CLOUD SECURITY WITH AWS IAM

Author: Mohammed Baqtiyar Ahmed Khan
Date Created: 11/2/2024
Version: 1

TABLE OF CONTENTS

1. INTRODUCTION TO CLOUD SECURITY

In today's world, security is essential not only for physical resources but also for virtual ones. As most organizations move to the cloud, it has become vital to enhance security measures to prevent unauthorized access. Adhering to best practices and protocols is crucial to avoid unintended access to the AWS Management Console. In this project, I have used AWS Identity and Access Management (IAM) to strategically implement security measures through AWS services.

AWS IAM controls who is authenticated (signed in) and authorized (has permission) to use your account's resources. This is achieved through IAM policies, where you can specify which resources should be accessed using JSON documentation. In real-world scenarios, companies commonly use this service to provide appropriate access to new employees.

Organizations create user accounts and user groups; for example, they may create a group named "Development" and add multiple users who share the same level of permissions. Next, permissions are attached to the group, granting all users in that group the same level of access.

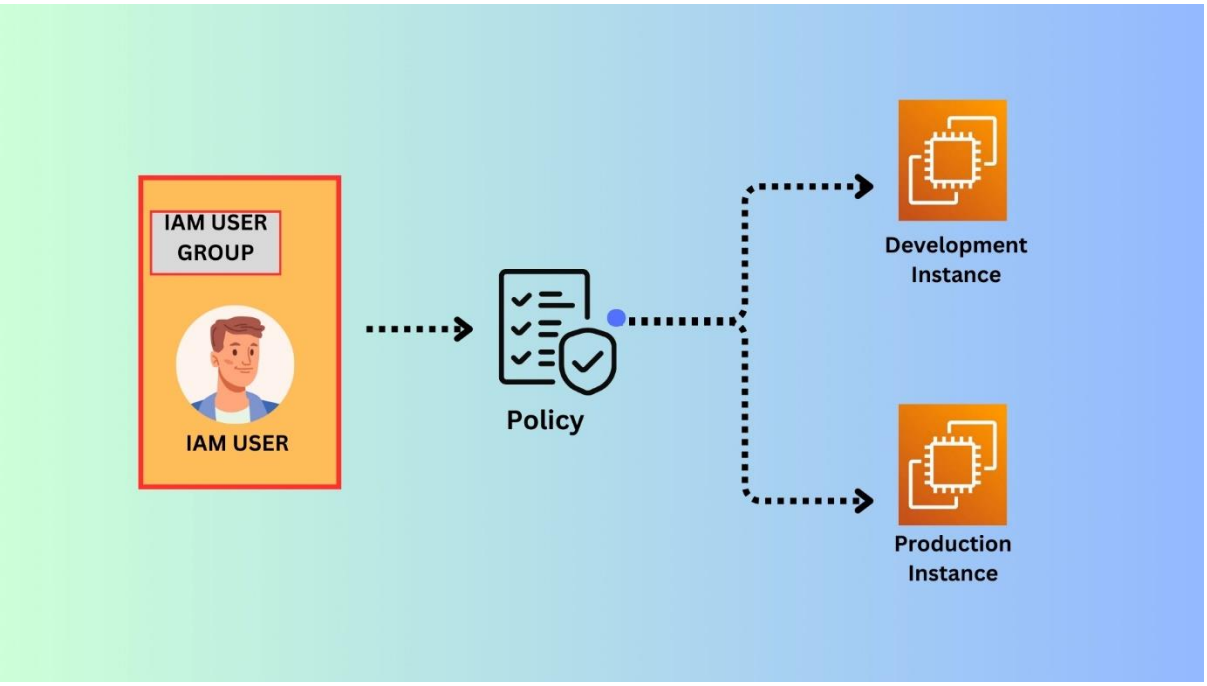
The AWS Services we are going to use are

- EC2 Instances
- IAM Policies
- IAM Users and User Groups
- AWS Account Alias

1. REQUIREMENTS

➔ **AWS Account:** To access AWS services
➔ **Amazon EC2:** Access this to create virtual servers on cloud
➔ **Amazon IAM:** Access to creating policies, users, groups.
➔ **Basic understanding of EC2 & IAM:** Familiarity with renting virtual server on cloud, creating IAM policies, users and groups.
➔ **Active Internet Connection:** Stable Internet connectivity is crucial to complete this project.

2. ARCHITECTURE



3. GETTING STARTED WITH CLOUD SECURITY

Let's get started with our project work

First, we are going to launch an EC2 Instance using AWS Management Console, but

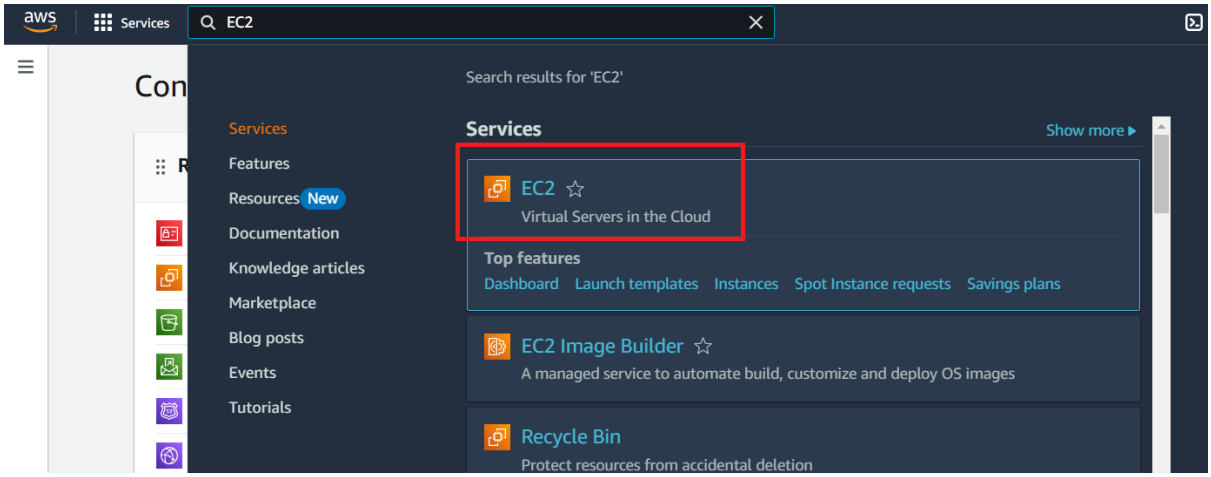
What is an EC2 Instance?

EC2 Instance stands for Elastic Compute Cloud, which is an AWS service to provide virtual computers, which in simple terms means that instead of physical computers in front of you, we use computers on the Internet.

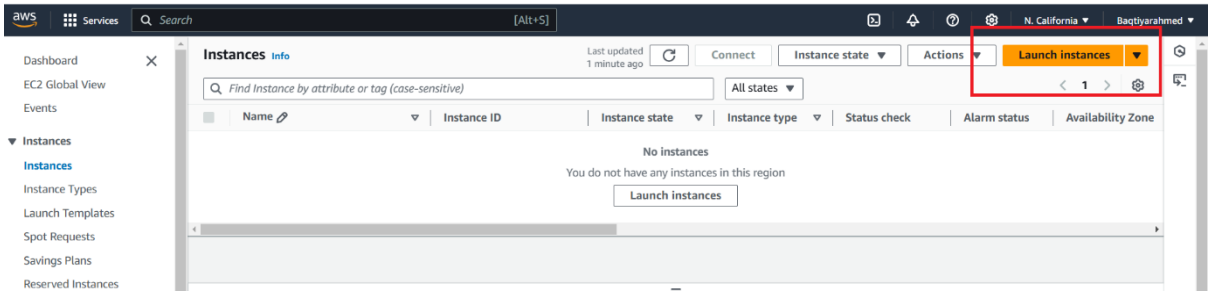
Most of us saw the movie "The Incredibles" in our childhood, in that movie there is character named Elastigirl, whose superpowers is her ability to be flexible and adapt to any challenge she faces. The same concept is applied here our EC2 Instance does the same job that's why it is called as "**Elastic**". **Compute** means computing power, and **Cloud** means over the Internet.

So now we are going to launch an EC2 Instance

In the AWS Management Console type EC2 and click on it



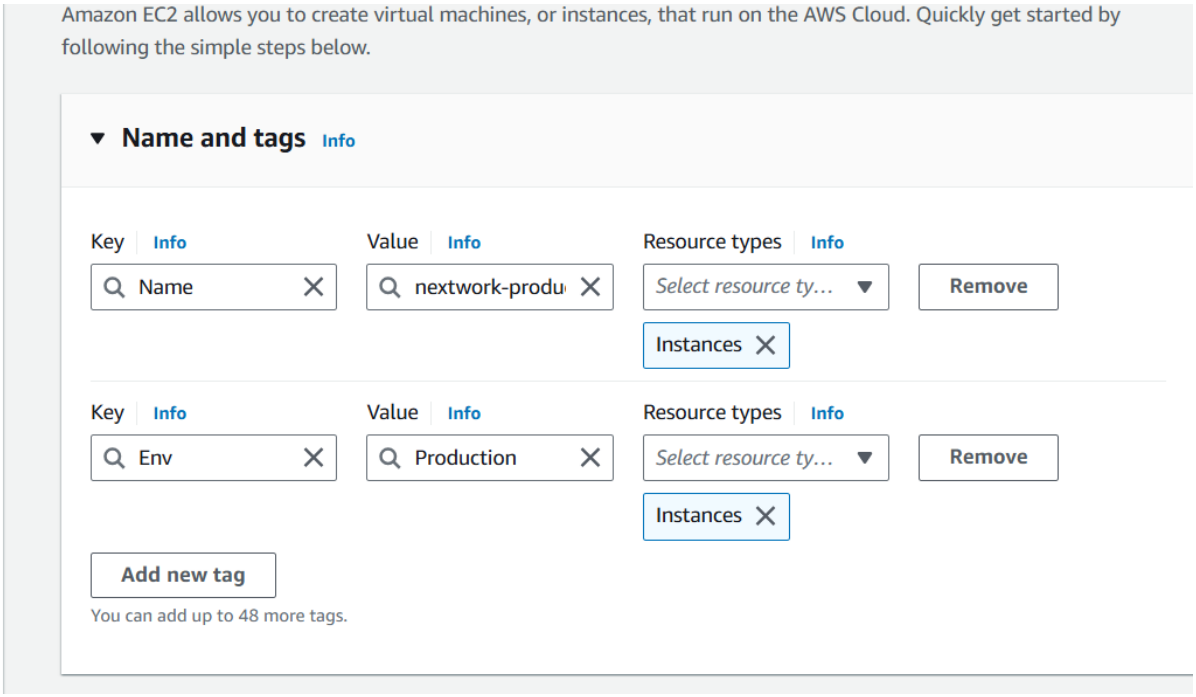
Now, click on Instances from the right-side panel, then launch instance button in the top right corner.



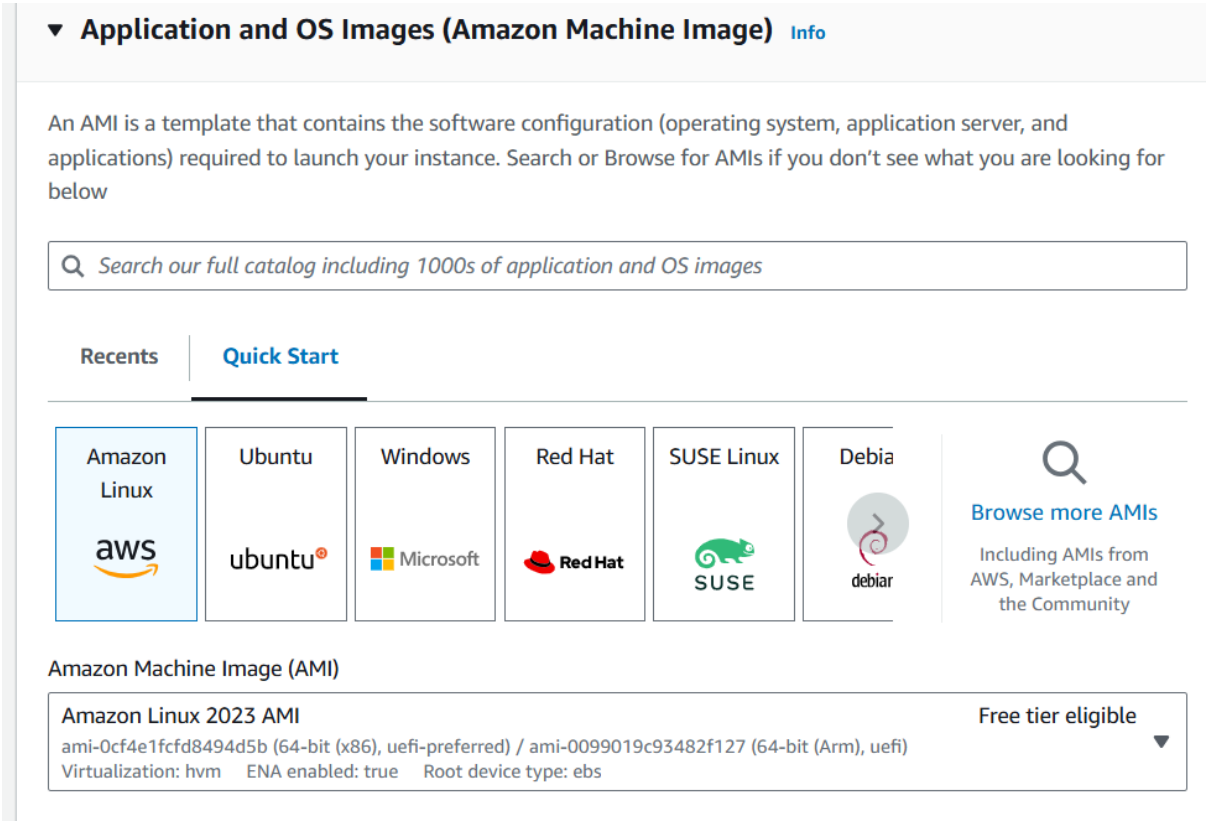
Here, you will find a couple of options, I will describe it for you

1. **Names & Tags** – You can name your virtual machine and put a tag on it, tags are like labels you can attach to your AWS resources, you will find tags in almost all AWS services, so get use to it.
2. **Amazon Machine Image (AMI's)** – This is a template or blueprint of your EC2 instance, when you buy a new computer, it comes with a pre-installed operating system and software (e.g. MacOS, Windows) already configured for you. Similarly, we have AMI.
3. **Instance Type** – This covers the hardware components, if AMI's give you pre-built software and OS, then Instance Type will give you CPU power, memory size, storage space and more!
4. **Key Pairs** – A key pair is primarily used for accessing your EC2 instance securely without going through the AWS Management Console. You use SSH (Secure Shell) Access with your key pairs.
5. **Network Settings** – This says how your instance (virtual machine) interacts over the Internet, it talks about IP addresses, router, firewall and security groups.
6. **Storage** – The type of hard drive that your EC2 instance will use to store data.

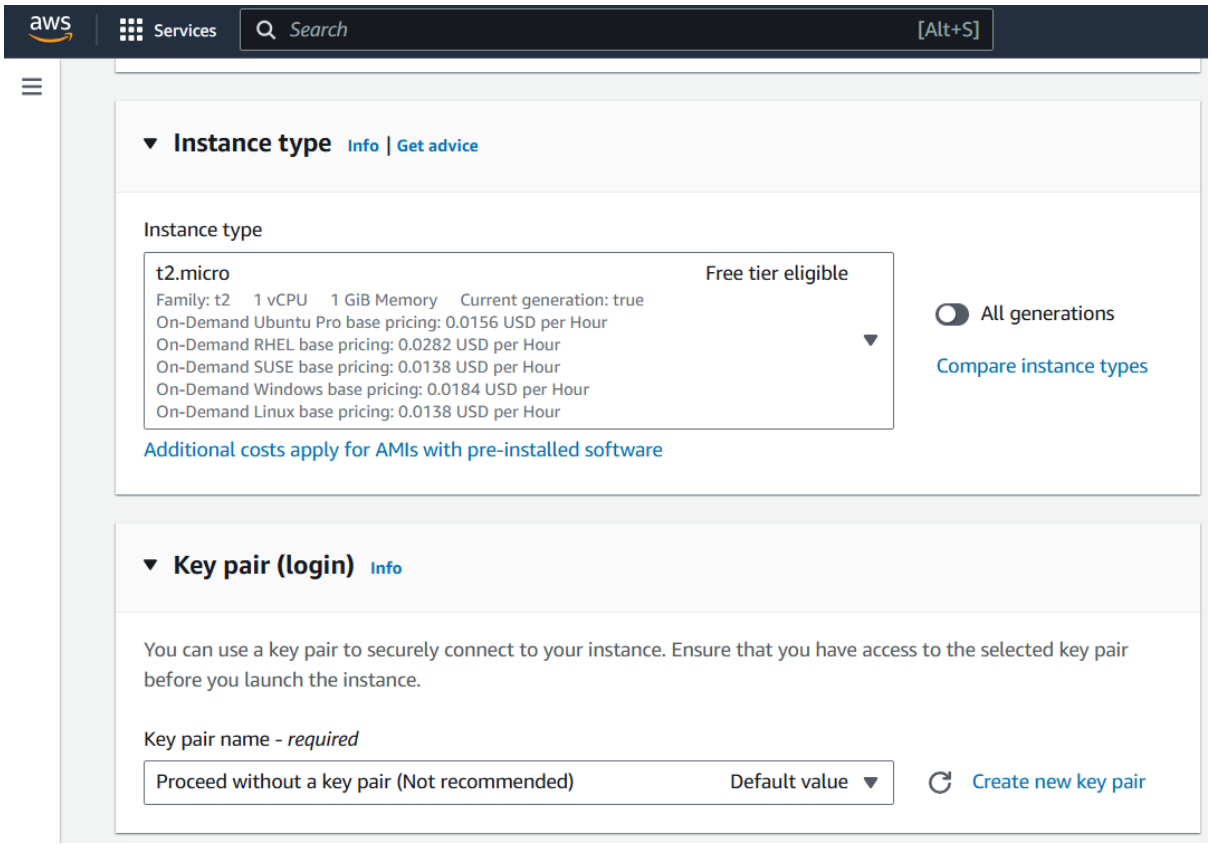
Now, give a name and tag the instance



Select the free tier AMI (Amazon Machine Image)



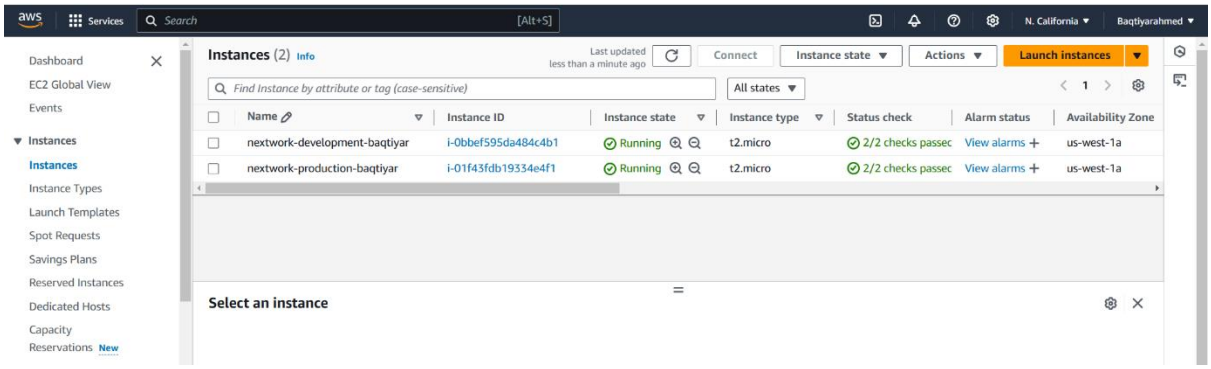
Select free tier Instance type & key pair as “proceed without key pair”. Keep the Network setting and Storage as it is.



Note: Its always recommended to create a key pair, because it will help you to access your ec2 instance outside of AWS Management Console through a secure shell.

Click Launch and it will show you success

Now create another ec2 instance called development instance, previously we created production instance. The screen should look something like this.



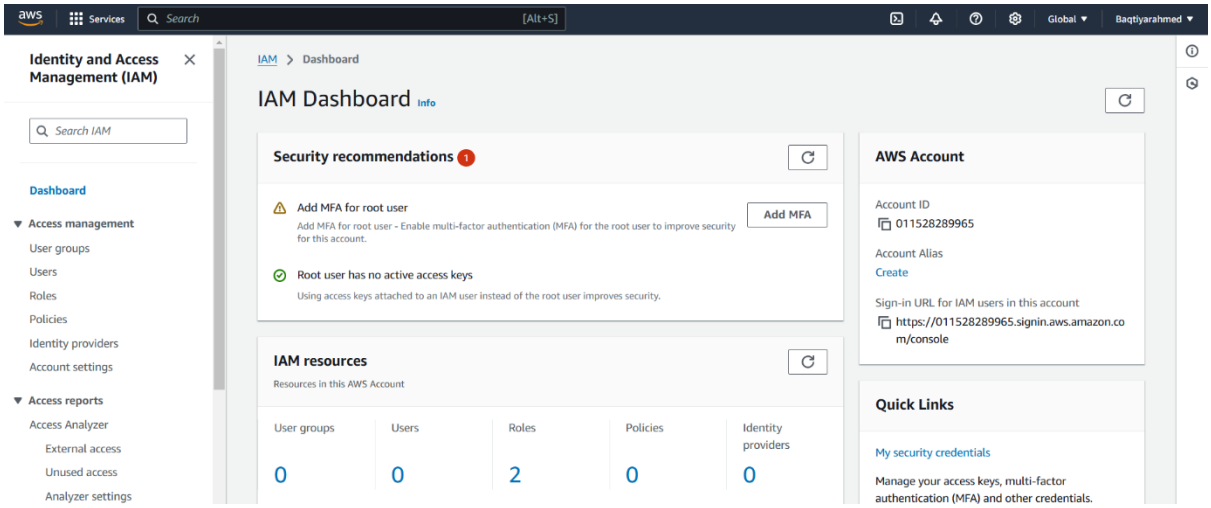
PART 2 OF THE PROJECT

Create an IAM Policy

Congratulations you have deployed two EC2 Instances, one for the development environment and another for production environment. These two environments are present in the software-development-life cycle in an organization.

Now, let's say we have hired an intern in a company, and we must provide him development instance access, this is what we are going to do in this step.

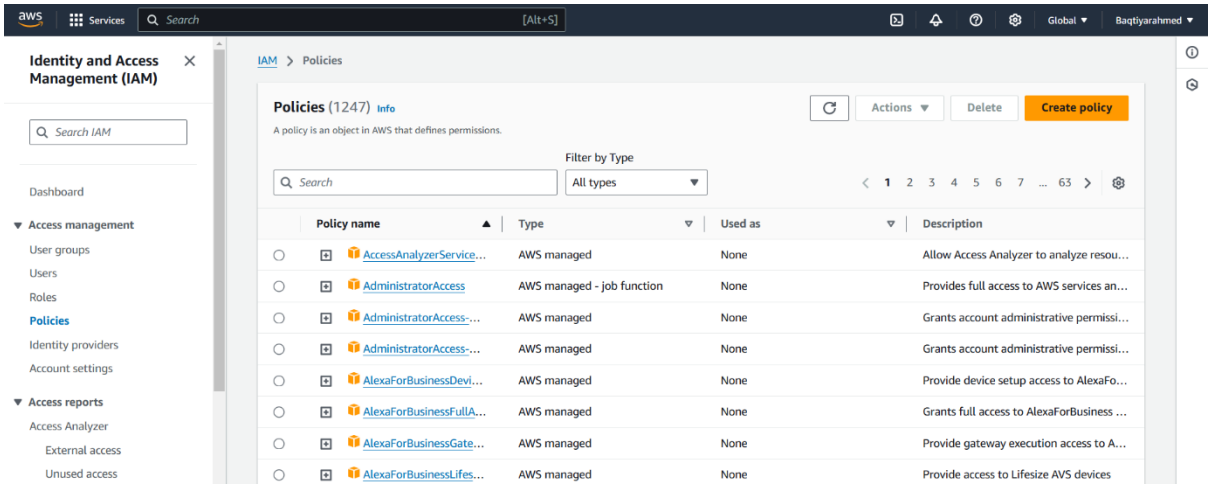
In the AWS Management Console search for IAM, the screen should look something like this.



What is IAM?

IAM stands for Identity and Access Management which is used to give permission to a user to use the AWS resources. You can create users, user-groups and roles with the help of IAM service.

Now let's create an IAM Policy, In the left-hand navigation panel click on Policies.



After that click on create policy, here you have two policy editors

1. Visual
2. JSON

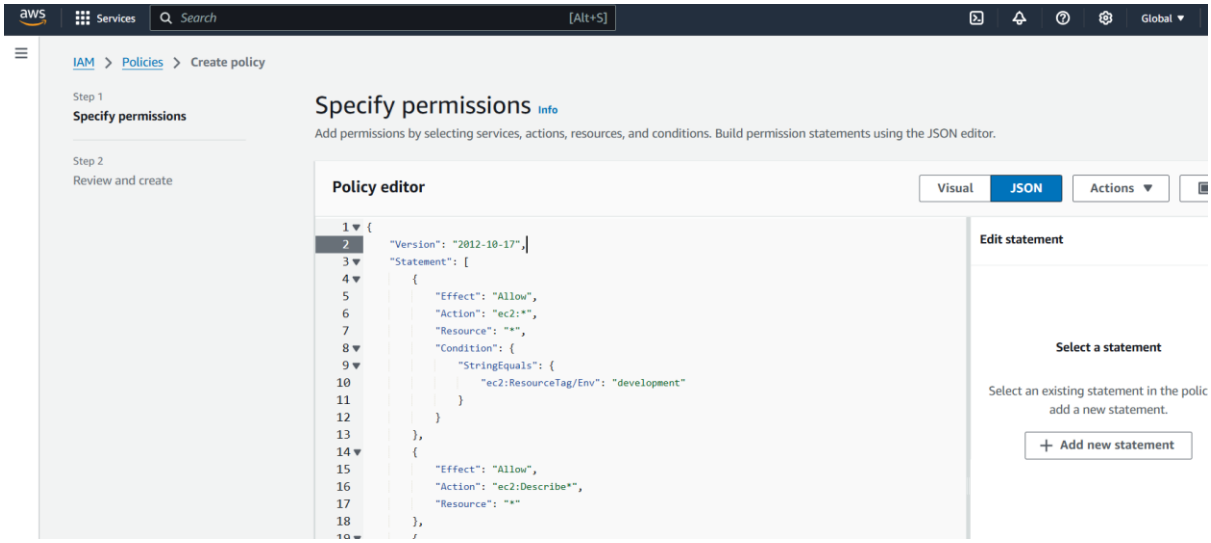
We will use JSON to create a policy, copy the code from the GitHub.

JSON stands for JavaScript Object Notion; I will explain you the JSON format in detail.

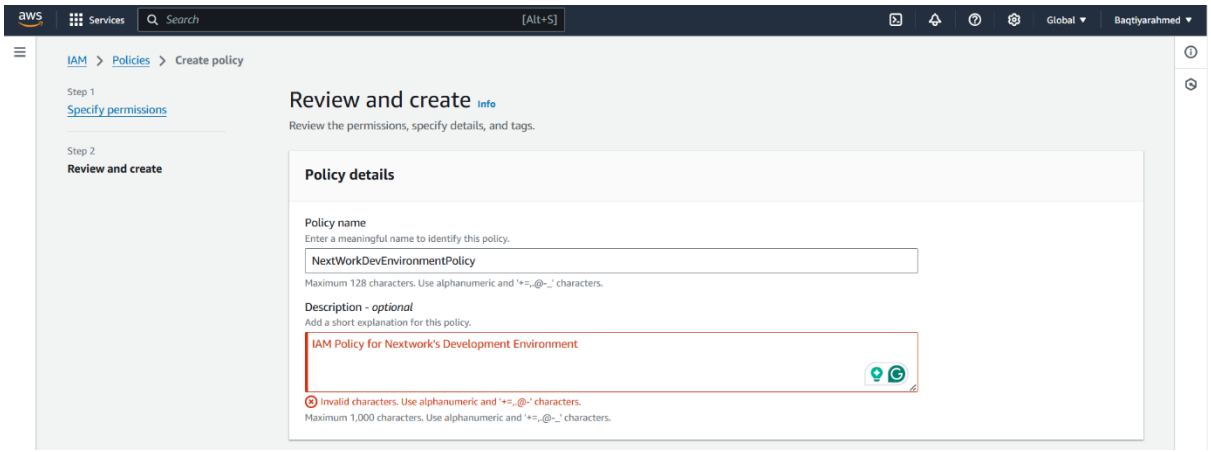
1. Version: This is the version of the policy language, for most AWS policies, this is 2012-10-17, which represents the latest policy.
2. Statement: The statement block is the core of a JSON policy. It contains one or more permissions that specify what actions are allowed or denied. It contains various subfields such as Effects, Action, Resource, and Conditions.
3. Effect: This has two states either ALLOW or DENY the specified actions. Deny has more priority over allow in cases of conflict.
4. Action: The Action subfield lists the actions that are affected by the policy. For example, “Action”: “ec2: *” indicates that all EC2 actions like (Start Instance, Stop Instance, terminate instance) are allowed. “*” means all possible actions are allowed for the specific service.
5. Resource: The resource section of JSON policy specifies what part of your cloud infrastructure the policy affects.
    a. Real-life example: Imagine your teacher is giving permission for students to use a computer lab. The Resource part is like specifying which computer the student is allowed to use. [“Resource”: “*”] = The students can use any computer in the lab. [“Resource”: “Computer1”] = The students can only use Computer1.
    b. In cloud terms, resources could be
        i. Specific EC2 Instances
        ii. S3 bucket
        iii. RDS database
6. Conditions: The condition block contains one or more conditions, The policy effect (Allow or Deny) is applied based on factors such as
    a. The time of day
    b. IP Address

c. Tags attached to resources
d. Whether the request is made through a secure connection (e.g., HTTPS)
    i. Real-life example: Think of **Conditions** as adding specific "rules" or "exceptions" to a general permission. For example, if you have a key to open a door (general permission), a condition would be like saying, "This key only works between 9 AM and 5 PM" or "The key works only if it's being used by someone wearing a uniform."

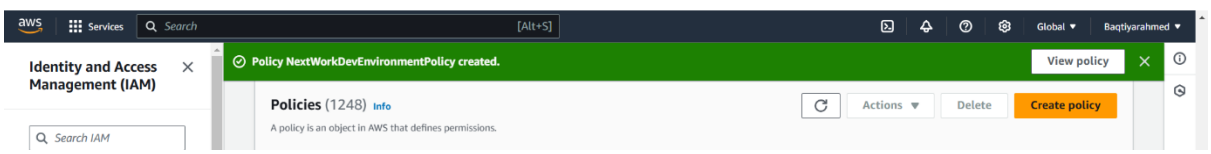
Copy the code from GitHub and paste it in the “specify permissions page”



Select Next and fill in the policy name and description



Turns out I got an error, this happened because I cannot use certain characters in the description text box. Remove the ‘symbol from the description and click create policy



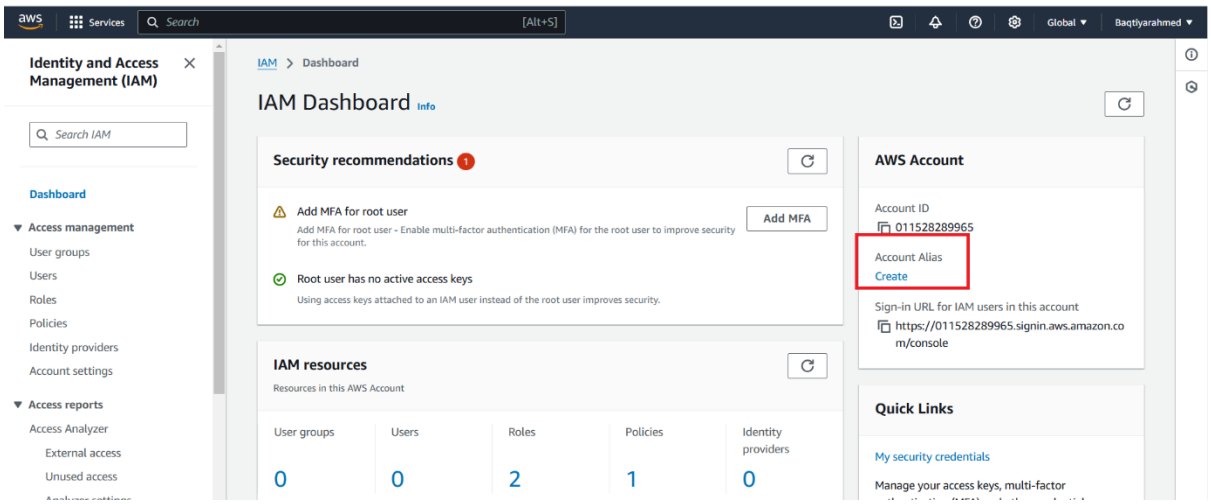
Now let's Create Account Alias

We are making it easier for users to log in to our AWS Account, using an Account Alias.

Once your on-board new intern to your AWS Account, these users get access through a unique log-in URL for your account. AWS Alias is a friendly name for your AWS account that you can use instead of your account id to sign in to the AWS Management Console.

Instead of Account ID you are using Account Alias, so that it would be easier to remember it.

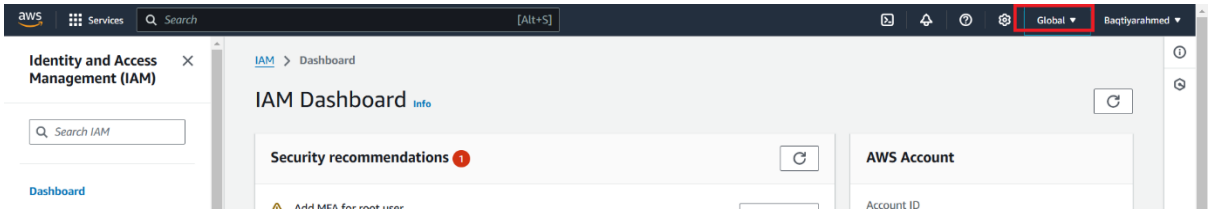
So, from the IAM dashboard click on create Alias



Successful page looks like this:

Now let's create IAM Users and IAM User-Groups

Before we create those, I want to mention one thing that, if you check the Region, it says Global because this service is available in every region not just a specific one.

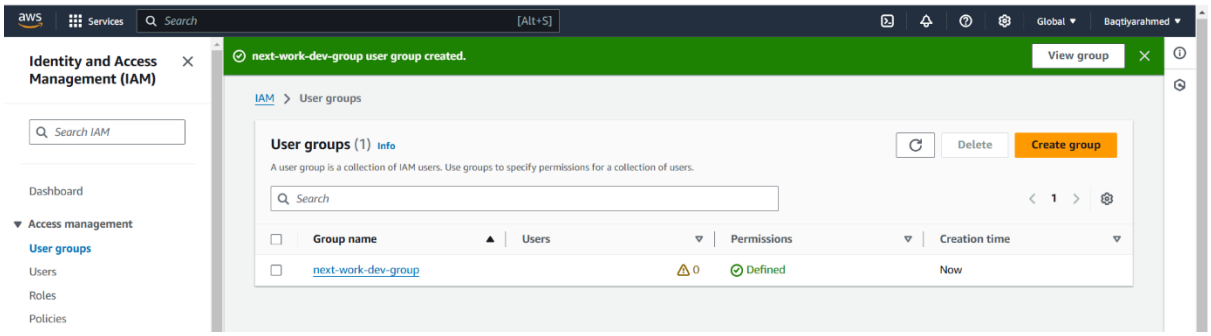


So now, we must create the username and password of our new employee, because you don't want to share your account details with them.

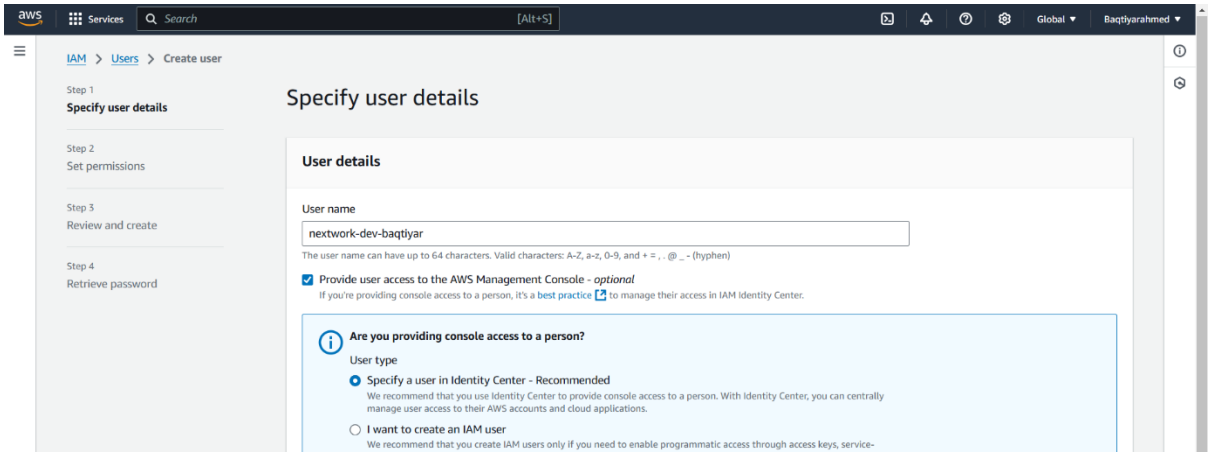
1. Create User group – for all the new employee with same level of permissions.
2. Create IAM users – so they have a way to log in.

So, from dashboard click on "User Groups" then click on "Create Group"

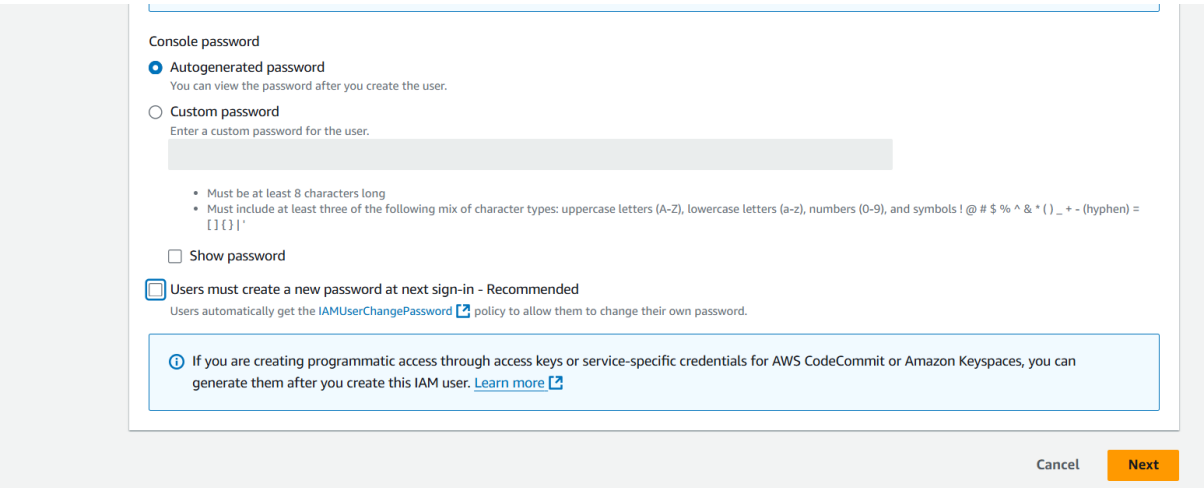
Attach the permission policy we created and select create user group



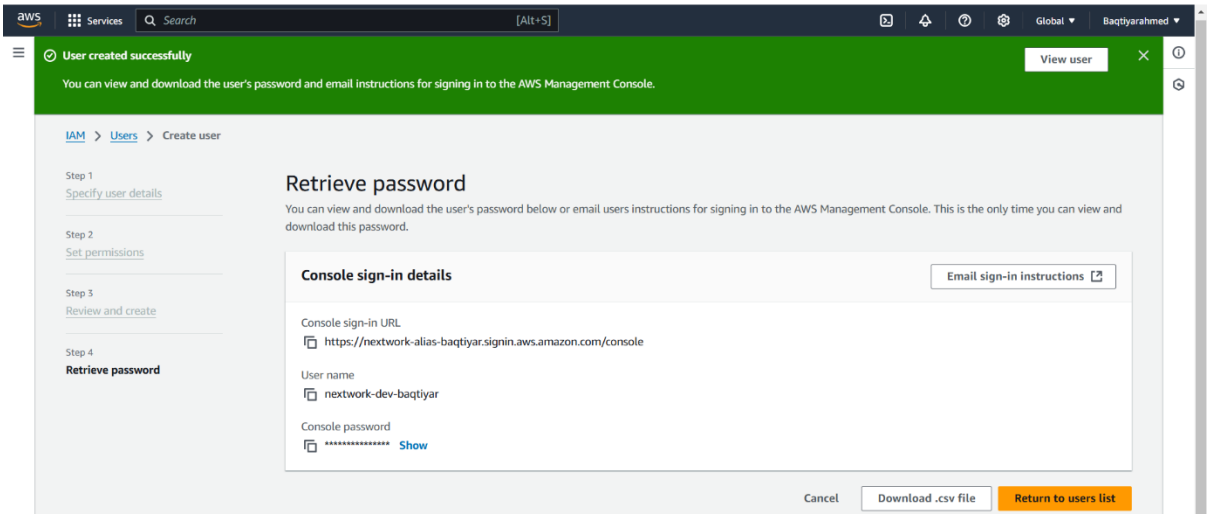
Now let's add users in this



In the image above you check after giving the name of the user, I have checked a box called provide user access to the AWS Management Console. If you don't check this box then users won't be able to access the AWS Management Console, however they can still access it through more advanced methods such as (AWS CLI, SDK's, API's).



In the above image I have unchecked the box which says "Users must create a new password at next sign-in, however in the real world we will always check the box.

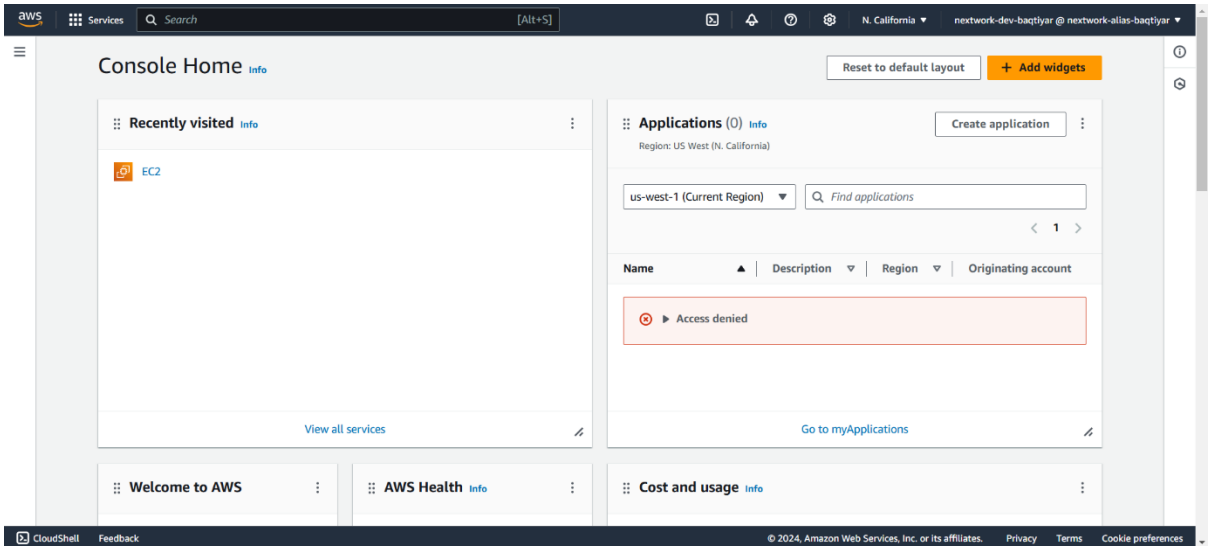


I have successfully created the user, follow the step to create it, now you can see the account credentials in the page above. You can also download the .csv file to your local machine.

Now before passing the account details to our new employee, we will test the Employee Access. That way you can make sure that you have given the right permissions to the account.

Now, copy the Console sign-in URL. Do not close the tab!

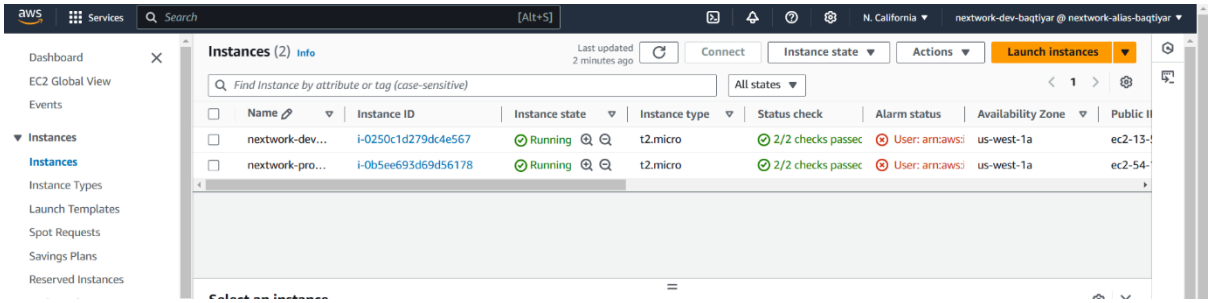
Open a new incognito window on your browser and paste the link address and try signing in using the details given.



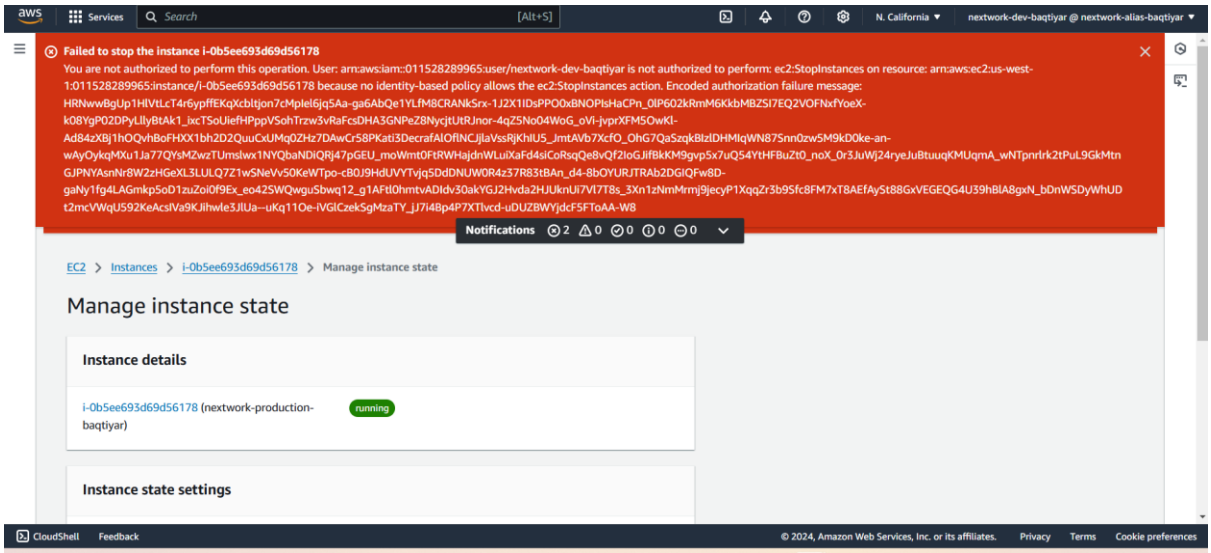
This is the interface of the new employee access level.

Head to the EC2 Instance, also make sure you are in the same region as before.

The screen should look something like this

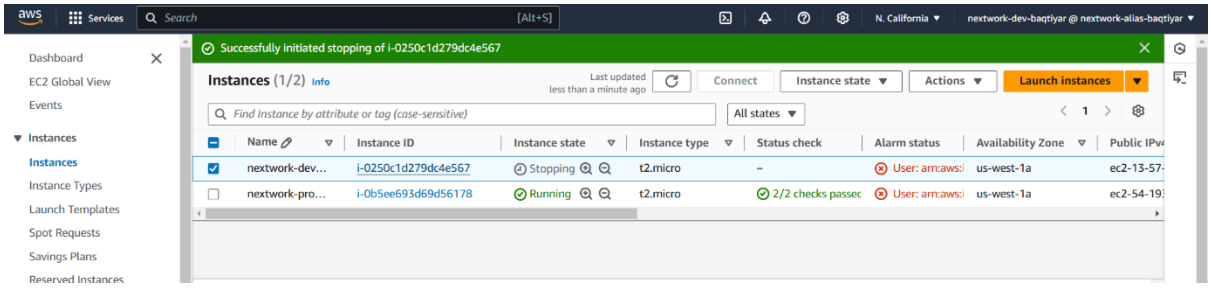


Select production server and go to Actions dropdown menu and click Manage Instance State. And try to stop the instance.



It displays are red warning sign saying that “you are not authorized to perform this operation”, This means that we have provided the right permissions to our new employee. The new employee cannot stop the production server with the tag: production.

Now let’s try this for development server, head back to the Instance page and select development server and try to stop it.



We can easily stop the development server because in the IAM Policy we mentioned that we are allowed to make any changes with the tag: development.

WOOHOOOOOOOO, YOU DID IT 👏

Congratulations on successfully using AWS IAM to control and test user permissions.

Wait before you go don’t forget to delete the AWS resources you have utilized.

- ➔ Terminate the two EC2 Instance.
- ➔ Delete the User Groups
- ➔ Delete the User
- ➔ Remove the IAM Policy we created.

QUICK RECAP

Today you have learned how to:

- ➔ Launch EC2 Instance
- ➔ Use tags for easy identification
- ➔ Set-up IAM policies accessing EC2 Instances (production or development)
- ➔ Create IAM user and assign them to appropriate user-group with necessary permissions.
- ➔ Test IAM access for the User you’ve created.

You are a Cloud Engineer Now

THANK YOU