

Practical - 4

Name: Sakshi Deshmukh

Roll no. 158

PRN: 202301040191

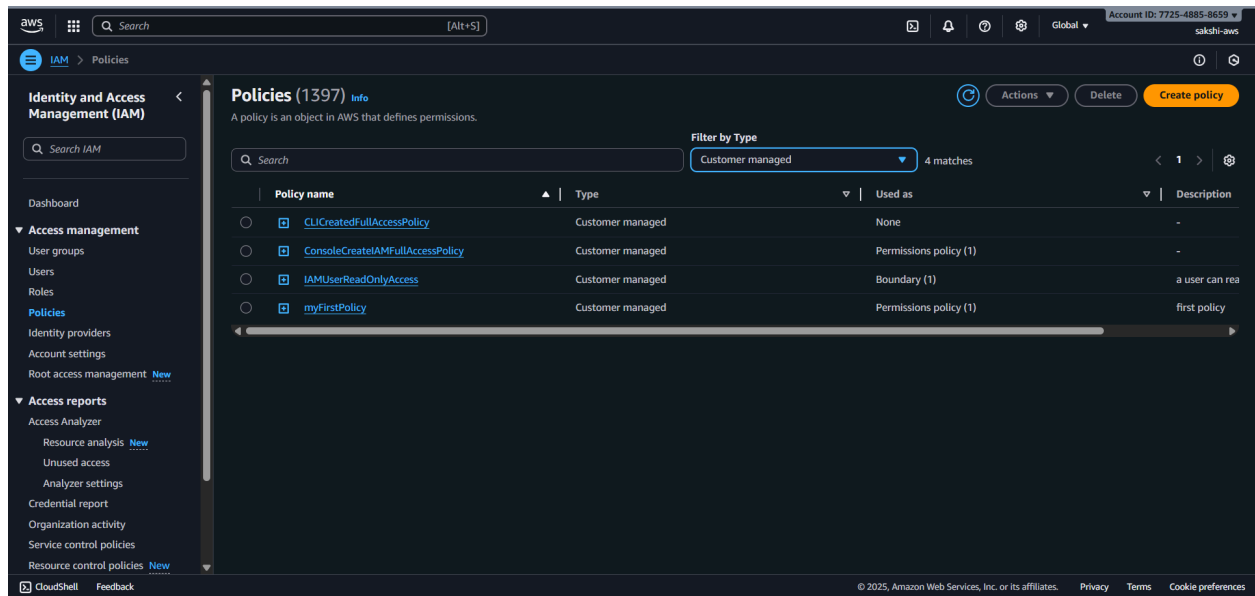
Permissions Boundary Configuration

In modern cloud environments, managing permissions at scale is a critical challenge. When development teams need to create resources for their applications, relying on a central security team for every IAM role creates a significant bottleneck. The AWS IAM Permissions Boundary is a feature designed to solve this problem by enabling safe delegation.

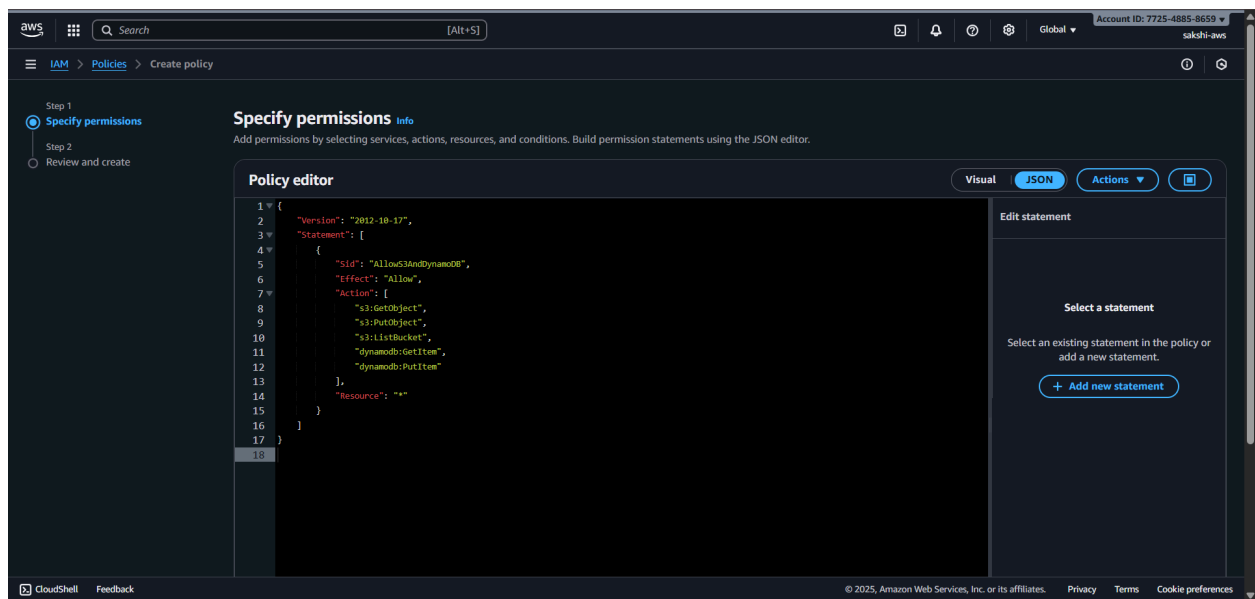
This practical demonstrates the primary use case for IAM Permissions Boundaries. We will simulate a real-world scenario where a central administrator empowers a developer to create IAM roles. The administrator will first create a "guardrail" policy (the Permissions Boundary) that defines the maximum possible permissions for any new role. They will then create a policy for the developer that allows them to create roles, but only if they attach the pre-defined boundary.

The objective is to show how the effective permissions of the new role become the intersection of the broad policy the developer attaches and the restrictive boundary set by the admin. This ensures that even if a developer tries to create a role with administrative privileges, its actual capabilities are safely limited by the boundary, thus maintaining security while enabling developer agility.

Admin View: Starting in the IAM Policies console to create two policies.



Creating WebAppBoundary, the policy that will act as our permissions guardrail.



Naming and reviewing the WebAppBoundary policy before creation.

The screenshot shows the 'Review and create' step in the AWS IAM console. The left sidebar indicates 'Step 1: Specify permissions' and 'Step 2: Review and create'. The main content area is titled 'Review and create' and includes a 'Policy details' section with a 'Policy name' field containing 'WebAppBoundary' and a 'Description - optional' field. Below this is a 'Permissions defined in this policy' section with a search bar and a table showing permissions for 'DynamoDB'.

Policy details

Policy name
Enter a meaningful name to identify this policy.
WebAppBoundary
Maximum 128 characters. Use alphanumeric and '+-=,@_.' characters.

Description - optional
Add a short explanation for this policy.
Maximum 1,000 characters. Use alphanumeric and '+-=,@_.' characters.

Permissions defined in this policy

Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it

Search

Allow (2 of 448 services) Show remaining 446 services

Service	Access level	Resource	Request condition
DynamoDB	Limited: Read, Write	All resources	None

Confirming WebAppBoundary is created and available in the policy list

The screenshot shows the 'Policies (1/1398)' list in the AWS IAM console. The left sidebar shows the 'Identity and Access Management (IAM)' section. The main content area displays a table of policies, with 'WebAppBoundary' highlighted. The table columns are 'Policy name', 'Type', 'Used as', and 'Description'.

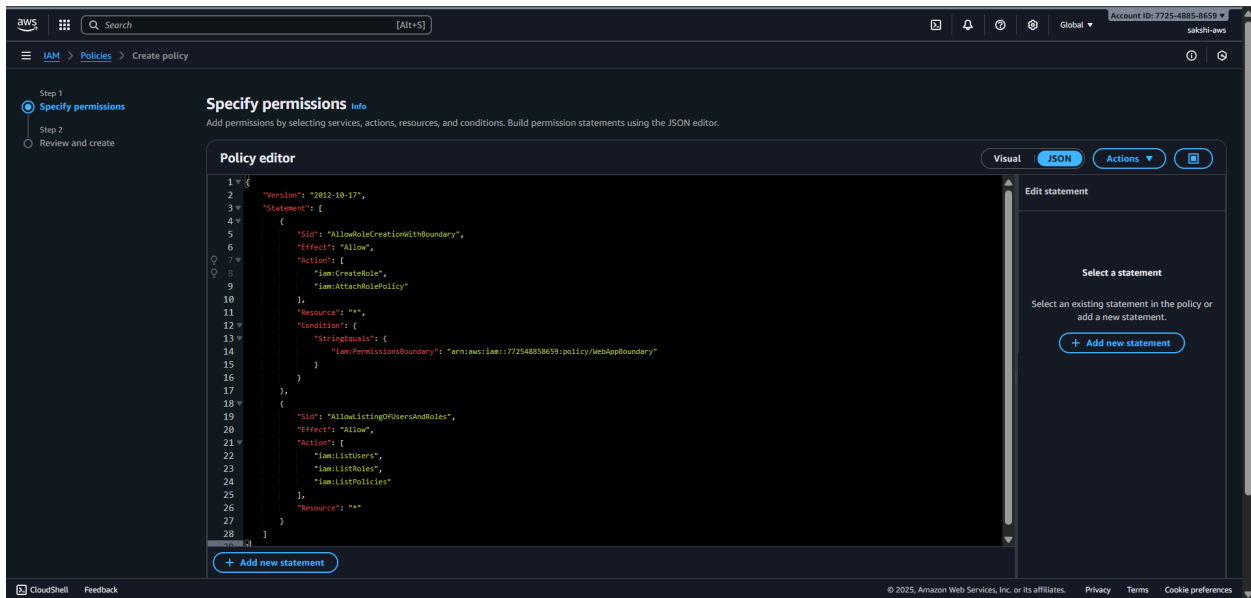
Policies (1/1398)

A policy is an object in AWS that defines permissions.

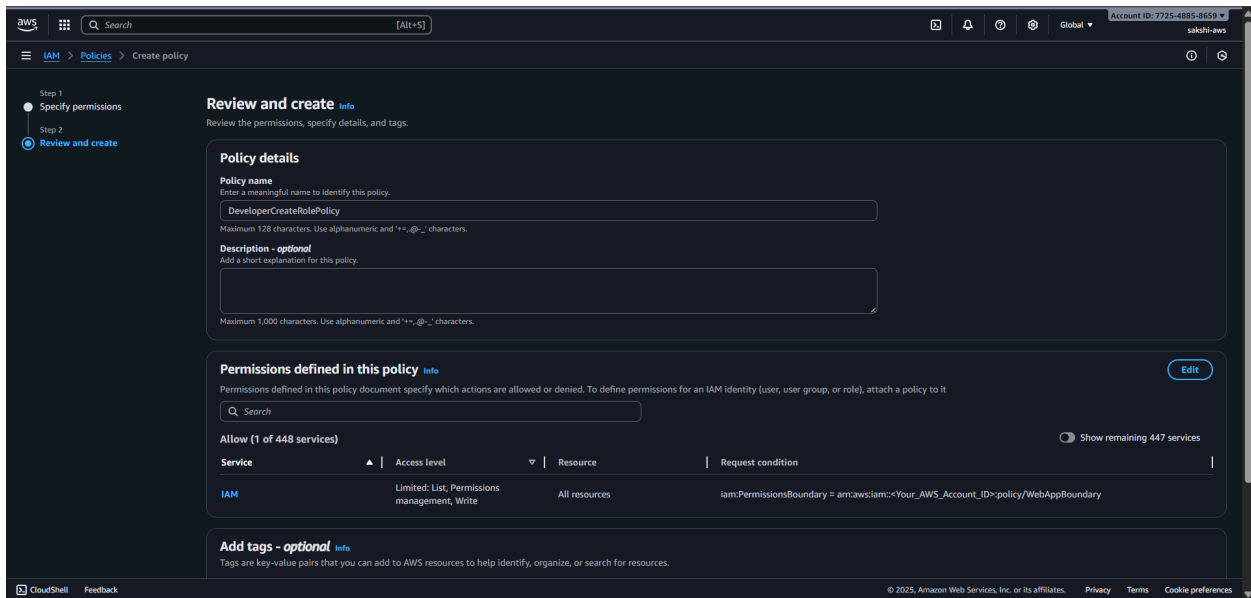
Filter by Type: Customer managed (5 matches)

Policy name	Type	Used as	Description
CLICreatedFullAccessPolicy	Customer managed	None	-
ConsoleCreatedIAMFullAccessPolicy	Customer managed	Permissions policy (1)	-
IAMUserReadOnlyAccess	Customer managed	Boundary (1)	a user can read and list
myFirstPolicy	Customer managed	Permissions policy (1)	first policy
WebAppBoundary	Customer managed	None	-

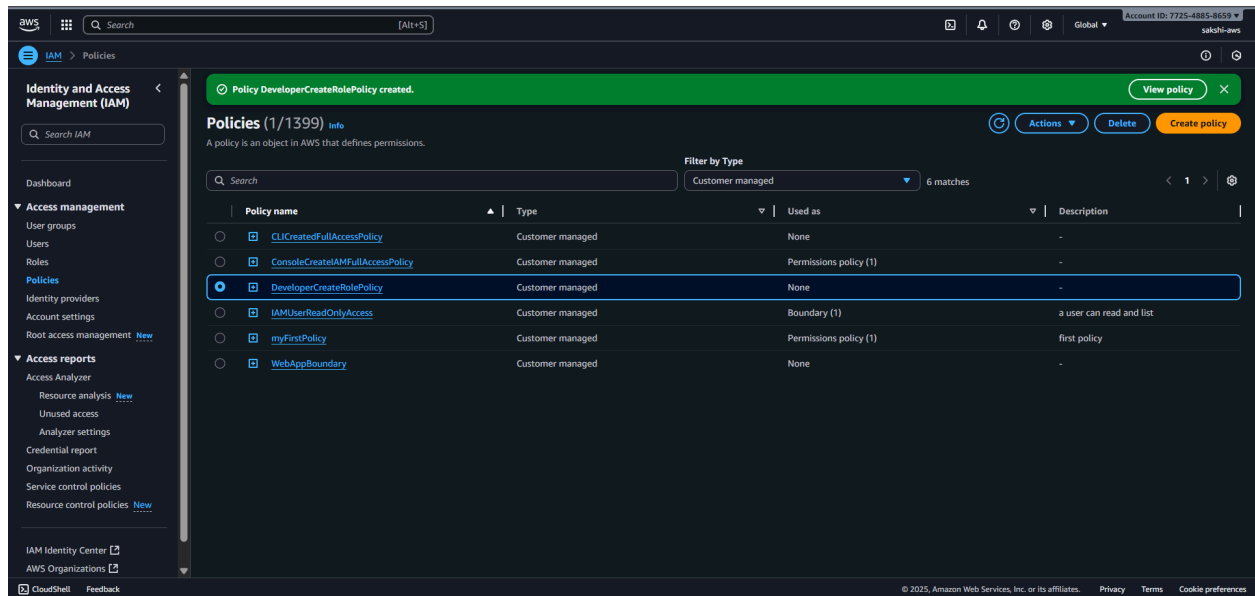
Creating DeveloperCreateRolePolicy to grant, yet control, developer actions.



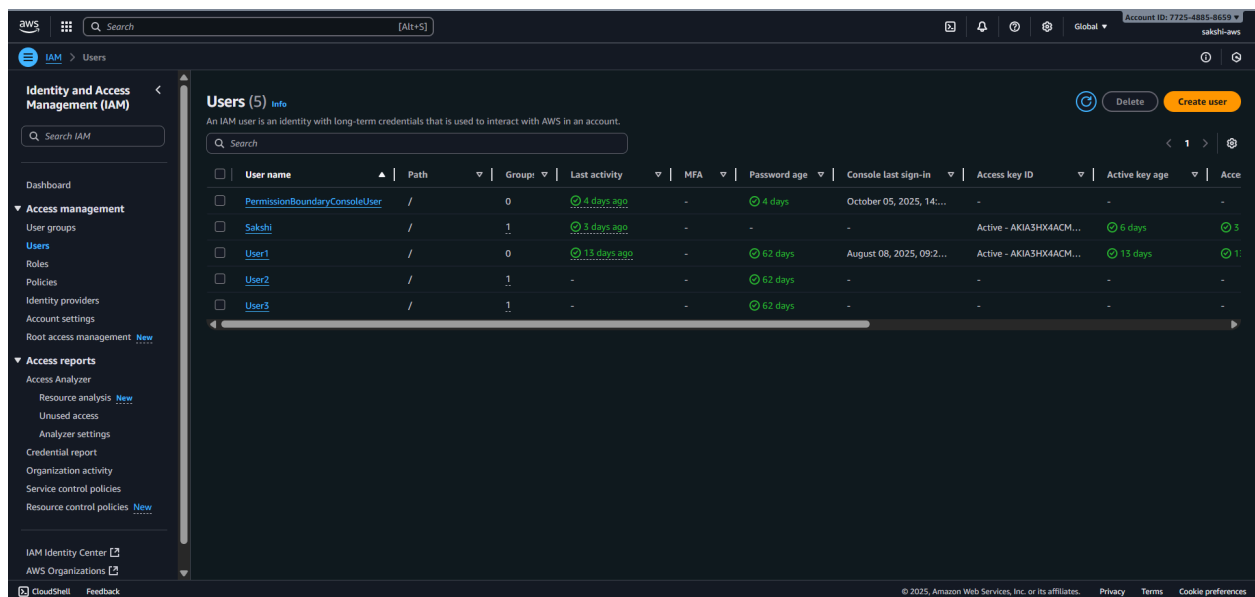
Reviewing the Condition that forces developers to use our boundary.



Admin prep complete: both WebAppBoundary and DeveloperCreateRolePolicy exist.



Admin View: Navigating to the IAM Users list before adding the developer.



Confirmation: MyWebAppRole is created and its permissions are now limited.

Role MyWebAppRole created with a few errors. See error description below.

Roles (7) Info

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.

<input type="checkbox"/>	Role name	Trusted entities	Last activity
<input type="checkbox"/>	AWSServiceRoleForResourceExplorer	AWS Service: resource-explorer-2 (Service-Linkage)	-
<input type="checkbox"/>	AWSServiceRoleForSupport	AWS Service: support (Service-Linkage)	-
<input type="checkbox"/>	AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service-Linkage)	-
<input type="checkbox"/>	CLITestRole	Account: 772548858659	-
<input type="checkbox"/>	IAMFullAccessForUser	Account: 772548858659	-
<input type="checkbox"/>	MyWebAppRole	AWS Service: ec2	-
<input type="checkbox"/>	S3FullAccessForSakshi	Account: 692977928139	-

Roles Anywhere Info

Authenticate your non AWS workloads and securely provide access to AWS services.

Access AWS from your non AWS workloads

Operate your non AWS workloads using the same authentication and authorization strategy that you use within AWS.

X.509 Standard

Use your own existing PKI infrastructure or use AWS Certificate Manager Private Certificate Authority to authenticate identities.

Temporary credentials

Use temporary credentials with ease and benefit from the enhanced security they provide.

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Defining the app-developer user and enabling console access.

Specify user details

User details

User name

app-developer

The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and +, -, @, _ (hyphen)

☒ Provide user access to the AWS Management Console - optional

If you're providing console access to a person, it's a best practice to manage their access in IAM Identity Center.

Are you providing console access to a person?

User type

☐ Specify a user in Identity Center - Recommended

We recommend that you use Identity Center to provide console access to a person. With Identity Center, you can centrally manage user access to their AWS accounts and cloud applications.

☒ I want to create an IAM user

We recommend that you create IAM users only if you need to enable programmatic access through access keys, service-specific credentials for AWS CodeCommit or Amazon Keyspaces, or a backup credential for emergency account access.

Console password

☒ Autogenerated password

You can view the password after you create the user.

☐ Custom password

Enter a custom password for the user.

Must be at least 8 characters long

Must include at least three of the following mix of character types: uppercase letters (A-Z), lowercase letters (a-z), numbers (0-9), and symbols (! @ # \$ % ^ & * () _ + - { } | ' " ~)

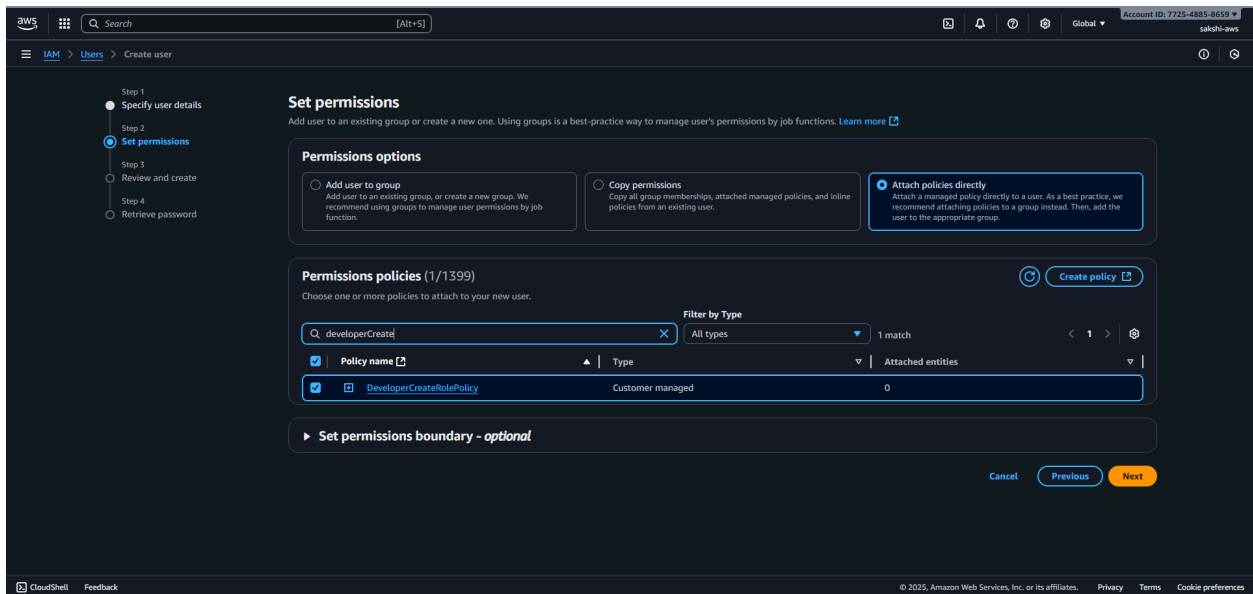
☐ Show password

☐ Users must create a new password at next sign-in - Recommended

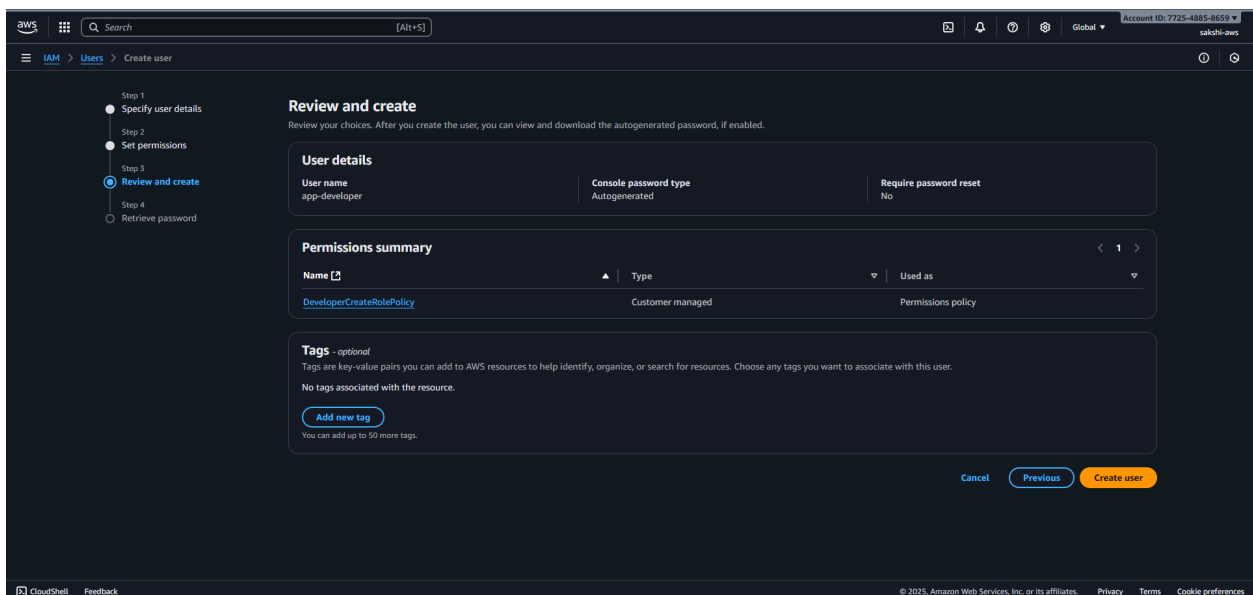
Users automatically get the IAMUserChangePassword policy to allow them to change their own password.

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

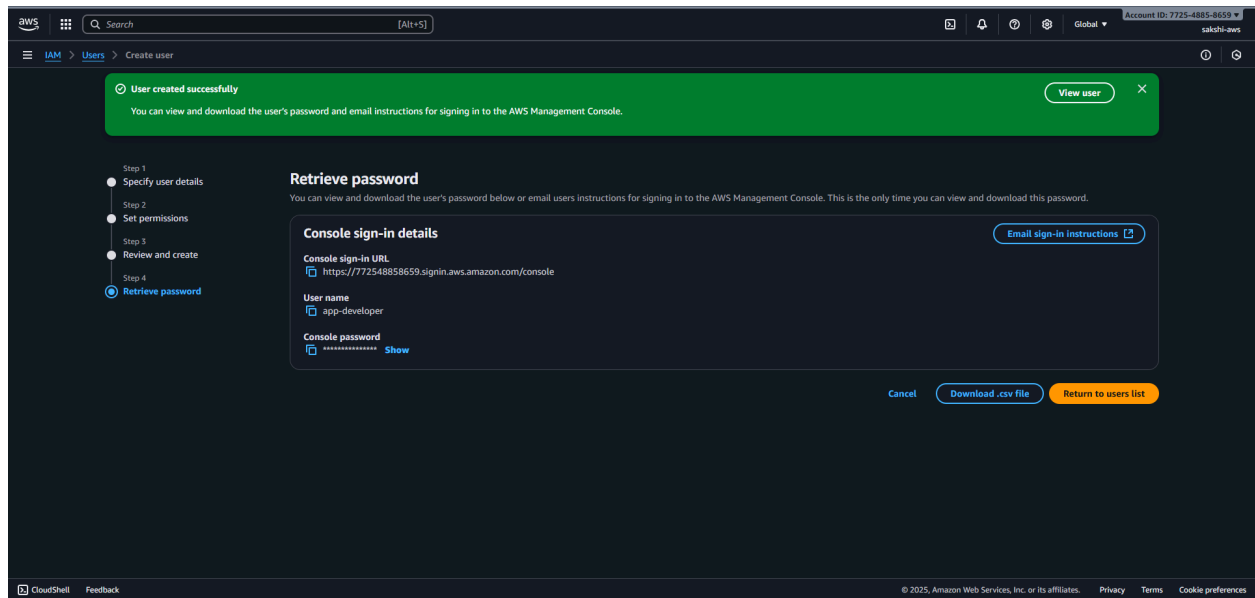
Attaching DeveloperCreateRolePolicy to the app-developer user.



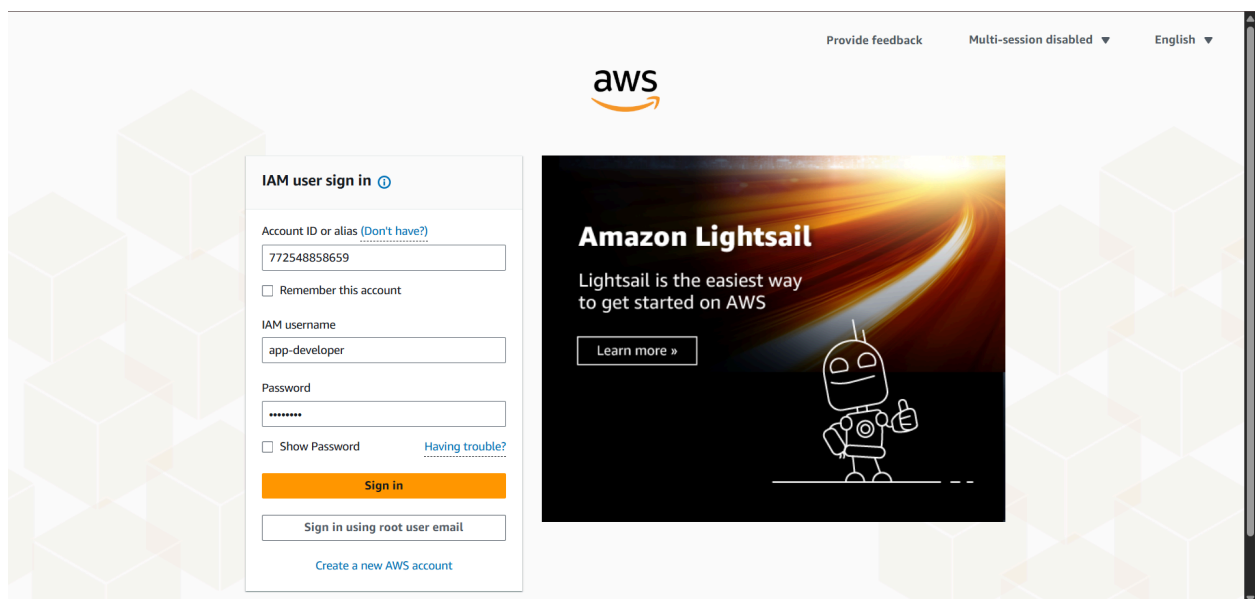
Final review of the app-developer user and its permissions.



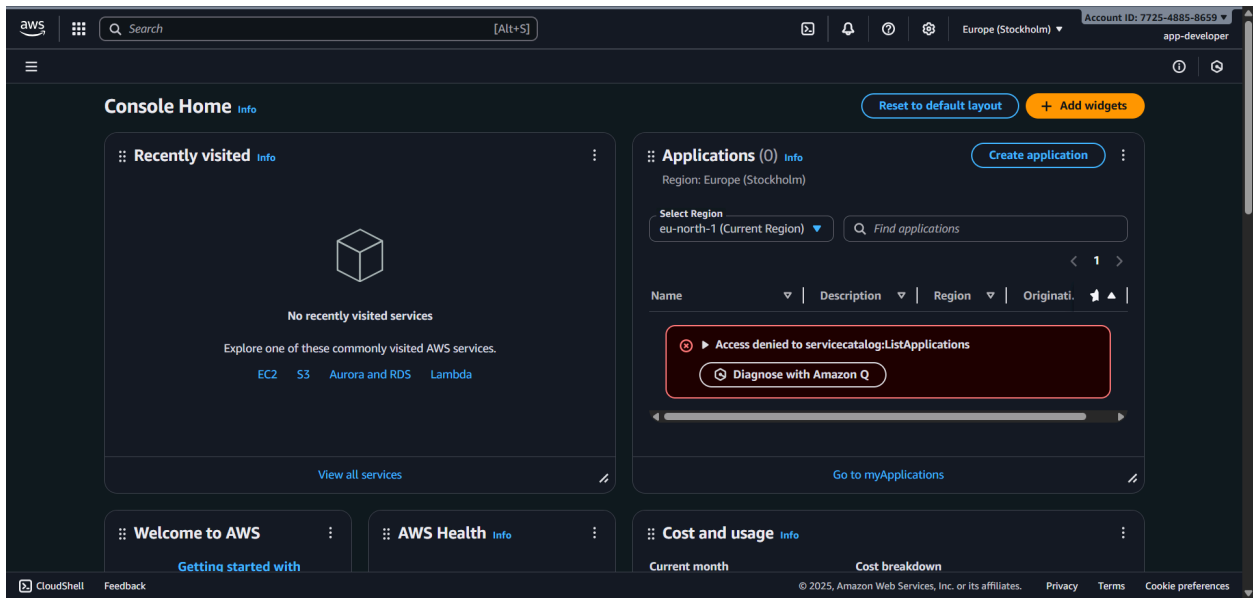
Retrieving the unique Console sign-in URL for the new user.



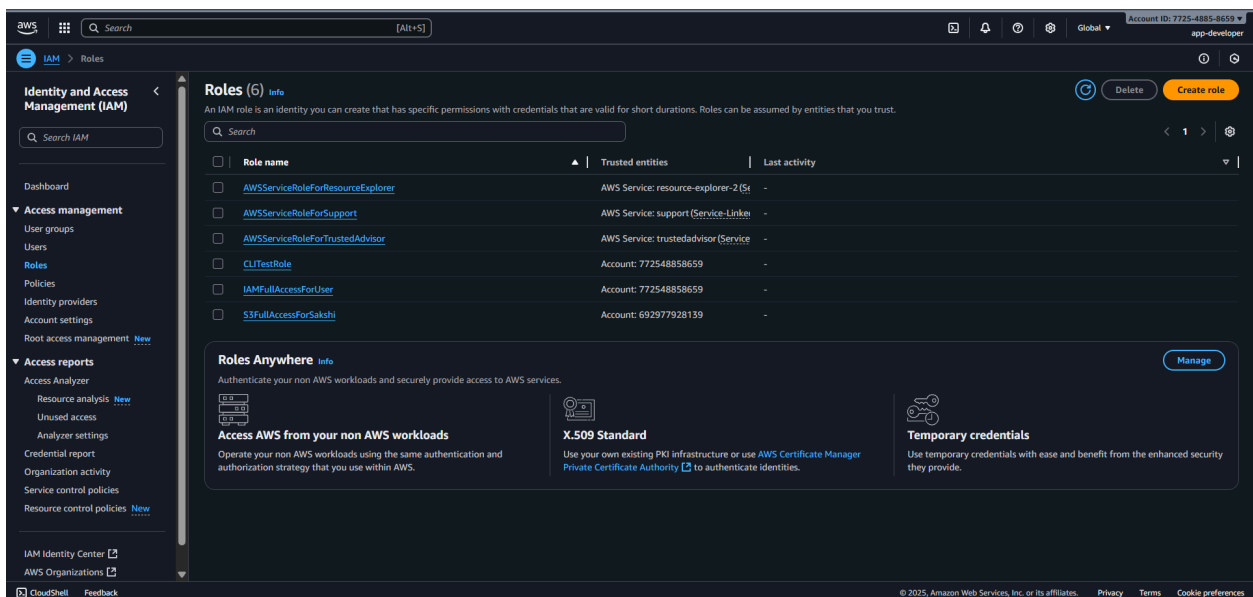
Developer View: Signing in with the app-developer credentials.



Developer is logged in; limited permissions are visible.



Navigating to the IAM Roles dashboard to perform their task.



Starting role creation by selecting EC2 as the use case.

Step 1: Select trusted entity

Select trusted entity

Trusted entity type

- ☒ **AWS service**
Allow AWS services like EC2, Lambda, or others to perform actions in this account.
- ☐ **AWS account**
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.
- ☐ **Web identity**
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.
- ☐ **SAML 2.0 federation**
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.
- ☐ **Custom trust policy**
Create a custom trust policy to enable others to perform actions in this account.

Use case
Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Service or use case
EC2

Choose a use case for the specified service.

Use case

- ☒ **EC2**
Allows EC2 instances to call AWS services on your behalf.
- ☐ **EC2 Role for AWS Systems Manager**
Allows EC2 instances to call AWS services like CloudWatch and Systems Manager on your behalf.
- ☐ **EC2 Spot Fleet Role**
Allows EC2 Spot Fleet to request and terminate Spot instances on your behalf.
- ☐ **EC2 - Spot Fleet Auto Scaling**
Allows Auto Scaling to access and update EC2 spot fleets on your behalf.
- ☐ **EC2 - Spot Fleet Tagging**
Allows EC2 to launch spot instances and attach tags to the launched instances on your behalf.

Attaching the AdministratorAccess policy to the role and setting the WebAppBoundary policy as the role's permission boundary.

Step 2: Add permissions

Add permissions

Choose one or more policies to attach to your new role.

Search: AdministratorAccess Filter by Type: All types 5 matches

Policy name	Type	Description
<input checked="" type="checkbox"/> AdministratorAccess	AWS managed - job function	Provides full access to AWS services and resources.
<input type="checkbox"/> AdministratorAccess-Ampify	AWS managed	Grants account administrative permissions while explicitly allowing direct access to resources needed b...
<input type="checkbox"/> AdministratorAccess-AWSElasticBeanstalk	AWS managed	Grants account administrative permissions. Explicitly allows developers and administrators to gain dire...
<input type="checkbox"/> AWSAuditManagerAdministratorAccess	AWS managed	Provides administrative access to enable or disable AWS Audit Manager, update settings, and manage ...
<input type="checkbox"/> AWSManagementConsoleAdministratorAccess	AWS managed - job function	Provides full access to configure and customize the AWS Management Console

Set permissions boundary - optional

Set a permissions boundary to control the maximum permissions this role can have. This is not a common setting but can be used to delegate permission management to others. [Learn more about permission boundaries](#)

☐ Create role without a permissions boundary

☒ Use a permissions boundary to control the maximum role permissions

Permissions policies (1/1081)

Select policy to set the permissions boundary.

Search: WebApp Filter by Type: All types 1 match

Policy name	Type	Description
<input checked="" type="radio"/> WebAppBoundary	Customer man...	-

Cancel Previous Next

Reviewing MyWebAppRole, which has both the admin policy and the boundary.

The screenshot shows the AWS IAM console interface for creating a new role. The breadcrumb navigation is IAM > Roles > Create role. On the left, a sidebar shows the progress: Step 1 (selected), Step 2, and Step 3. The main content area is titled 'Name, review, and create' and contains two sections: 'Role details' and 'Step 1: Select trusted entities'.

Role details

Role name
Enter a meaningful name to identify this role.
MyWebAppRole
Maximum 64 characters. Use alphanumeric and '+', '@', '-' characters.

Description
Add a short explanation for this role.
Allows EC2 instances to call AWS services on your behalf.
Maximum 1000 characters. Use letters (A-Z and a-z), numbers (0-9), tab, new lines, or any of the following characters: '_', '+', '@', '/', '[]' and '%': '"~"

Step 1: Select trusted entities

Trust policy

```
1. {
2.   "Version": "2012-10-17",
3.   "Statement": [
4.     {
5.       "Effect": "Allow",
6.       "Action": [
7.         "sts:AssumeRole"
8.       ],
9.       "Principal": {
10.        "Service": [
11.          "ec2.amazonaws.com"
12.        ]
13.      }
14.    ]
15.  }
16. }
```

Reviewing MyWebAppRole, which has both the admin policy and the boundary.

The screenshot shows the AWS IAM console interface for adding permissions to a role. The breadcrumb navigation is IAM > Roles > Create role. The main content area is titled 'Step 2: Add permissions' and contains two sections: 'Permissions policy summary' and 'Step 3: Add tags'.

Step 2: Add permissions

Permissions policy summary

Policy name	Type	Attached as
AdministratorAccess	AWS managed - job function	Permissions policy
WebAppBoundary	Customer managed	Boundary policy

Step 3: Add tags

Add tags - optional info

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

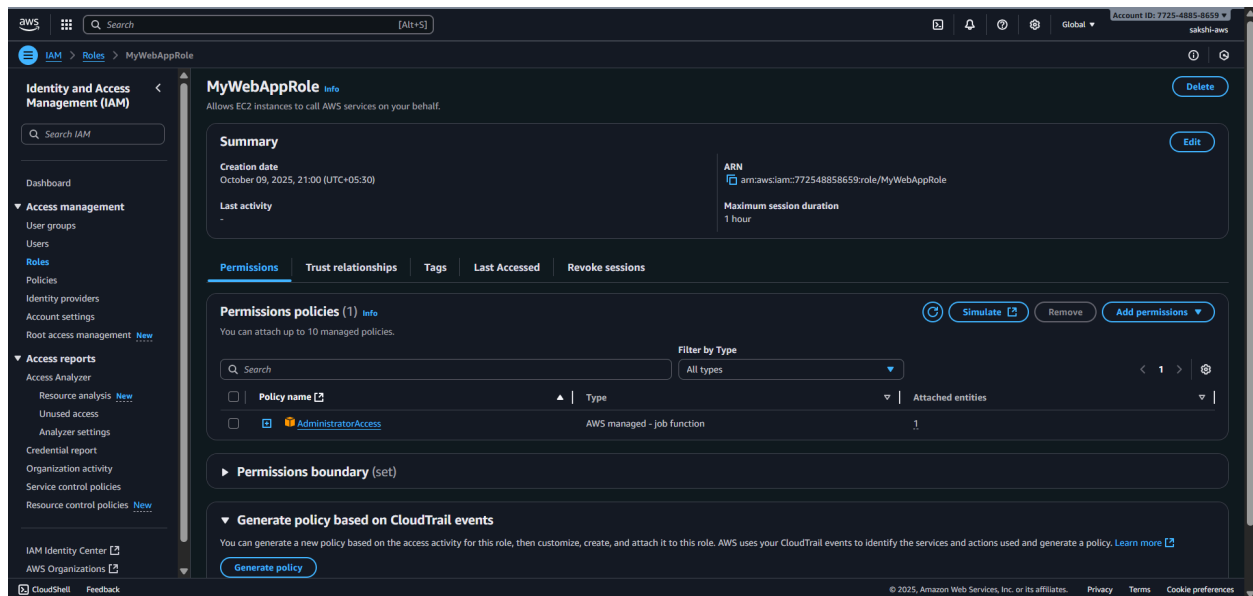
No tags associated with the resource.

[Add new tag](#)

You can add up to 50 more tags.

[Cancel](#) [Previous](#) [Create role](#)

The summary page for MyWebAppRole visually confirms that both policies are active. The AdministratorAccess policy is listed under "Permissions policies," but the WebAppBoundary is also set, acting as the ultimate permissions guardrail for the role.



This practical has successfully demonstrated the use of AWS IAM Permissions Boundaries as a powerful feature for secure delegation. The objective, as stated in the introduction, was to simulate a scenario where a developer could be empowered to create IAM roles without posing a security risk.

The exercise was conducted in distinct phases: the administrator prepared the environment by creating a restrictive WebAppBoundary policy and a special DeveloperCreateRolePolicy that enforced its use. Subsequently, the app-developer user was created and granted these limited permissions. The final test involved the developer creating the MyWebAppRole, where they attempted to attach the highly permissive AdministratorAccess policy while being forced to apply the boundary. The result, confirmed by the final verification screenshot, is a new role (MyWebAppRole) that has both policies attached simultaneously. This proves the core concept: the role's effective permissions are the intersection of both policies. Therefore, MyWebAppRole is prevented from performing any administrative actions outside of those explicitly allowed by the WebAppBoundary. This practical confirms that permissions boundaries are an essential tool for enabling developer agility and scaling cloud operations without compromising on security governance.