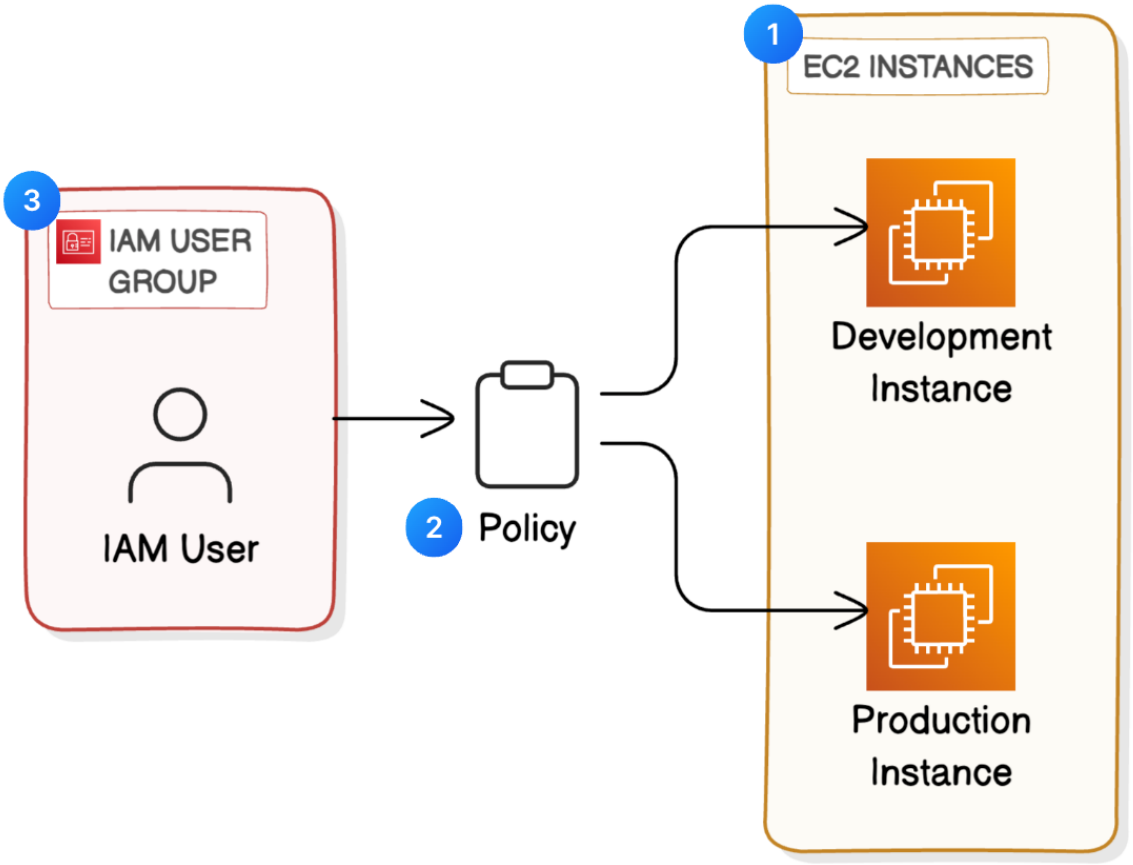
In AWS, a **user** refers to a person or application that interacts with AWS resources. To manage and secure this interaction, AWS provides **Identity and Access Management (IAM)**, a service that controls who can authenticate (sign in) and what actions they can perform (authorization) within your AWS account. IAM allows you to define fine-grained permissions, ensuring that users only have access to the resources necessary for their roles, enhancing overall security in your cloud environment.

Get ready to create (and learn from scratch):

1. 💻 EC2 instances
2. 📏 IAM Policies
3. 👥 IAM Users and User Groups
4. 🚀 AWS Account Alias



What is EC2?

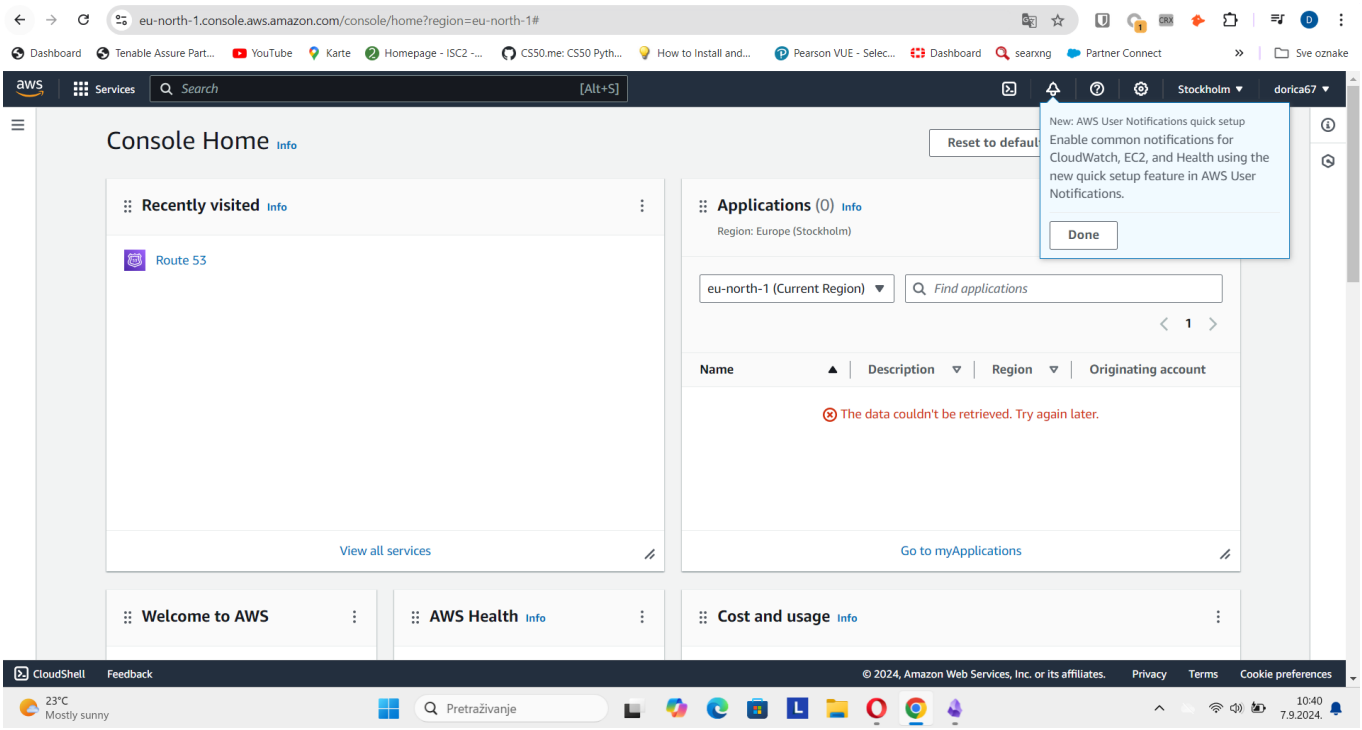
A legendary AWS service! Amazon EC2 is a service that lets you rent and use virtual computers in the cloud. They're like your personal computers, but they exist on the internet instead of

being physically in front of you. You can create, customize, and use these computers for all different reasons, from running applications to hosting websites.

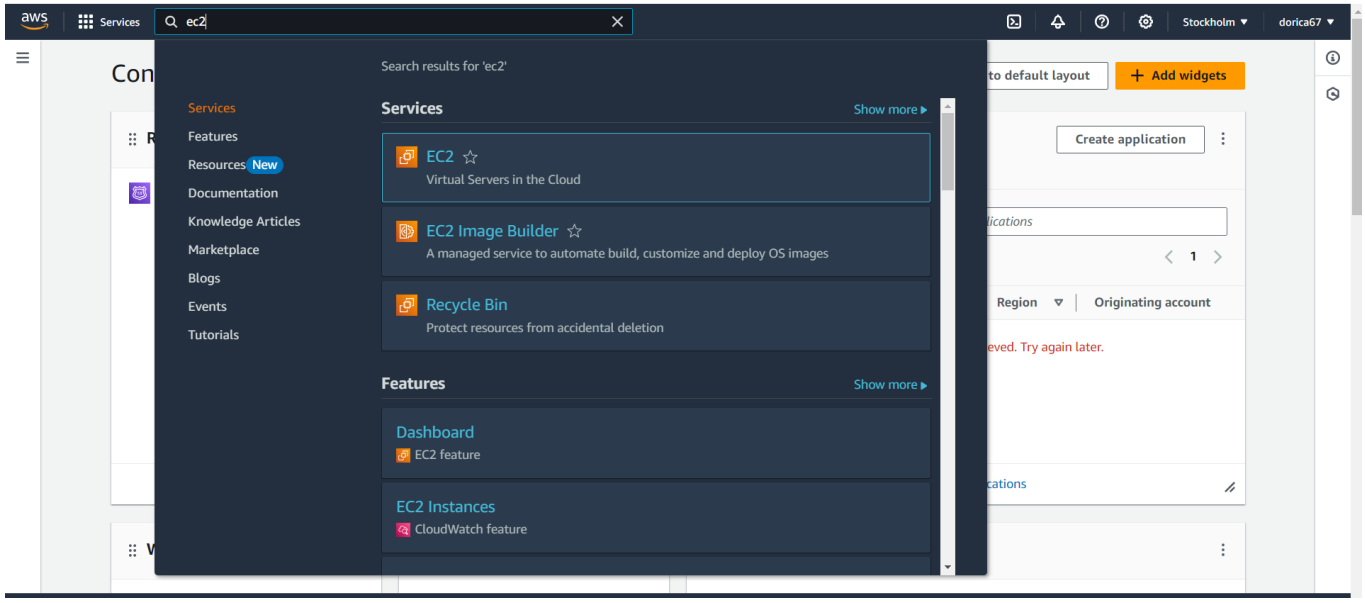
Psssst... EC2 = **Elastic Compute Cloud.**

Here's what the three words mean:
Elastic = flexible. This service can easily adapt and change in size and power to fit your needs.
Compute = computing power. EC2 provides virtual computers that can do various tasks, just like your personal computer.
Cloud = available over the internet.

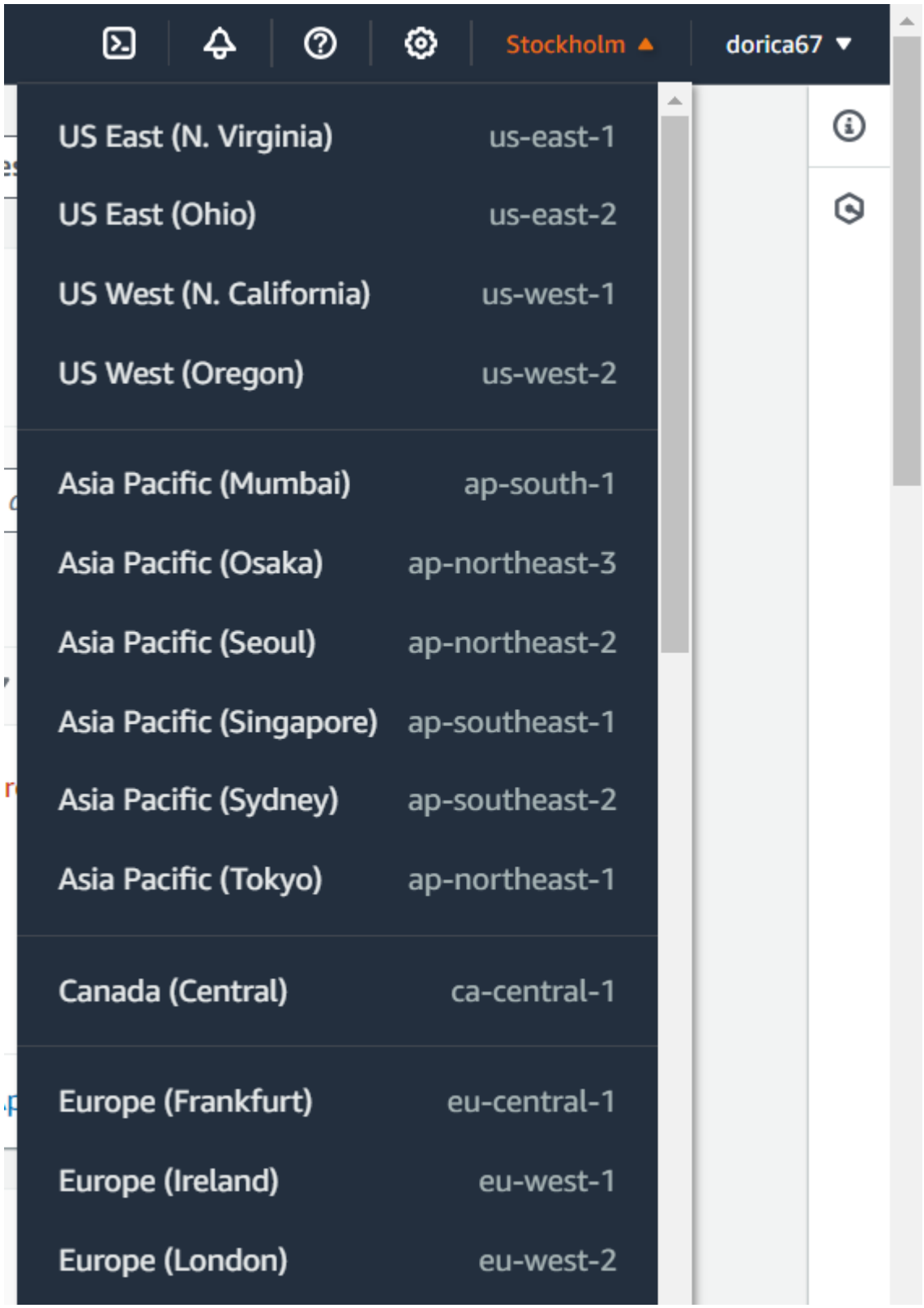
Login to AWS console:



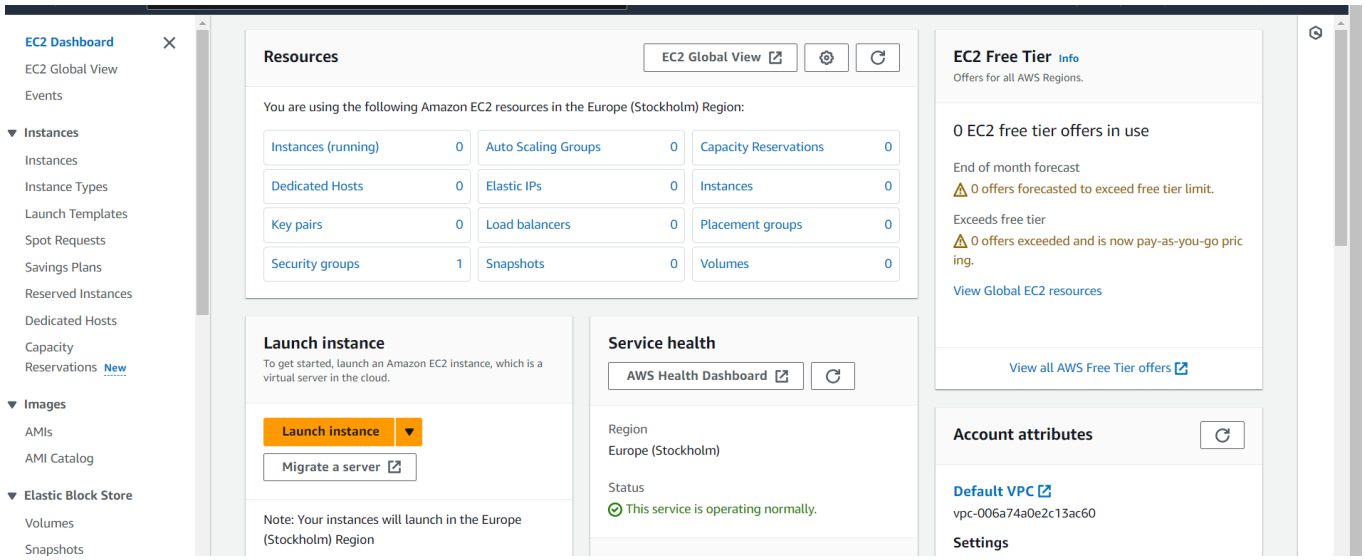
Head to EC2:



Switch to region to choose one closes to you:



In EC2 instance choose launch instance:



💡 What are EC2 instances?

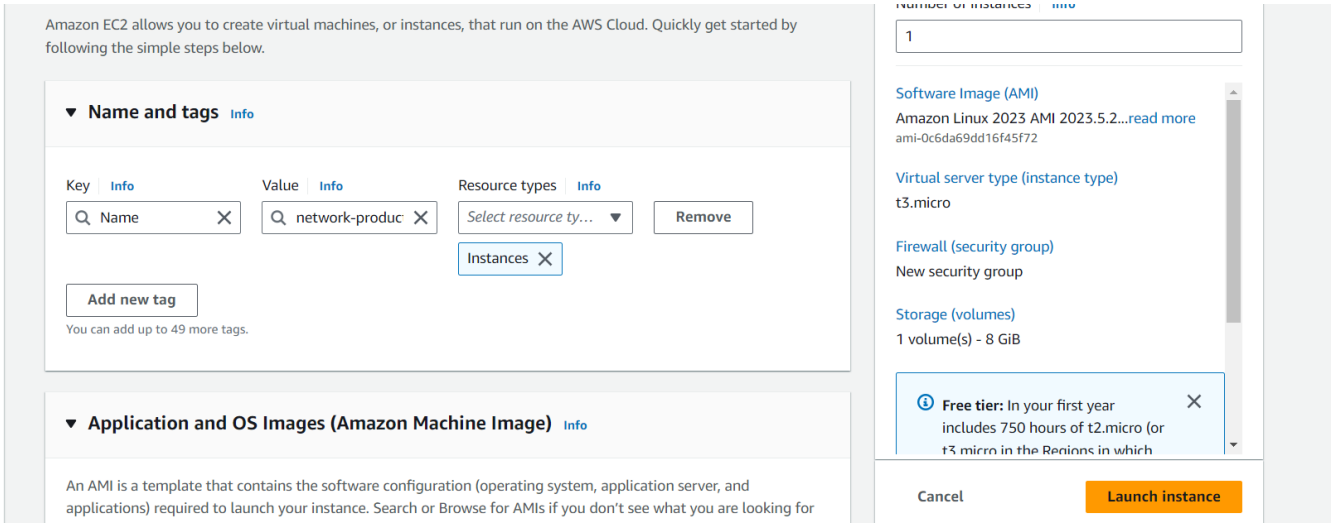
If EC2 is the service that provides virtual computers/servers, each instance is one of those computers/servers that gets produced.

Just like you can choose a computer with more memory or a faster processor when you buy a laptop, with EC2 instances, you can pick a virtual computer that fits what you need for your projects. You can customize your EC2 instance's CPU, memory, storage, and networking capacity and more!

- Let's set up your EC2 instance!
- In **Name**, enter the value `nextwork-production-yourname` . Yup, replace **yourname** with your name.

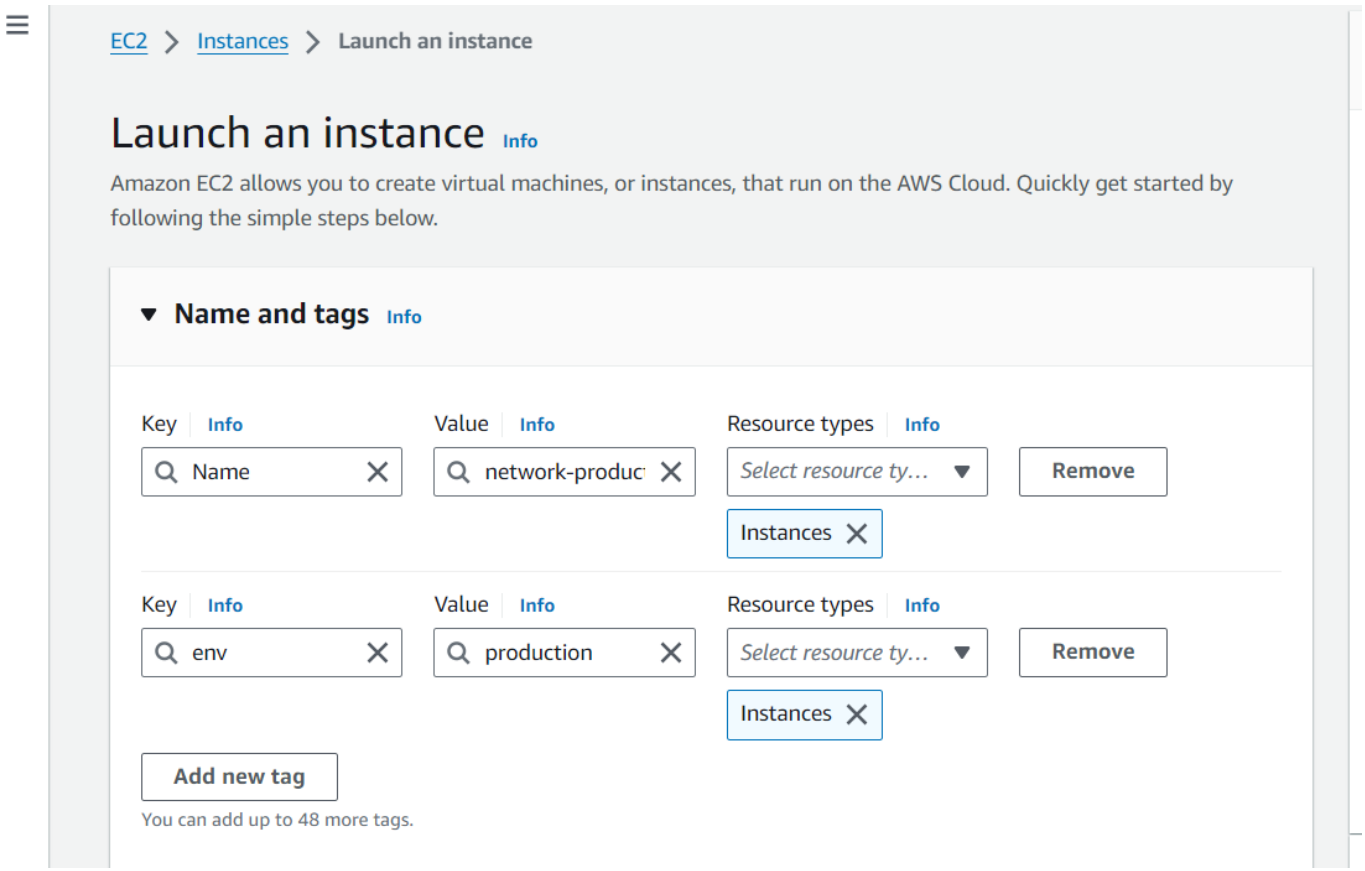
> 💡 Every EC2 instance must have a unique name in its AWS Region.

You can add aditional tags



- Choose `Add new tag` .

- For the next tag, use this information:
  - Key: `Env`
  - Value: `production`



Why are we creating a new tag? What does this tag mean, how will it be useful later?

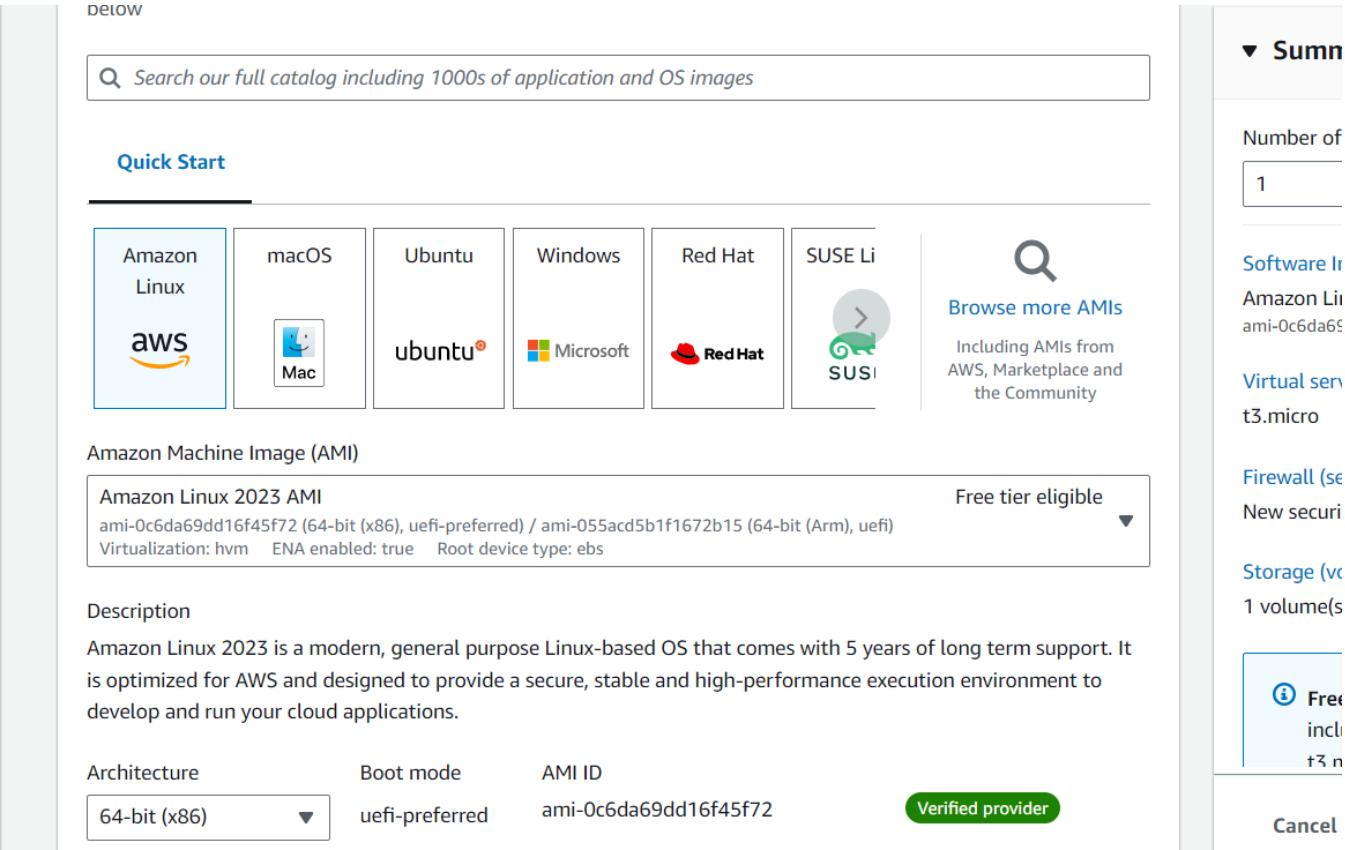
Tags are like labels you can attach to AWS resources for organization.

In this case, we're creating a tag called "Env" with a value of "production" or "development" to label the instances used in production vs development environments.

This tagging helps us with identifying all resources with the same tag at once (they are useful filters when you're searching for something), cost allocation, and applying policies based on environment types. You'll see the last point about policies in action soon!

- Head on down to see your EC2 settings and make sure the **Amazon Machine Image (AMI)** is using a **Free tier eligible** option.

We see that free tier eligble option is used in this example



What is AMI? What is Free tier eligible?
When you buy a new computer off the shelf, most computers already have some software and the operating system (e.g. MacOS, Windows) already configured and set up for you!

AMI stands for Amazon Machine Image, and it's very similar to those pre-built computers. An AMI is a template or blueprint used to create EC2 instances and contains the operating system along with the applications needed to launch the instance.

Free tier eligible AMIs are those that qualify for the AWS Free Tier, so you won't get charged for using it.

What is AMI? What is Free tier eligible?
When you buy a new computer off the shelf, most computers already have some software and the operating system (e.g. MacOS, Windows) already configured and set up for you!

AMI stands for Amazon Machine Image, and it's very similar to those pre-built computers. An AMI is a template or blueprint used to create EC2 instances and contains the operating system along with the applications needed to launch the instance.

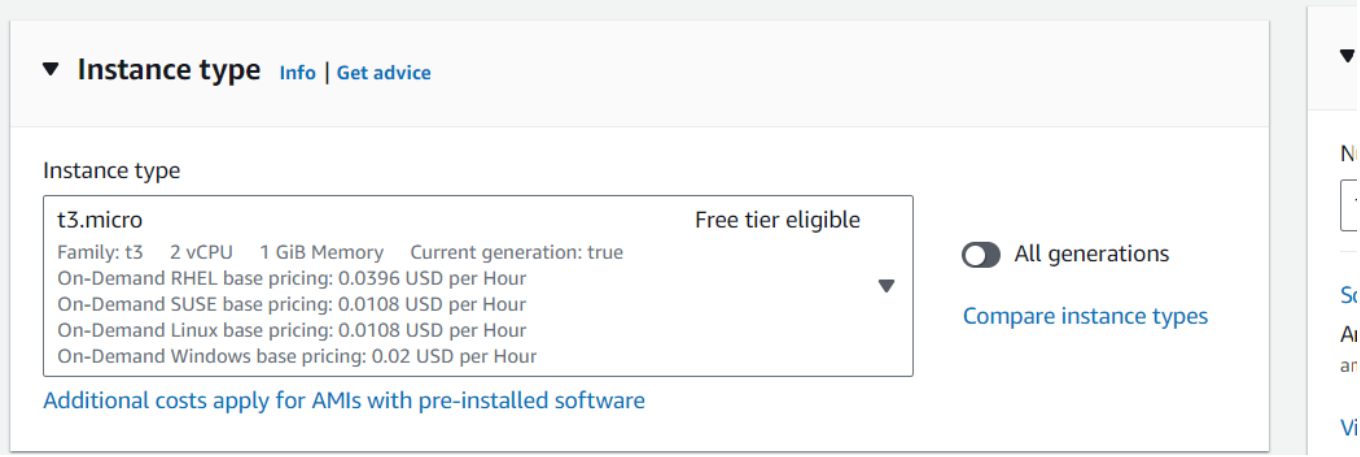
Free tier eligible AMIs are those that qualify for the AWS Free Tier, so you won't get charged for using it.

- For the instance type, also make sure you're using a **Free tier eligible** option!

- **What is instance type?**
  If AMIs give you pre-built software and operating systems, instance types cover the '**hardware**' components.

CPU power, memory size, storage space and more!

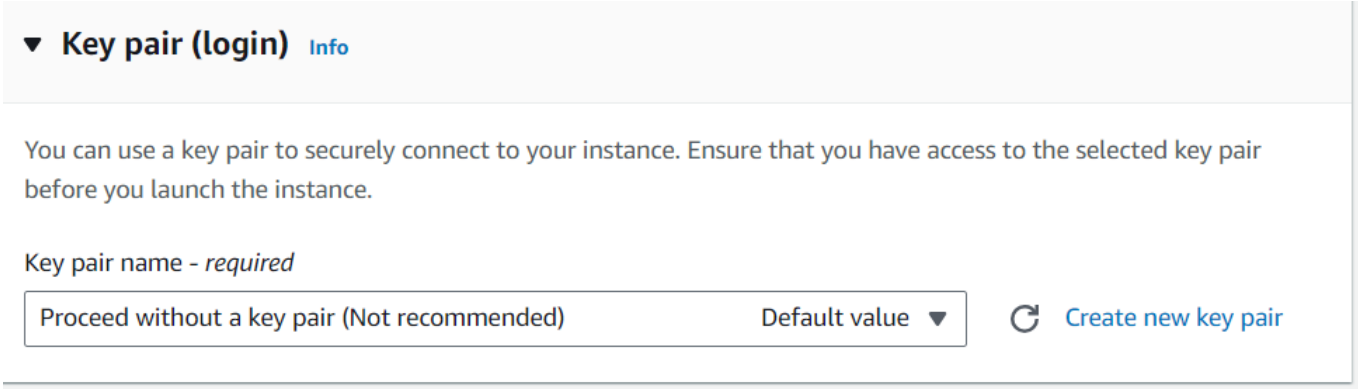
So, while the AMI decides what operating system your server runs, the instance type determines how fast and powerful it performs.



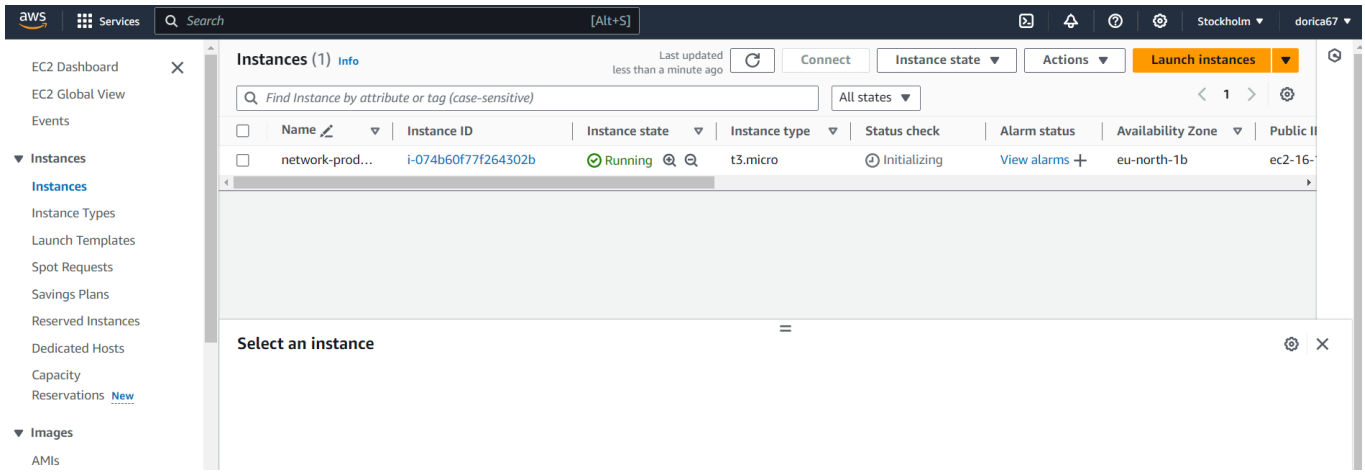
What is a key pair? Why does it say (Not recommended) next to proceeding without one?

A key pair is primarily used for accessing your EC2 instance securely **without going through the AWS Management Console**. Instead of the Management Console, you're using SSH (Secure Shell) Access with your key pairs - this is out of scope for this project, but you'll learn more about SSH and key pairs in a networking or compute-themed project!

Proceeding without a key pair means you won't have SSH (Secure Shell) access to your instance, which is generally not recommended because it limits your ability to troubleshoot or manage your EC2 instance through a secure way outside of the Console. It's always safer and more flexible to have a key pair set up, so you would create a key pair for bigger projects that you work on over a longer period of time.



Click on launch instance



💡 **Hold up... what are the settings we've skipped just now?**

We skipped configuring network and storage settings for simplicity in this project. These settings are crucial for fine-tuning your EC2 instances' performance, security, and connectivity, but for this project, we'll focus on the basic steps of launching instances with minimal configuration.

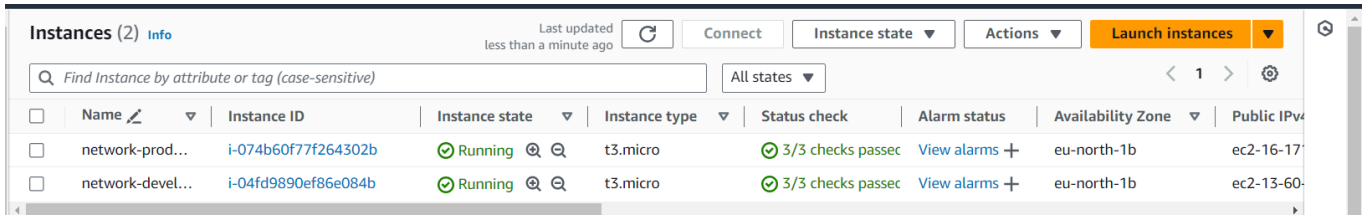
Network settings define how your instances interact with the internet and other AWS resources, determining factors like IP addresses and network routing.

Storage settings involve choosing the type and size of storage volumes (like hard drives) that your EC2 instance will use to store data.

- Now let's create one more EC2 instance for the **development environment**.

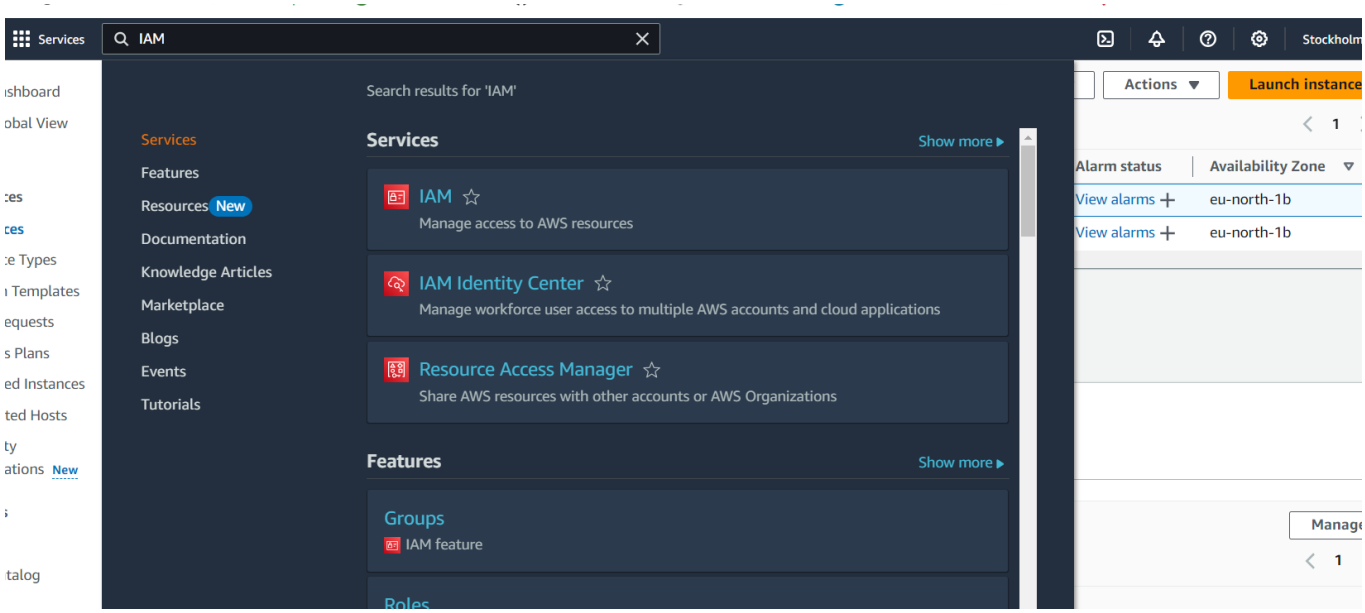
> 💡 **What do the development vs production environments mean?**
> Development and production environments refer to different stages in the software development lifecycle.
>
> The **development** environment is where developers write, test, and debug code before it's deployed to **production**, which is the live environment that your end users can use!

Now the instance can be launched:



Create an IAM policy

Head to your IAM console



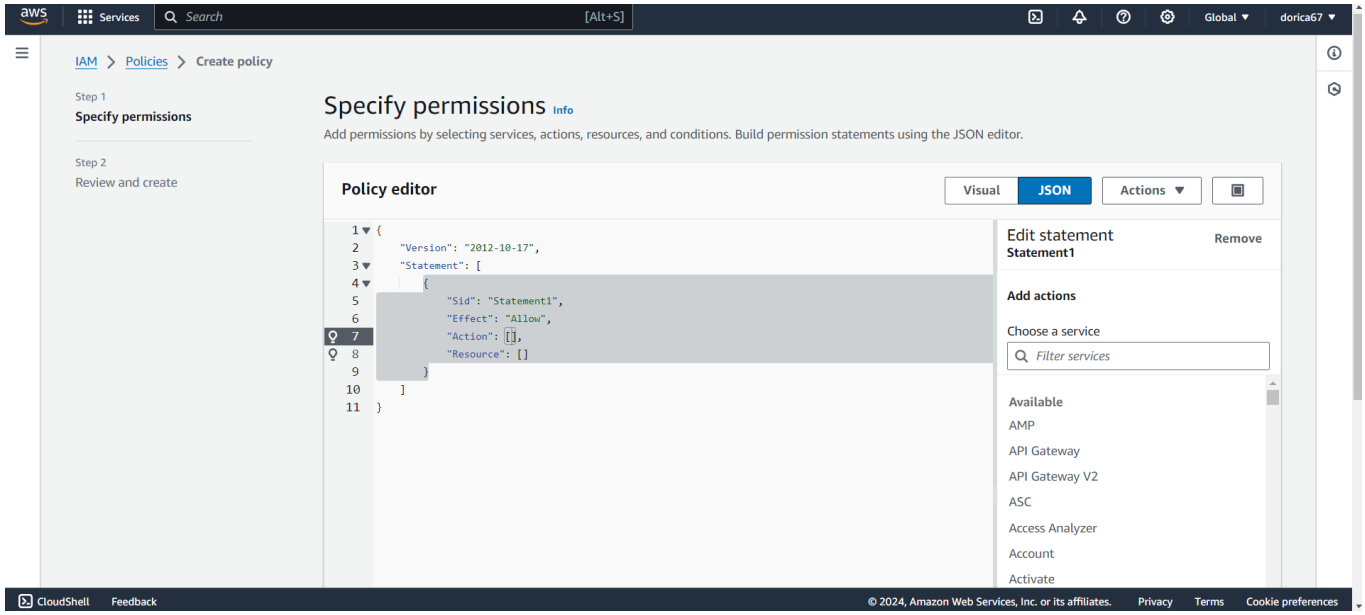
What is IAM?

IAM stands for Identity and Access Management. You'll use AWS IAM to manage the access level that other users and services have to your resources.

- Now on the left-hand navigation panel of your IAM console, choose **Policies**.

What is a Policy?

An IAM policy is a rule 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.



- Choose `Create policy`.
- Switch your Policy editor tab to JSON.

Let's unpack this spicy policy
This policy 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.

An extra for the curious: how JSON policies are structured
Version
This means 2012-10-17 is the date of the latest policy version. This tells you whether the policy is up to date with the latest standards and practices.

Statement
The main part of the policy structure and defines a list of permissions.

Effect
This can have two values - either **Allow** or **Deny** - to indicate whether the policy allows or denies a certain action. **Deny** has priority. Looking at the first statement, "Effect": "Allow" means this statement is trying to allow for an action.

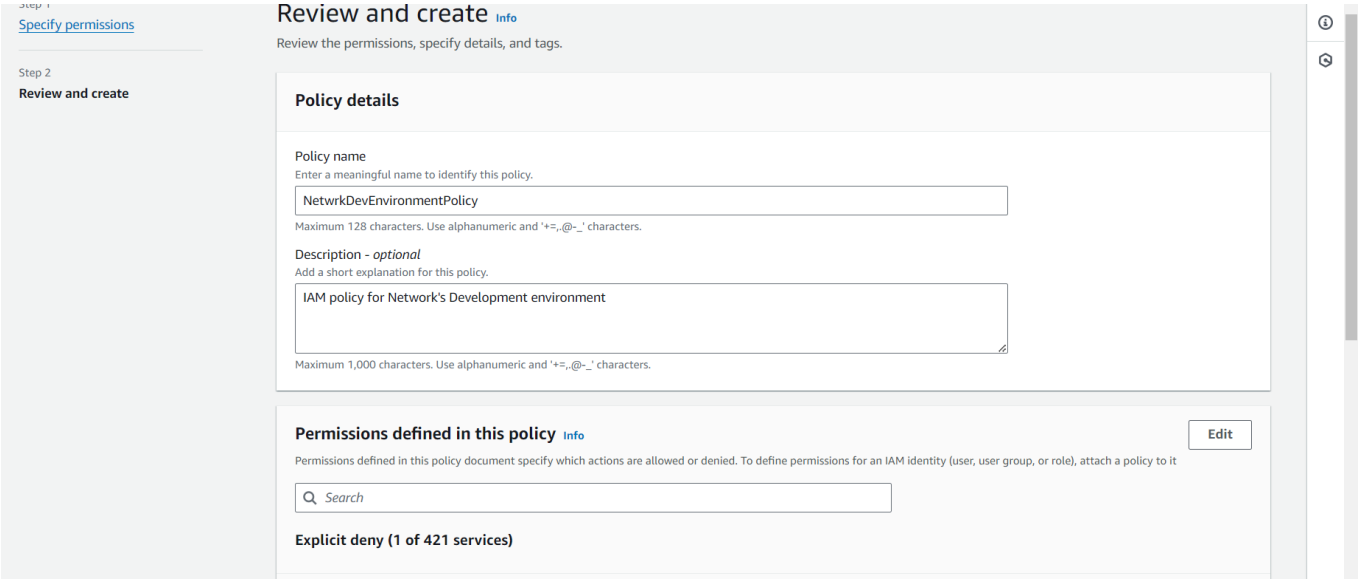
Action
A list of the actions that the policy allows or denies. In this case, "Action": "ec2:*" means all actions that you could possibly take on EC2 instances are allowed. Woohoo!

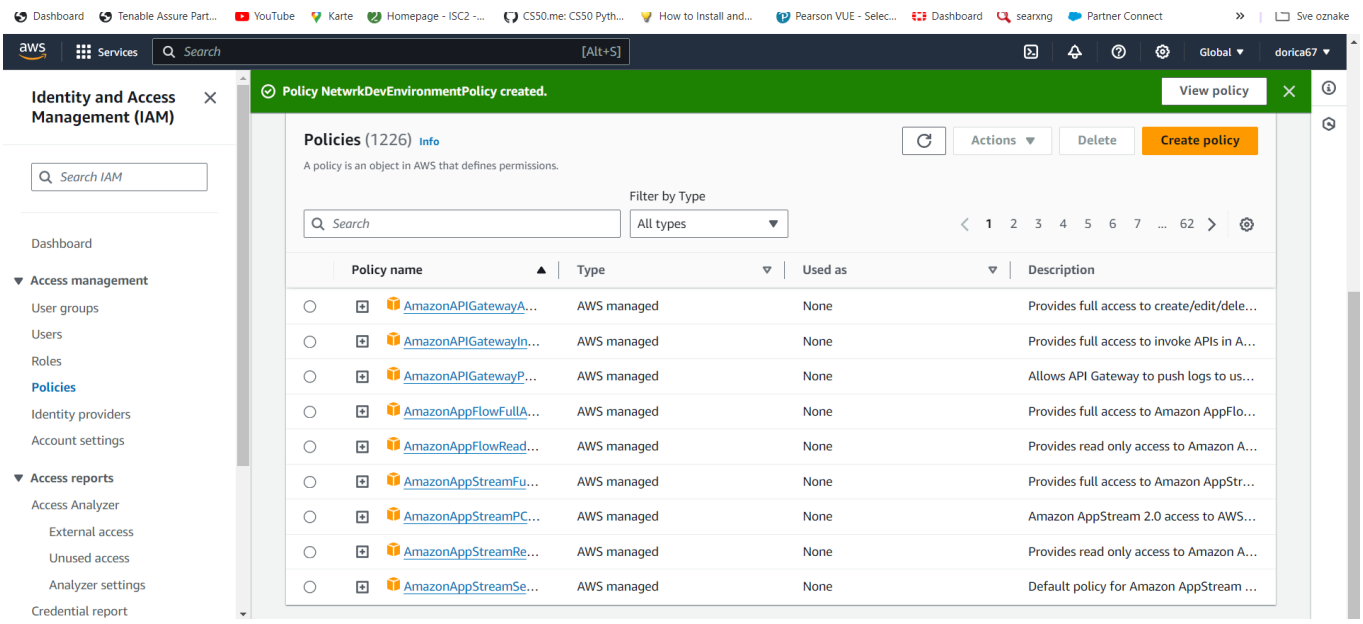
Resource
Which resources does this policy apply to? Specifying "*" means all resources within the defined scope (see the next point).

Condition Block (optional)
The circumstances under which the policy is in action. In this case, the condition is that the resource is tagged **Env - development**. This means specifying "Resource": "*" in the line above means all resources with the **Env - development** tag are impacted by your statement.

- Select `Next` when you're ready.
- Fill in your policy's details:
  - Name: `NextWorkDevEnvironmentPolicy`
  - Description: `IAM Policy for NextWork's development environment.`
- Choose `Create policy`.
- Oh no! Turns out there's a rule for the characters allowed in your Policy description. Edit this description to get rid of that error (can you tell which character is not valid? There's a hint given to you right underneath the Description's text box)!

- Choose `Create policy` again when you're done.





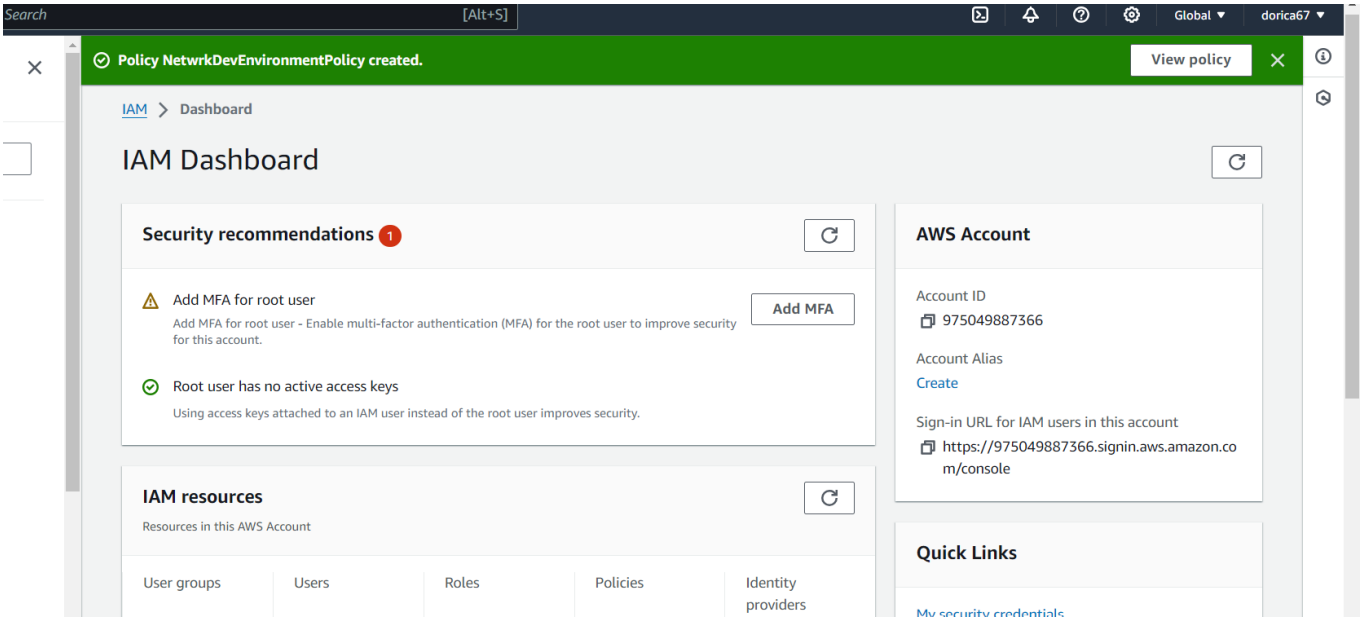
💡 Feeling stuck? Remove the apostrophe (i.e. ') 😉

Create an AWS account alias

- Head to your IAM dashboard.
- In the right-hand side of the dashboard, choose `Create` under **Account Alias**.

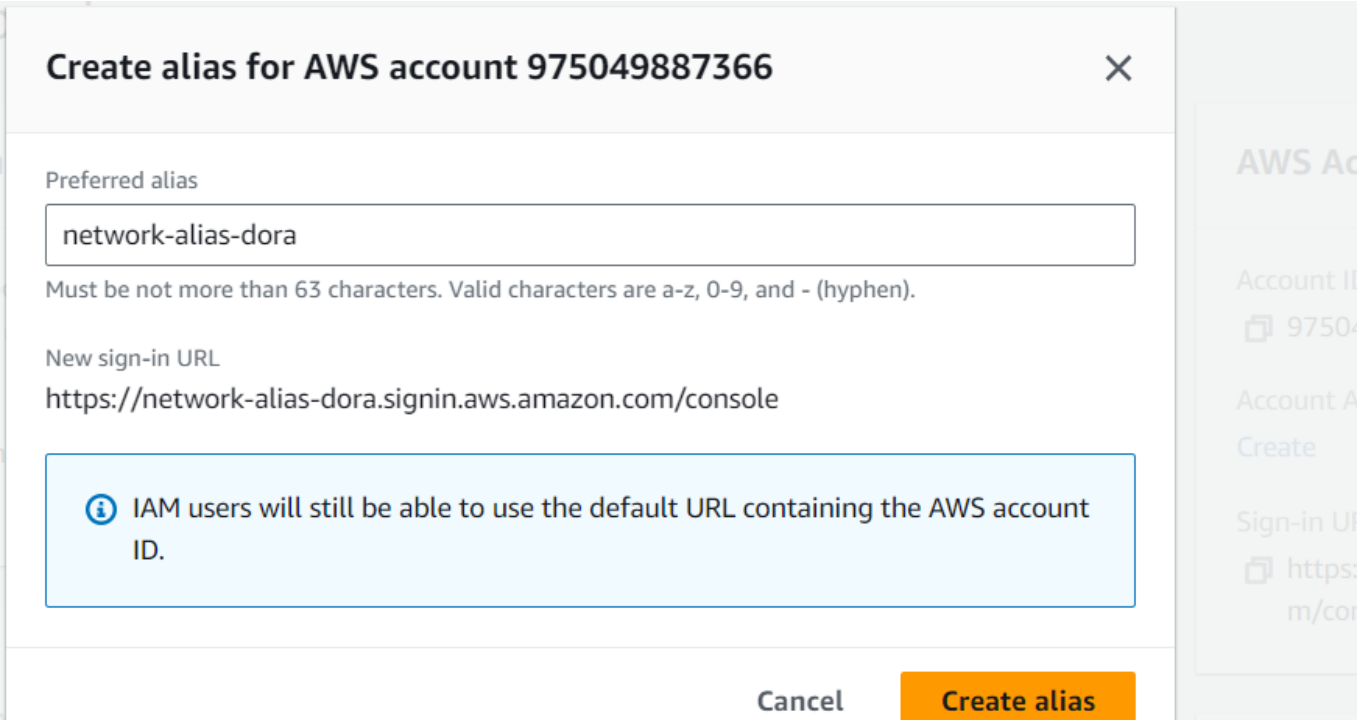
> 💡 **What is an Account Alias? Why are we creating one?**
> Once you onboard new users into your AWS account (which we'll do for our new NextWork intern), these new users get access through a unique log-in URL for your account.
>
> An Account Alias is a friendly name for your AWS account that you can use instead of your account ID (which is usually a bunch of digits) to sign in to the AWS Management Console.

> Your AWS account's sign-in page has this URL by default: https://**Your_Account_ID**.signin.aws.amazon.com/console/
>
> If you create an AWS account alias for your AWS account ID, your sign-in page URL looks more like: https://**Your_Account_Alias**.signin.aws.amazon.com/console/
>
> You would create an alias to make it easier to remember and share your AWS console's login URL with others e.g. NextWork's new intern. Companies often use this so that their AWS account sign-in page is more user-friendly for their users!



We have to choose create in Account alias

- In the **Preferred alias** field, enter `nextwork-alias-yourname` . Yup, replace `yourname` with your name!

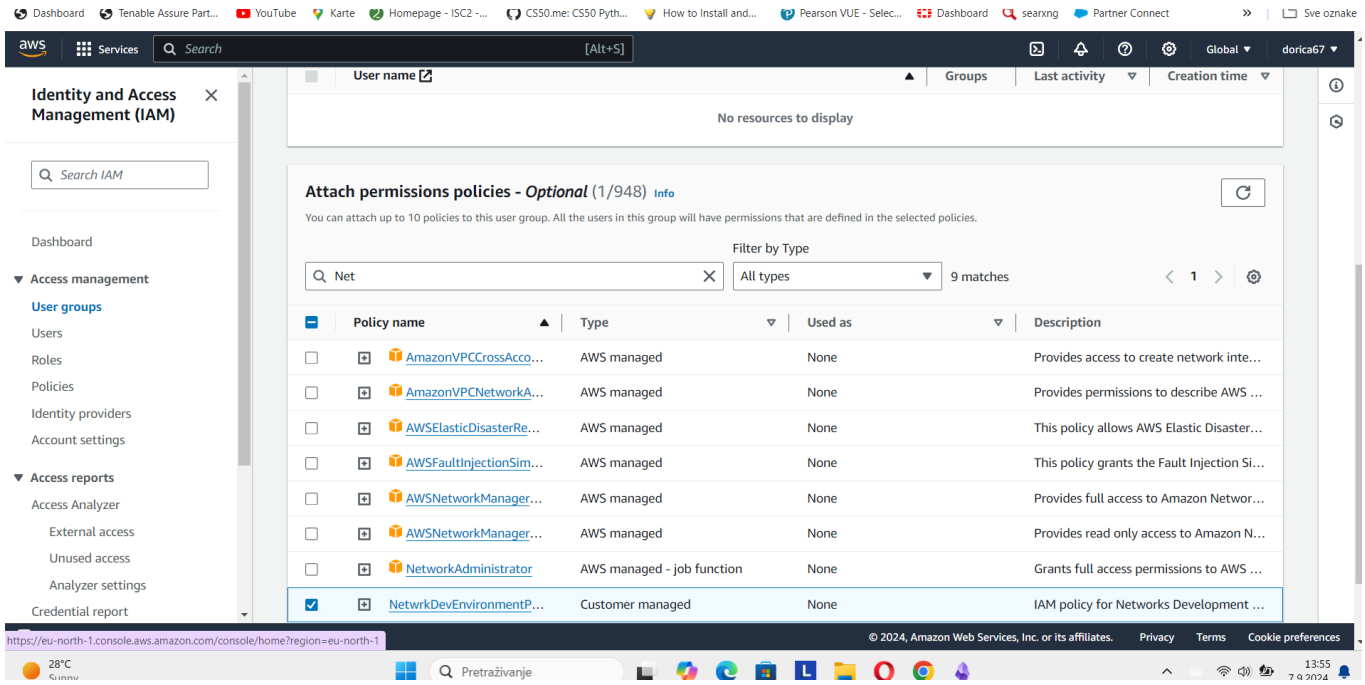


Create IAM users and Users group

- Choose `User groups` in your left-hand navigation panel.
- Choose `Create group` .
- Let's create your first user group!

> 💡 **What is an IAM user group?**
> An IAM user group is a collection/folder of IAM users. It allows you to manage permissions for all the users in your group at the same time by attaching policies to the group rather than individual users.

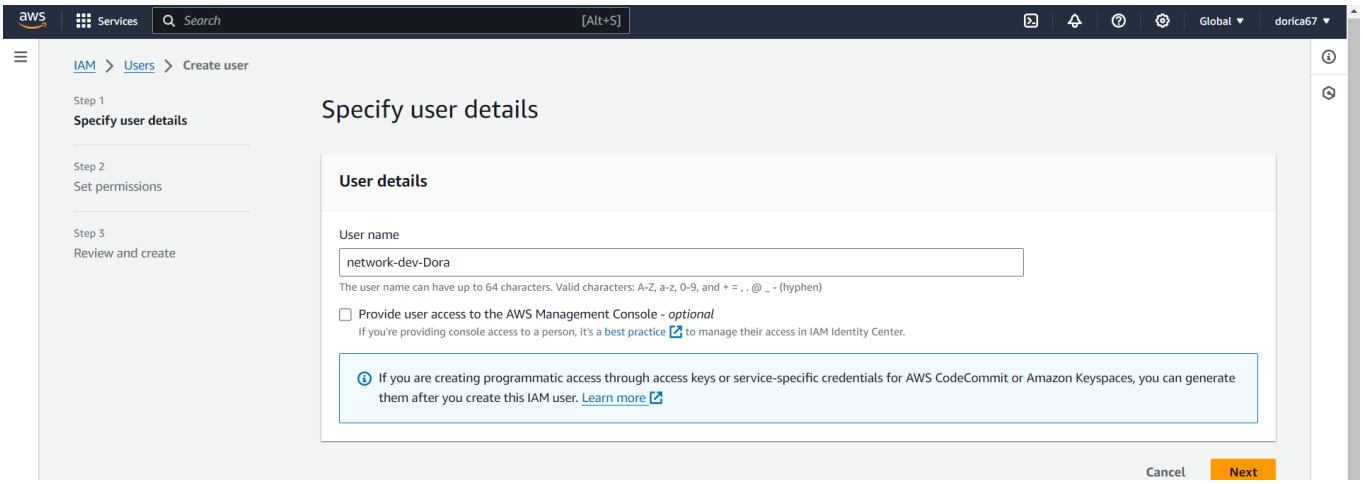
- To set up your user group:
  - Name: `nextwork-dev-group`
  - Attach permission policies: `NextWorkDevEnvironmentPolicy`



- Now let's add Users to your user group.

> 💡 **Why do we need users in our user group?**
> IAM users are the people that will get access to your resources/AWS account, whereas user groups are the collections/folders of users for easier user management.
>
> We're adding users to
>
> `nextwork-dev-group`
>
> to grant them the permissions associated with that group.
>
> This simplifies managing permissions and ensures consistency across users who have similar access to AWS resources. Imagine if you have a whole team of 5 interns (users) that need the same permission settings next Summer!

- oose `Users` from the left-hand navigation panel.
- Choose `Create user` .
- Let's set up this user! Under **User name**, enter `nextwork-dev-yourname` .
- Tick the checkbox for `Provide user access to the AWS Management Console`

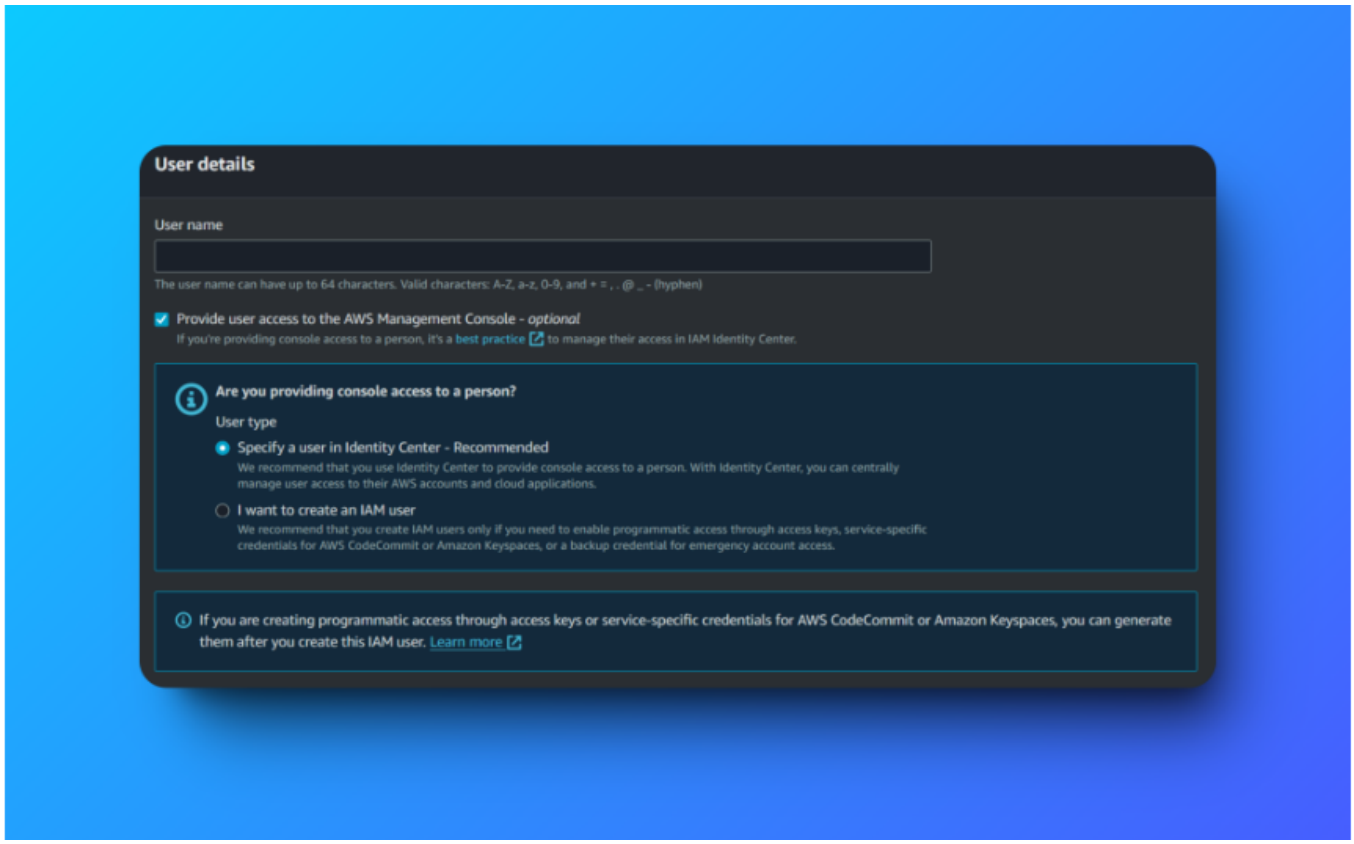


> **Why are we ticking this box?**
> If you don't tick this box, your new user won't get to sign in and access AWS services through the Console. They'll have to access AWS services through other, more advanced methods - we'll cover those advanced methods (e.g. AWS CLI, SDKs, APIs) in a future project!

- Uncheck the box for **Users must create a new password at next sign-in - Recommended**.

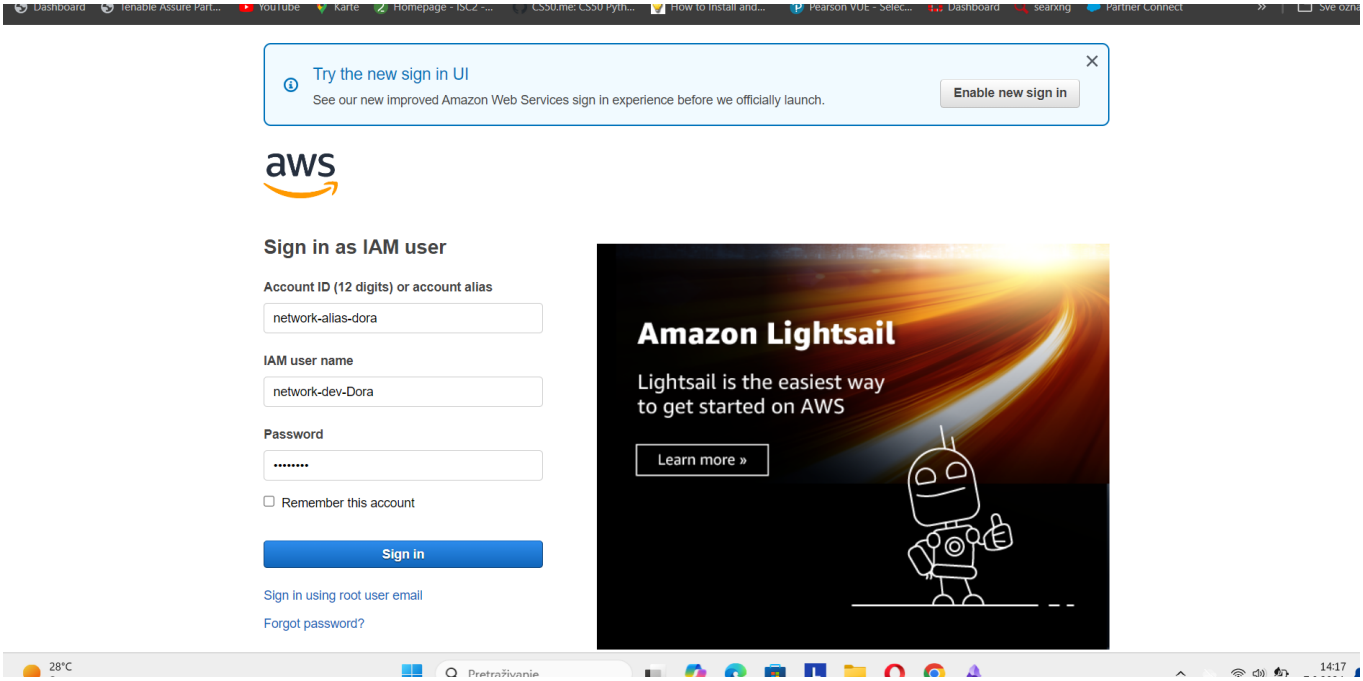
> 💡 Ahem... in the real world, you should absolutely leave this box checked! We are leaving it unchecked because you'll have to create a new password for this user, which is irrelevant to our learning objectives for today.

Note: This does not show up for every AWS Account, but if you see a highlighted pop-up that asks "**Are you providing console access to a person?**" - select **I want to create an IAM user**.



- Select `Next` when you're ready!
- To set permissions for your user, we'll simply add it to the user group you've created. Select the checkbox next to `nextwork-dev-group`.
- Select `Next`.
- Select `Create user`!

Test your user's access



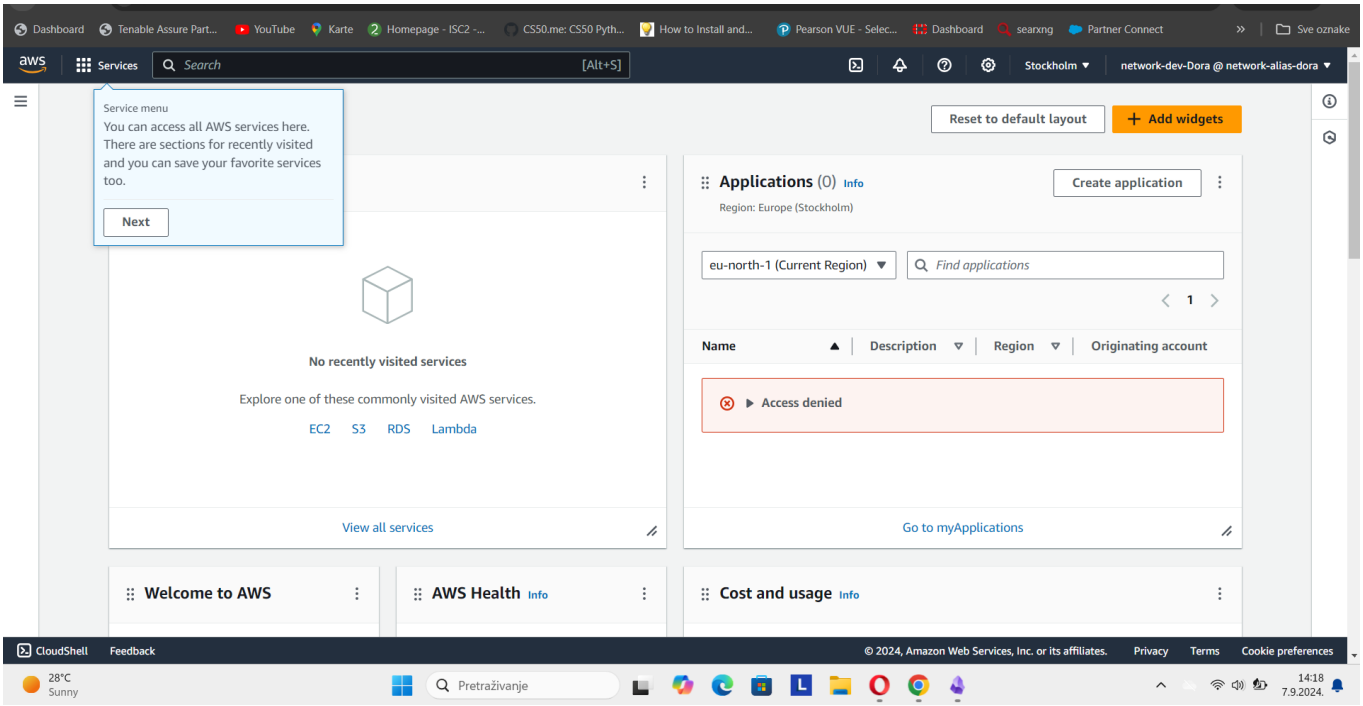
- Copy the

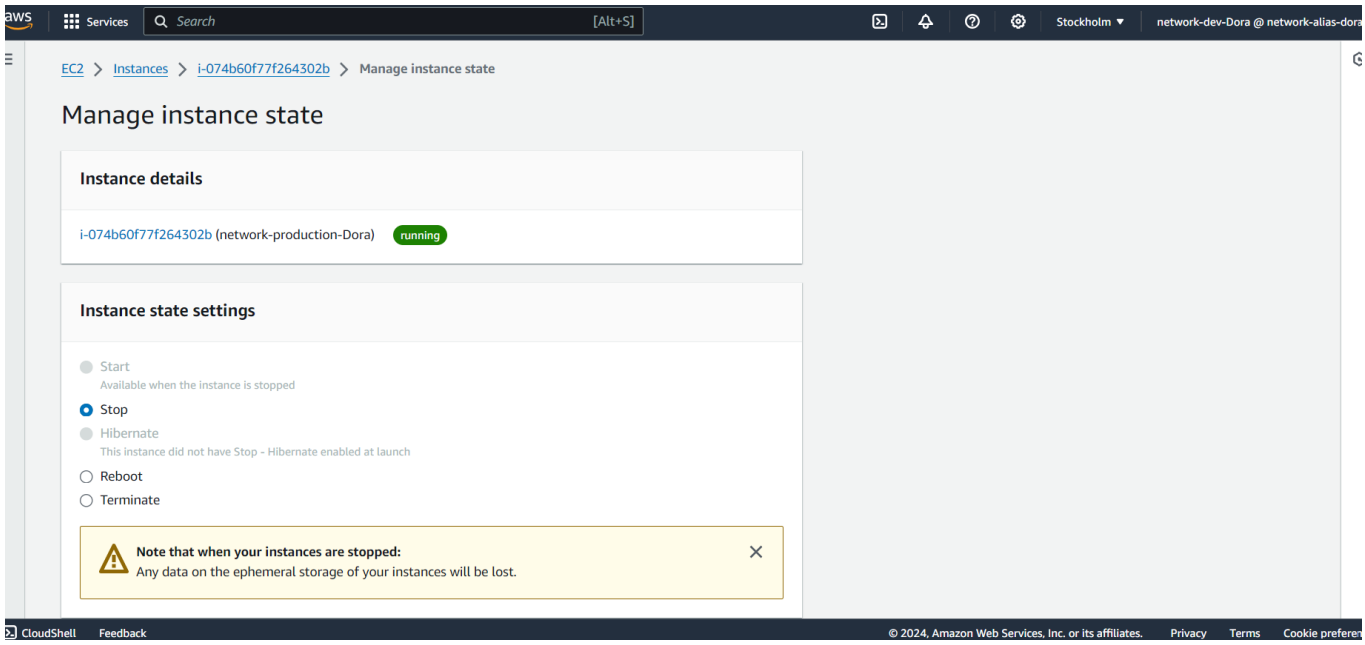
  `Console sign-in URL`

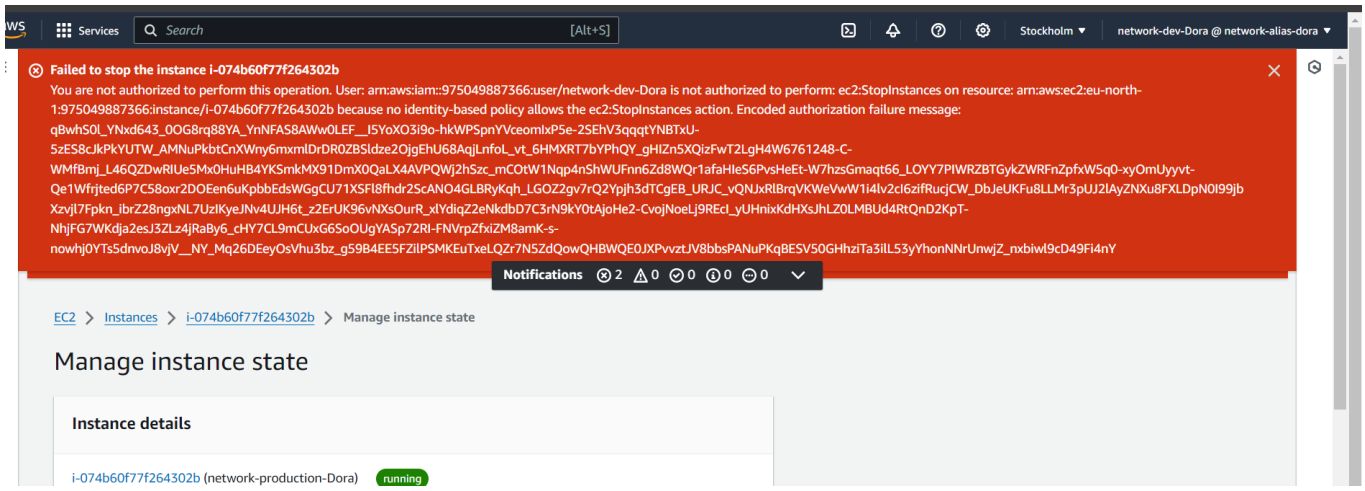
  . Do not close this tab!
- Open a new **incognito window** on your browser.
- Open the new console sign-in URL in your incognito window.
- Using the **User name** and **Console password** given in your IAM tab, let's log in!
- Woah! Welcome back to your AWS console, but this time as the dev user that you've created for yourself.

> 💡 As a new user, the AWS console will treat you as someone that is starting from 0 again. Awesome for the new team member that you'll be giving this User to!

- As a new user, you'll notice that some of your dashboard panels are showing `Access denied` already!



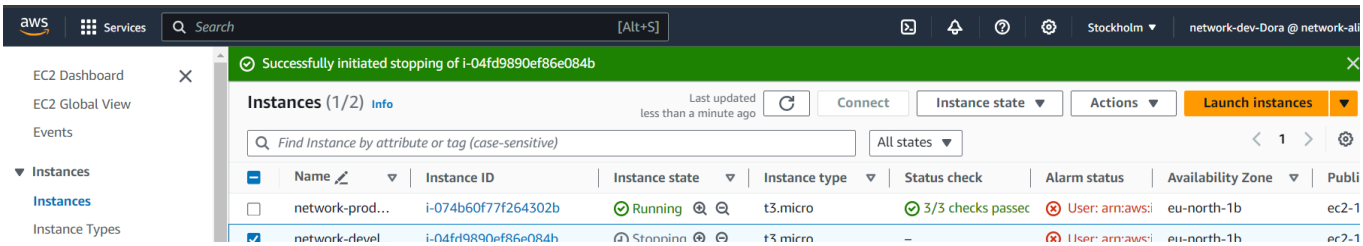




Yoikes! At the top of your page, an angry-looking banner tells us we've failed to stop this instance. The banner tells us it's because we're not authorized! We don't have permission to stop any instance with the production tag.

- Now let's try to stop the development instance.
- Head back to the **Instances** page, and select the checkbox next to `nextwork-development-yourname`.
- Under the `Actions drop-down`, select `Manage instance state`.
- Select `Stop`, then `Change state`. Select `Stop`.
- Success!

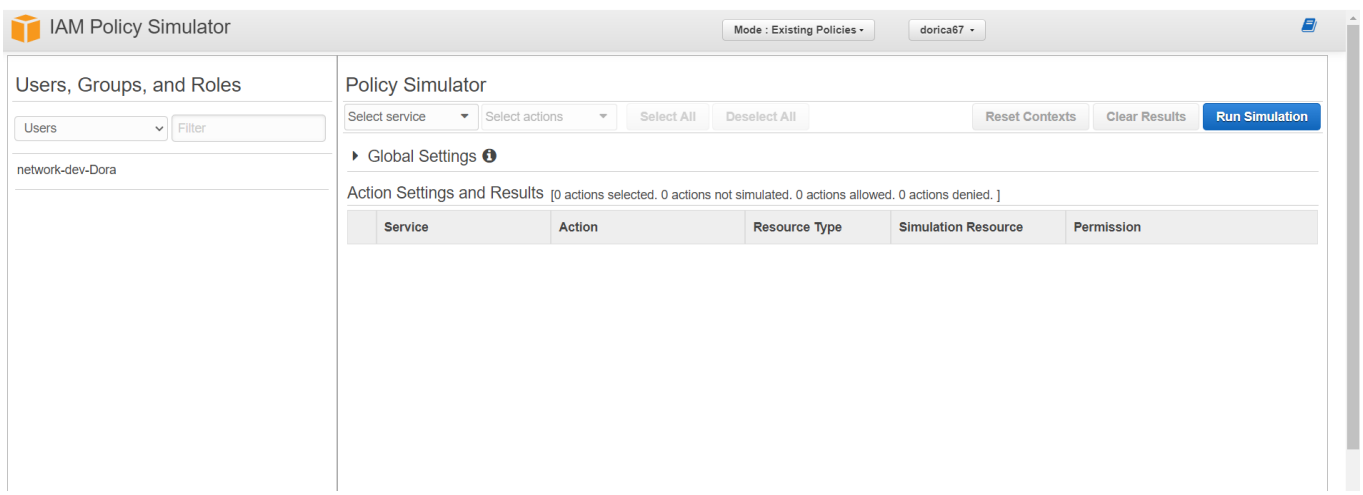
We can see it was successfull:



IAM Policy Simulator

> 💡 Have you noticed that to test out the effectiveness of our policy, we had to shut down the instance? In the real world, you might not want to actually shut down your EC2 instances just to test your custom IAM policy (this could get pretty disruptive).
>
> **The IAM Policy Simulator** lets you test and validate your policies without affecting your actual AWS resources. Let's try it out!

- Head back to your main AWS account (not the dev user!).
- In your IAM **dashboard**, look for the **Policy Simulator** link under the **Tools** panel.
  - If you can't find it, here's a quick link: [IAM Policy Simulator]



- Select your dev user group.
- Under the

  `Select service`

  drop-down, select

  `EC2`

  .
- Under the

`Select actions`

drop-down, select

`DeleteTags`

and

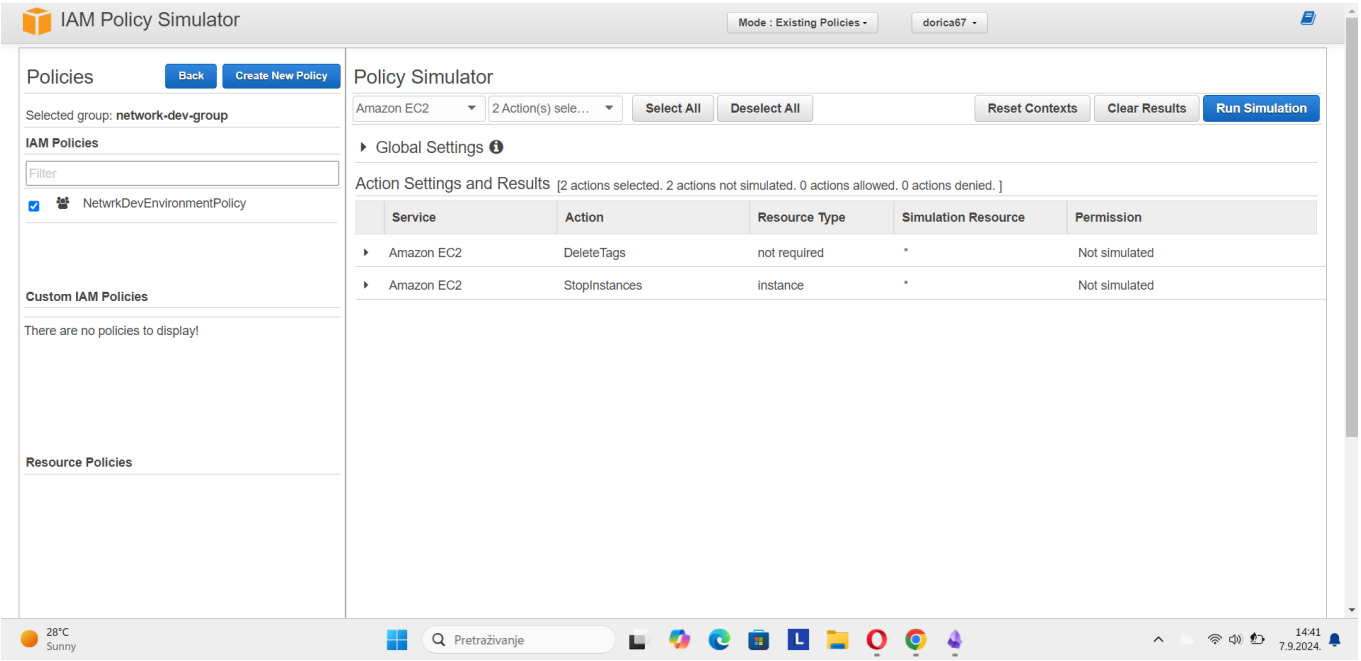
`StopInstances`

.

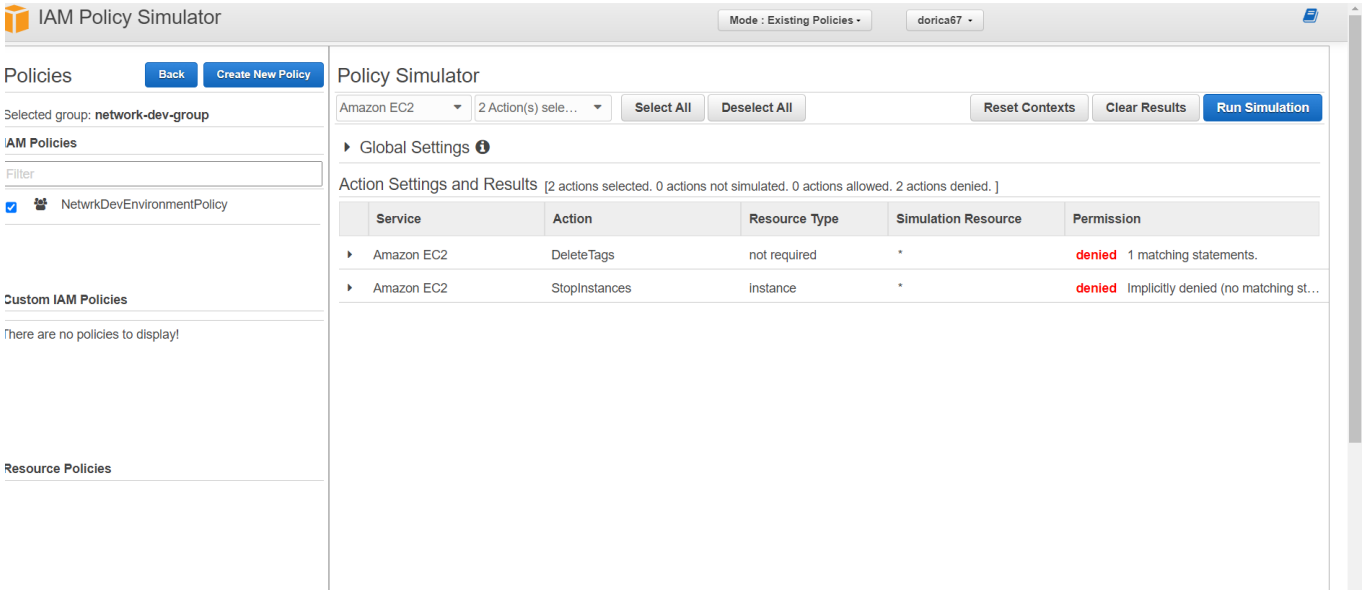
- Select

  `Run Simulation`

  when you're ready!



- You'll see that both are denied. ❌
- Expand the **toggle** for DeleteTags, and select `Show statement` .



- Woahhhhh, you even get to see exactly which statement in

  `NextWorkDevEnvironmentPolicy`

  is blocking your user from deleting tags. Pretty handy!
- Do you notice that StopInstances is denied too?

> 💡 **Strange... because your dev user could stop the dev instance in their EC2 console. Why does it say denied?**
>
> The action was denied because the simulation resource is "*", which means all resources. Note that your development user can only stop EC2 instances with the
>
> `Env - development`
>
> tag (not all EC2 instances!).

- Expand the `StopInstances` toggle, and in the Instance field, add `development` to indicate that you want to run the simulation for the instances with that tag.
- Select `Run simulation` again