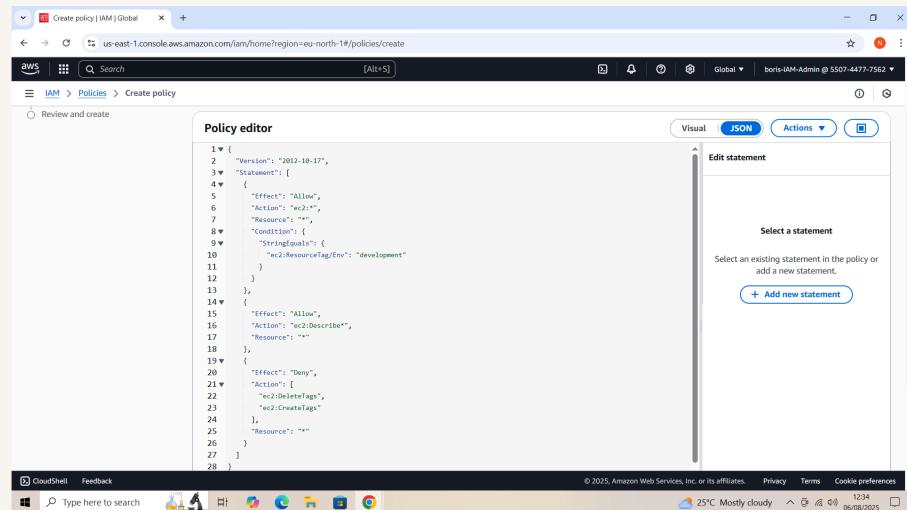




# Cloud Security with AWS IAM

N

Nchindo Boris



The screenshot shows the AWS IAM Policy Editor interface. The top navigation bar includes links for IAM, Policies, Create policy, and a search bar. The main area is titled "Policy editor" and displays a JSON code block representing a policy. 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:DeleteTags",
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 has tabs for "Visual" and "JSON" (which is selected), and a "Actions" dropdown. A sidebar titled "Edit statement" contains the message "Select a statement" and a button "+ Add new statement". The bottom of the window shows standard browser controls and a status bar indicating the date and time.

# Introducing Today's Project!

Today, we'll be using the AWS Identity and Access Management (IAM) service to control who is authenticated (signed in) and authorized (has permissions) in your AWS account. We'll launch an EC2 instance, then control who has access to it

## Tools and concepts

Services I used were; - EC2 - IAM and policy  
Key concepts I learnt include; - JSON policy for IAM users - Account Alias creation

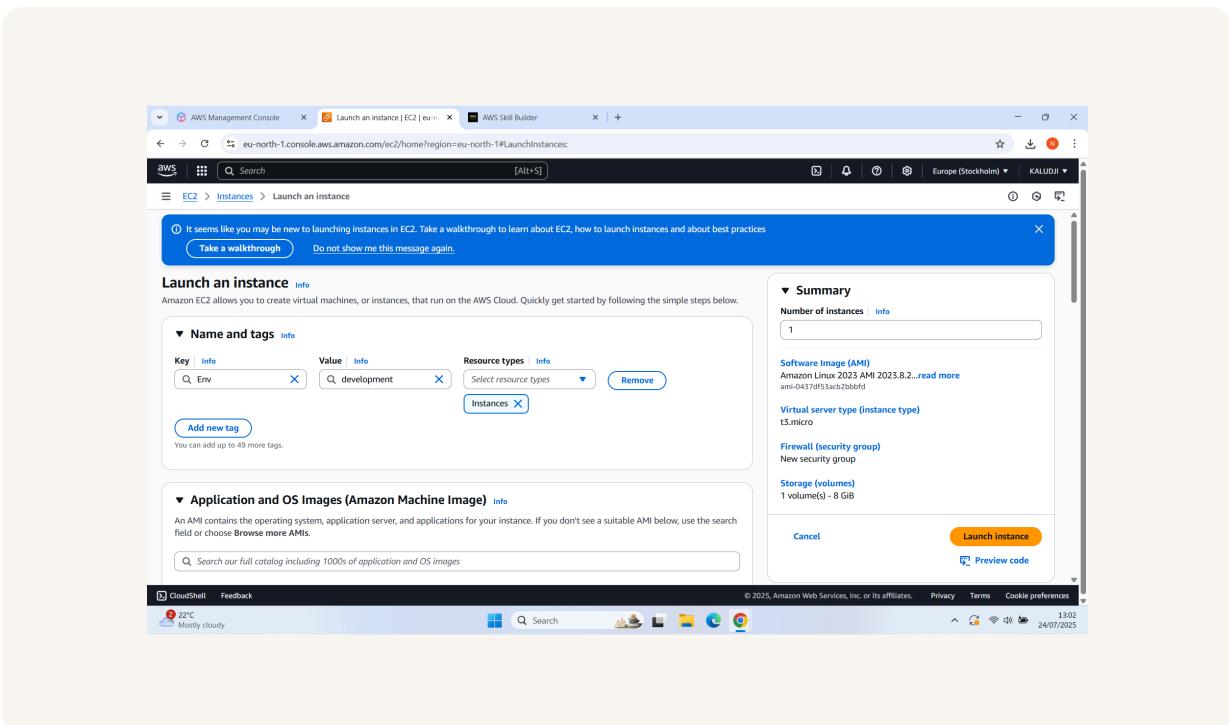
## Project reflection

This project took me approximately 1hr 30mins. The most challenging part was to enforce the security policies correctly. It was most rewarding to see the policies work effectively

# Tags

Tags are like labels you can attach to AWS resources for organization. This tagging helps us with identifying all resources with the same tag at once

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. It's all about giving permissions to IAM users, groups, or roles, saying what they can or can't do on certain resources, and when those rules kick in.

## 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; Effect = either Allow or Deny Action = A list of the actions that the policy allows or denies. Resource = Specifying "\*" means all resources within the defined scope



# My JSON Policy

The screenshot shows the AWS IAM Policy Editor interface. The left pane displays the JSON code of the policy, and the right pane shows a visual editor with tabs for Visual, JSON, and Actions. A modal window titled "Edit statement" is open, prompting the user to "Select a statement" or "Add new statement".

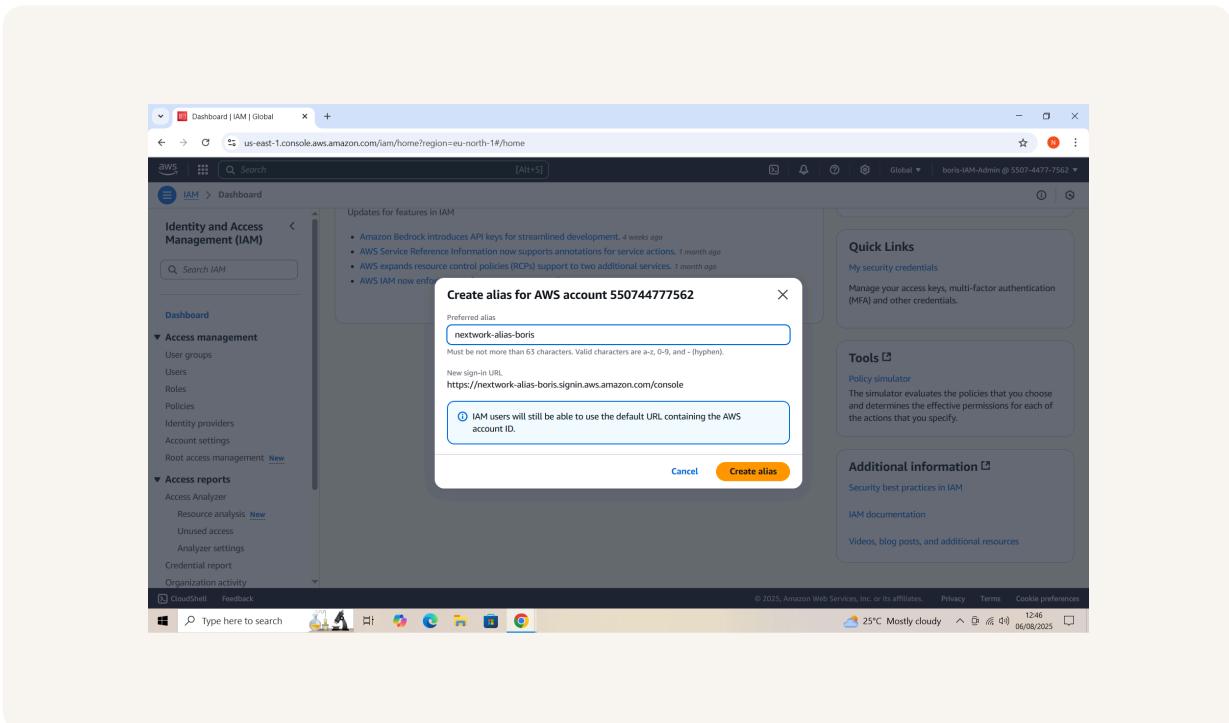
```
1▼ {
2  "Version": "2012-10-17",
3  "Statement": [
4    {
5      "Effect": "Allow",
6      "Action": "ec2:*",
7      "Resource": "*",
8      "Condition": {
9        "StringEquals": {
10          "ec2:ResourceTag/Env": "development"
11        }
12      }
13    },
14    {
15      "Effect": "Allow",
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 }
```



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





# IAM Users and User Groups

## Users

IAM users are the people that will get access to your resources/AWS account

## User Groups

IAM user groups are collections/folders of IAM users. They allow me to manage permissions for all the users in my group at the same time by attaching policies to the group rather than individual users

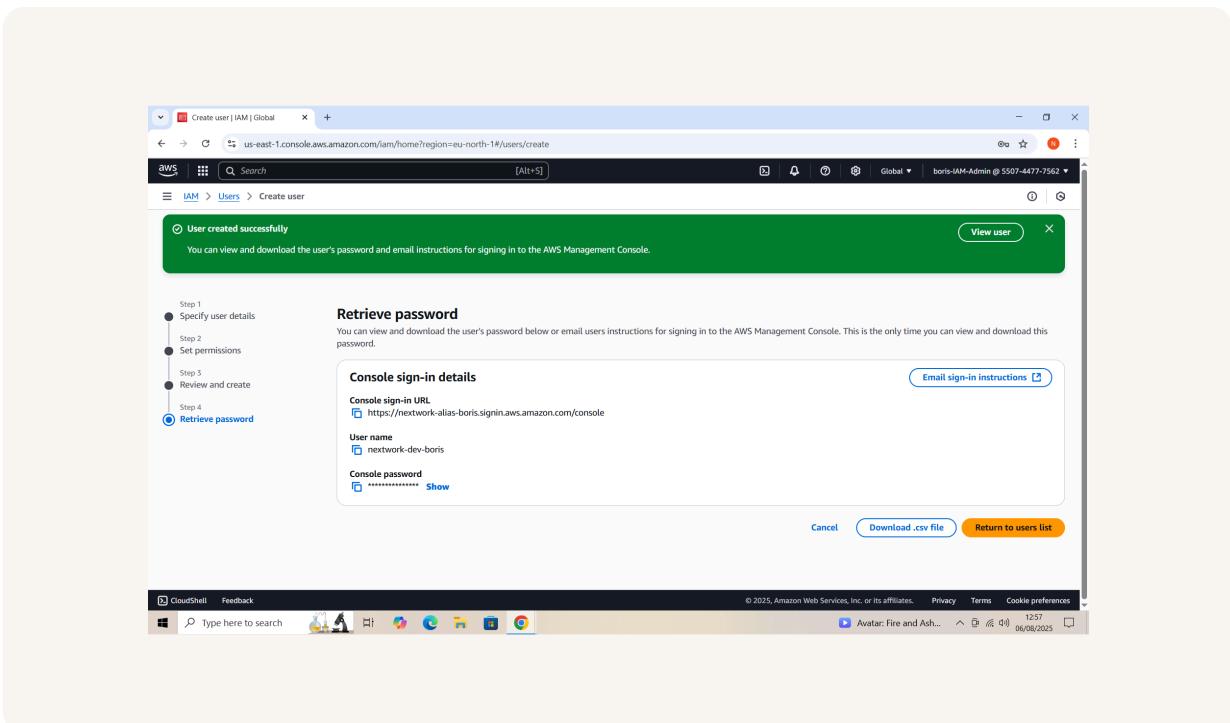
I attached the policy I created to this user group, which means simplifying the management of permissions and ensures consistency across users who have similar access to AWS resources.



# Logging in as an IAM User

The first way is to download the Credentials (.csv File) and manually send them. The second way is to email the sign-in instructions

Once I logged in as my IAM user, I noticed that some of my dashboard panels are showing Access denied already.

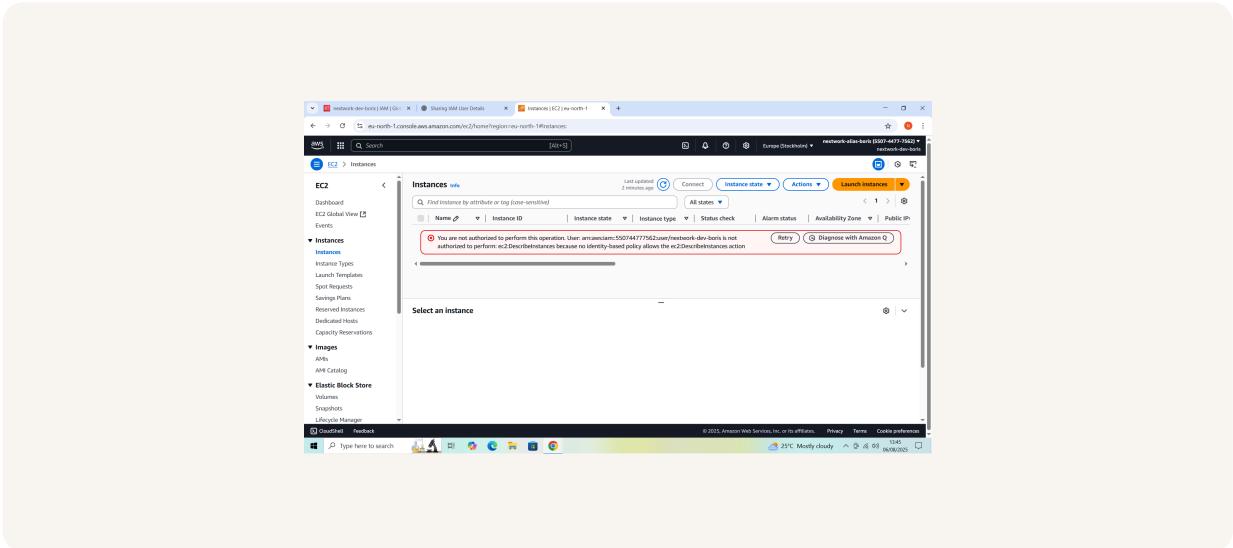


# Testing IAM Policies

I tested my JSON IAM policy by trying to stop the EC2 Instances

## Stopping the production instance

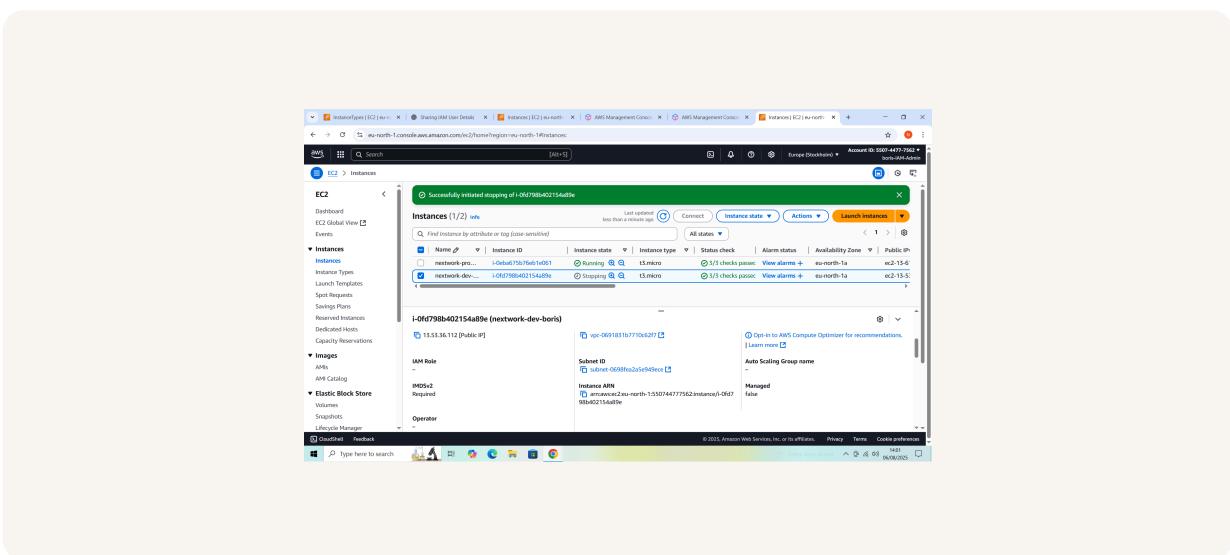
When I tried to stop the production instance I got an error message because this user is not authorized to perform this operation.



# Testing IAM Policies

## Stopping the development instance

Next, when I tried to stop the development instance it stopped. This was because the JSON policy allowed permission for that





[nextwork.org](https://nextwork.org)

# The place to learn & showcase your skills

Check out [nextwork.org](https://nextwork.org) for more projects

