



# Cloud Security with AWS IAM



nihitha40633@gmail.com

The screenshot shows the AWS IAM 'Create policy' interface. The left sidebar has 'Step 1: Specify permissions' selected. The main area is titled 'Specify permissions' with an 'Info' link. It contains a 'Policy editor' section with a JSON editor. The JSON code is as follows:

```
1  {
2     "Version": "2012-10-17",
3     "Statement": [
4         {
5             "Effect": "Allow",
6             "Action": "ec2:Describe",
7             "Resource": "*",
8             "Condition": {
9                 "StringEquals": {
10                     "ec2:ResourceTag/Env": "development"
11                 }
12             }
13         },
14         {
15             "Effect": "Allow",
16             "Action": "ec2:DescribeTags",
17             "Resource": "*"
18         },
19         {
20             "Effect": "Deny",
21             "Action": [
22                 "ec2:DeleteTags",
23                 "ec2:CreateTags"
24             ],
25             "Resource": "*"
26         }
27     ]
28 }
```

The JSON editor has tabs for 'Visual', 'JSON' (which is selected), and 'Actions'. A right-hand panel titled 'Edit statement' says 'Select a statement' and 'Select an existing statement in the policy or add a new statement' with a '+ Add new statement' button.

# Introducing today's project!

## What is AWS IAM?

AWS IAM (Identity and Access Management) is a service that enables you to manage users and their permissions within AWS. It's useful because it allows fine-grained access control, ensuring that users only have the permissions they need, enhancing sec

## How I'm using AWS IAM in this project

Manage access level that other users and services have to my resources.

## One thing I didn't expect...

We can test IAM policy with simulator and other user can access my account.

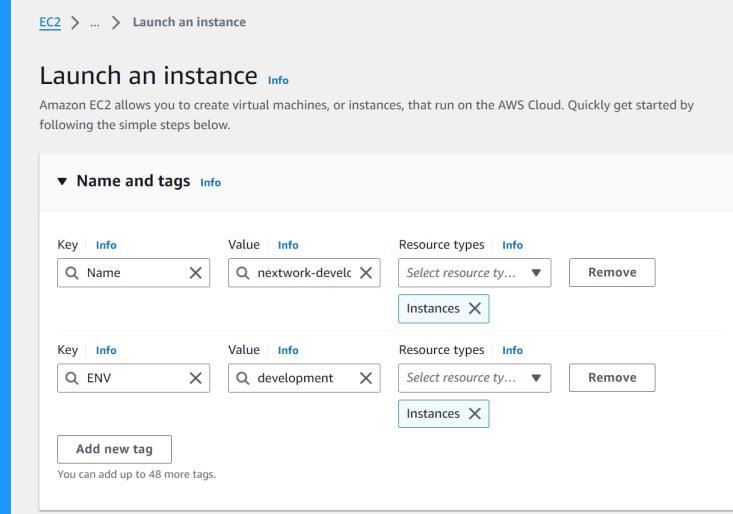
## This project took me...

It took me 120 minutes to finish this project.

# Tags

Tags are like labels you can attach to AWS resources for organization. This tagging helps us identify all resources with the same tag at once cost allocation and apply policies.

The tag I've used on my EC2 instances is called ENV. The value I've assigned for my instances are production and development



# IAM Policies

IAM Policies are rules for who can do what with your AWS resources.

## The policy I set up

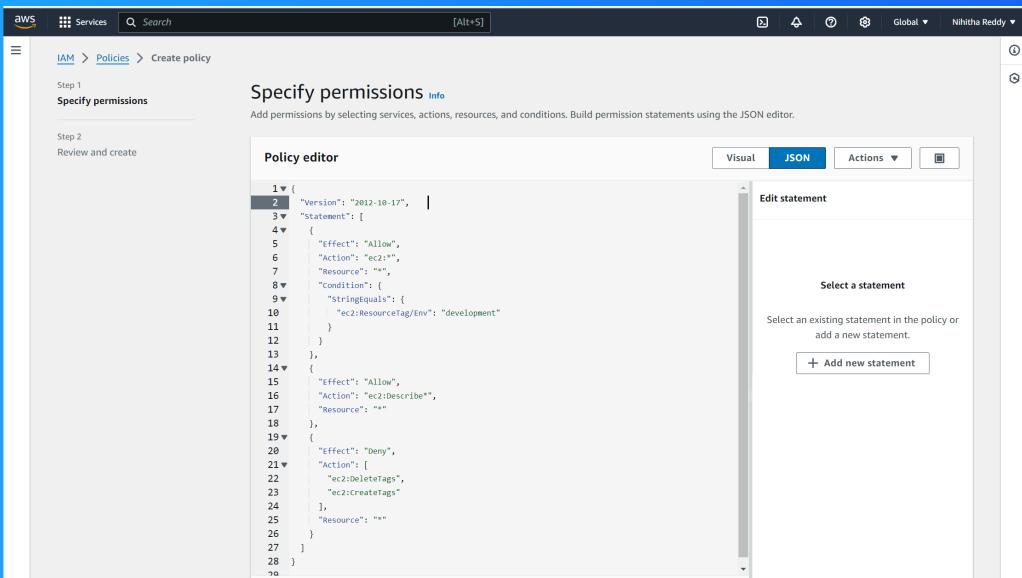
For this project, I've set up a policy using JSON.

I've created a policy that allows some actions (like starting, stopping, and describing EC2 instances) for instances tagged with "Env = development" while denying the ability to create or delete tags for all instances.

## When creating a JSON policy, you have to define its Effect, Action and Resource.

The Effect, Action, and Resource attributes of a JSON policy mean they collectively define permissions, determining what actions can be performed on which resources and whether they are allowed or denied.

# My JSON Policy



The screenshot shows the AWS IAM Policy Editor interface. The top navigation bar includes the AWS logo, Services, Search, and Global dropdown. The main title is "Specify permissions" with an "Info" link. Below it, a sub-header says "Add permissions by selecting services, actions, resources, and conditions. Build permission statements using the JSON editor." The "JSON" tab is selected in the toolbar. The "Policy editor" section contains the following JSON code:

```
1 * {
2     "Version": "2012-10-17",
3     "Statement": [
4         {
5             "Effect": "Allow",
6             "Action": "ec2:*",
7             "Resource": "*",
8             "Condition": {
9                 "StringEquals": {
10                     "ec2:ResourceTag/Env": "development"
11                 }
12             }
13         },
14         {
15             "Effect": "Allow",
16             "Action": "ec2:Describe",
17             "Resource": "*"
18         },
19         {
20             "Effect": "Deny",
21             "Action": [
22                 "ec2:DeleteTags",
23                 "ec2:CreateTags"
24             ],
25             "Resource": "*"
26         }
27     ]
28 }
```

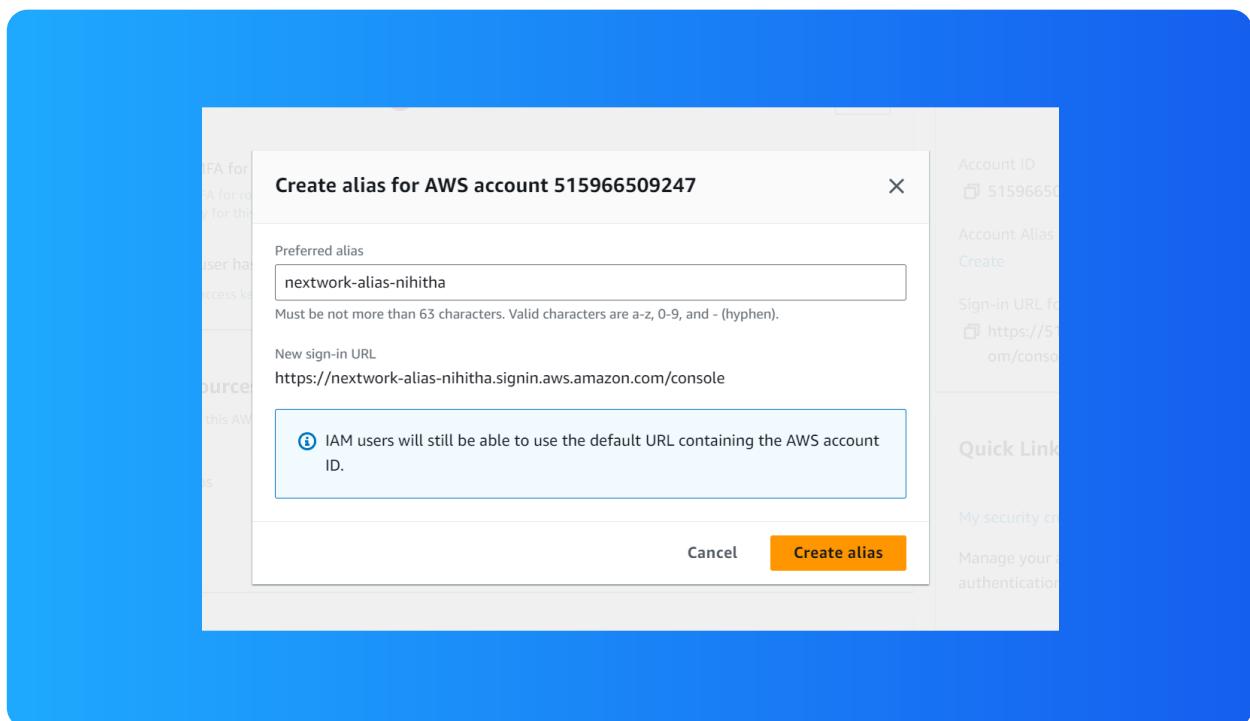
The right side of the editor shows a sidebar with "Edit statement" and "Select a statement" sections, along with a button to "+ Add new statement".

# Account Alias

An account alias is a friendly name for your AWS account that you can use instead of your account ID (which is usually a bunch of digits) to sign in to the AWS Management Console.

Creating an account alias took me two minutes.

Now, my new AWS console sign-in URL is <https://nextwork-alias-nihitha.signin.aws.amazon.com/console>



# IAM Users and User Groups

## Users

IAM users are people who get access to your resources/ AWS account.

## User Groups

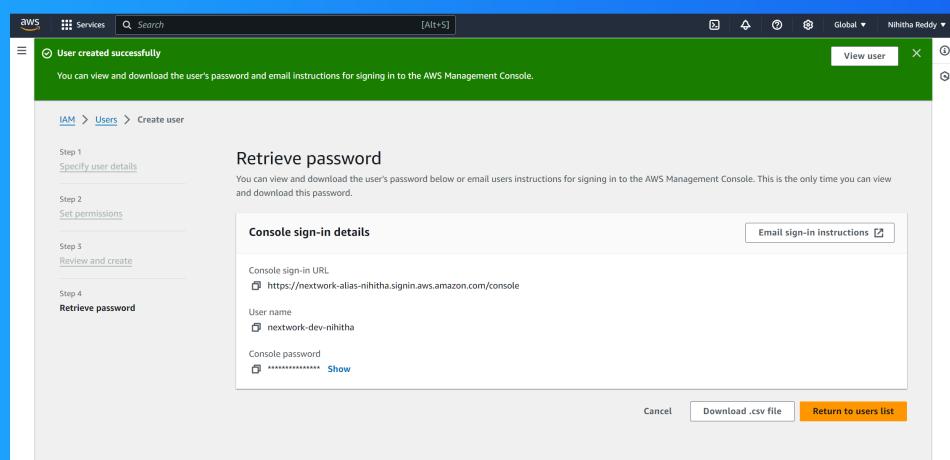
IAM user groups are collections/folders of IAM users.

I attached the policy I created to this user group, which means it sets the permissions to all the users in the group rather than individual users.

# Logging in as an IAM User

The first way is to send the user their sign-in link, username, and temporary password securely via encrypted email. The second way is to use AWS IAM's invitation feature, which automatically emails the sign-in details for secure access.

Once I logged in as my IAM user, I noticed access was denied to a few dashboard panels.





nihitha40633@gmail.com  
NextWork Student

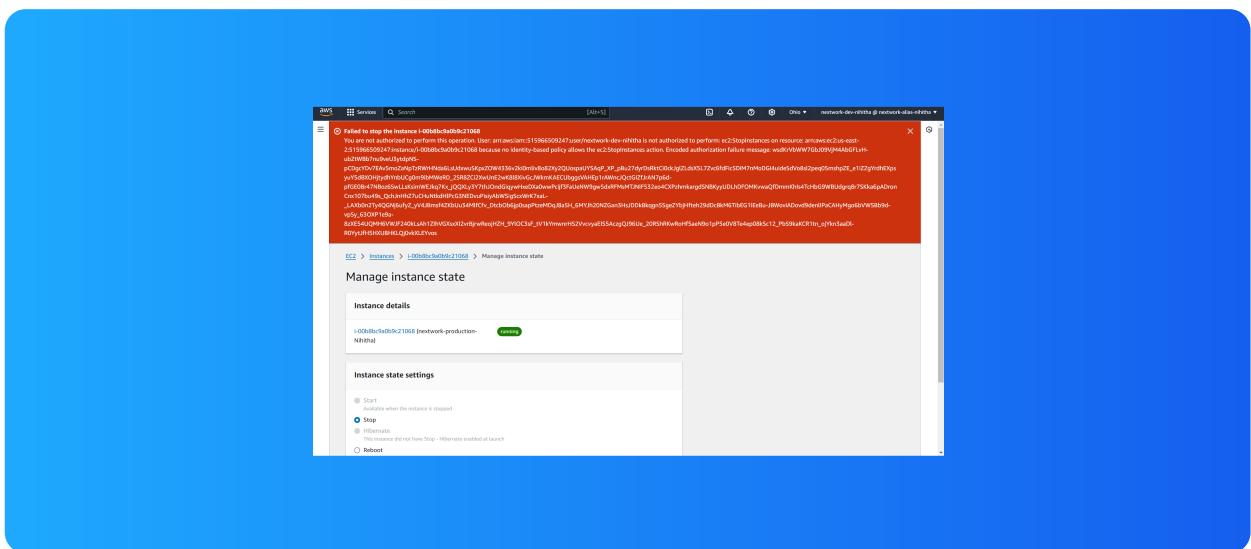
[NextWork.org](http://NextWork.org)

# Testing IAM Policies

I tested my JSON IAM policy by stopping the two instances.

# Stopping the production instance

When I tried to stop the production instance I got an error banner stating access denied.





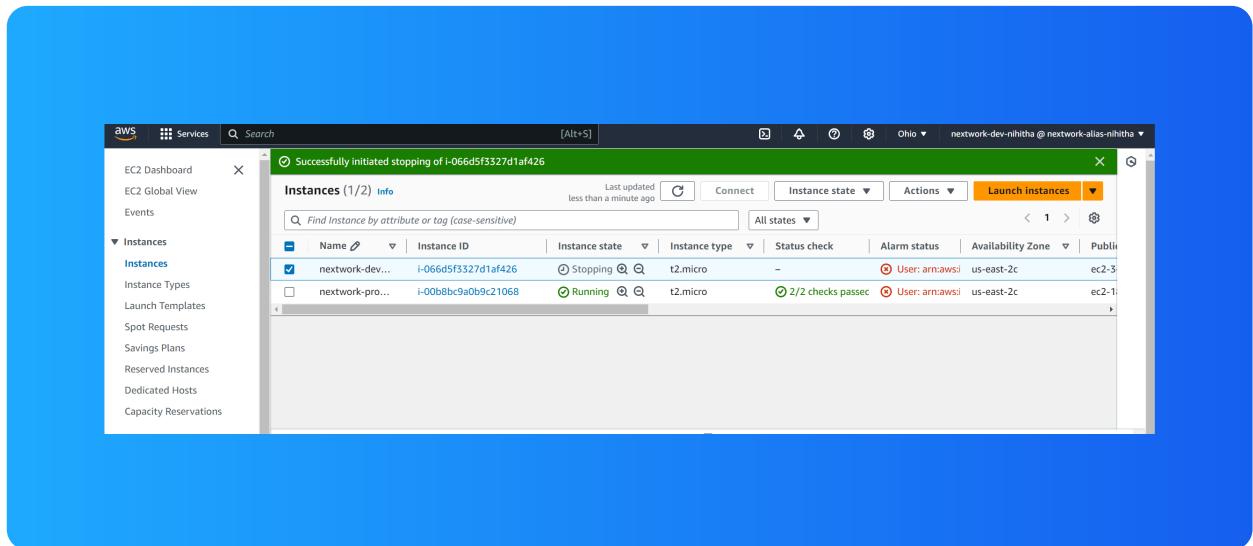
nihitha40633@gmail.com  
NextWork Student

[NextWork.org](http://NextWork.org)

# Testing IAM Policies

# Stopping the development instance

Next, when I tried to stop the development instance it was successful.





NextWork.org

# Everyone should be in a job they love.

Check out nextwork.org for  
more projects

