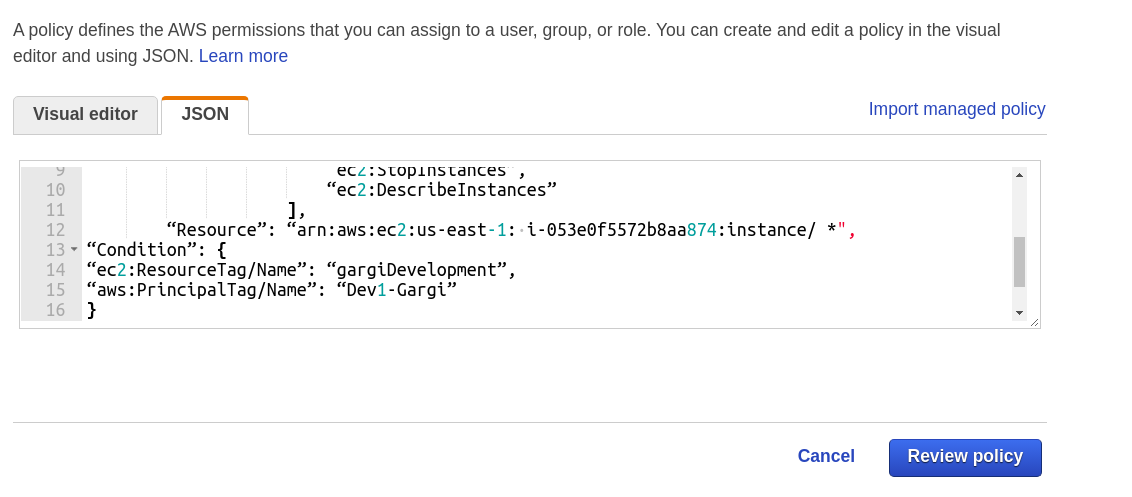
Step2: Now create two users: Dev1-gargi and Prod1-gargi



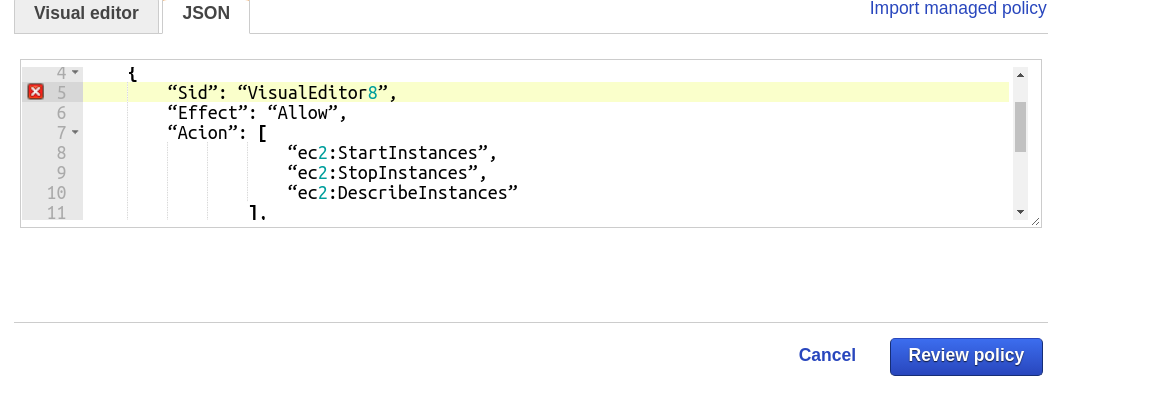


Now create a policy for the development server





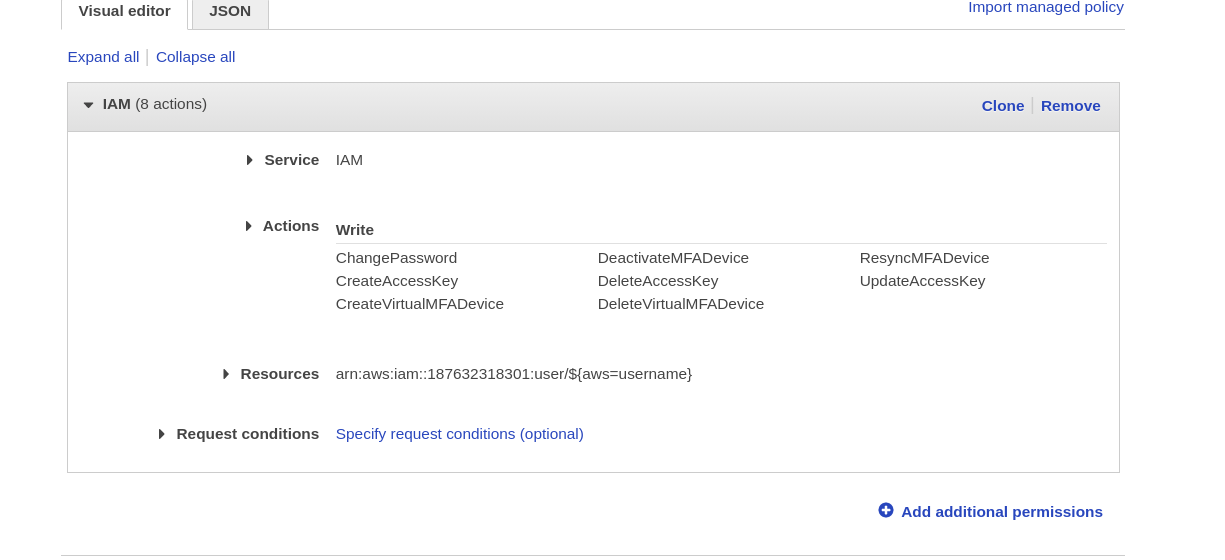
Create policy in the same way for Prod1-Gargi also.

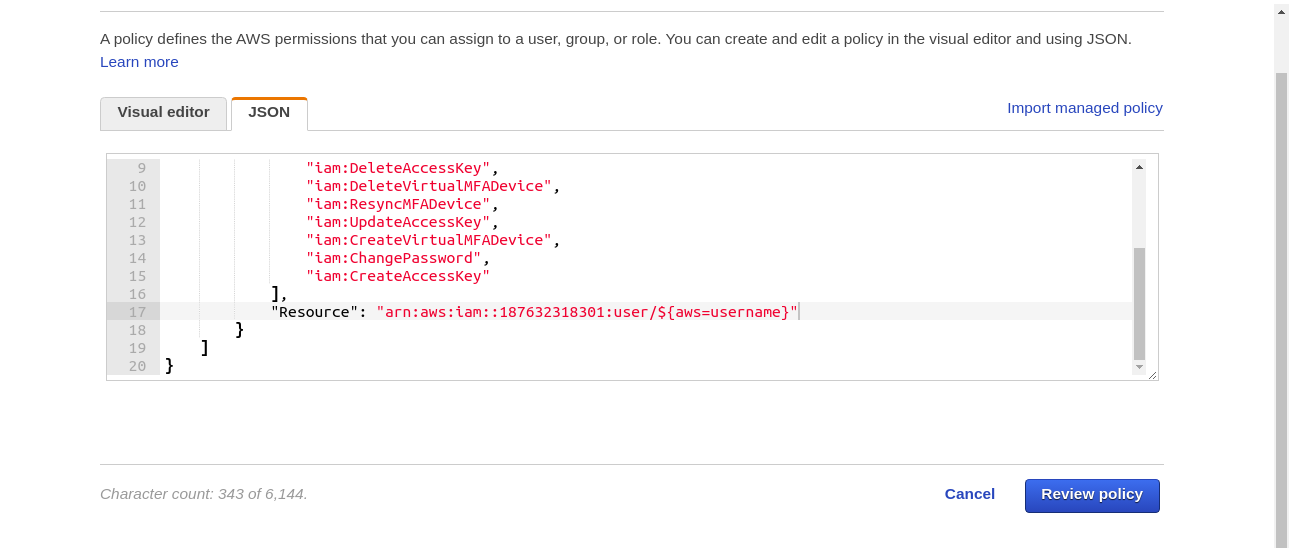




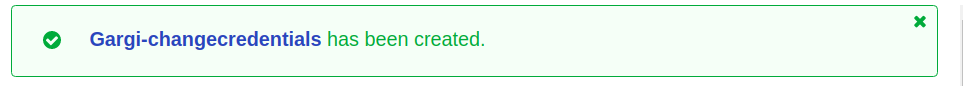
**10. Create a policy for allowing users to set or rotate their credentials, such as their console password, their programmatic access keys, and their MFA devices.**

STEP 1: Create a policy and set service=IAM and give actions as per the question

****

****

Policy has been created

****