**IAM Identity Center features & use cases**

Contents

[Scope 2](#_Toc114825506)

[Problem Statement 2](#_Toc114825507)

[Introduction 2](#_Toc114825508)

[How AWS Single Sign On Works? 2](#_Toc114825509)

[How to create and manage users within AWS Single Sign-On 3](#_Toc114825510)

[Diagram Representation 3](#_Toc114825511)

[Steps Enable single sign-on access to your AWS EC2 Windows instances: 4](#_Toc114825512)

[1. Add users and groups in AWS SSO 4](#_Toc114825513)

[2. Create permission sets: 5](#_Toc114825514)

[3. Secondly, simply assign groups to accounts and permission sets: 7](#_Toc114825515)

[4. Assign the group: 8](#_Toc114825516)

[5. Assign group to Account 8](#_Toc114825517)

[6. Check the permission set to users 9](#_Toc114825518)

[7. Users sign into the User Portal to access accounts: 10](#_Toc114825519)

[8. Sign in and Account UI: 10](#_Toc114825520)

[Steps Enable single sign-on access to your AWS applications: 12](#_Toc114825521)

[1. Provision and configure AWS SSO 12](#_Toc114825522)

[2. Jenkins side SAML configuration 14](#_Toc114825523)

[3. Add user to Jenkins SSO application 16](#_Toc114825524)

[4. Add access privileges in Jenkins portal 18](#_Toc114825525)

**IAM Identity Center features & use cases**

Document Name: IAM Identity Center features & use cases

Owner: Bhagyashree Patle

# Scope

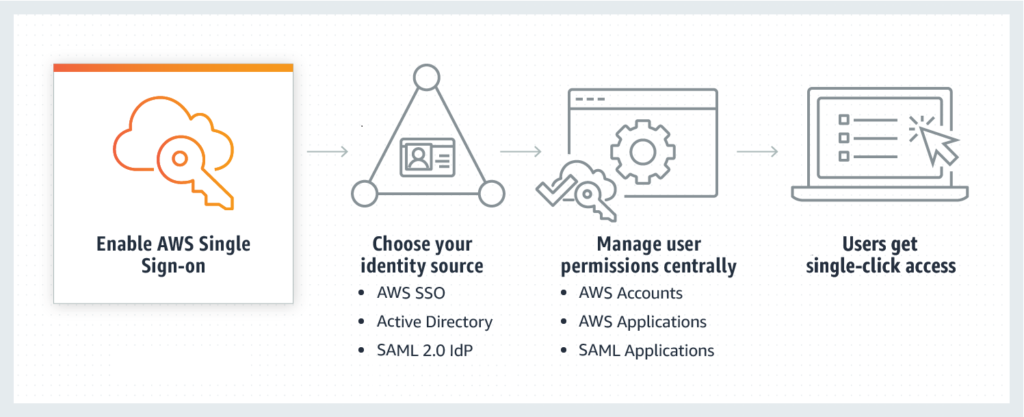
This allows you to securely access your EC2 Windows instances with existing corporate user names, passwords, and MFA devices. You are no longer required to share administrator credentials, access credentials multiple times, or configure remote access client software.

# Problem Statement

Create AWS directory that you can use to manage users and groups within AWS SSO. At the same time, grant user permissions to resources in multiple AWS accounts and business applications

# Introduction

## How AWS Single Sign On Works?

AWS Single Sign On

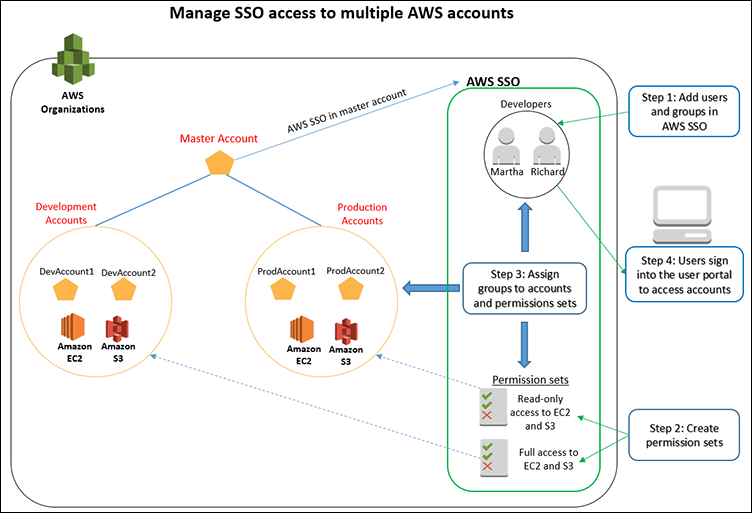
First of all, You have the option to create your users’ identities and groups in AWS SSO. Or, you can connect to your existing users and groups from Microsoft Active Directory Domain Services, Okta Universal Directory, Azure AD, or another standards-based identity provider.

## How to create and manage users within AWS Single Sign-On

SSO allows you to grant your users access to AWS resources, such as Amazon EC2 instances, across multiple AWS accounts. By default, AWS SSO now provides a directory that you can use to create users, organize them in groups, and set permissions across those groups.

Thus, you can also grant the users that you create in AWS SSO permissions to applications such Salesforce, Box, and Office 365.

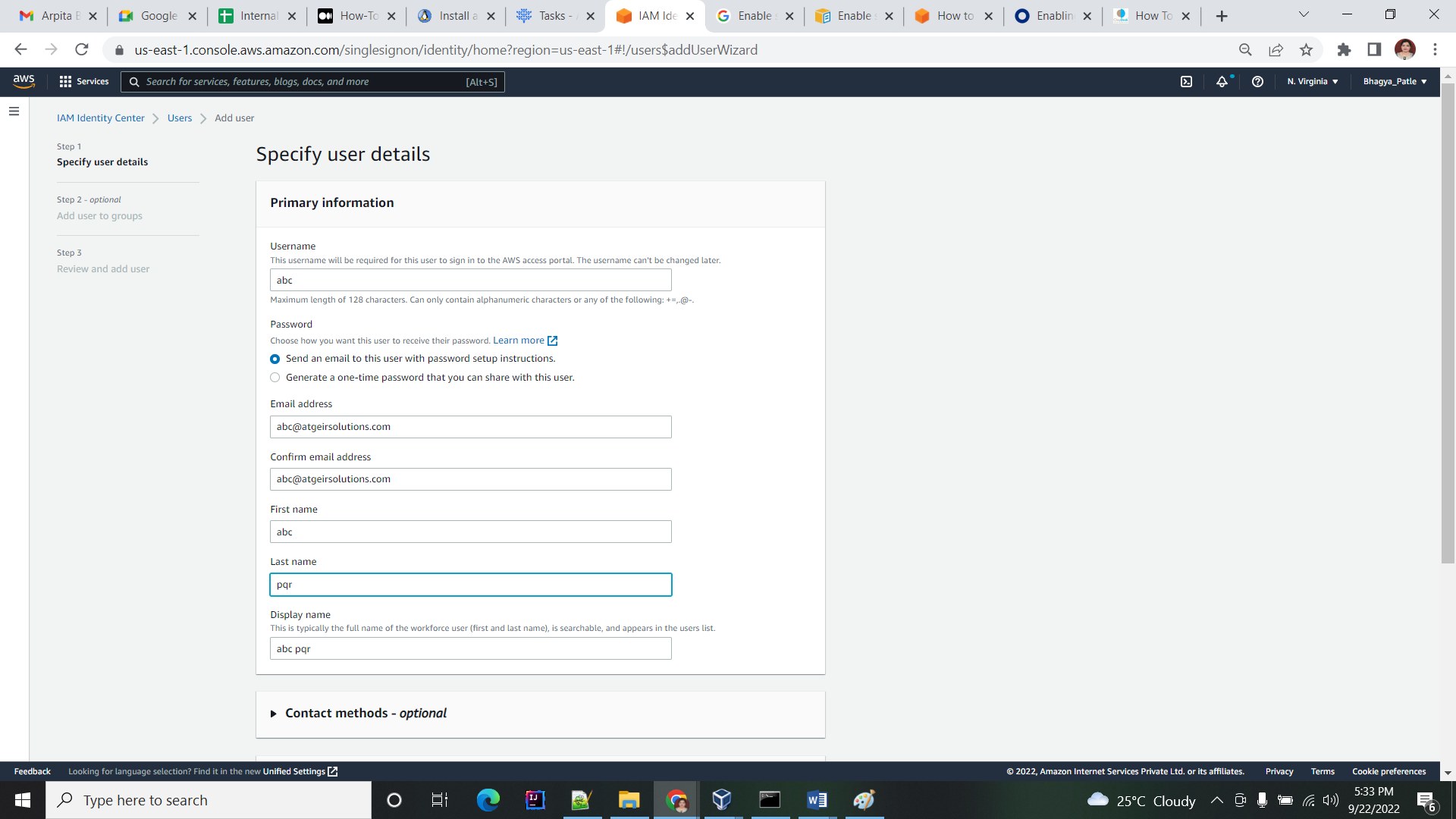
## Diagram Representation

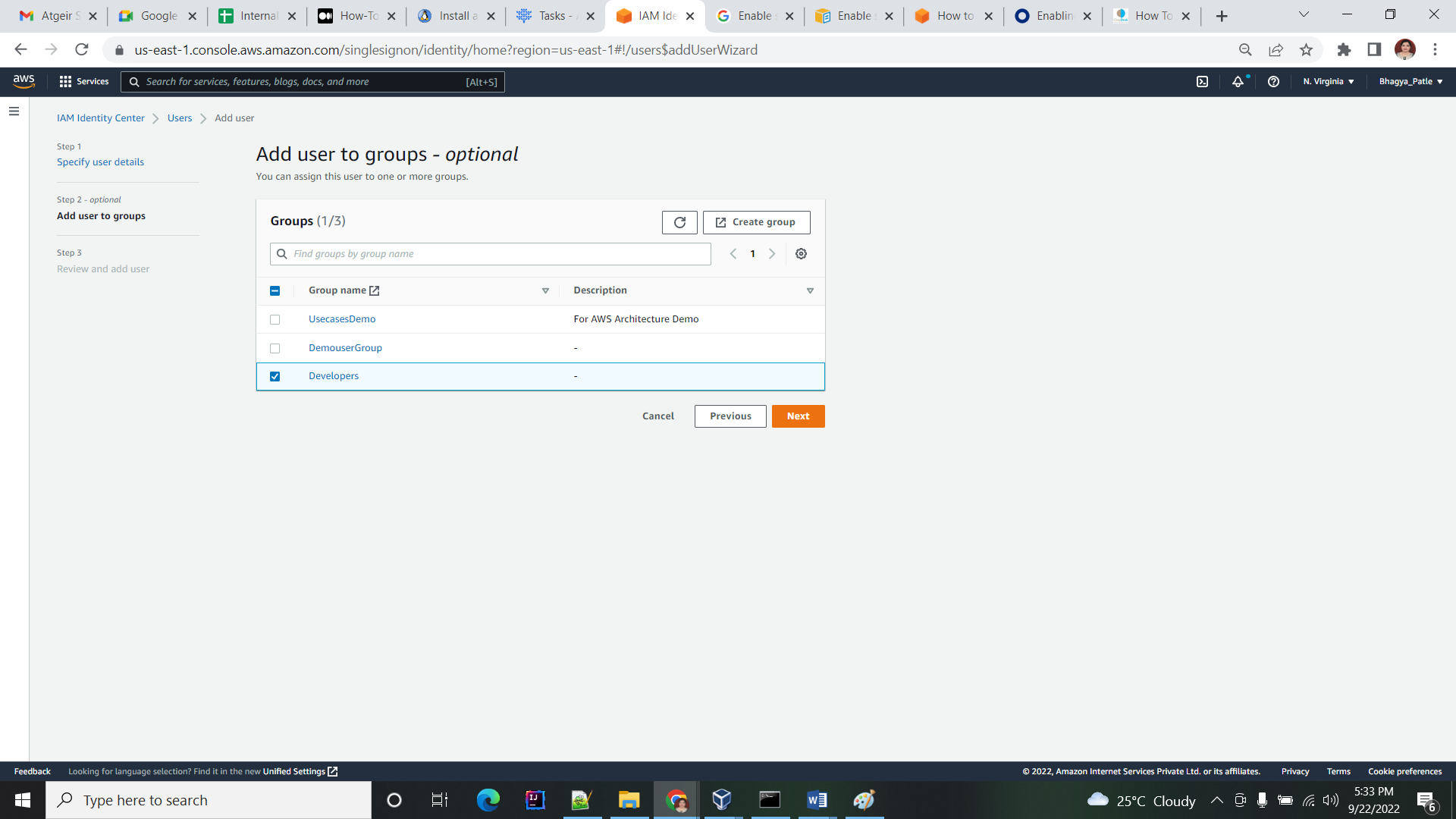


# Steps Enable single sign-on access to your AWS EC2 Windows instances:

1. Add users and groups in AWS SSO:

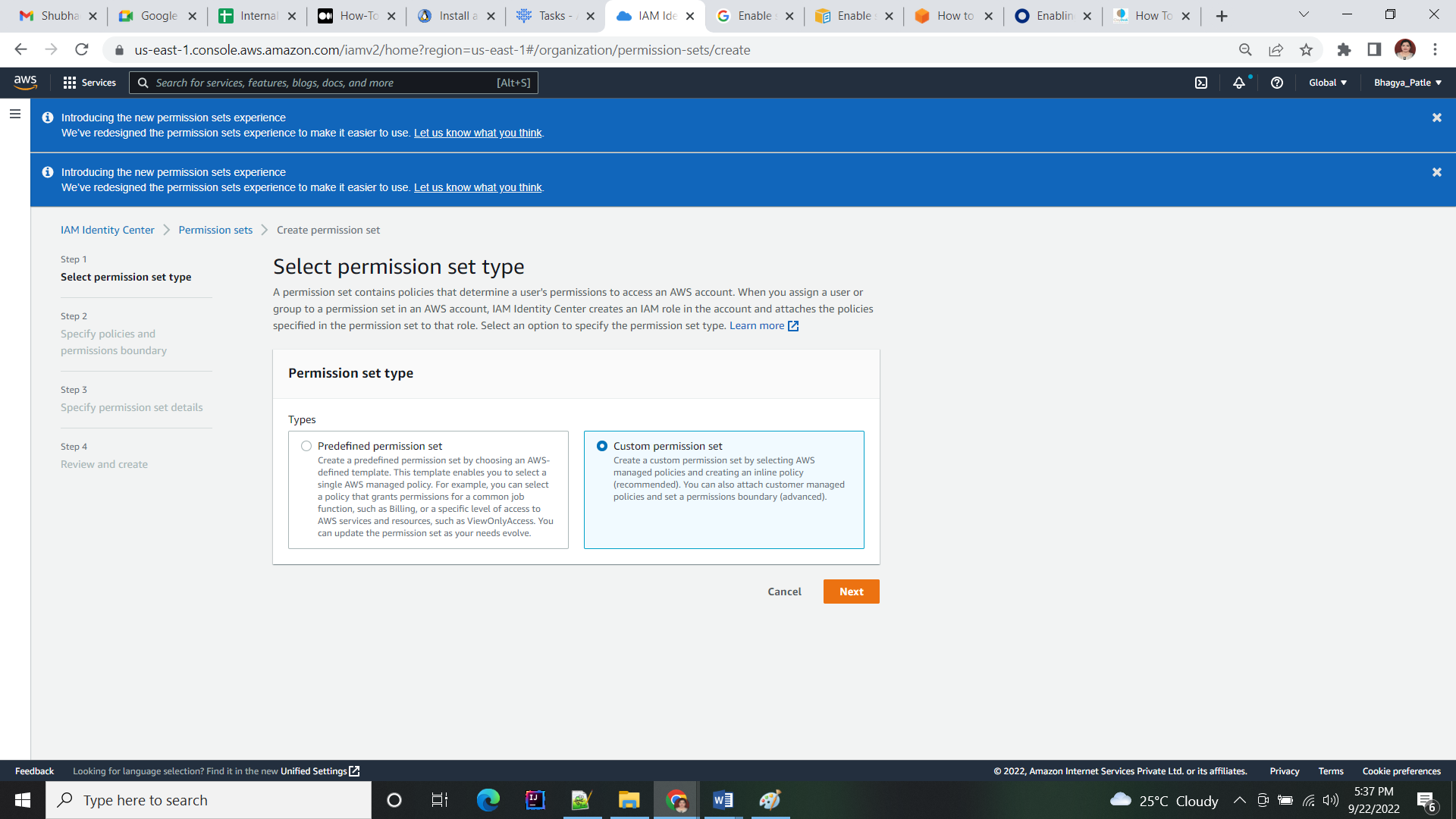
Add users cdd and bhagya in AWS SSO. Add a group called Developers in AWS SSO and add users cdd and bhagya to the Developers group.

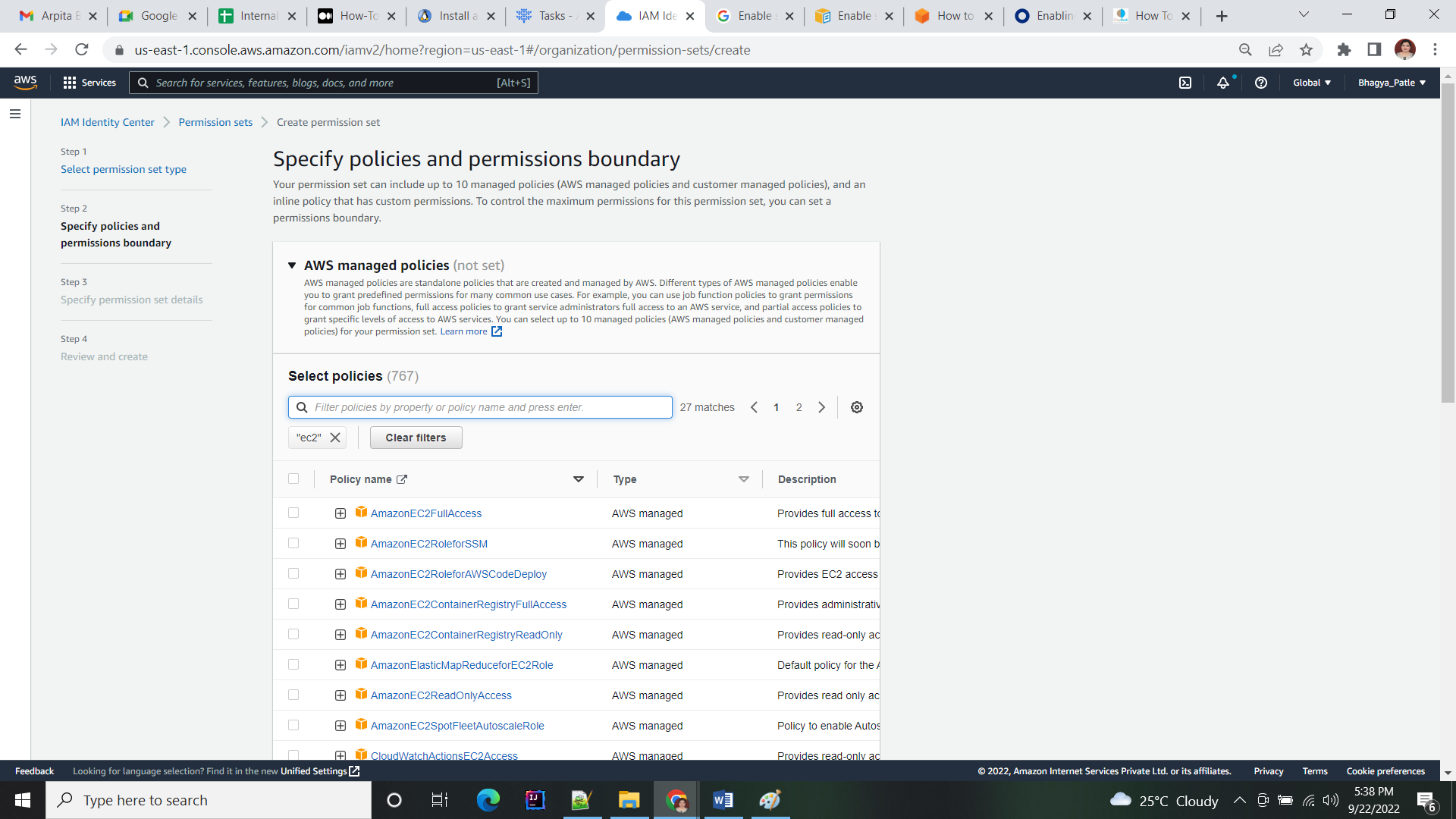


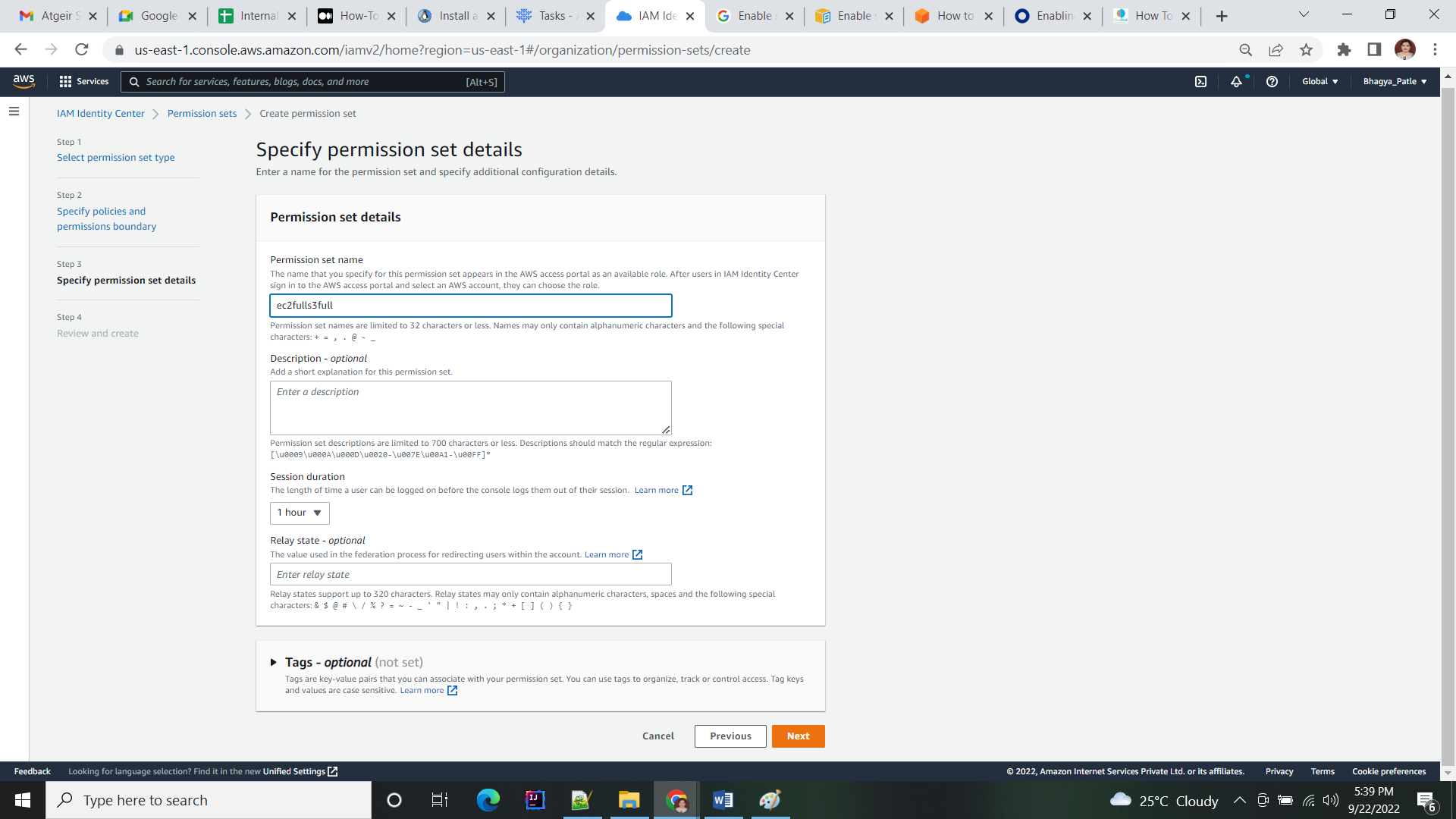


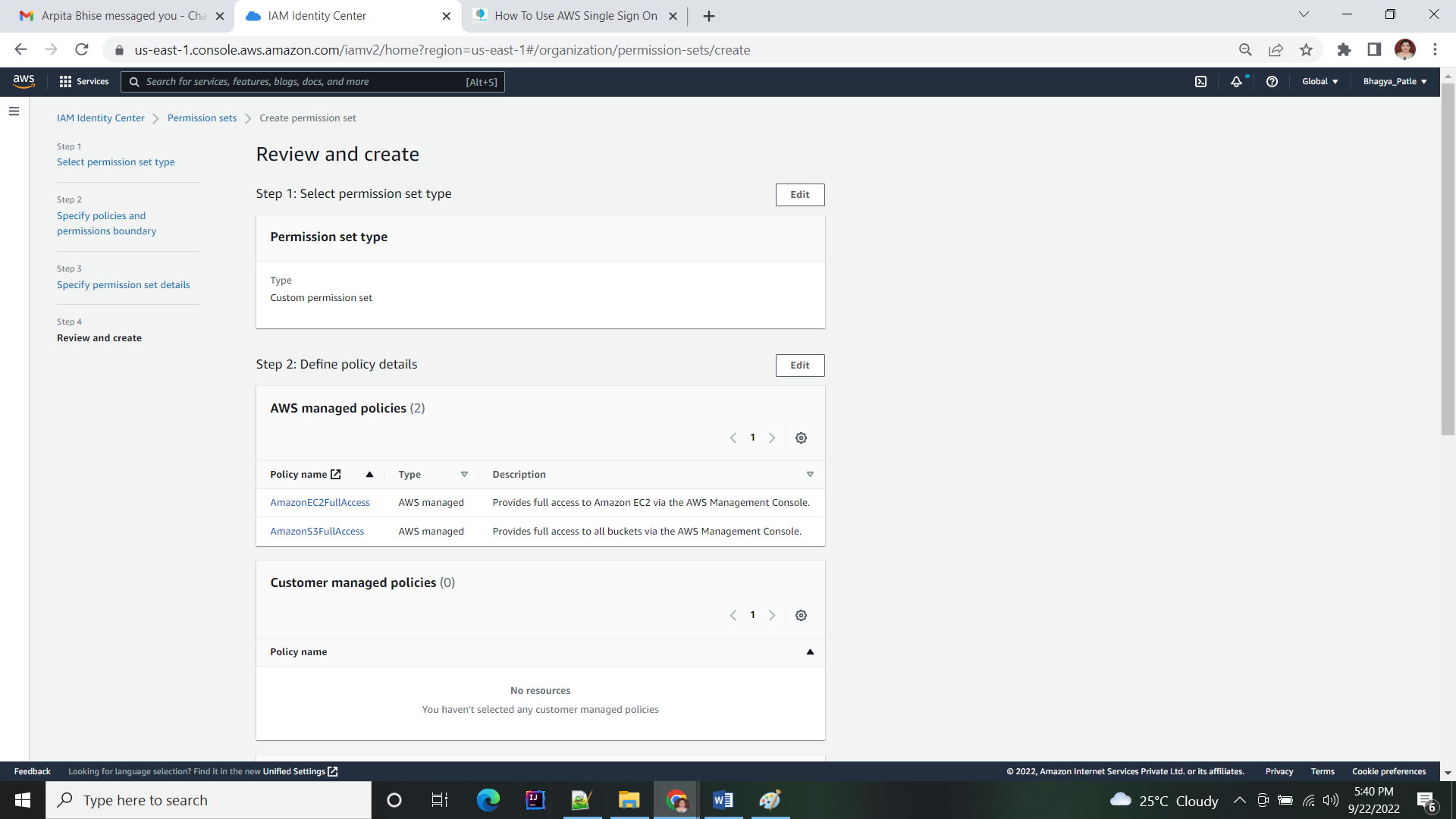
1. Create permission sets:

Create two permission sets. In the first permission set, include policies that give full access to Amazon EC2 and Amazon S3. In second permission set, include policies that give read-only access to Amazon EC2 and Amazon S3.

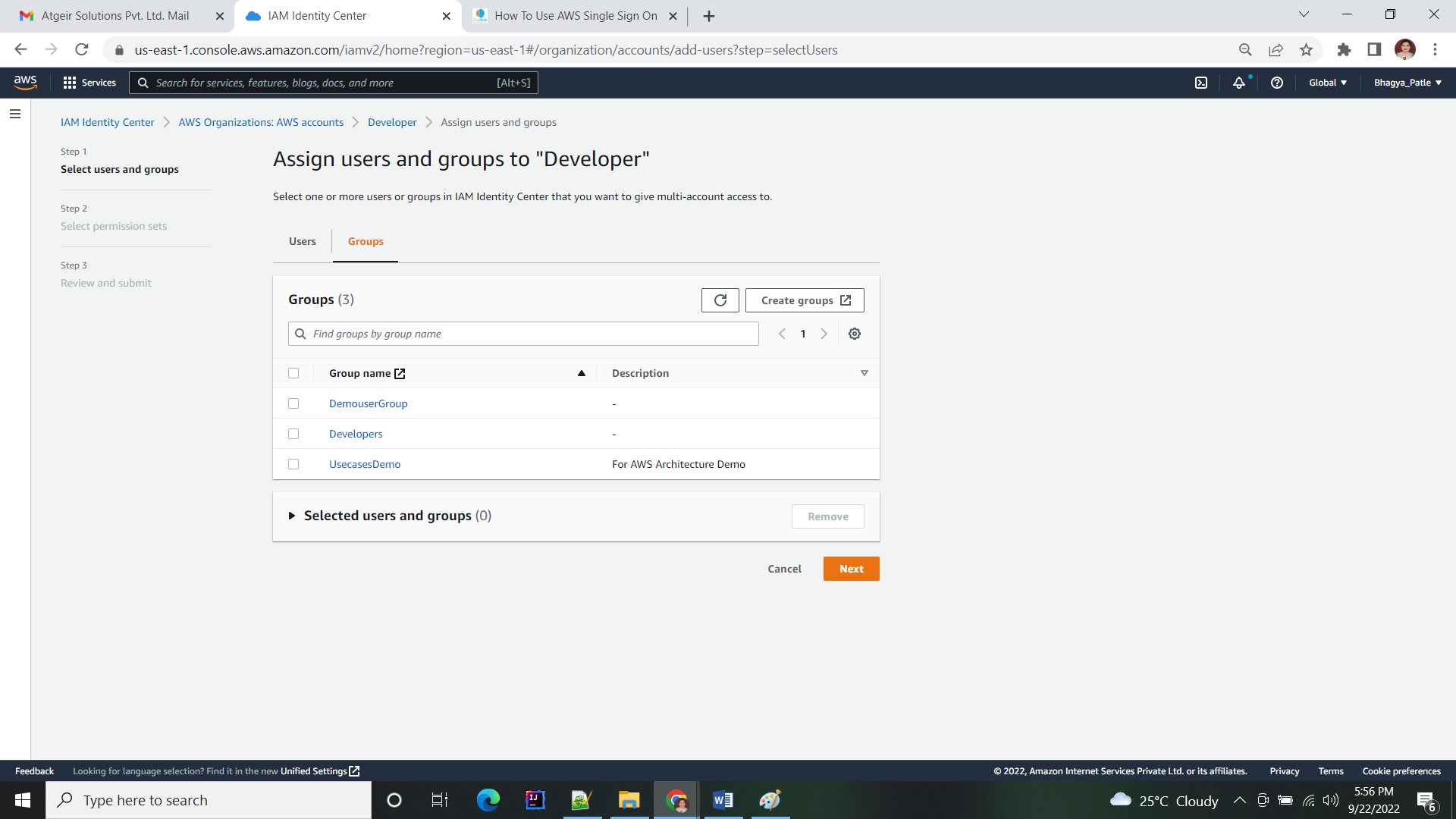






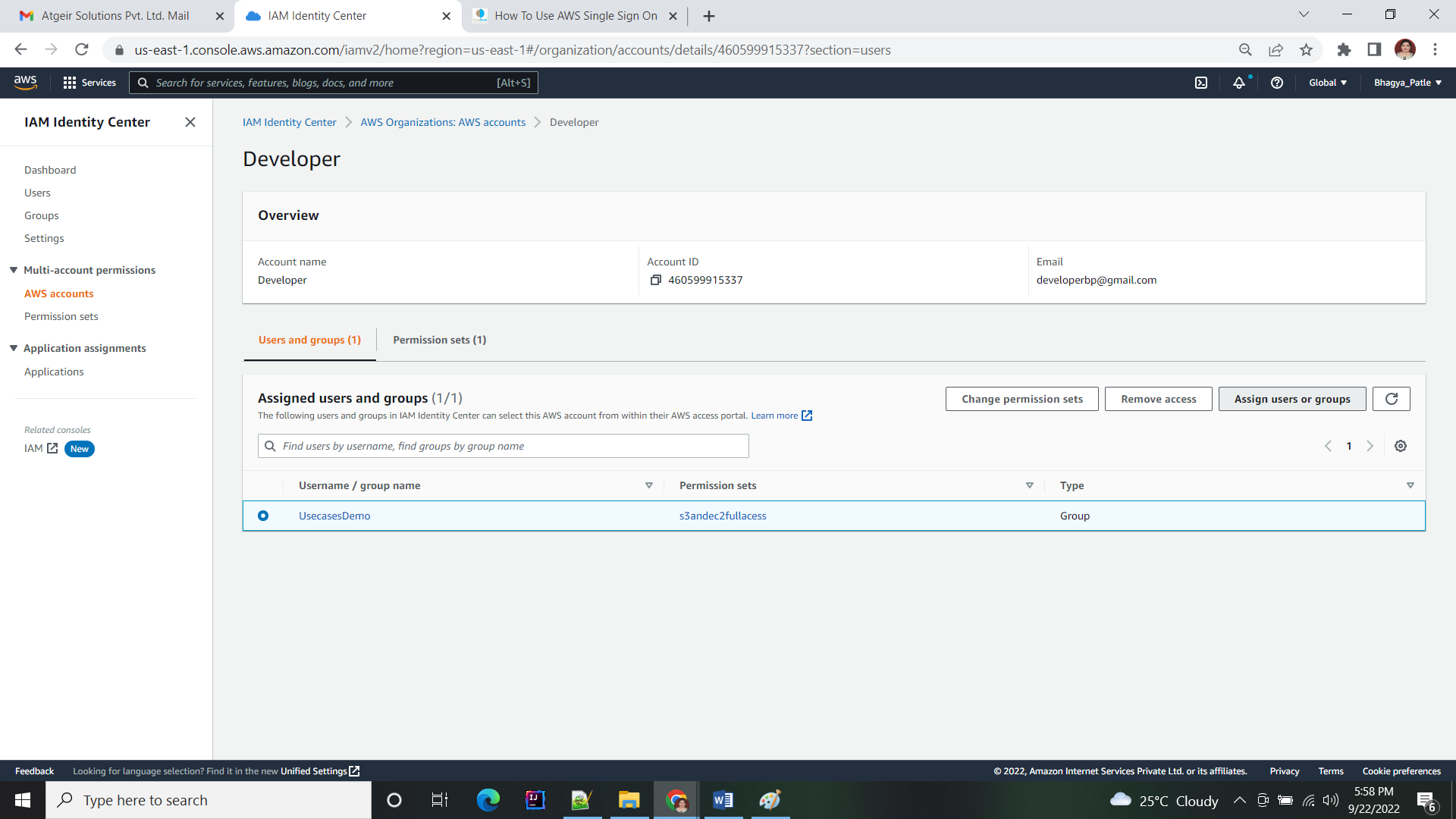


## Secondly, simply assign groups to accounts and permission sets:



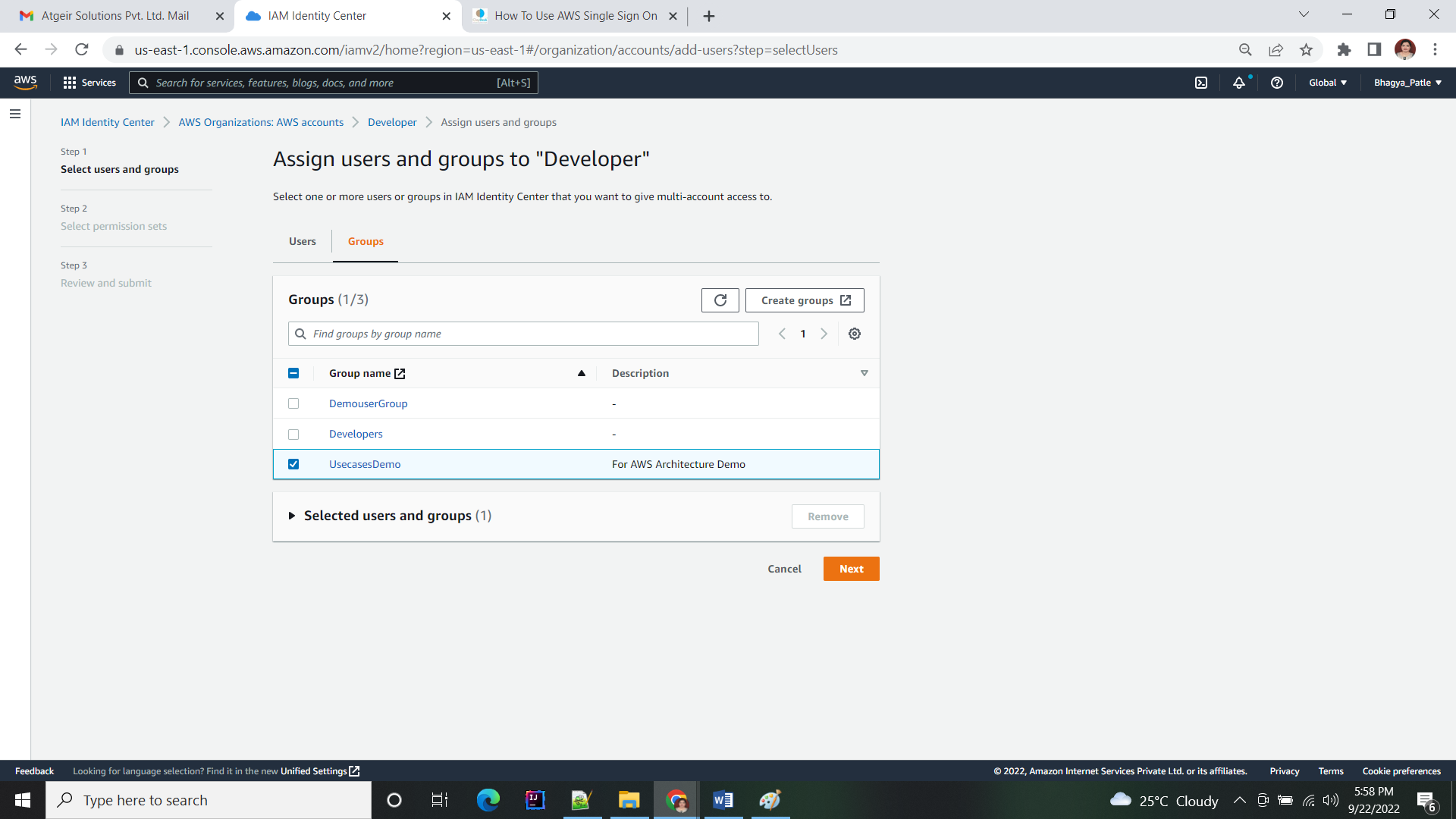
## Assign the group:

Assign the UsecaseDemo group to your developer accounts and assign the permission set.



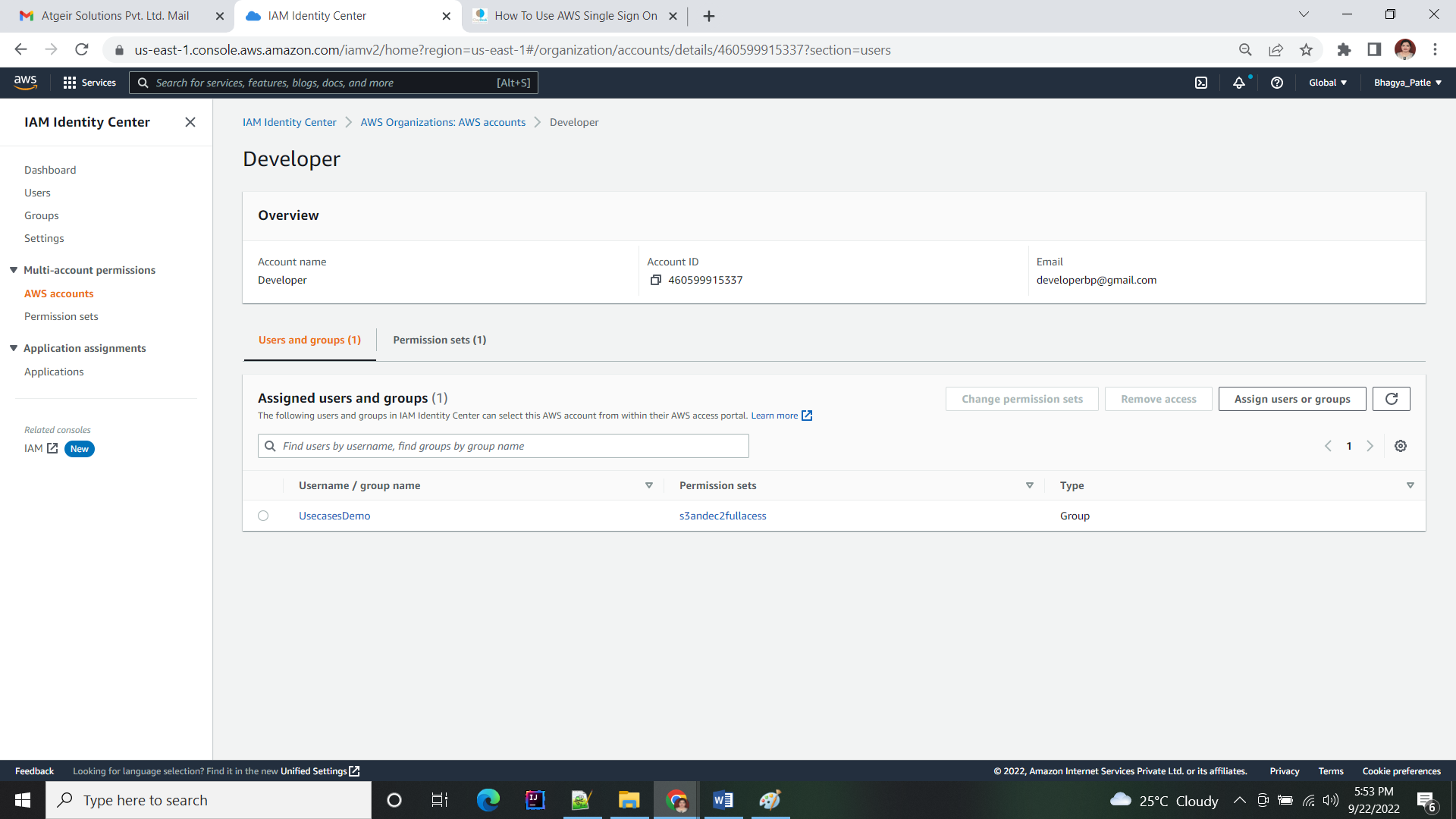
## Assign group to Account

Hence, next, assign the Developers group to your production accounts. Then assign the permission set that gives read-only access to Amazon EC2 and Amazon S3.



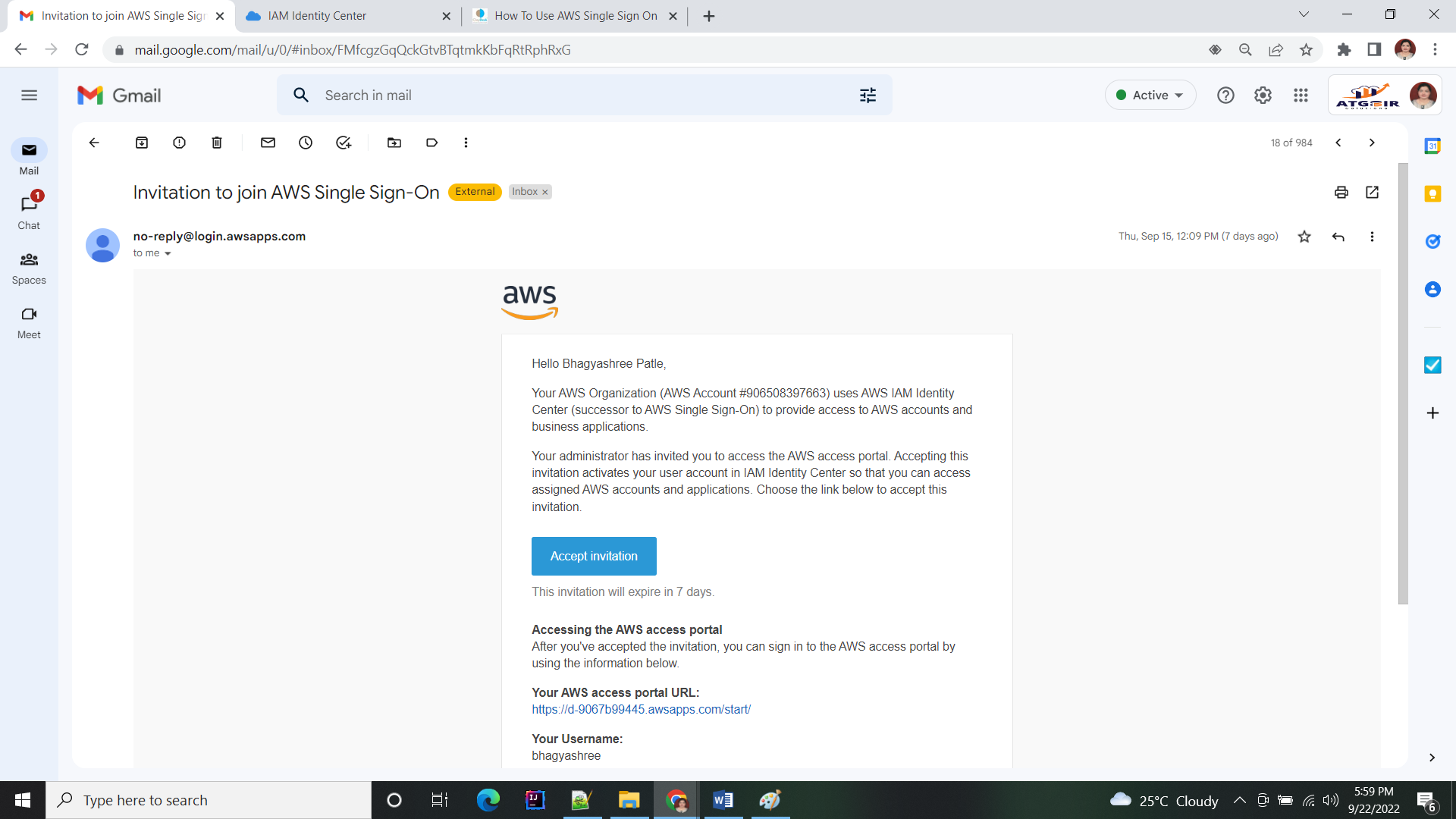
## Check the permission set to users

Basically, User now have full access to Amazon EC2 and Amazon S3 in the developer accounts. Also, they have read-only access in the production accounts.



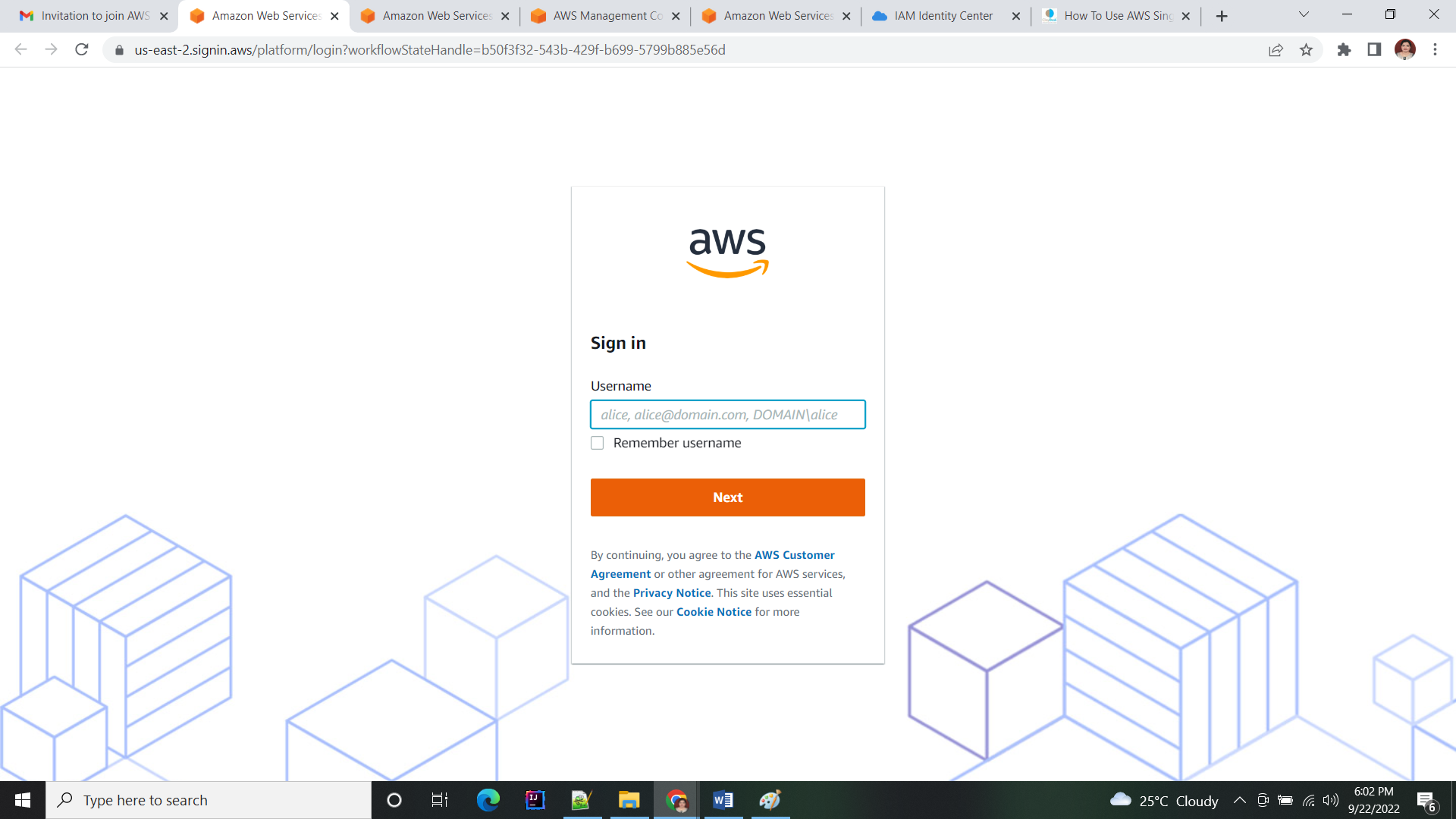
## Users sign into the User Portal to access accounts:

User receive email from AWS to set their passwords with AWS SSO.



## Sign in and Account UI:

Basically, User can now sign into the AWS SSO User Portal using their email addresses and the passwords they set with AWS SSO, allowing them to access their assigned AWS accounts.

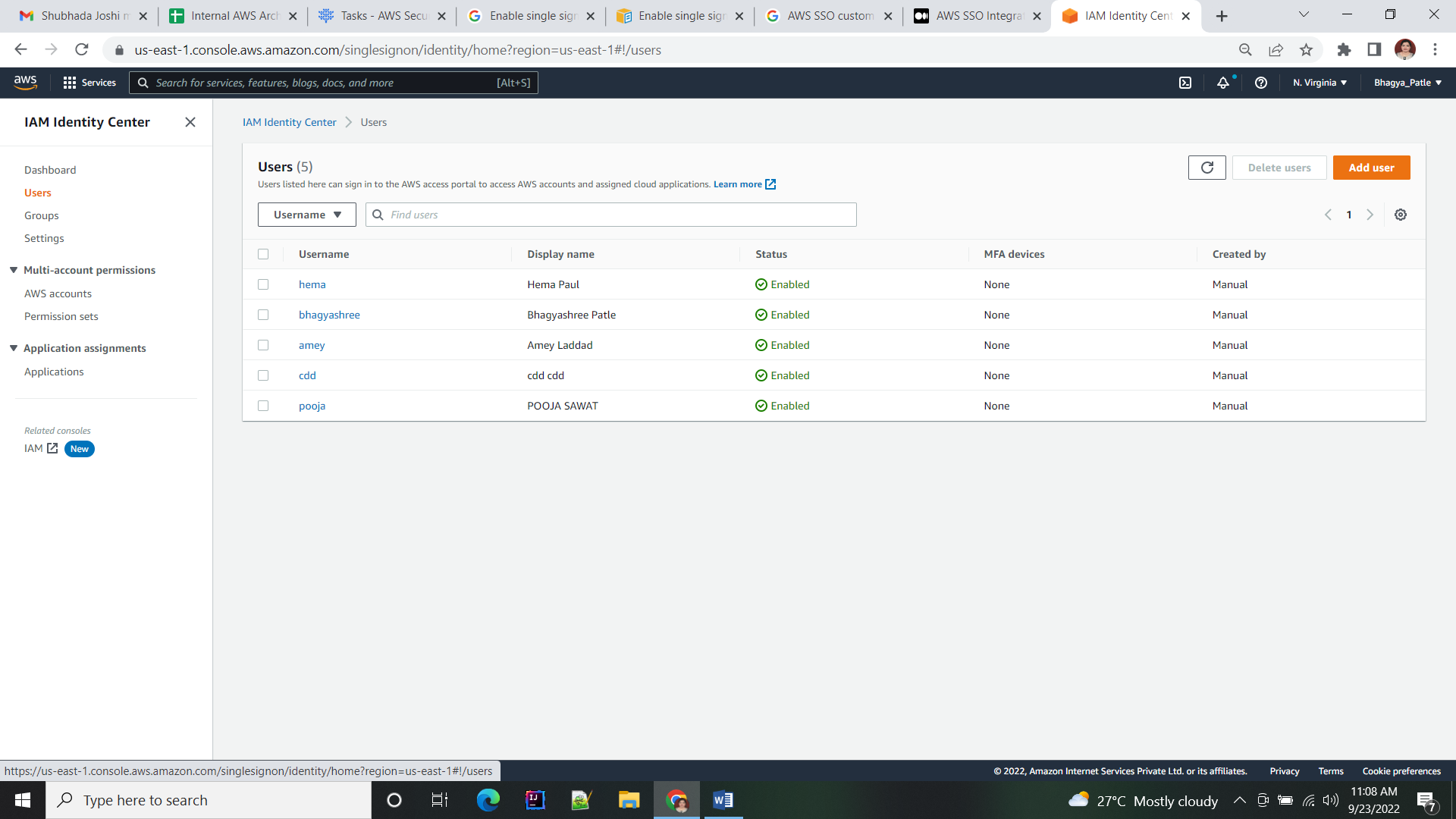




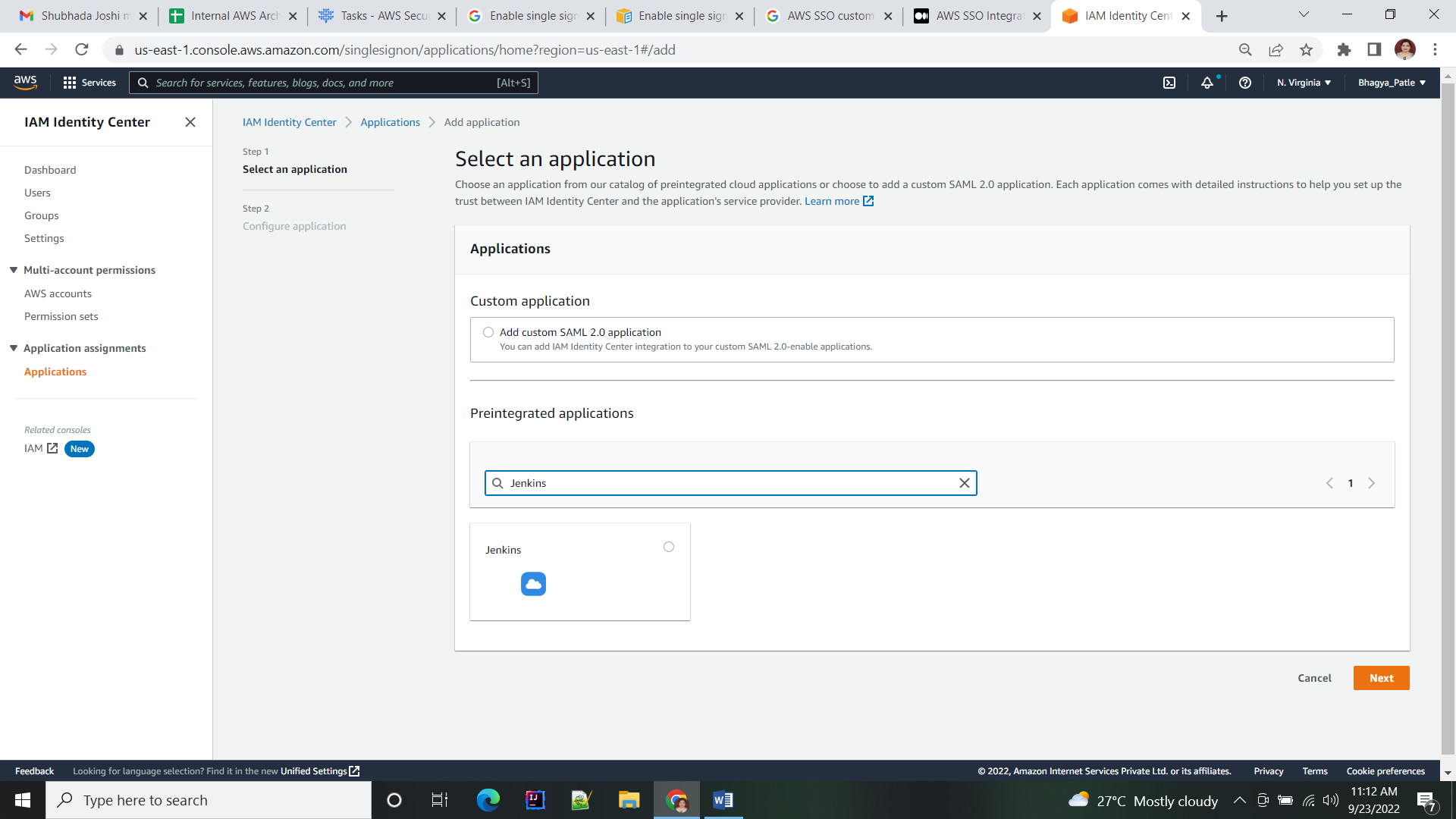
# Steps Enable single sign-on access to your AWS applications:

## Provision and configure AWS SSO

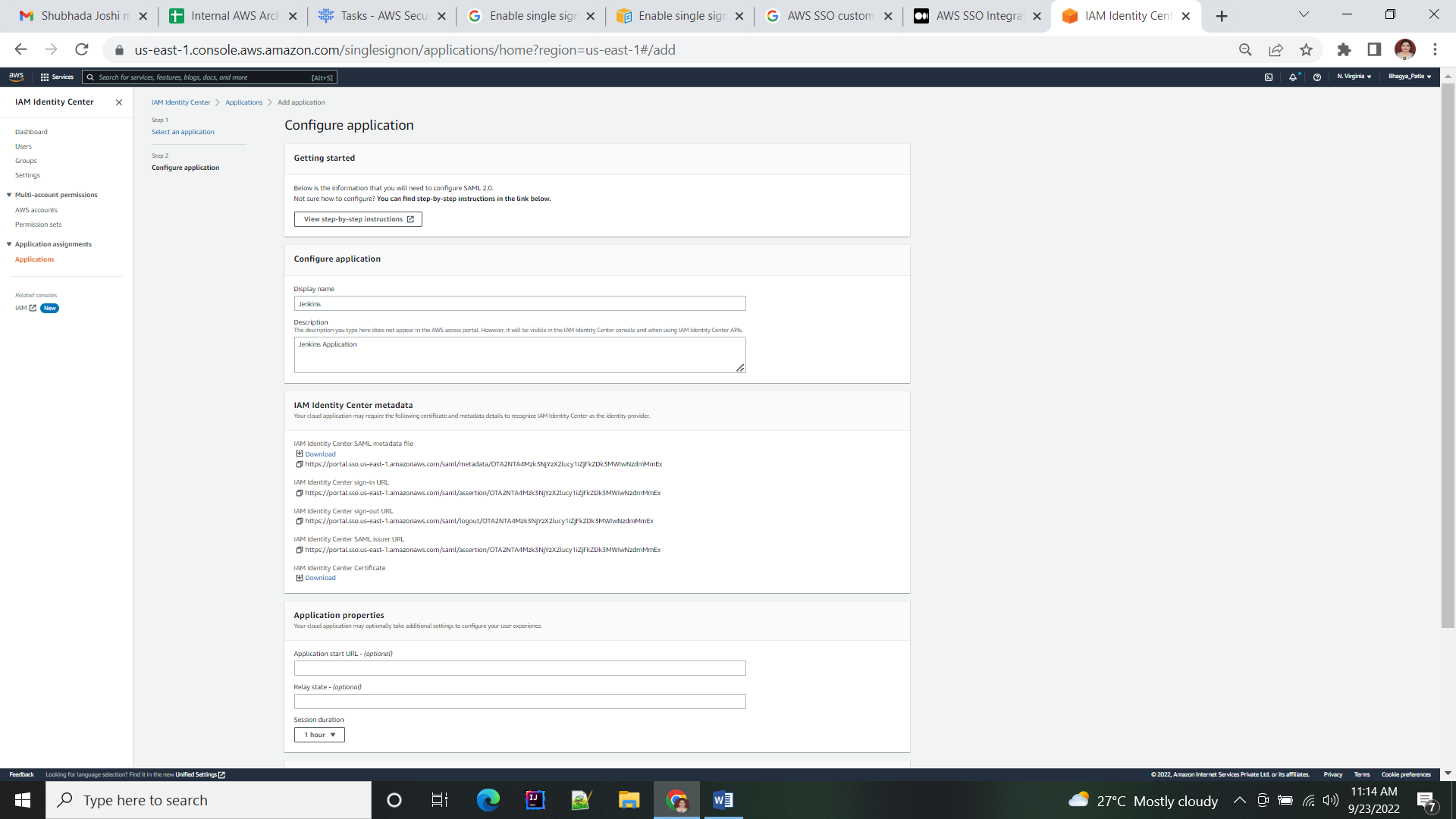
* Log in to your master account or organization root account.
* Enable AWS SSO with Enable single sign-on access to your AWS EC2 Windows instances steps.
* Once enabled, create a few users by going in the user section.



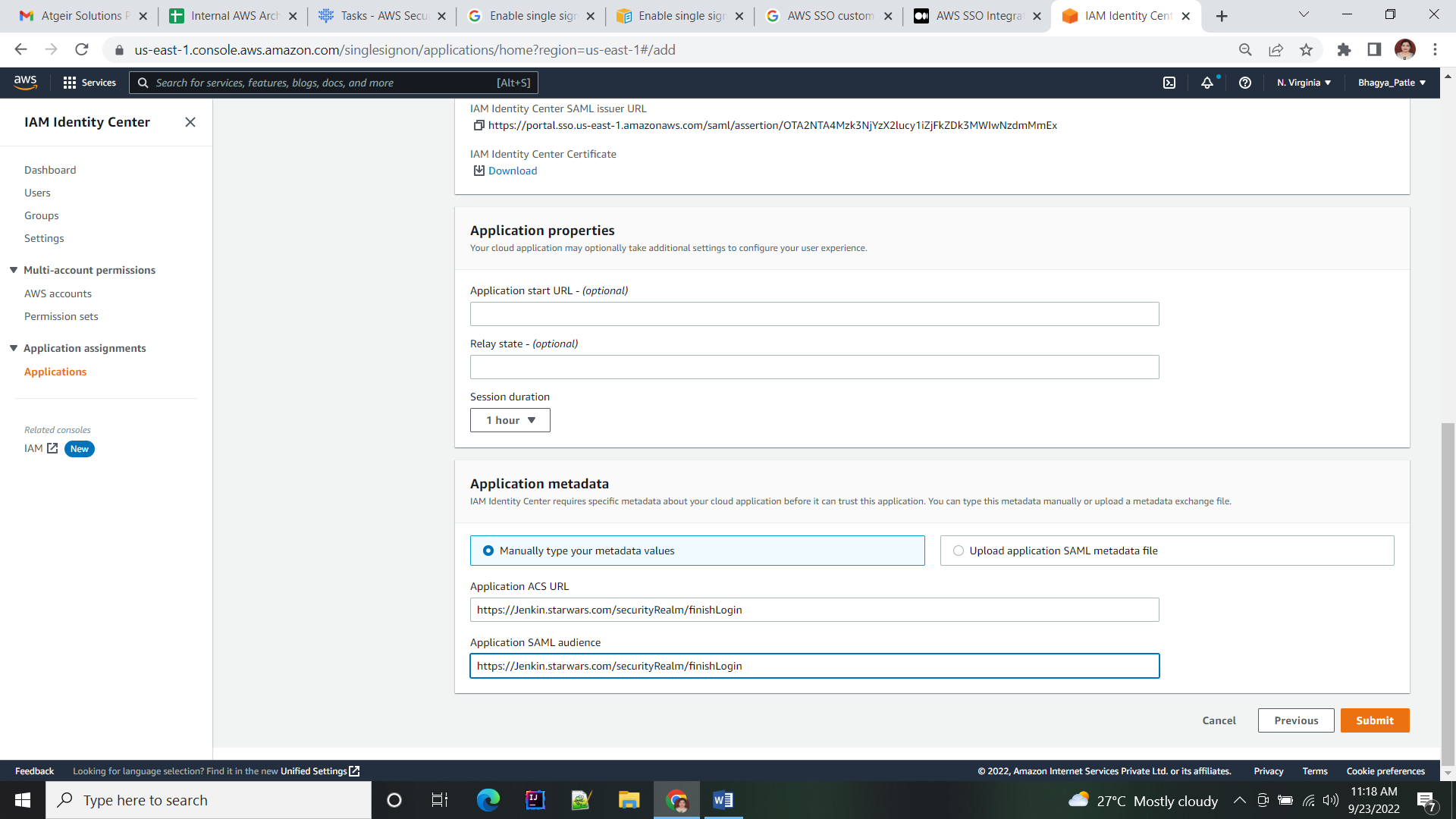
* Under the application section click on “Add a new application”, search for “Jenkins”



* Keep session duration less or equal to 1 hours, so that you can enforce users to re-authenticate after 1 hours.



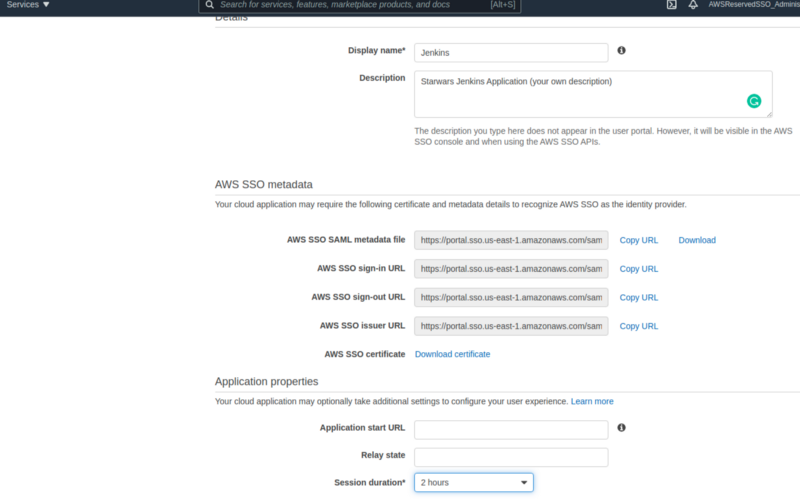
* Make sure you have enabled SSL certificate for Jenkins machine and opted for custom DNS. Replace “Application ACS URL” and “Application SAML audience” with your Jenkins URL example (https://<replace with you Jenkins url>/securityRealm/finishLogin)
* *Note: This ACS URL “https://Jenkin.starwars.com/securityRealm/finishLogin" will be used to redirect from AWS SSO once users get authenticated.*



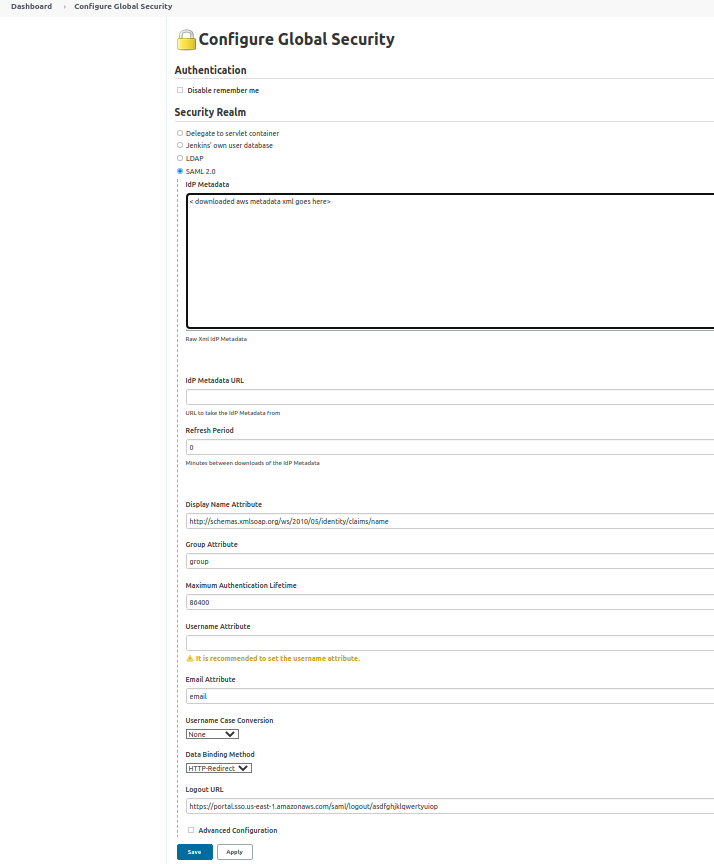
* Click on “Submit”

## Jenkins side SAML configuration

* Login to your Jenkins as administrator
* Goto manage Jenkins > install plugins
* Install the “SAML Plugin” and restart the Jenkins
* Goto manage Jenkins > Configure global security
* Under Security Realm select the SAML option
* From AWS SSO dashboard download XML metadata (click download next to AWS SSO metadata file)

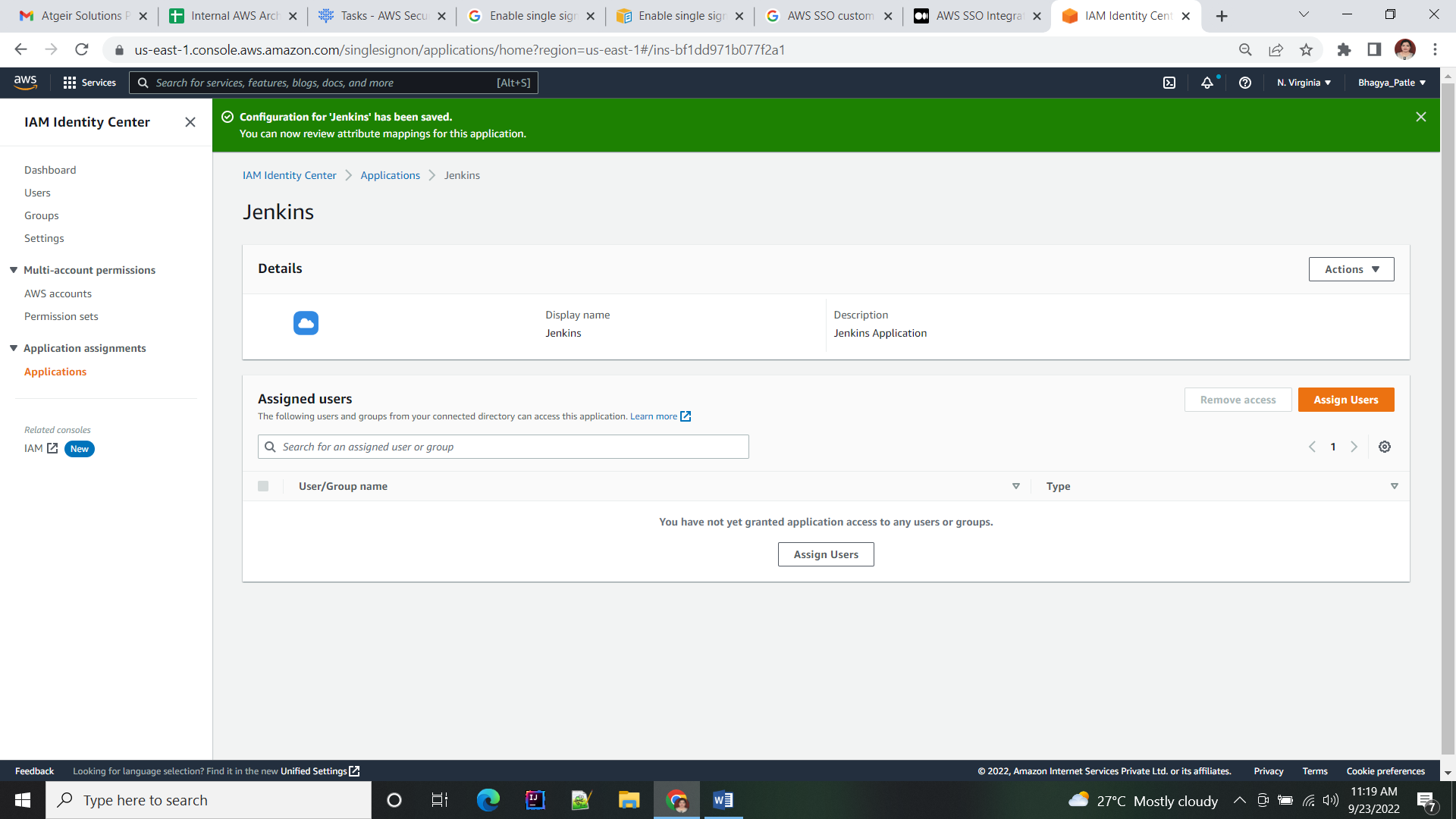


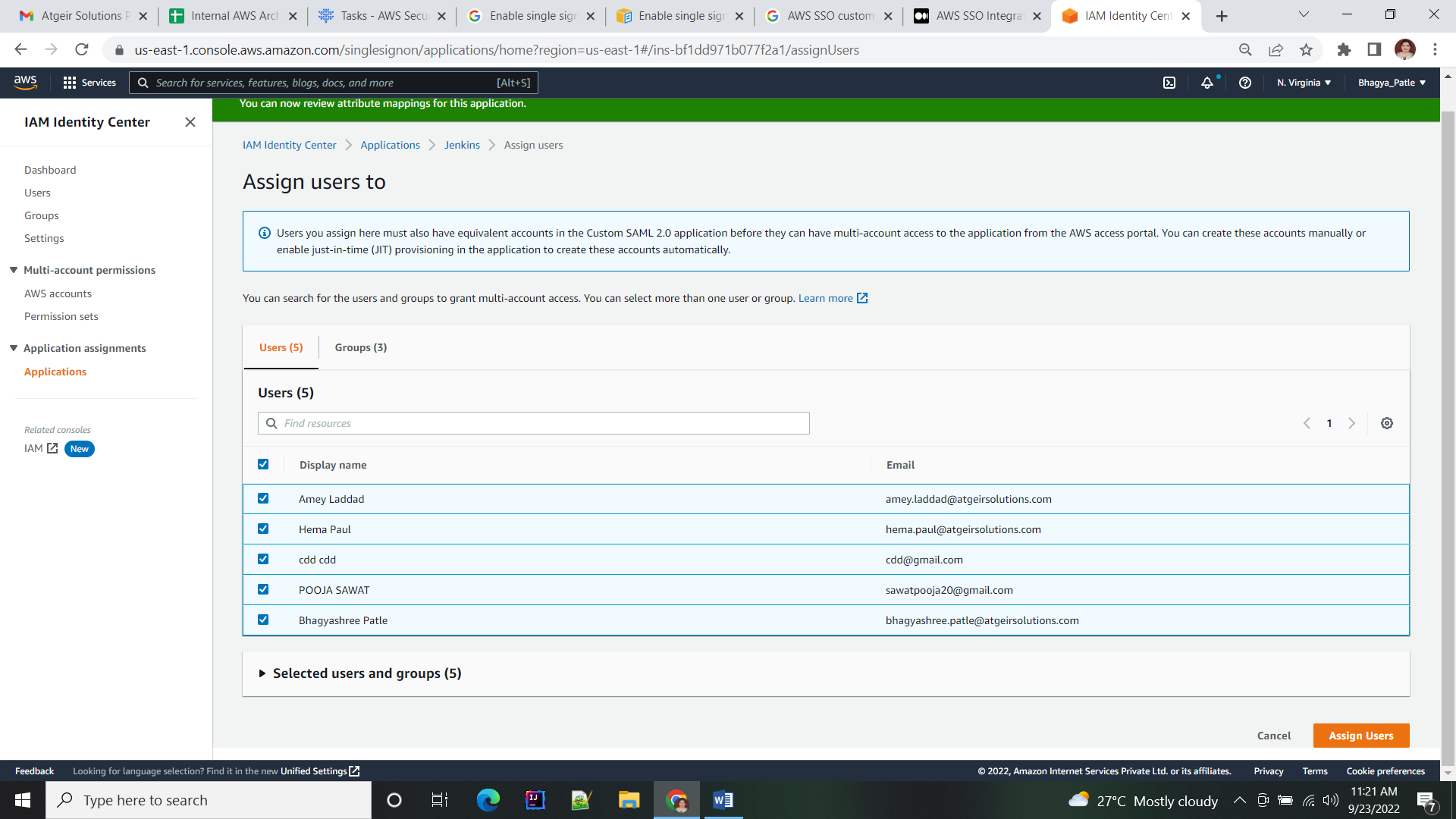
* Paste the XML content into the IDP metadata text box
* Specify Group attribute as “group”
* Enter Logout URL as “AWS SSO sign-out URL” so that once logged out from Jenkins you will land on AWS SSO URL.
* Leave the rest fields as-is.
* Reduce the maximum authentication lifetime to your convenience.
* Save and apply.
* If you missed any configuration and got locked out in Jenkins, disable authentication in the Jenkins configuration file and restart the Jenkins (restart the configuration again)
* The resulting configuration should look something like this:

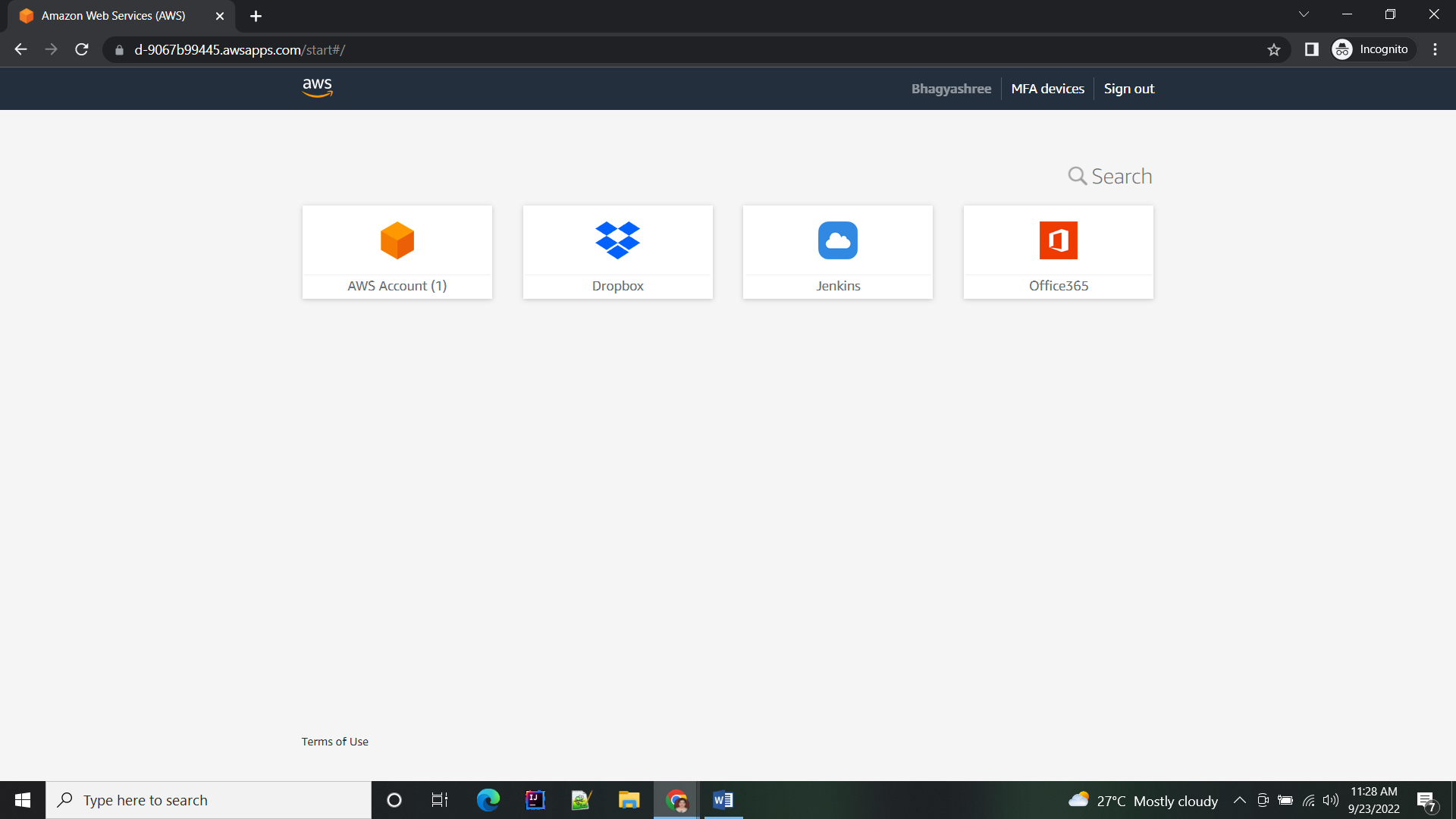


## Add user to Jenkins SSO application

* Go to the AWS SSO dashboard and select your application
* Switch to the “Assign users” button
* Select the users and/or groups that you want to be able to log into Jenkins.







## Add access privileges in Jenkins portal

* Login to your Jenkins as administrator
* Goto manage Jenkins > Configure global security
* Remove access to anonymous users by unselecting all the checkboxes
* Remove access to the authenticated user, add the group or users manually (in this case “Kylo Ren” user), and add appropriate access based on your requirement.

