AWS Cloud Infrastructure Overview

In this cloud environment, the **AWS Management Console** serves as the central control panel for managing all cloud-based resources.

At the highest level, a **Root User** is responsible for administrative oversight managing all assets, permissions and configurations across the organization's AWS account.

To simulate an enterprise structure, multiple **departments** such as Production and Deployment are represented through isolated environments, each managing its own set of cloud assets. These assets include **virtual servers** and **workstations** provisioned as **Amazon EC2 instances**.

EC2 Instances – Virtual Infrastructure

Each **EC2 instance** functions like a dedicated virtual computer or server hosted in the AWS cloud. These instances vary in CPU cores, memory (RAM), and performance capacity, defined by the **Instance Type** (for example t2.micro, t3.medium).

Within this setup, I will be deploying two key environments:

- **Deployment Server** – Used for staging, testing, and preparing updates before release.
- **Production Server** – Hosts the live environment used by end users or departments.

Each instance is built from an **Amazon Machine Image (AMI)**, which acts as the base operating system and configuration blueprint for the virtual machine.

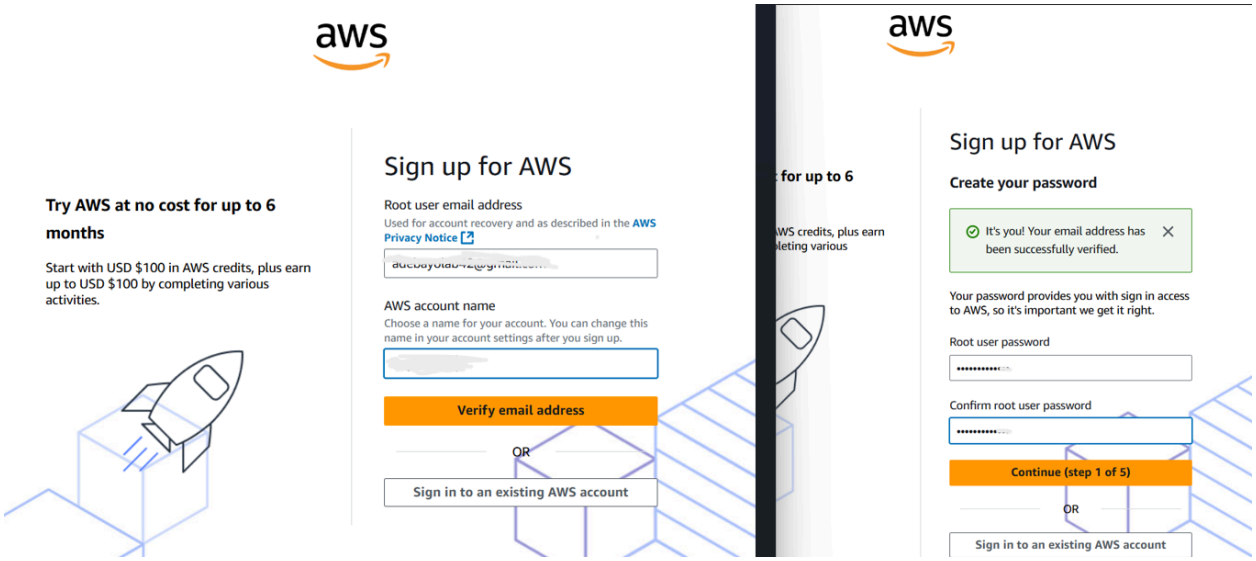
Cloud Security Implementation Steps (IAM)

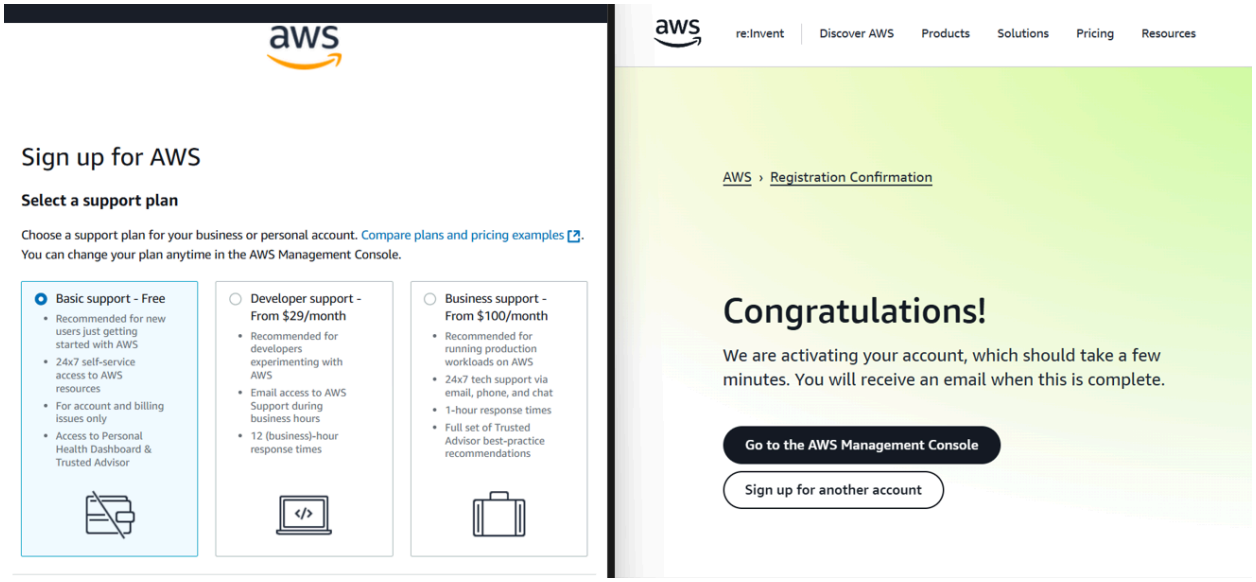
- **Set up the AWS Management Console** → Access and configure the AWS Management Console to manage cloud resources and permissions.
- **Launch Two EC2 Instances** → Deploy two virtual machines to simulate separate environments (e.g., *Production* and *Deployment* servers).
- **Create an IAM Policy** → Define a custom JSON-based policy to control permissions and enforce least-privilege access for users and groups.
- **Create an AWS Account Alias** → Customize the AWS account alias for easier identification and secure access management.
- **Create an IAM Group and User** → Establish user groups based on roles or departments and create IAM users with appropriate permissions.

- **Test IAM User Access**→ Log in as the newly created IAM user to verify that permissions and restrictions are working as intended.
- **CloudTrail & Config**→ This ensures every user action, configuration change and policy update in the AWS environment is tracked.

STEP 1 → CREATING AWS ACCOUNT

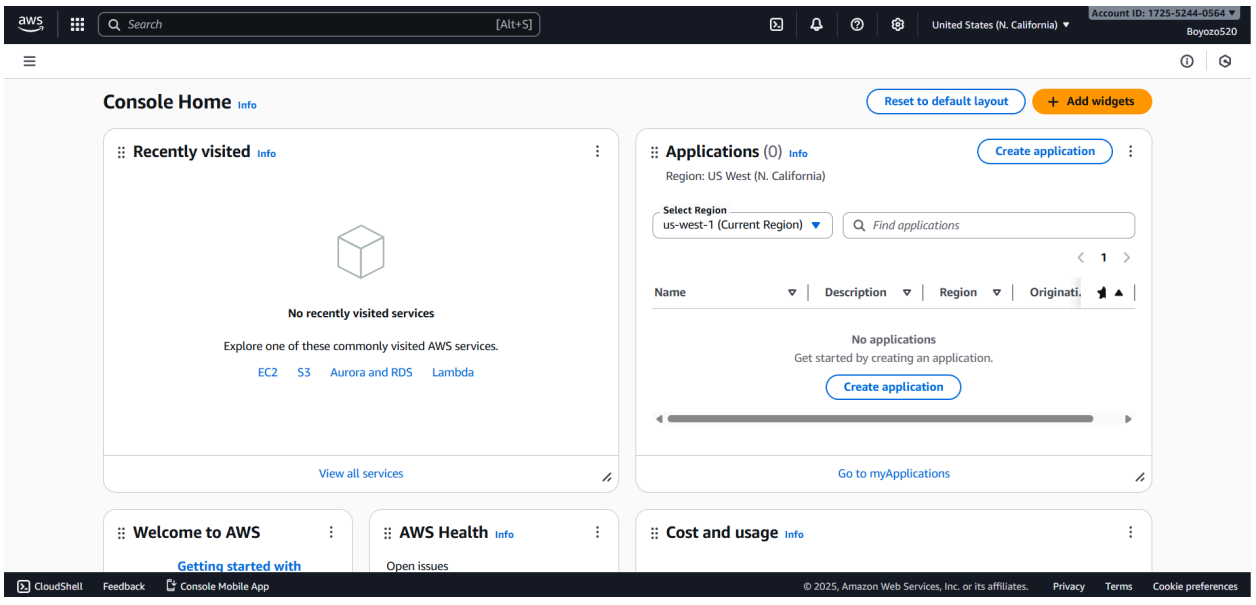
- Open your preferred web browser and navigate to **https://aws.amazon.com**
- Click on the **link** titled **"Amazon Web Services (AWS)"** to access the official AWS homepage.
- On the homepage, click **"Create an AWS Account."**
- You'll be redirected to the AWS sign-up console where you can begin the account creation process.
- Register using a **root user email address** (this will serve as the main administrative account).
- When prompted, select **"Personal use"** for the account type.
- Choose the **Basic Support Plan**, then click **"Complete Sign Up."**
- Once registration is complete, click **"Go to the AWS Management Console"** to access your AWS environment.



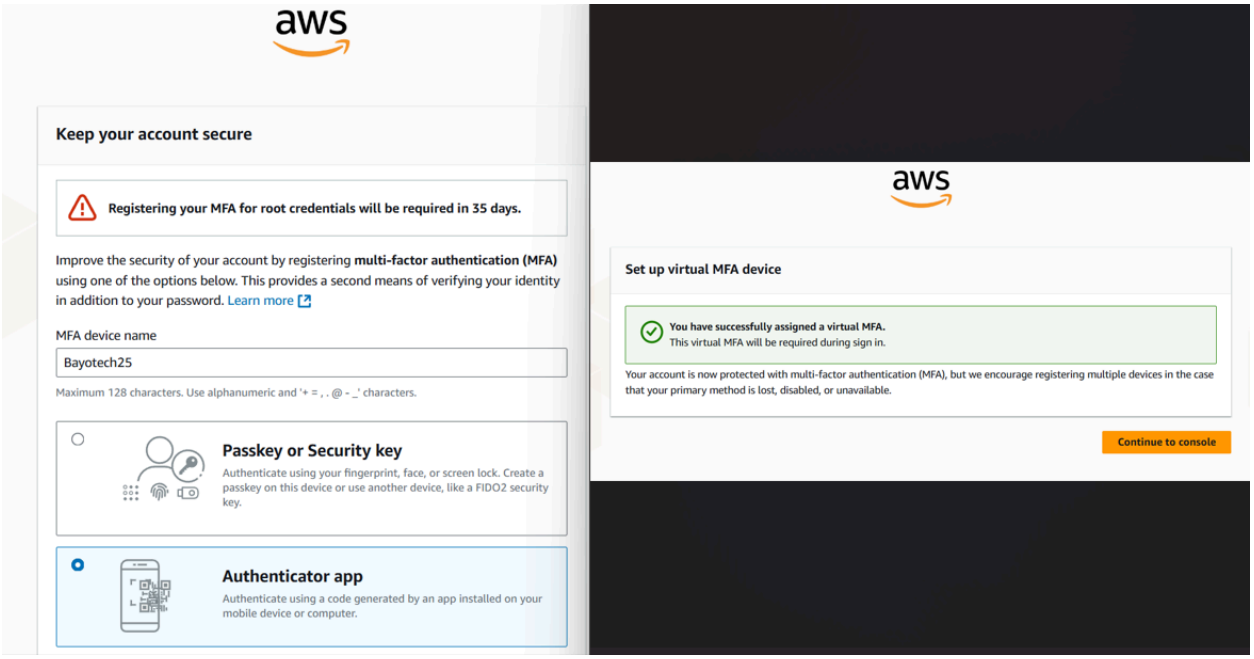


Once logged into the **AWS Management Console**, you'll see the Services Menu, which provides access to all available AWS services such as **EC2, S3, IAM,** and more.

From the top-right corner of the console, select the **region closest to your geographical location**. Choosing a nearby region helps ensure lower latency, faster performance, and compliance with data residency requirements.



As best practice ,I will be enabling MFA for the Root user, as this is one of AWS top security recommendations.



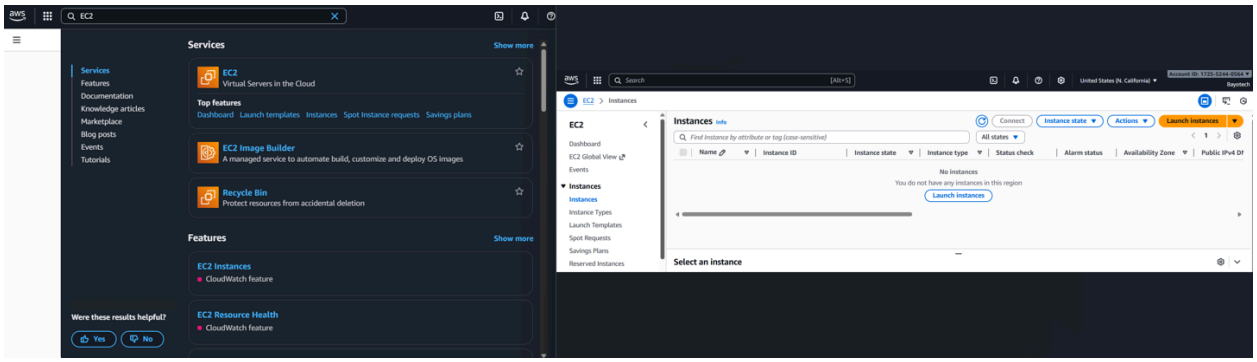
STEP 2 → Creating EC2 Instances (Deployment Server)

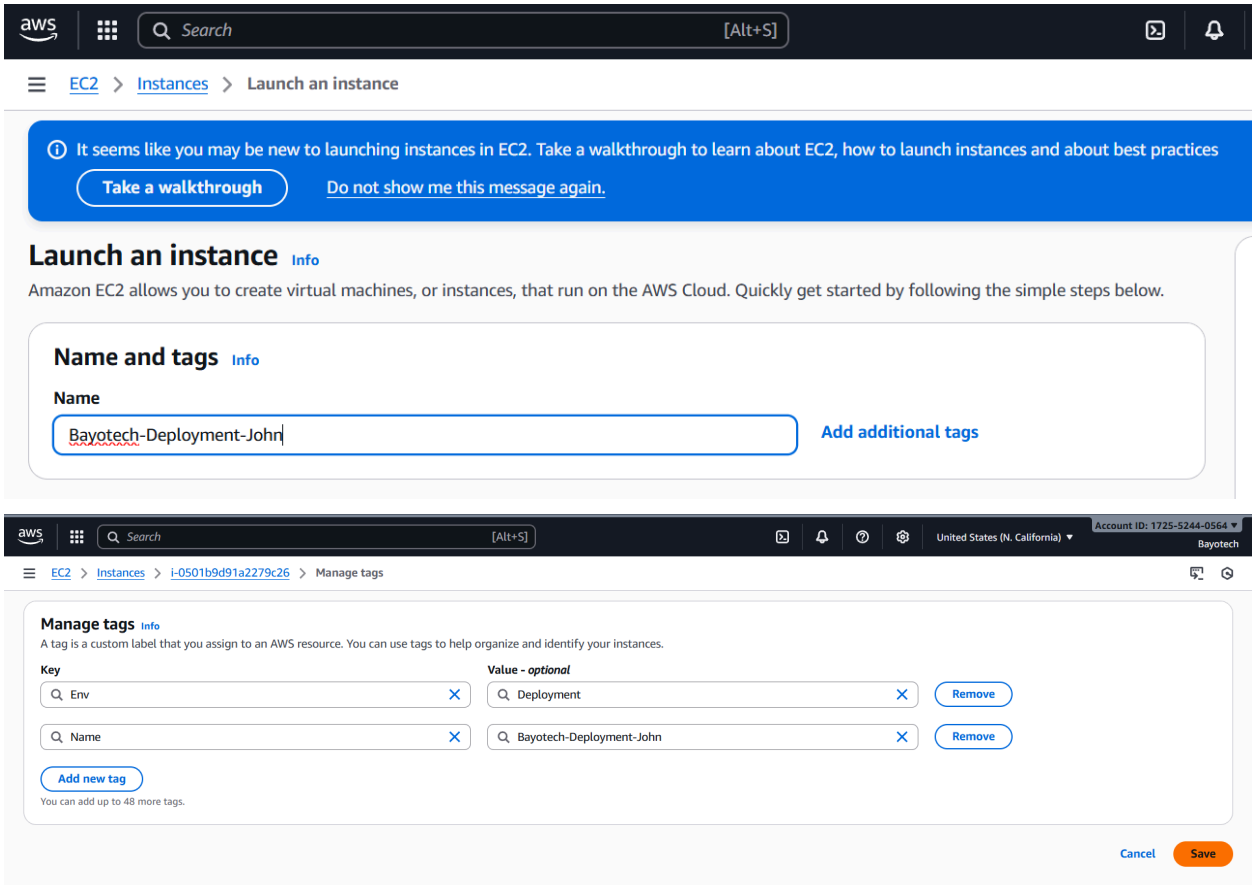
To create a Deployment EC2 instance, follow these steps:

- In the AWS Management Console, use the search bar at the top and type "EC2."
- Click on EC2 this is where you can create and manage virtual servers (instances) in the AWS cloud.
- In the EC2 dashboard, select **"Instances"** from the left-hand menu.

Note: By default, no instances will appear if you haven't created any in the current region.

- Click **"Launch Instance"** to begin creating a new virtual server.
- Under the Name and Tags section:
  - In the Name field, enter;
    1. **Bayotech-Deployment-John**
    2. Click **"Add additional tag."**
- **Key:** `Env` (short for Environment) → **Value:** `Deployment` → Leave **Resource Type** as default (blank).



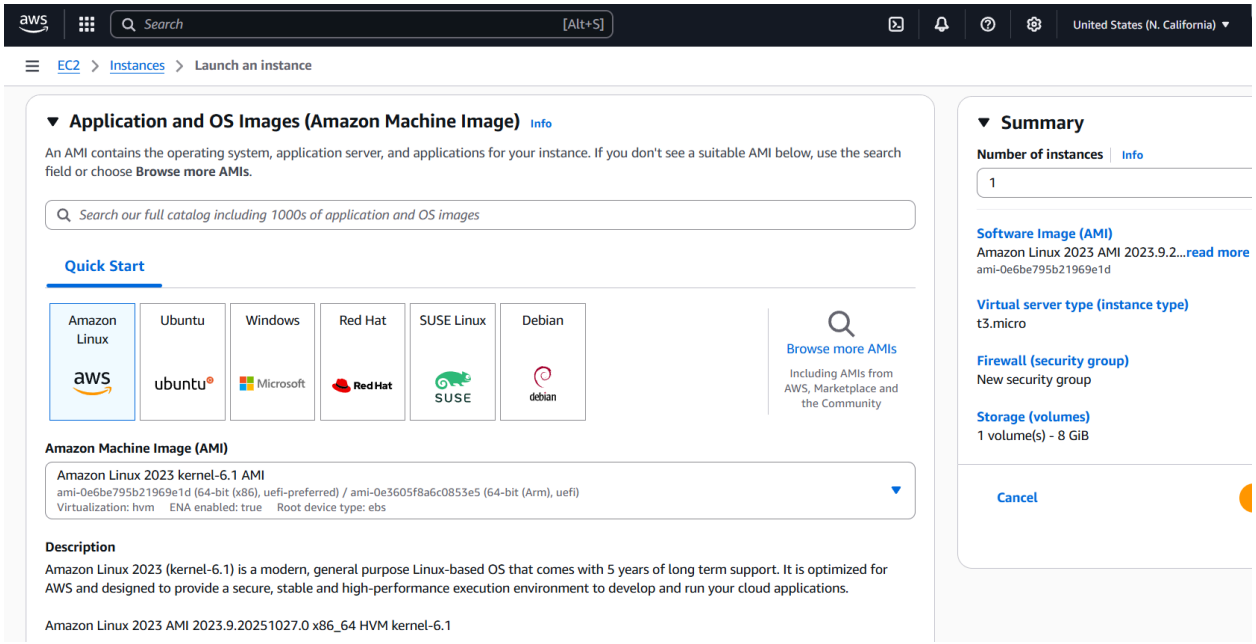


Configuring the EC2 Instance

1) Select the Application and OS Image (AMI)

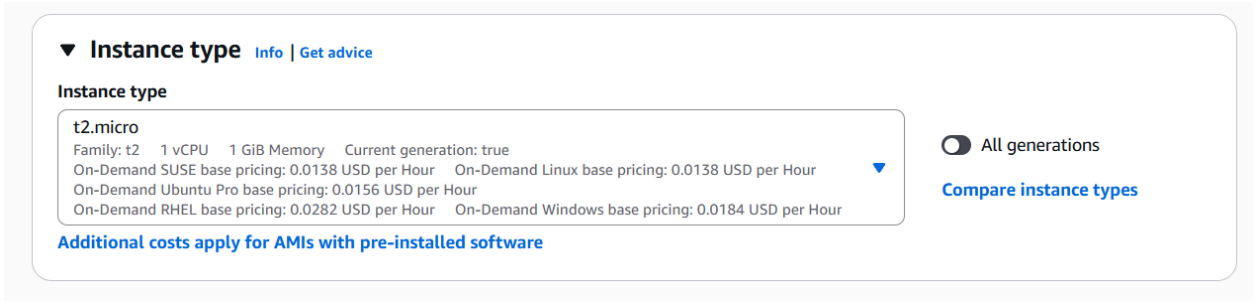
- Scroll down to the **"Application and OS Images (Amazon Machine Image)"** section.
- From the dropdown menu, select the **Operating System** you want to use.
    - For this lab, I chose **Amazon Linux** (free tier eligible).

NOTE→ You can also choose other OS options such as **Ubuntu**, **Windows Server**, or **Red Hat Enterprise Linux**, depending on your project requirements.

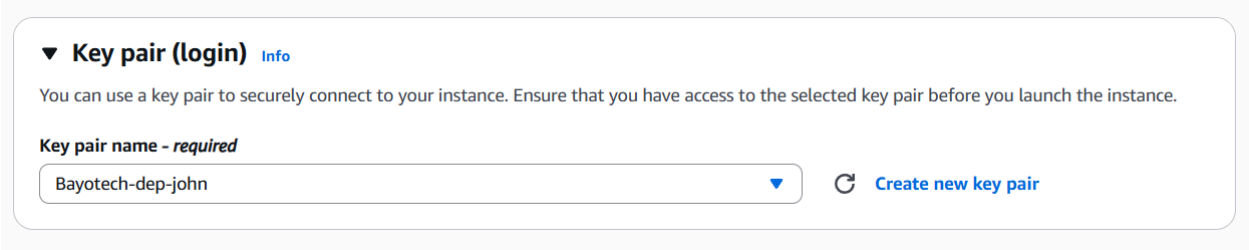


2) Choose Instance Type

- Select the **Instance Type**, which determines the virtual hardware (CPU and memory) of your EC2 instance. (For example: `t2.micro` (1 vCPU, 1 GiB RAM) — suitable for testing and free-tier environments). –> Once selected, click **Next** to proceed.

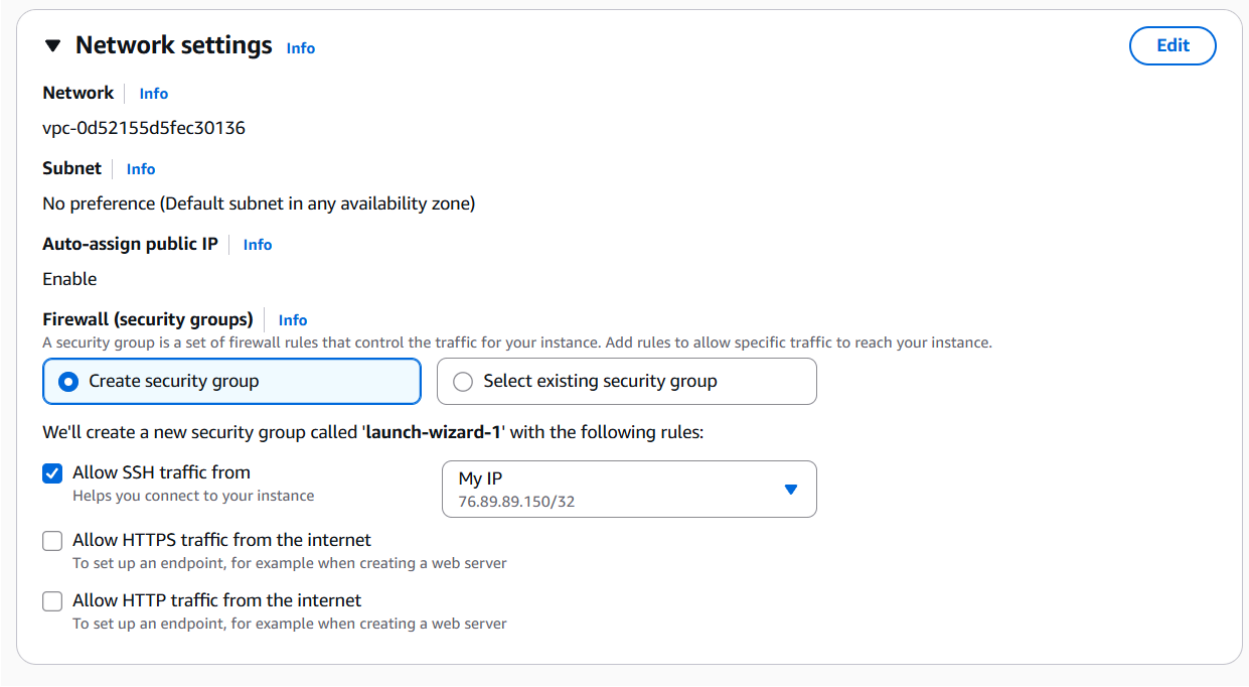


3. **Key Pair (Login)**
   - Under Key pair (login), select “Proceed without a key pair.”



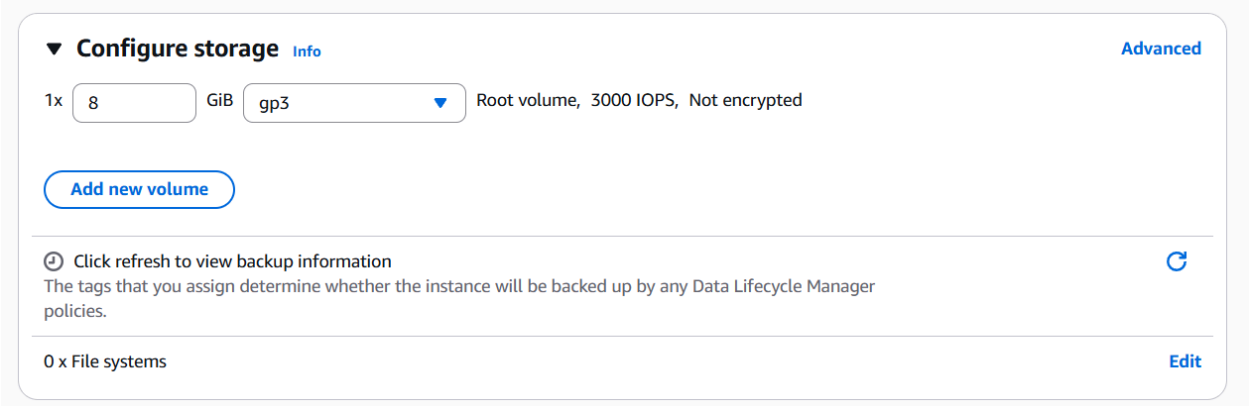
4) Under Network Settings, For this project I will check on **'Create security group'** (slect existing security group can be checked if there is an existing one) → Check **"Allow SSH traffic from"** and select Anywhere (I will be using 'myip') to enable remote access.

NOTE → (For enhanced security, restrict this to a specific IP range in real-world deployments.)



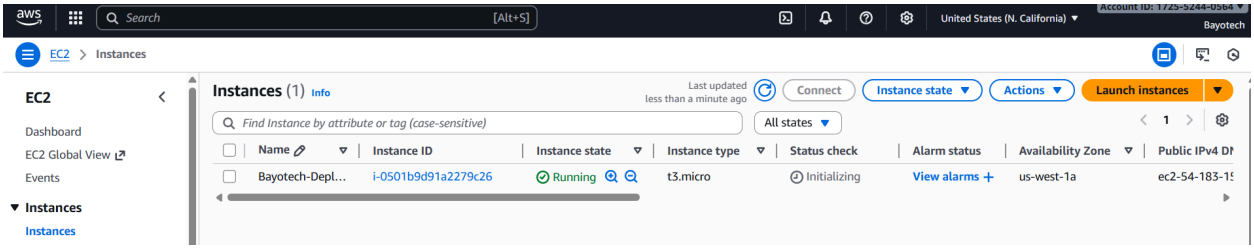
5. **Configure Storage**

- The default storage size is 8 GiB (General Purpose SSD).
- You may adjust this based on your use case, but 8 GiB is sufficient for basic testing.

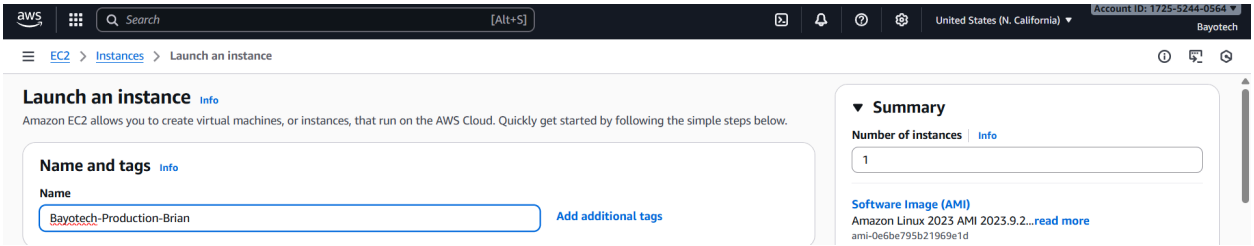


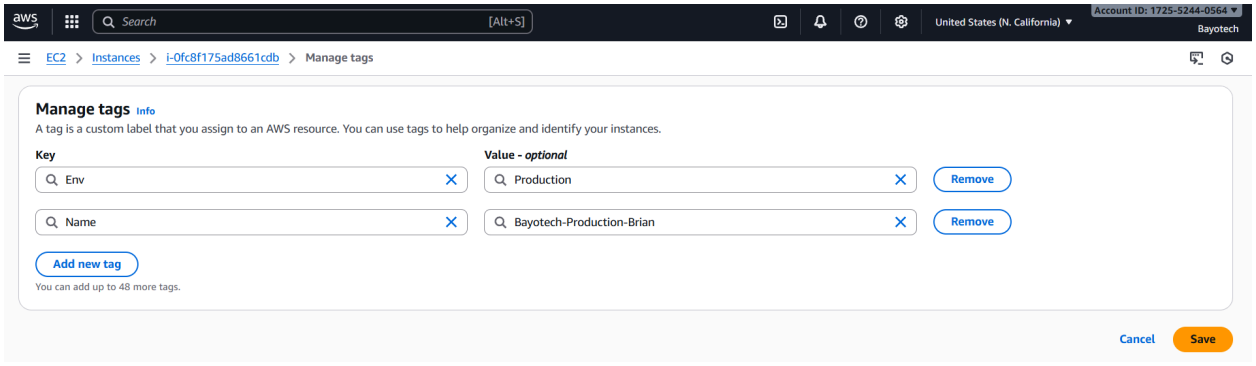
6. **Launch the Instance**

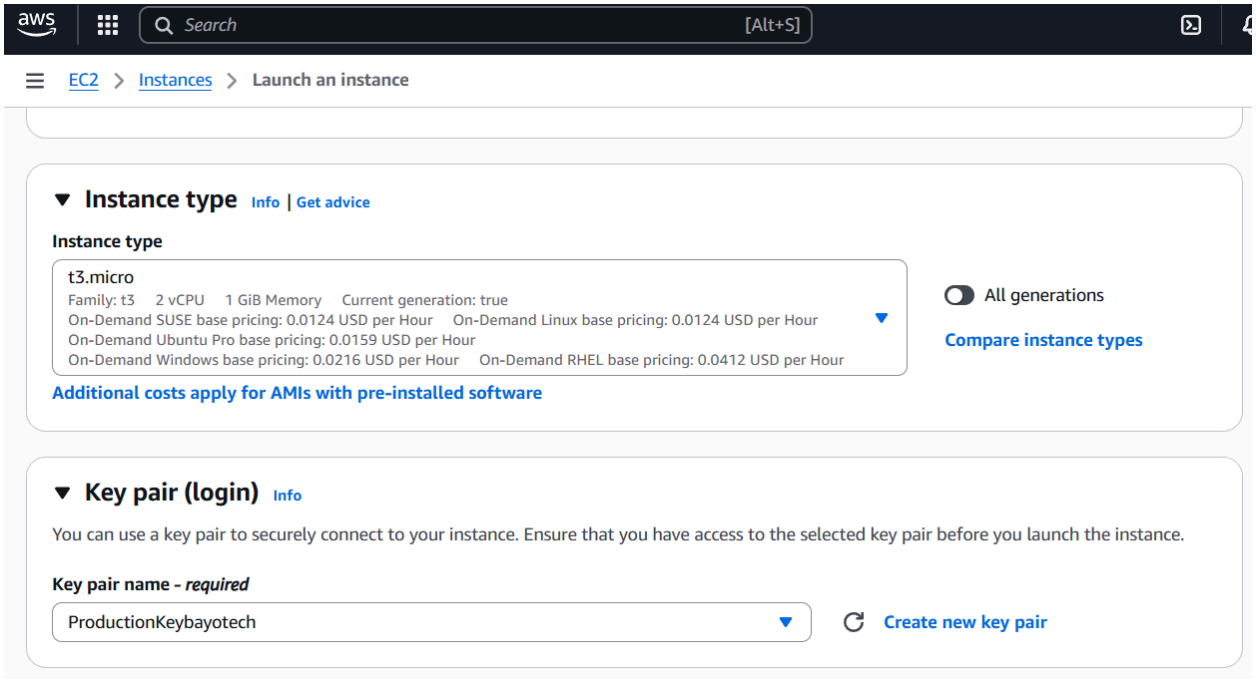
- Review your configuration and click "Launch Instance."
- Once launched, navigate back to EC2 → Instances to view your newly created server.
- If it doesn't appear immediately, click Refresh to update the instance list.

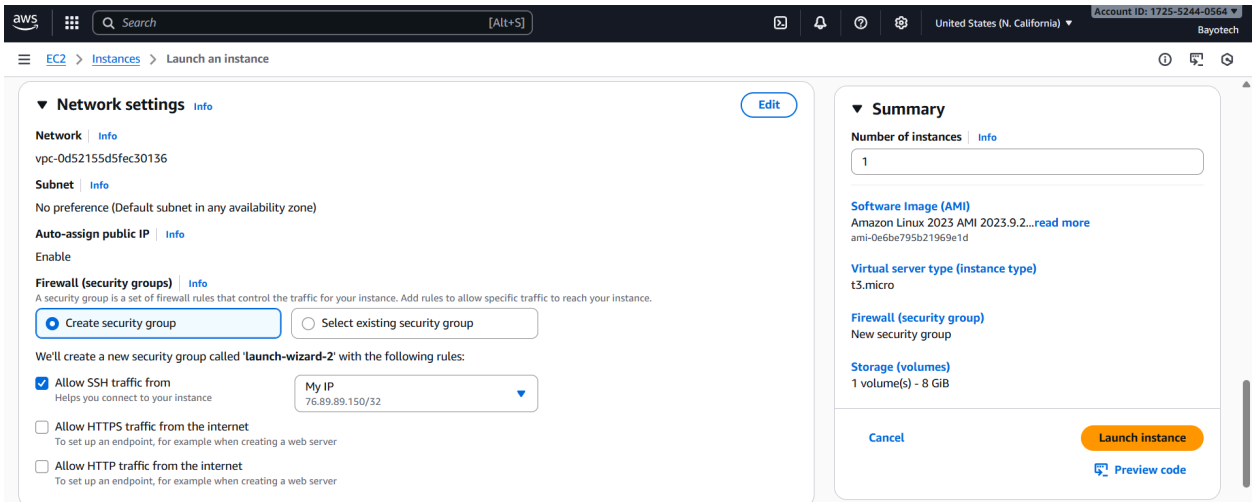


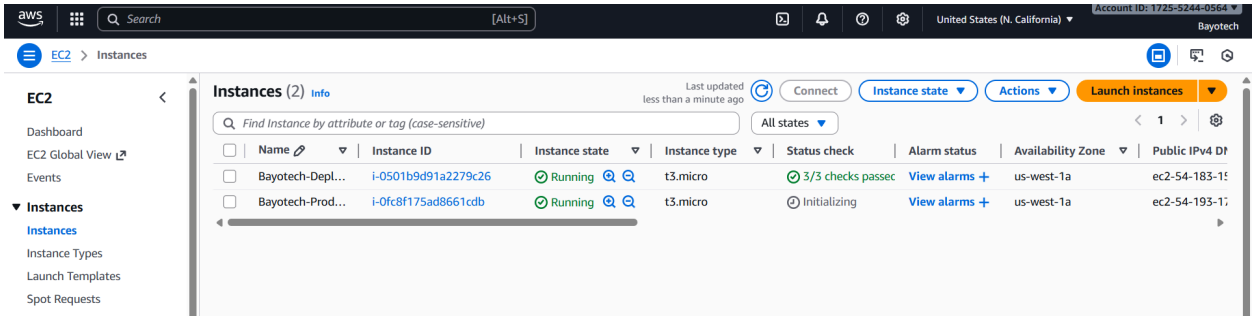
***REPEAT THE SAME PROCESS TO CREATE PRODUCTION INSTANCE

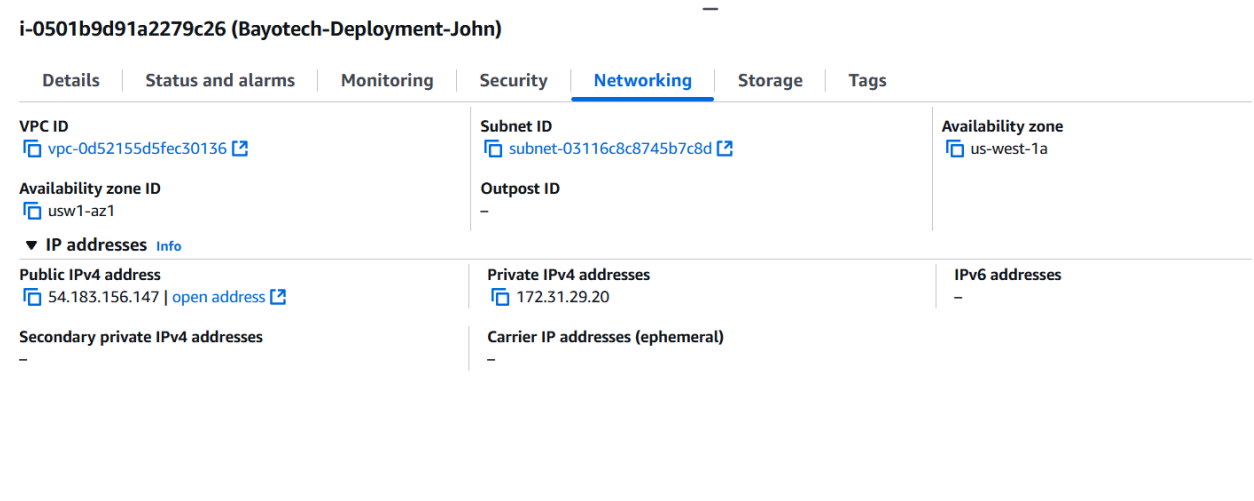


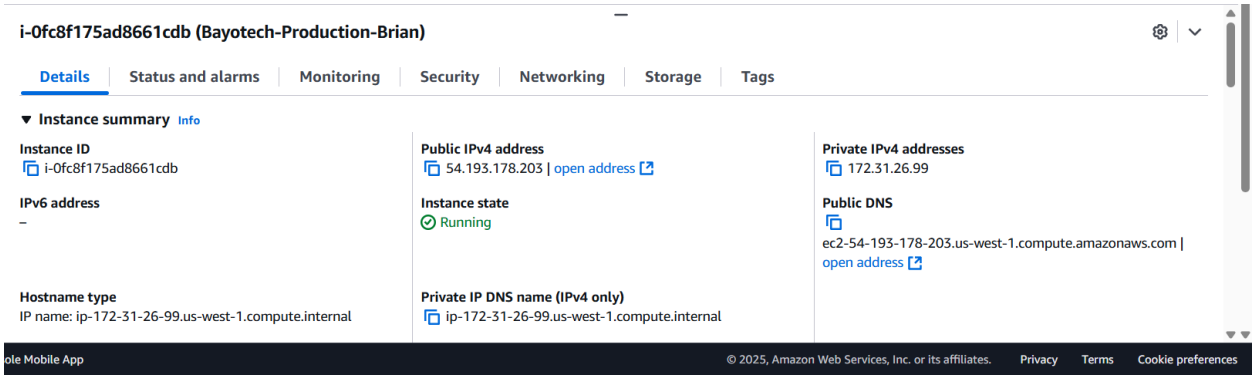

Search [Alt+S]
United States (N. California)
Account ID: 1725-5244-0564
Bayotech
EC2 > Instances > i-0fc8f175ad8661cdb > Manage tags
Manage tags Info
A tag is a custom label that you assign to an AWS resource. You can use tags to help organize and identify your instances.
Key
Value - optional
Env
Production
Remove
Name
Bayotech-Production-Brian
Remove
Add new tag
You can add up to 48 more tags.
Cancel
Save


Search [Alt+S]
EC2 > Instances > Launch an instance
Instance type Info | Get advice
Instance type
t3.micro
Family: t3 2 vCPU 1 GiB Memory Current generation: true
On-Demand SUSE base pricing: 0.0124 USD per Hour On-Demand Linux base pricing: 0.0124 USD per Hour
On-Demand Ubuntu Pro base pricing: 0.0159 USD per Hour
On-Demand Windows base pricing: 0.0216 USD per Hour On-Demand RHEL base pricing: 0.0412 USD per Hour
All generations
Compare instance types
Additional costs apply for AMIs with pre-installed software
Key pair (login) Info
You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.
Key pair name - required
ProductionKeybayotech
Create new key pair


Search [Alt+S]
United States (N. California)
Account ID: 1725-5244-0564
Bayotech
EC2 > Instances > Launch an instance
Network settings Info
Edit
Network Info
vpc-0d52155d5fec30136
Subnet Info
No preference (Default subnet in any availability zone)
Auto-assign public IP Info
Enable
Firewall (security groups) Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.
Create security group
Select existing security group
We'll create a new security group called 'launch-wizard-2' with the following rules:
Allow SSH traffic from
Helps you connect to your instance
My IP
76.89.89.150/32
Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server
Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server
Summary
Number of instances Info
1
Software Image (AMI)
Amazon Linux 2023 AMI 2023.9.2...read more
ami-0e6be795b21969e1d
Virtual server type (instance type)
t3.micro
Firewall (security group)
New security group
Storage (volumes)
1 volume(s) - 8 GiB
Cancel
Launch instance
Preview code





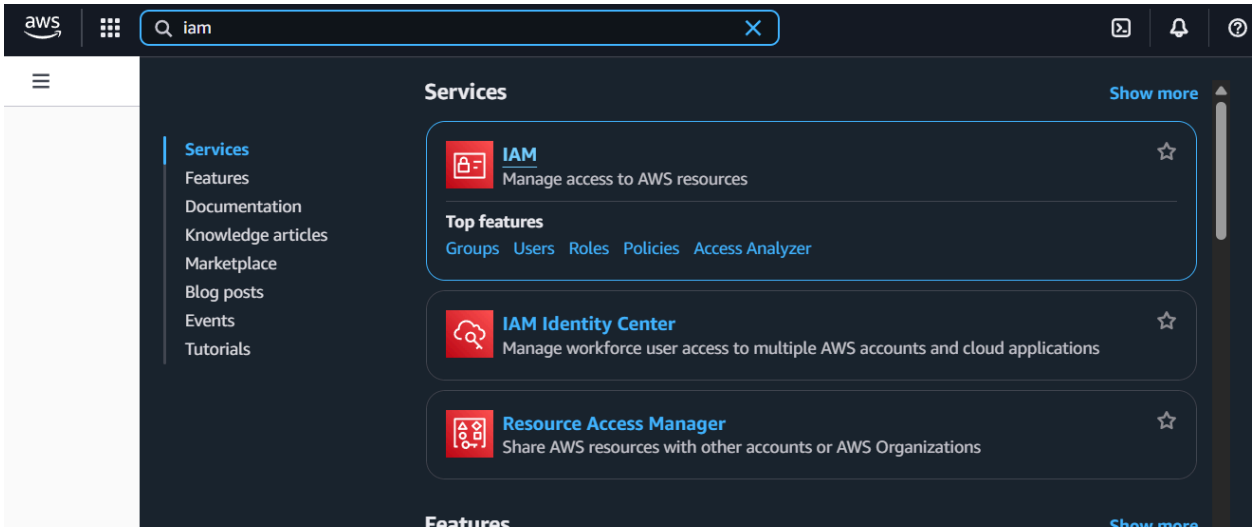


STEP 3→ Create an IAM Policy

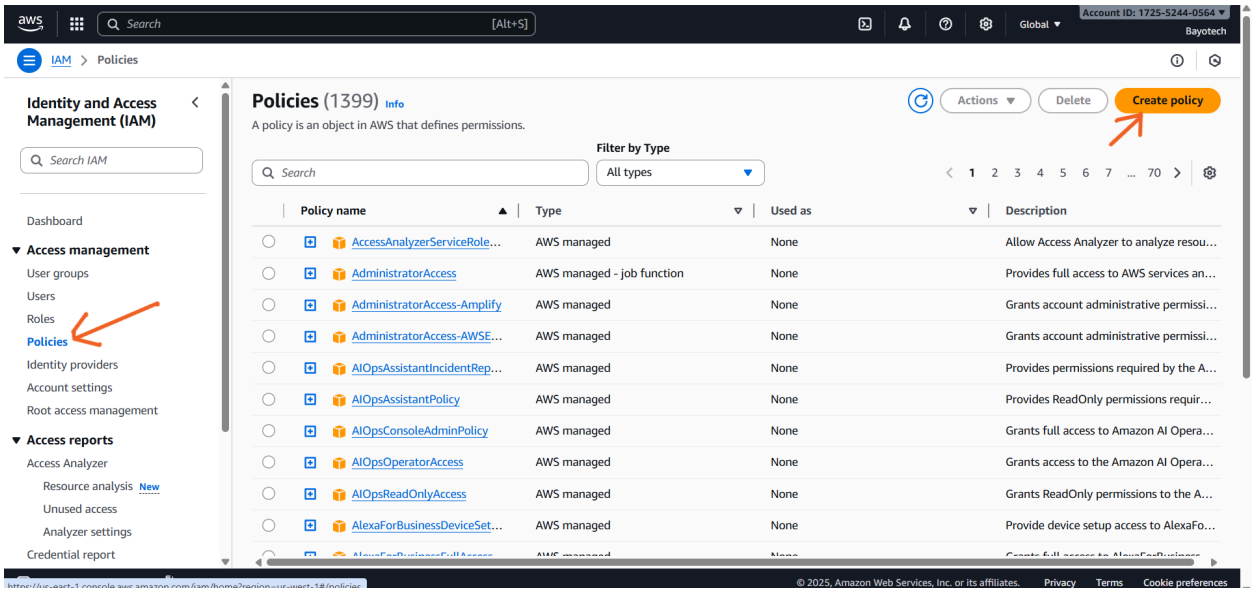
IAM policies define the rules that determine who has access to what within the AWS environment. These policies specify which actions users, groups, or roles can perform on specific

AWS resources.There are available policies but i will be creating a custom IAM policy, follow these steps:

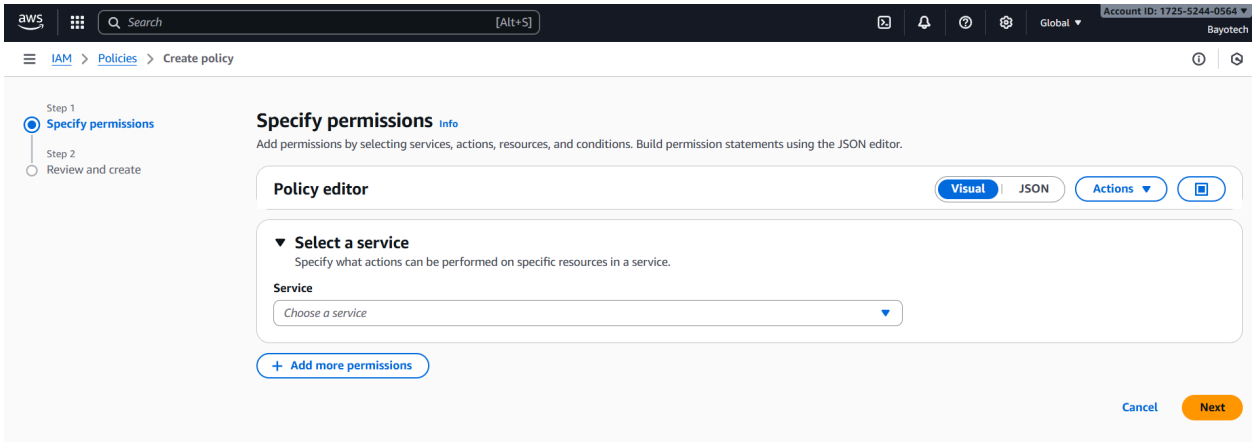
- Navigate back to the AWS Management Console.
- In the search bar, type “IAM” and select IAM from the results.



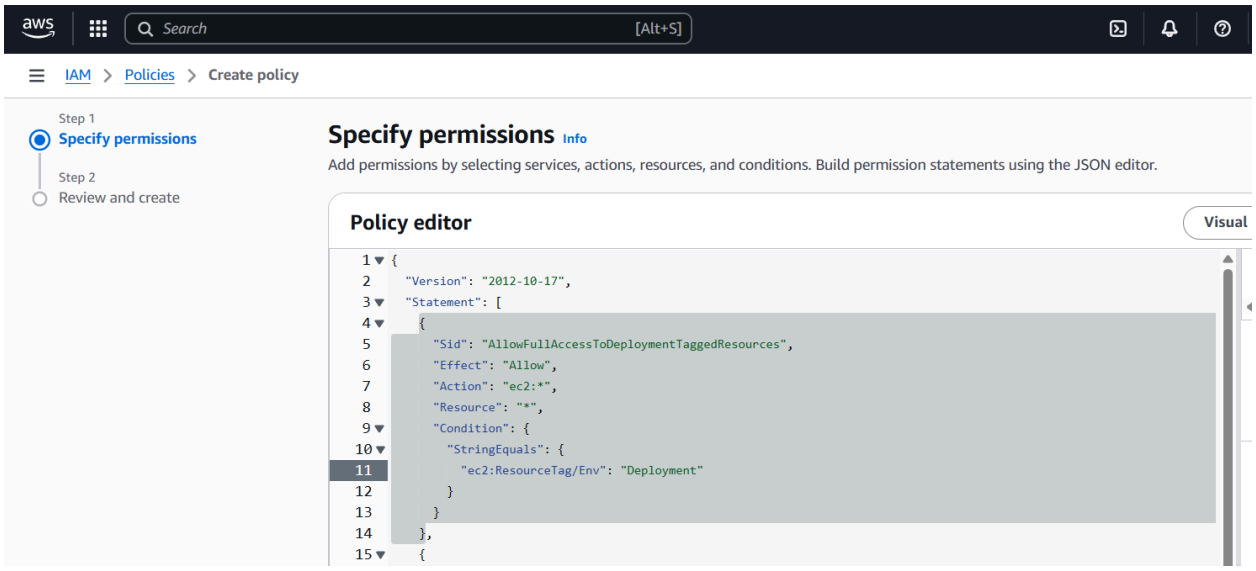
- In the IAM dashboard, click on “Policies” in the left-hand menu, You’ll see a list of existing AWS-managed and customer-managed policies.
- To create a new custom policy, click “Create policy.”

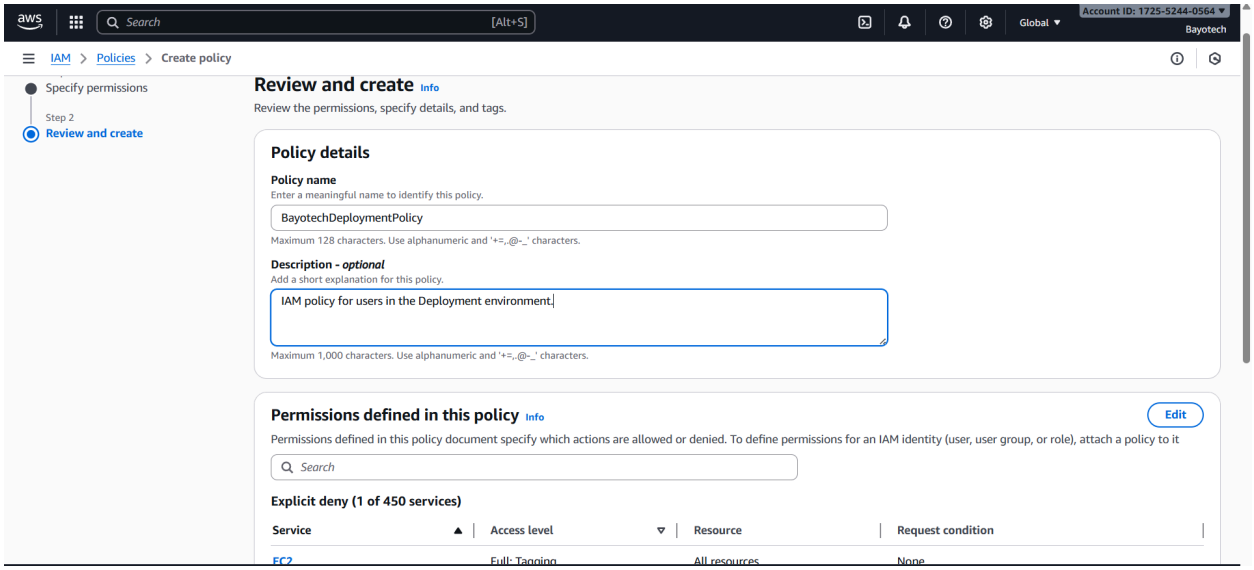


- The Policy Editor will open this interface and allow me to define custom permissions.

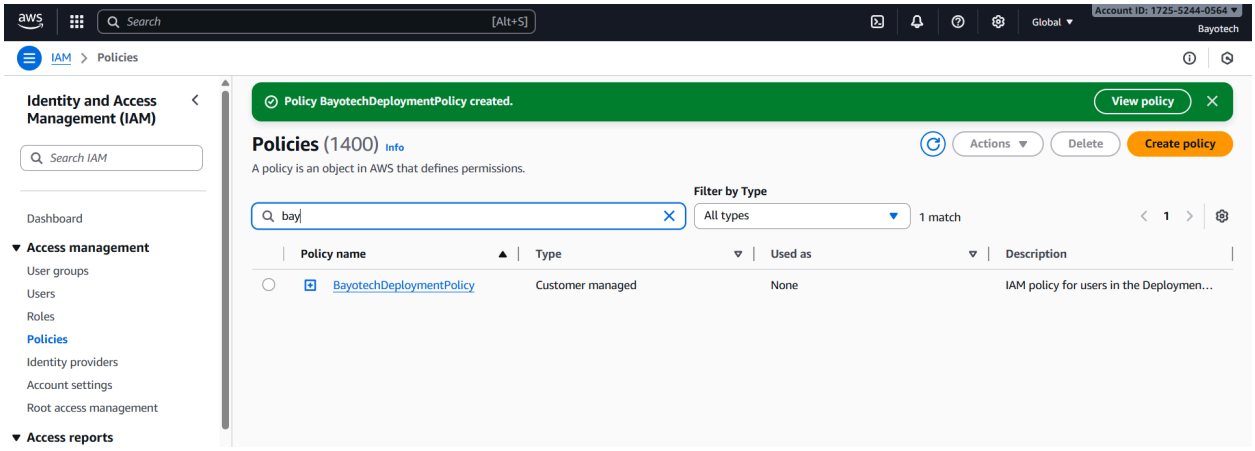


- Switch to the JSON tab to enter my policy code. Input the custom JSON policy script (for example, the one granting EC2 access to specific tagged resources).
- Review the policy structure it consists of:
    i. Root permissions: Define which AWS services and actions are allowed or denied.
    ii. Group permissions: Apply policies to multiple users within a group.
    iii. Identity permissions: Attach directly to a specific IAM user or role.

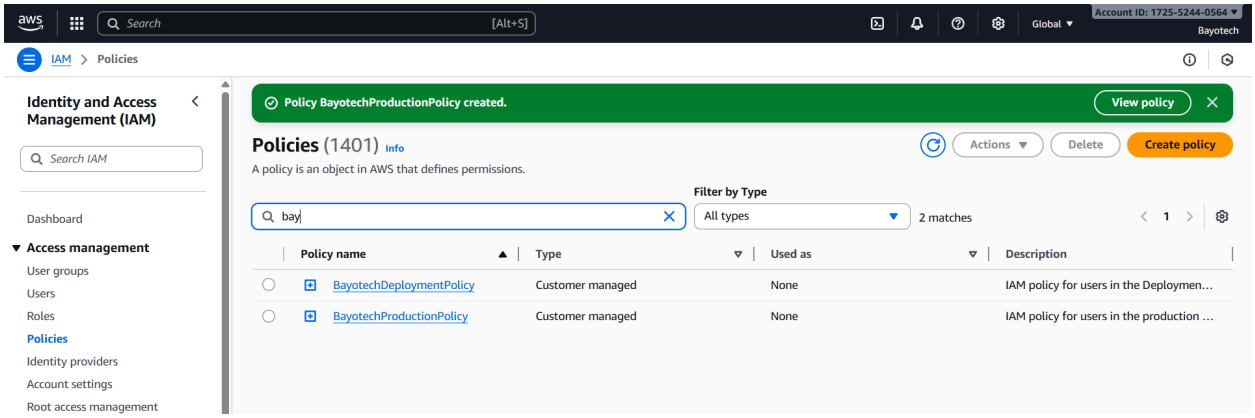




Once your policy is complete → click “Next Tags” → “Next: Review”, give the policy a clear name (e.g., EC2DeploymentAccessPolicy) → then click “Create policy”.

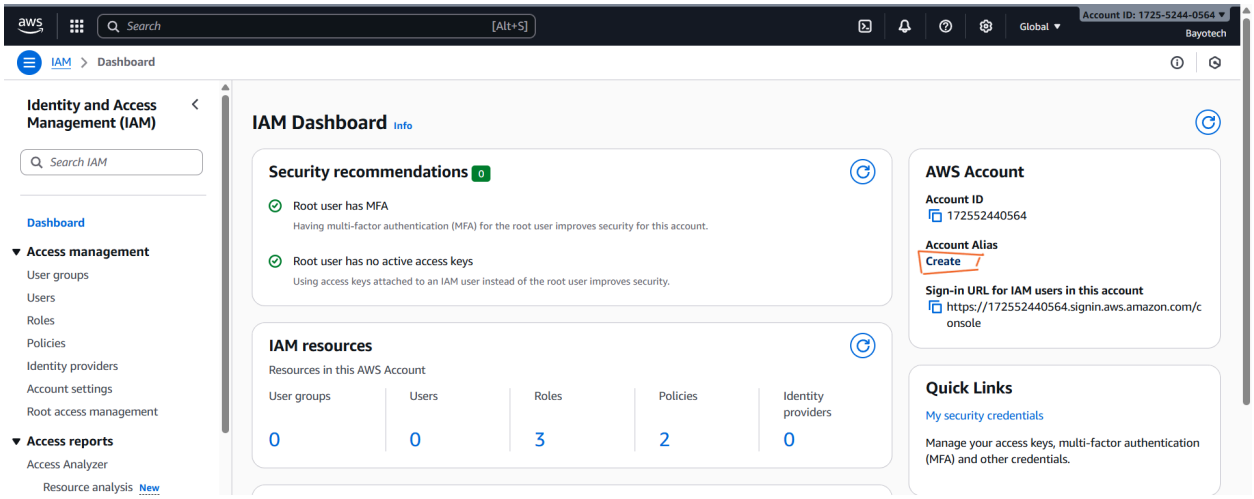


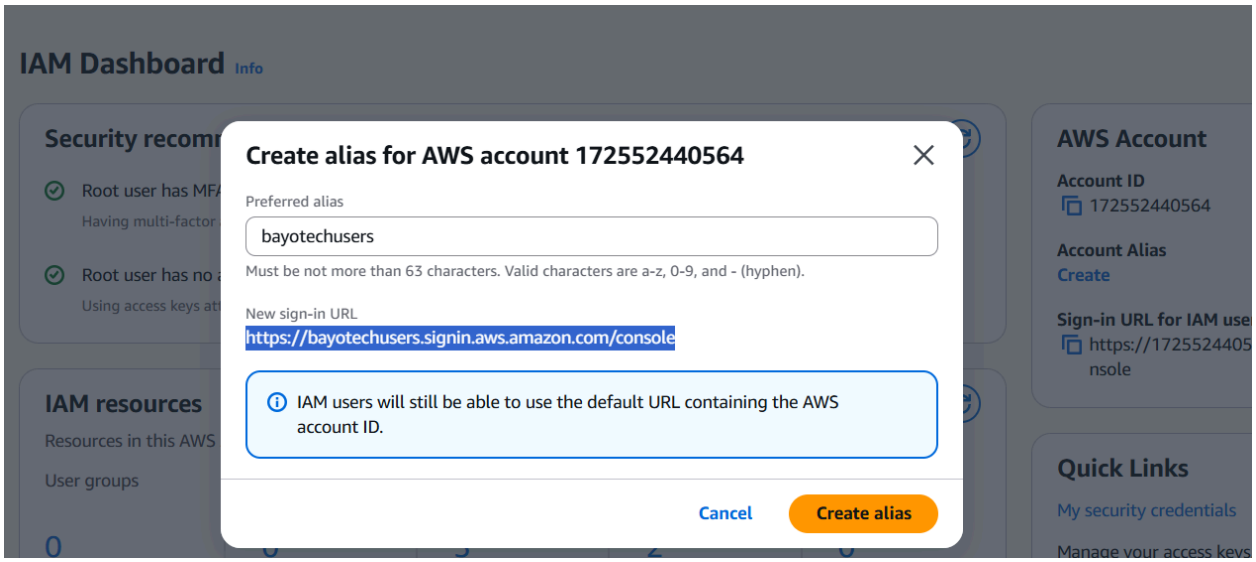
REPEAT THE-SAME PROCESS TO CREATE A PRODUCTION POLICY.

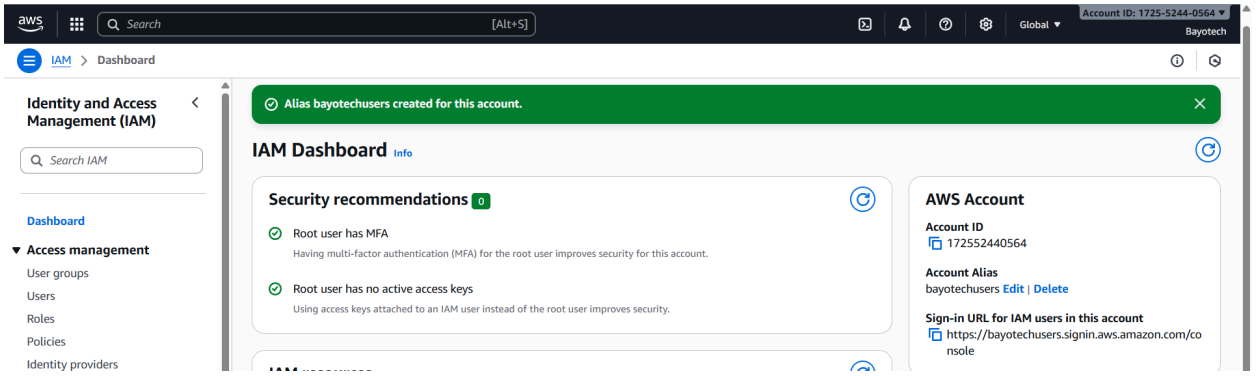


STEP 4→ Create AWS ALIAS

On IAM Dashboard → Click on **'create'** under **'Account Alias'** . Next, to customize your AWS sign-in URL. Enter your 'preferred alias', for example:Bayotechusers → Click 'create Alias'







This alias creates a more user-friendly login URL**(e.g.,https://Bayotechusers.signin.aws.amazon.com/console)** making it easier and more secure for team members to access the AWS environment.

STEP 5→ CREATING IAM GROUP AND USERS

CREATING IAM GROUP

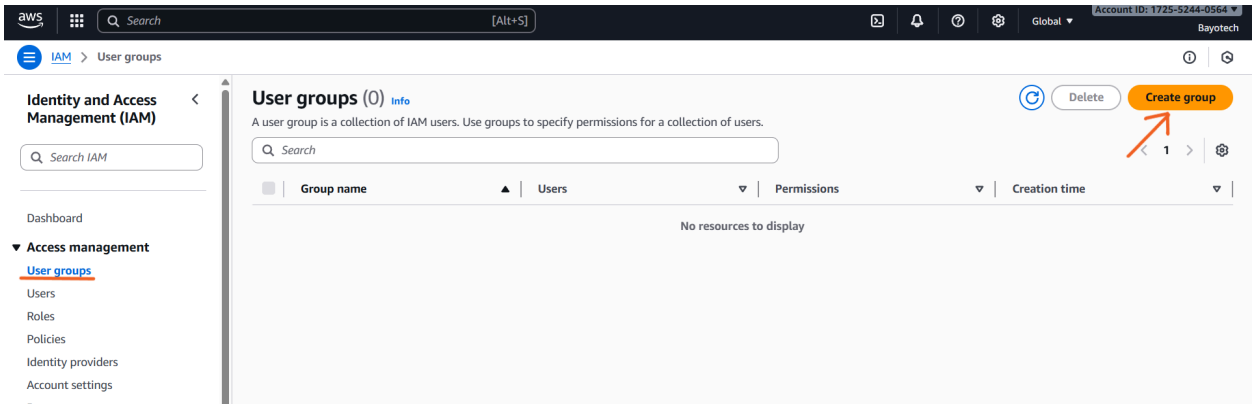
An IAM Group is a collection of multiple IAM users. The main advantage of using groups is simplified permission management ,instead of assigning policies to each user individually, you can apply a policy once to the group, and all members automatically inherit those permissions.

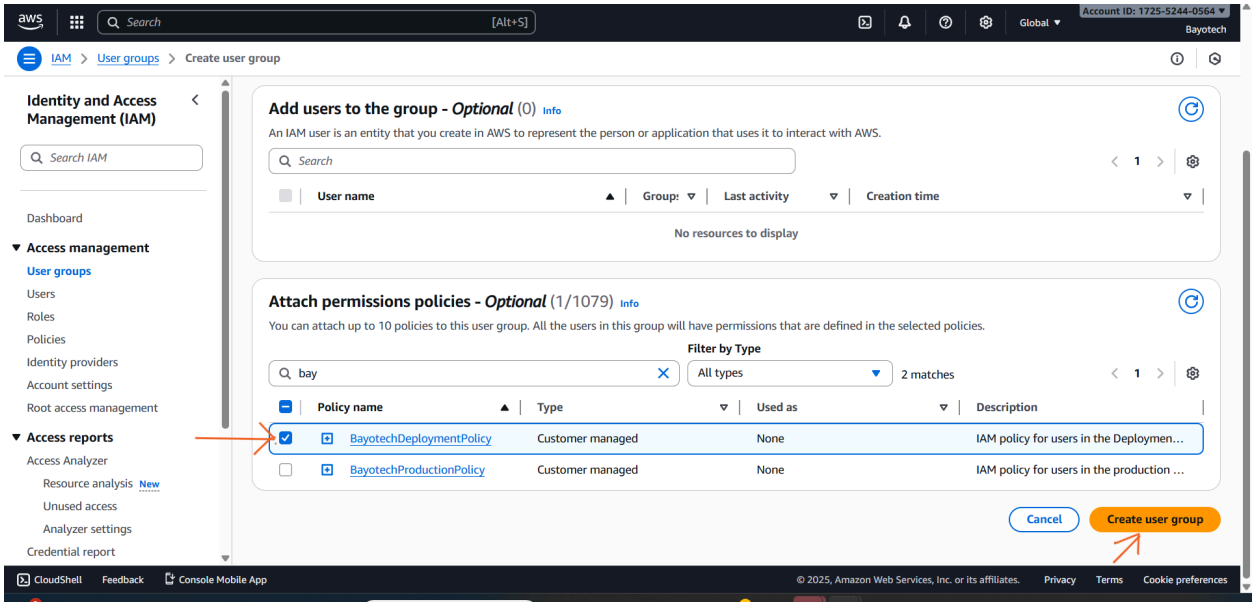
For example, in an organization with 20 employees, managing access for each user separately would be inefficient. By creating a single group (e.g., DeploymentTeam), you can attach one or more policies to that group, and every member of the group will receive the same access privileges.

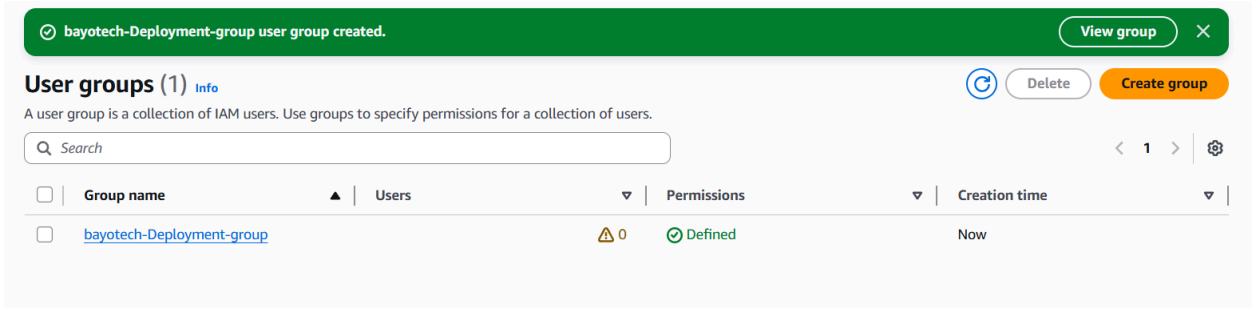
Each IAM User is a member of one or more groups, allowing for structured, scalable, and secure access management within the AWS environment.

Steps To create Groups;

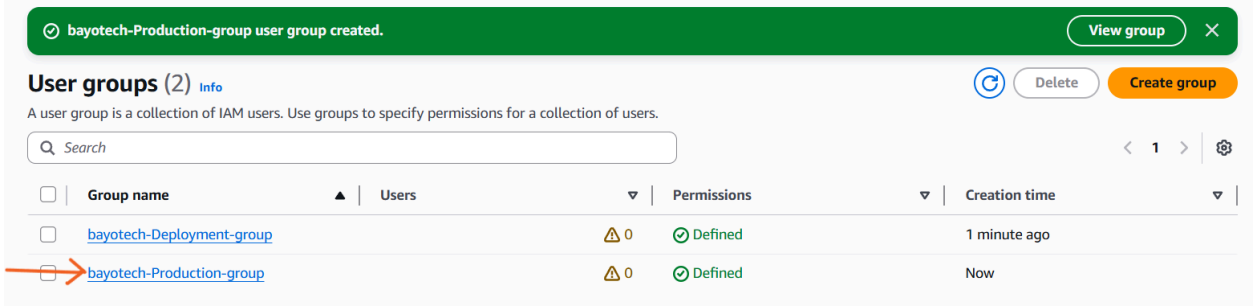
- Navigate to the IAM service by typing “IAM” into the search bar at the top of the AWS Management Console.
- In the IAM Dashboard, select **“User groups”** from the left-hand menu → Click “Create group.”
- In the Group name field, enter:**Bayotech-Deployment-Group**
- Under **Attach permissions policies (optional)**, search for the custom policy you previously created for example:**BayotechDeploymentPolicy**
- Select the checkbox next to the policy name to attach it to the group –> Finally, click **“Create user group.”**







REPEAT THE SAME STEPS TO CREATE A PRODUCTION GROUP .

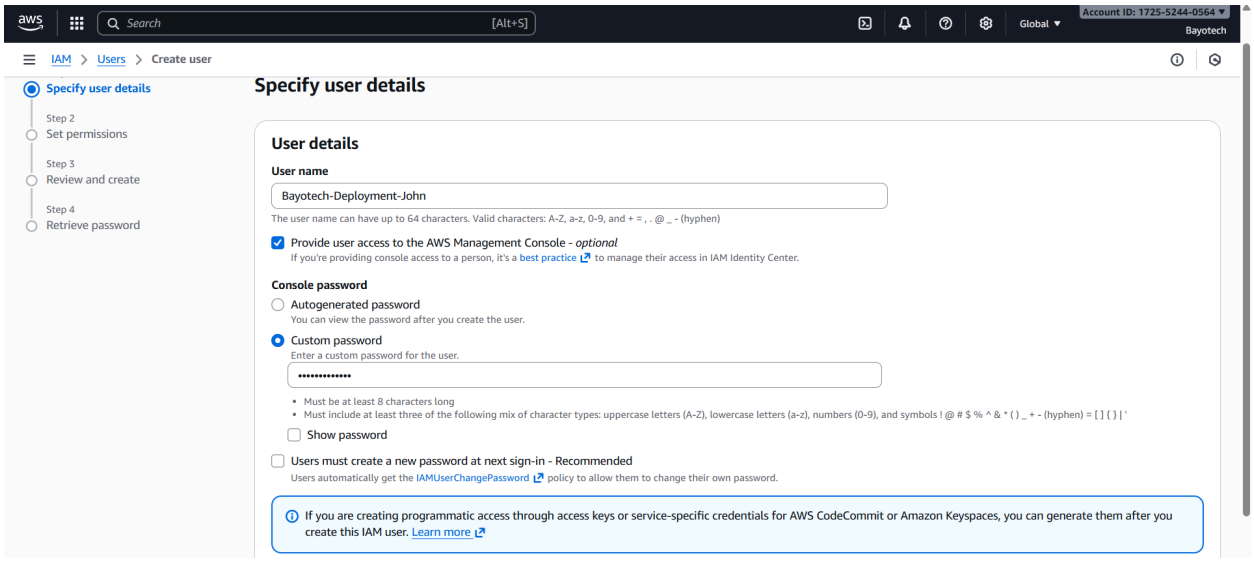


Creating an IAM User and Assigning Group Access

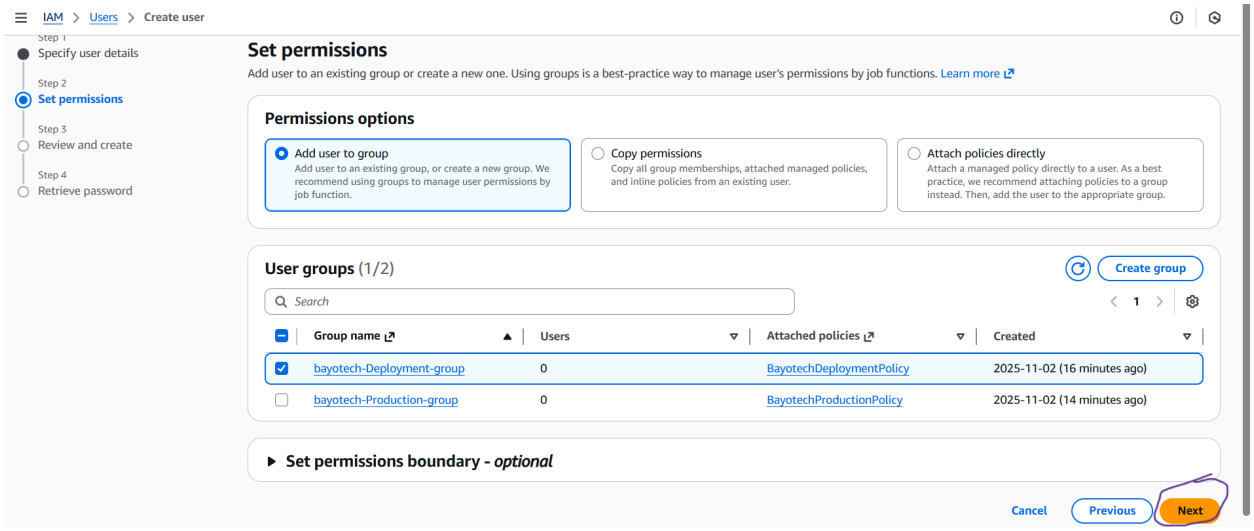
- In the **IAM Dashboard**, click on **"Users."** → Select **"create user"**
- Under Specify user details, enter a username, for example:**Bayotech-Deployment-John**
- Check the box for "Provide user access to the AWS Management Console".

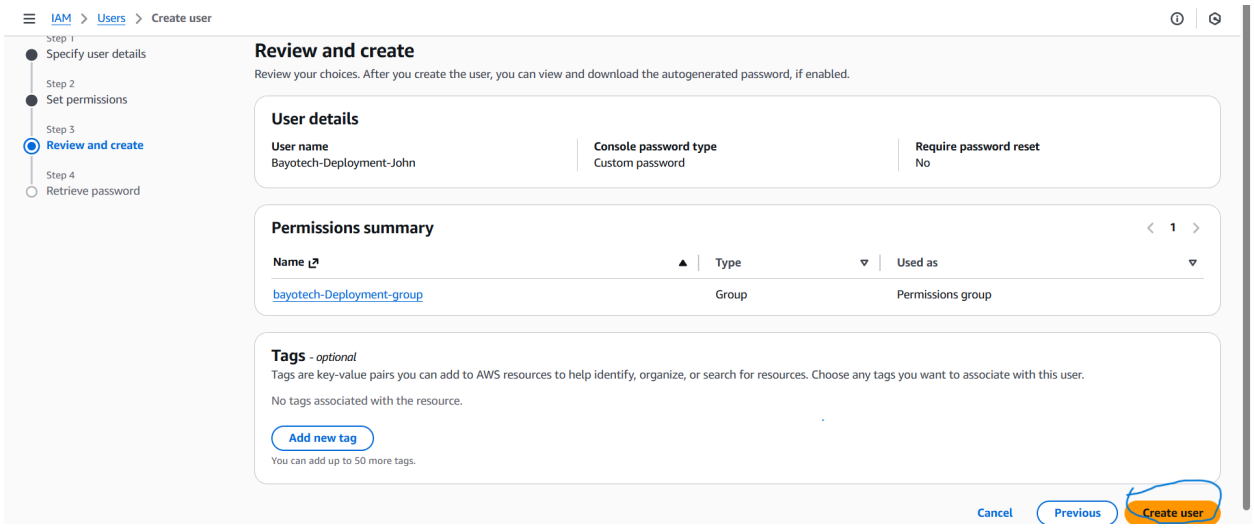
- (This allows the user to access the AWS Console via a web interface instead of only through the CLI or terminal.)

- Create a custom password, and (optional) uncheck the box labeled “User must create a new password at next sign-in.” → Click “Next.”

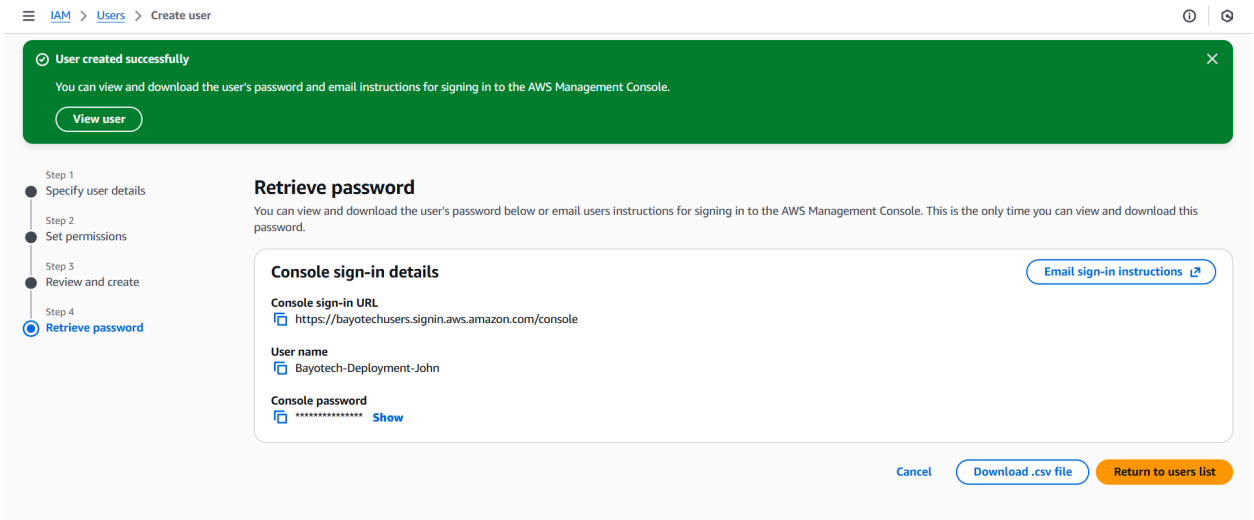


- On the ‘Set permission’ screen check **‘Add user to group’**, select the group you created earlier **(e.g., Bayotech-Deployment-Group).** → Click “Next.” → On ‘Review and create’ leave as default→ Click “Create user.”

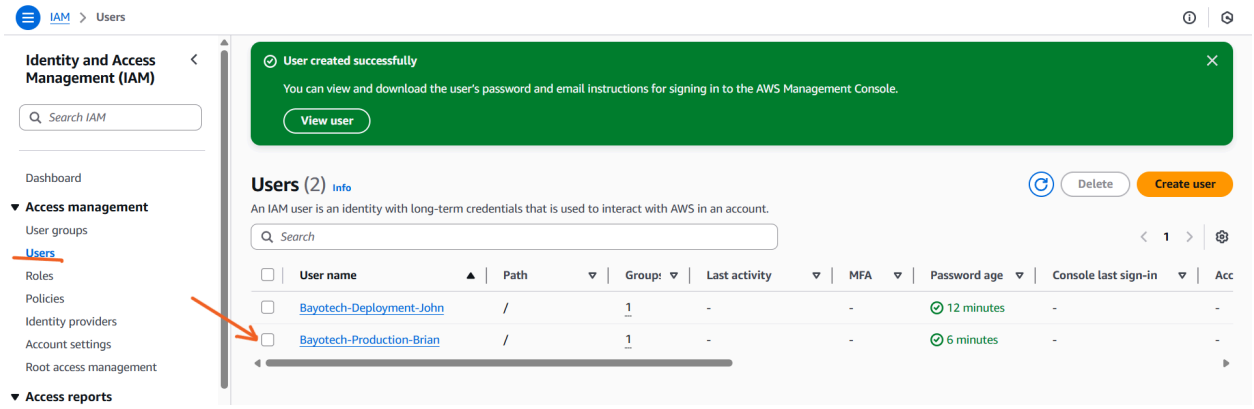




- On the Retrieve password page, download the generated .csv file this file contains the user's login credentials (username, password, and console sign-in URL).
- Copy the AWS Console Sign-In URL, which will reflect your custom alias (e.g., https://Bayotechusers.signin.aws.amazon.com/console).

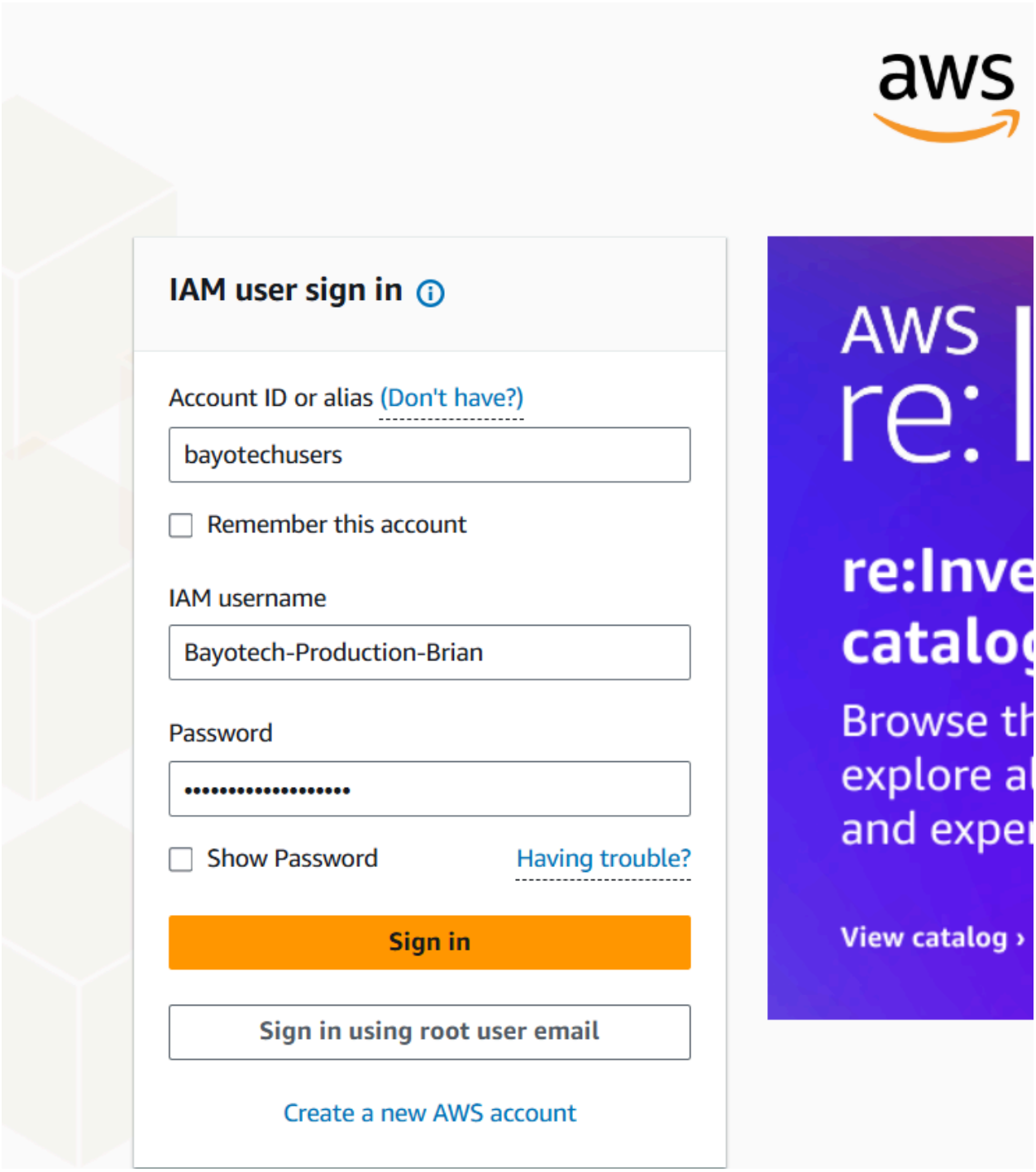


Repeat the same process to create a user assigned to the Distribution group . Return to the Users tab ,you'll now see the newly created users along with the assigned group and attached policies.



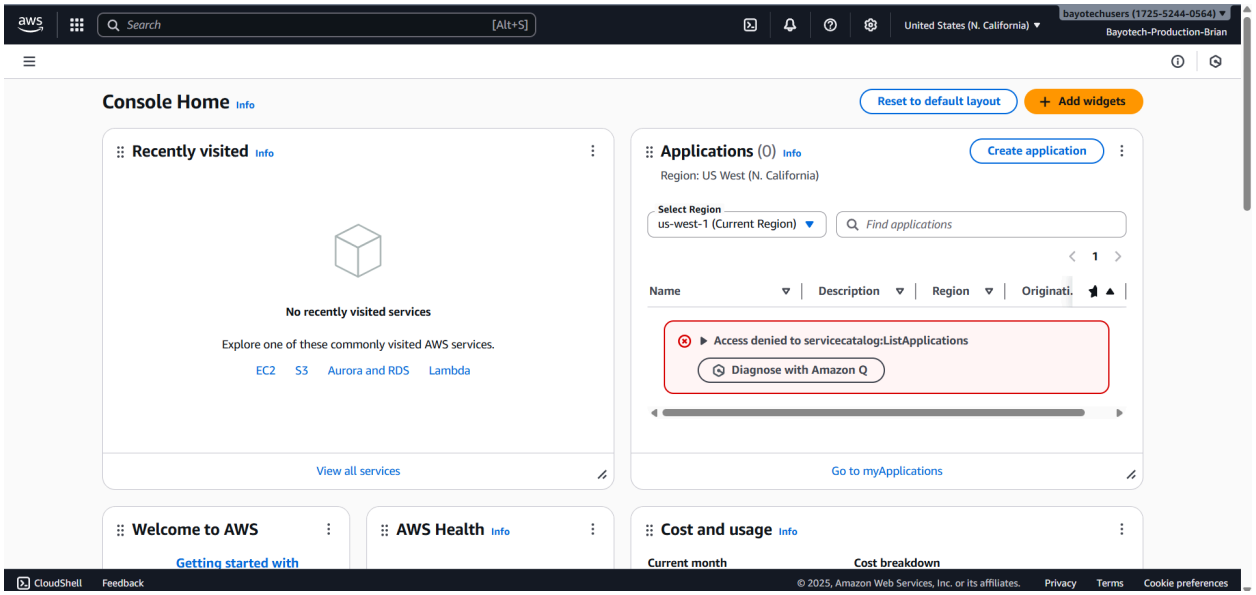
STEP 6→ Test IAM User Access

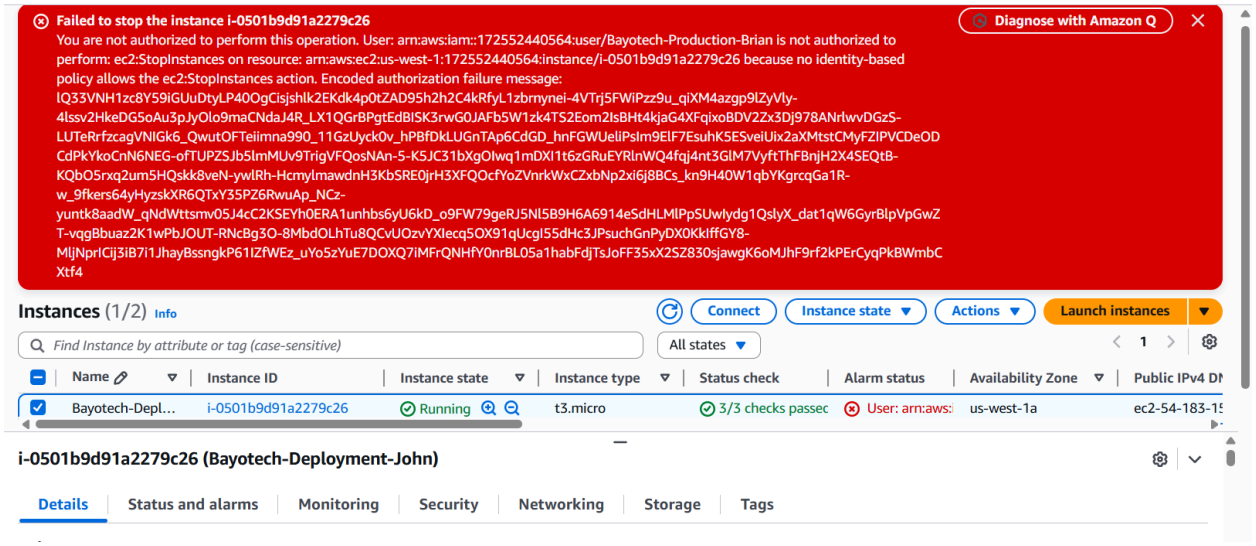
- Open a new browser window→ Go to the AWS Console (I will be implementing the Sign-In URL i copied earlier, for example:(https://Bayotechusers.signin.aws.amazon.com/console)
- Log in using the IAM user credentials from the downloaded .csv file:
    - Username: Bayotech-Deployment-John
    - Password: (xxxxxxx)

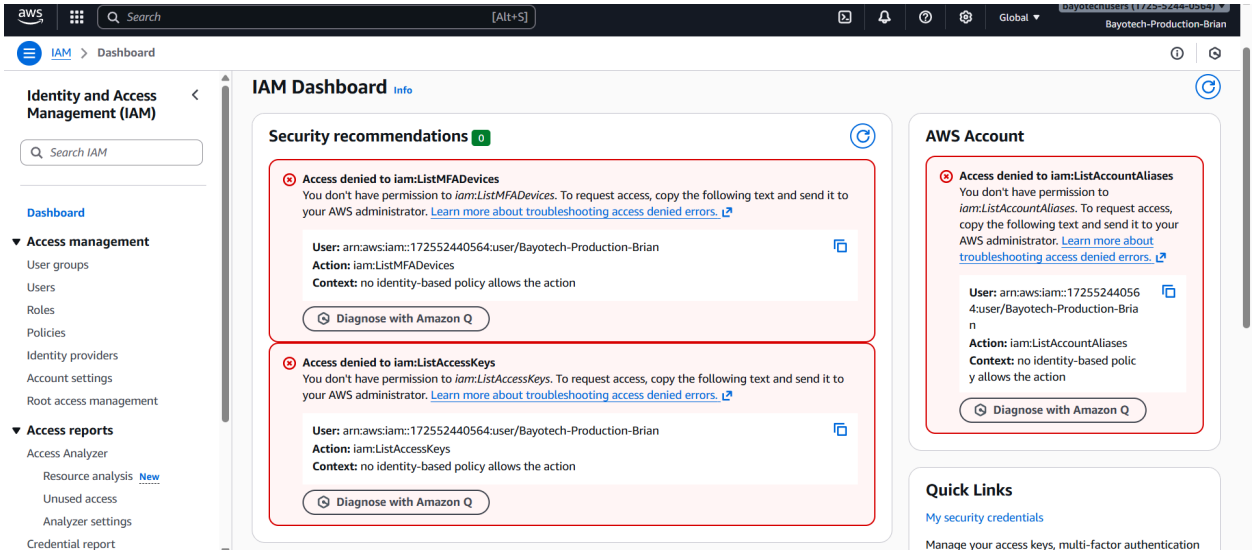


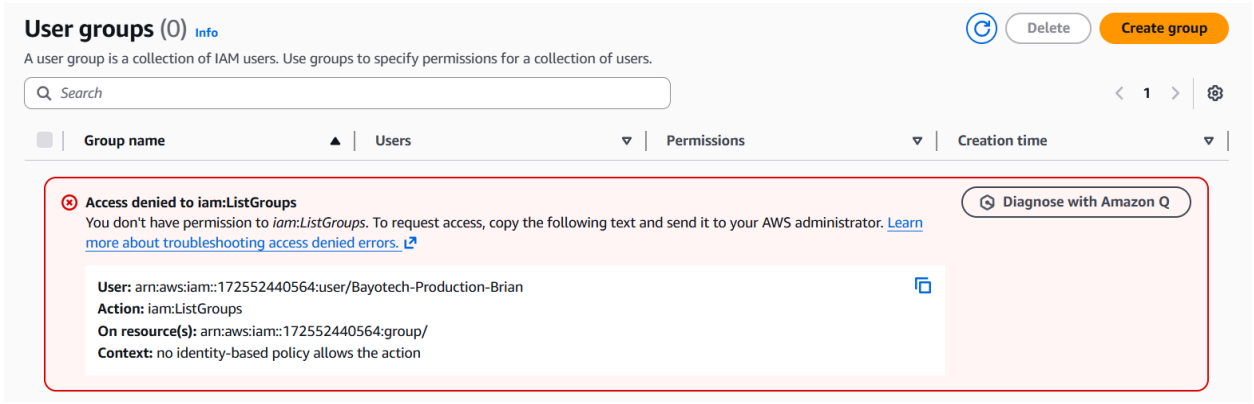
- Once logged in, you'll see the AWS Management Console ,but with limited access based on the policy attached to the user's group.
- I tried performing actions such as stopping EC2 instances and view IAM.
- If the IAM policy restricts certain actions (e.g.,Creating user group, launching untagged instances or modifying tags), AWS will display an "Access Denied" message.

This confirms that the IAM policy and group permissions are working as intended, enforcing the Principle of Least Privilege.









SUMMARY

This IAM configuration demonstrates how to:

- Securely manage user identities and permissions in AWS.
- Enforce policy-based access control using JSON configurations.
- Maintain account security and compliance with AWS best practices.

By combining IAM policies, groups, and user testing, I've established a scalable and secure structure the foundation for any enterprise-level Cloud Security and Governance strategy.