Introduction to Cloud Computing — Security & Identity Management (IAM)

Author: Oluwaseun Osunsola
Environment: AWS
Project Link: https://github.com/Oluwaseunoa/DevOps-Projects/blob/main/Cloud%20Computing/Security%20%26%20Identity%20Management%20(IAM)Security%20%26%20Identity%20Management%20(IAM)/README.mdes

---

1. Introduction

This project focuses on **Amazon Web Services (AWS) Identity and Access Management (IAM)** and its application in securing cloud resources for Zappy e-Bank, a fintech startup.
The objective is to learn how IAM can help enforce **authentication and authorization** for users, ensuring that sensitive financial data is properly secured.

Through this project, I gained hands-on experience in creating **policies, groups, users, and multi-factor authentication (MFA)**, while applying the **principle of least privilege**.

---

2. Importance of IAM for Zappy e-Bank

For a fintech company like Zappy e-Bank, **security and compliance** are critical. IAM helps to:

- Create and manage AWS **users and groups**.
- Assign fine-grained **permissions** through IAM **policies**.
- Implement **multi-factor authentication (MFA)** for stronger access control.
- Enforce the **principle of least privilege**, ensuring users only have access to what they need.

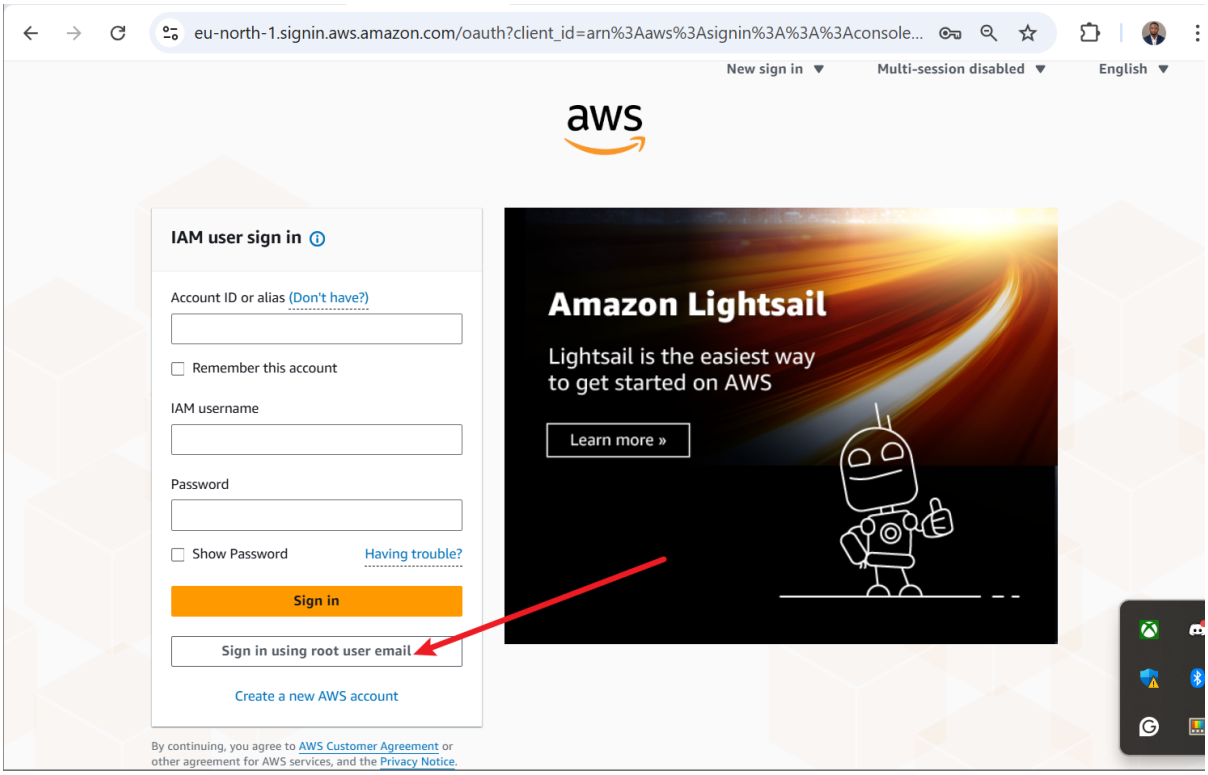
This reduces the risk of insider threats, unauthorized access, and compliance violations.

---

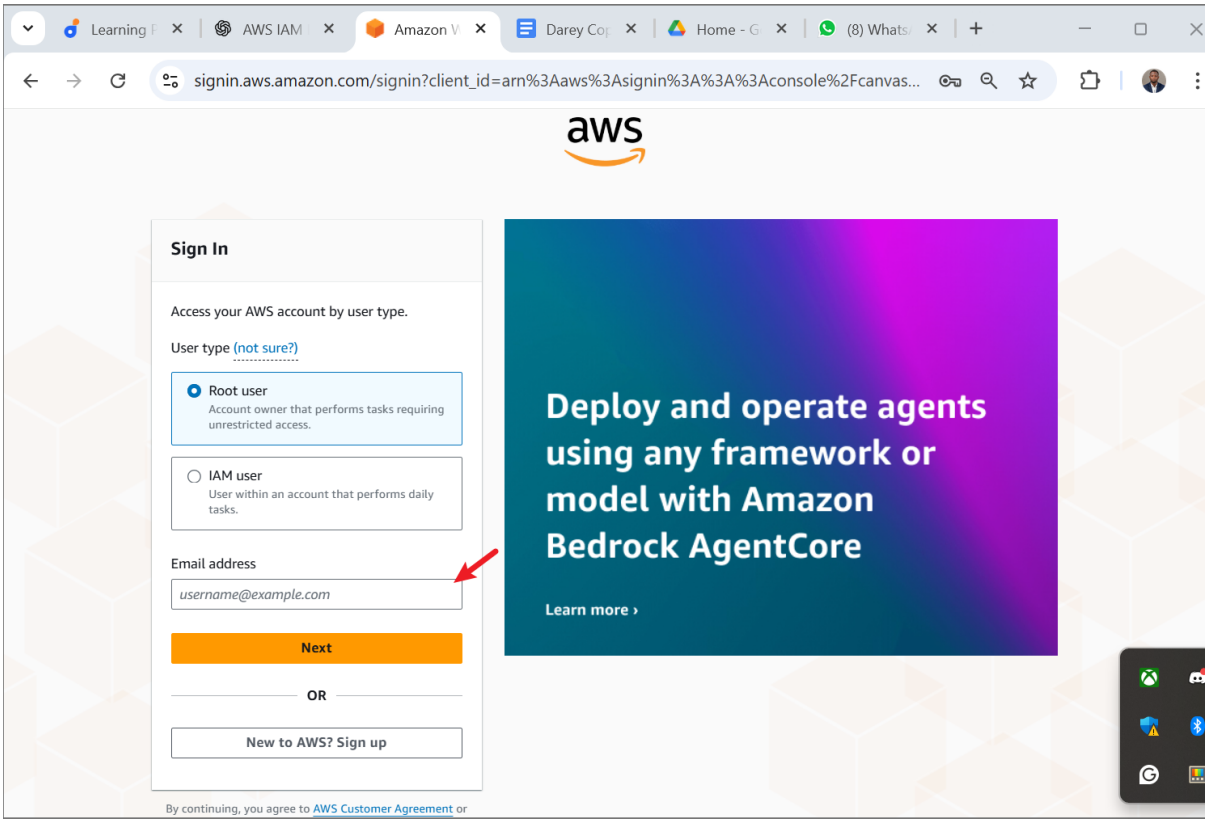
3. Project Setup

1. Logged into **AWS Management Console** using the administrator account.

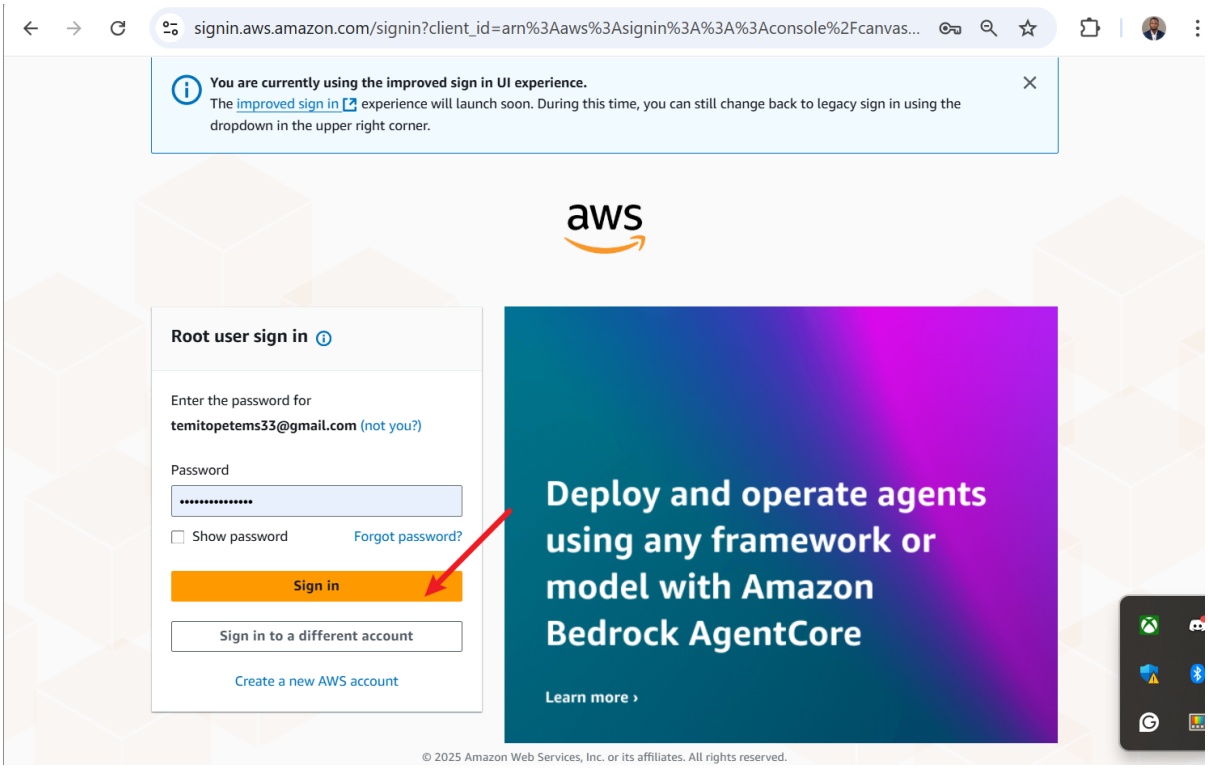
- Log in using root email.



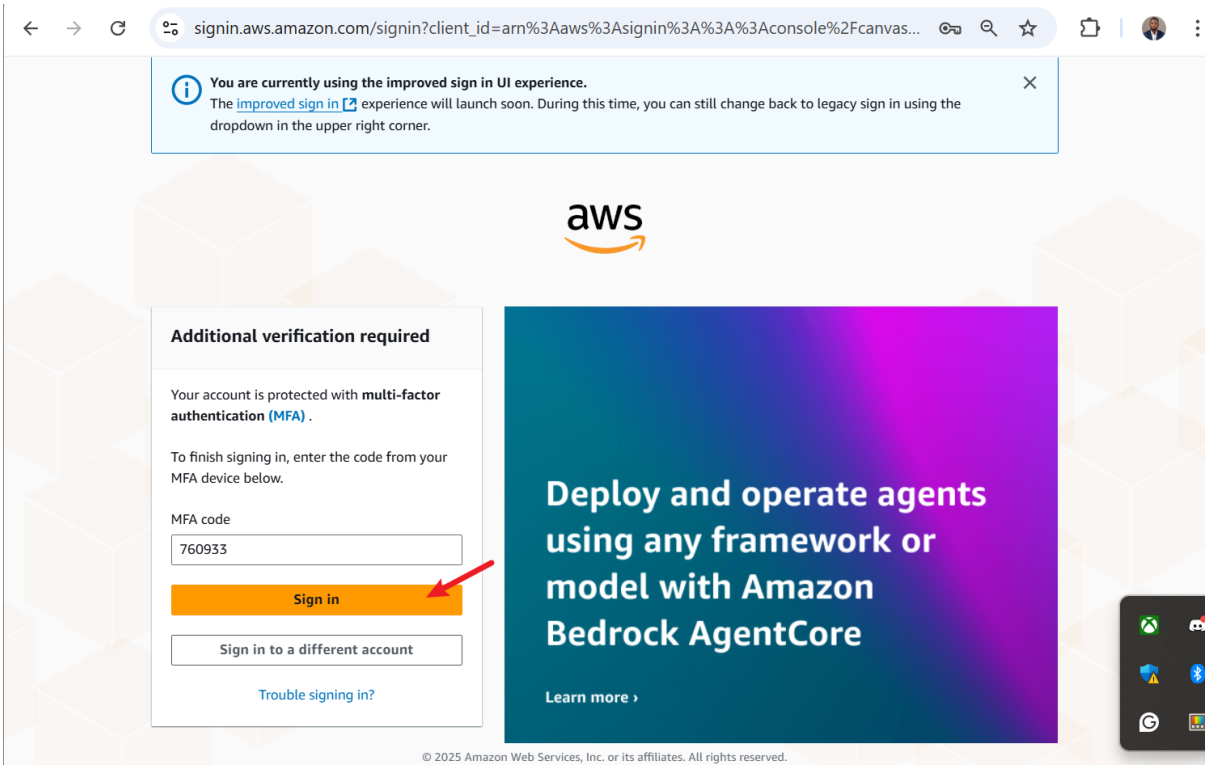
- Supply root user email.



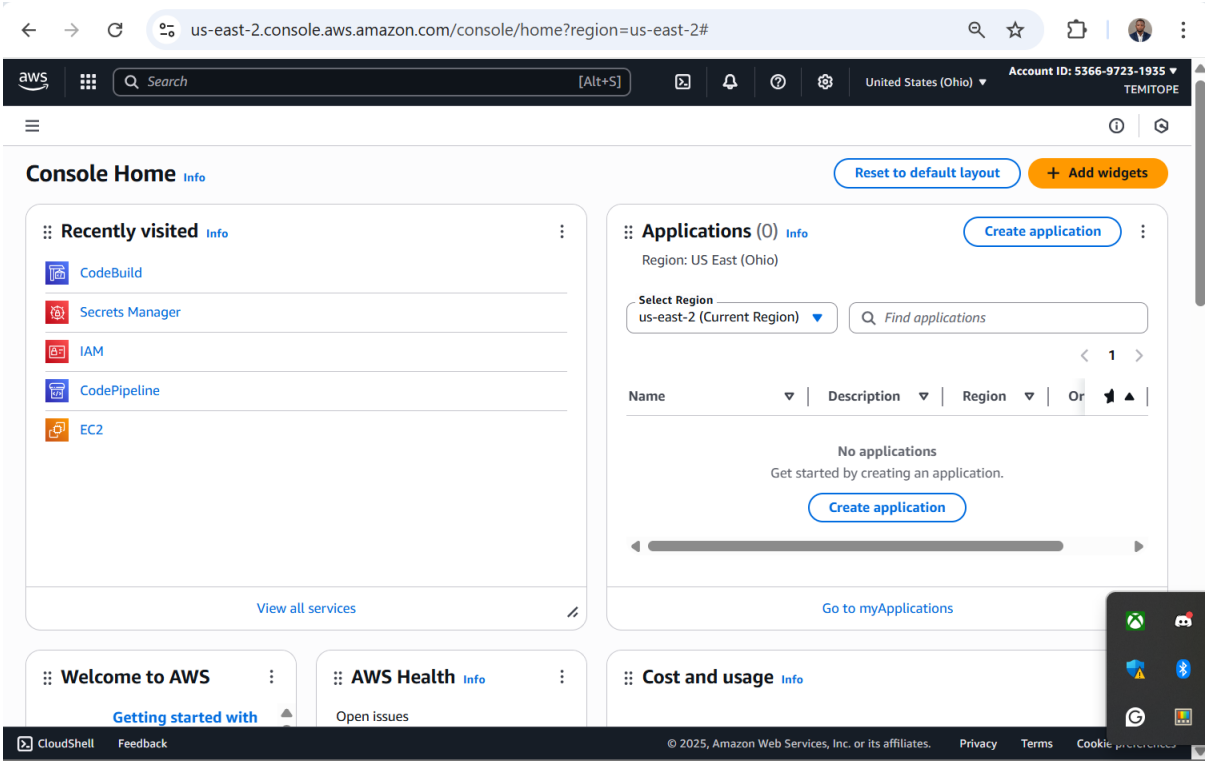
- Supply root user password and click sign in.



- Supply MFA code for additional verification.

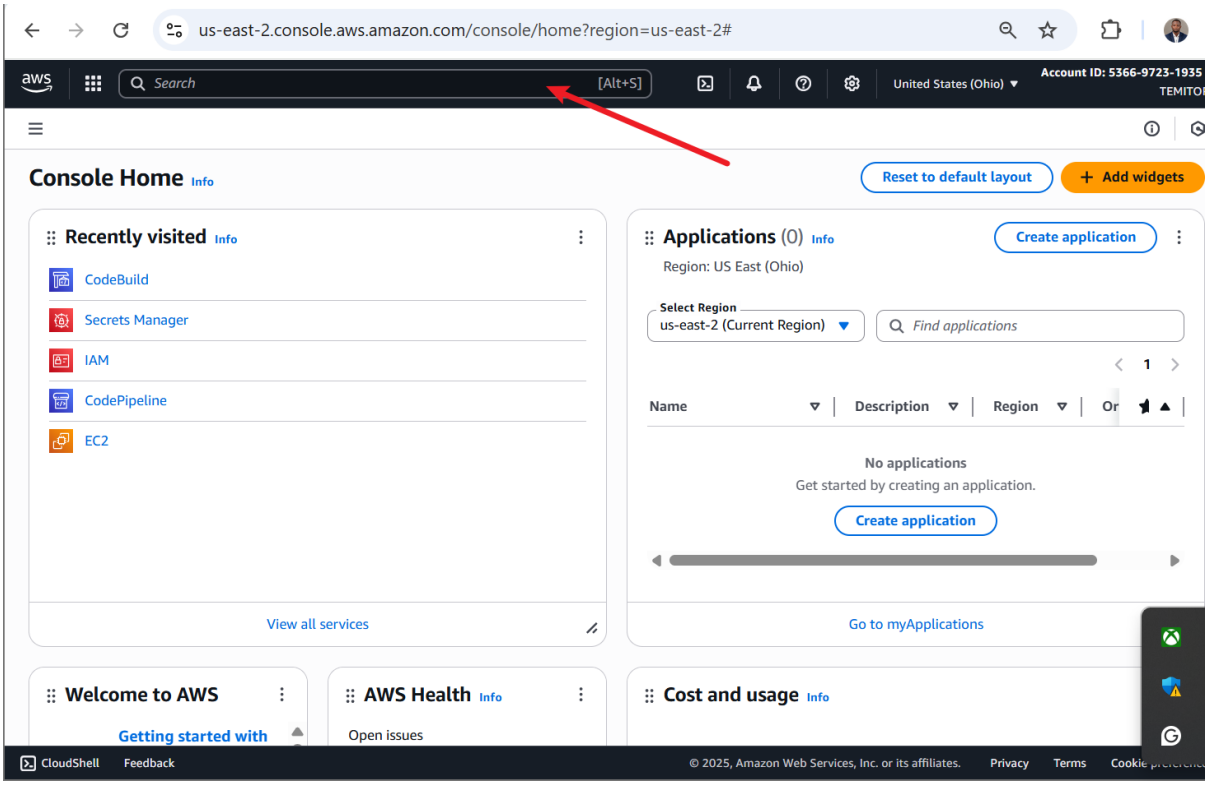


- Landed on the console dashboard.

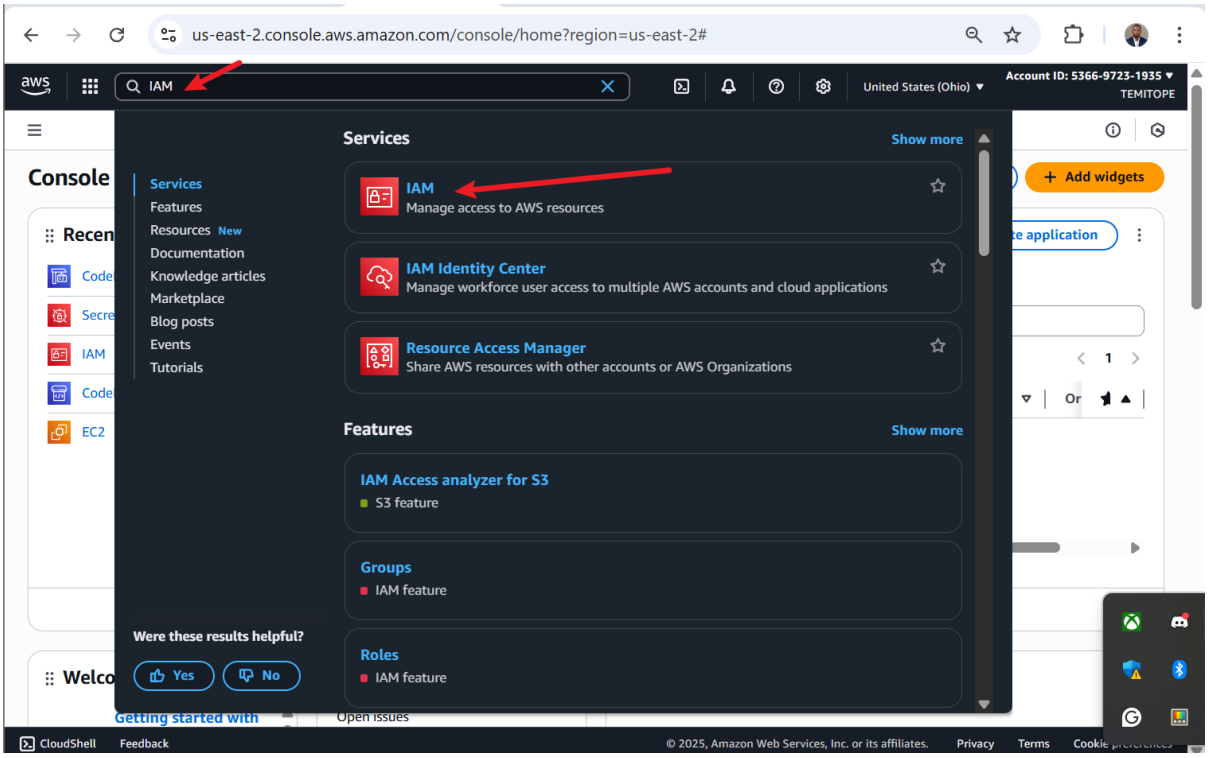


2. Navigated to the **IAM Dashboard**.

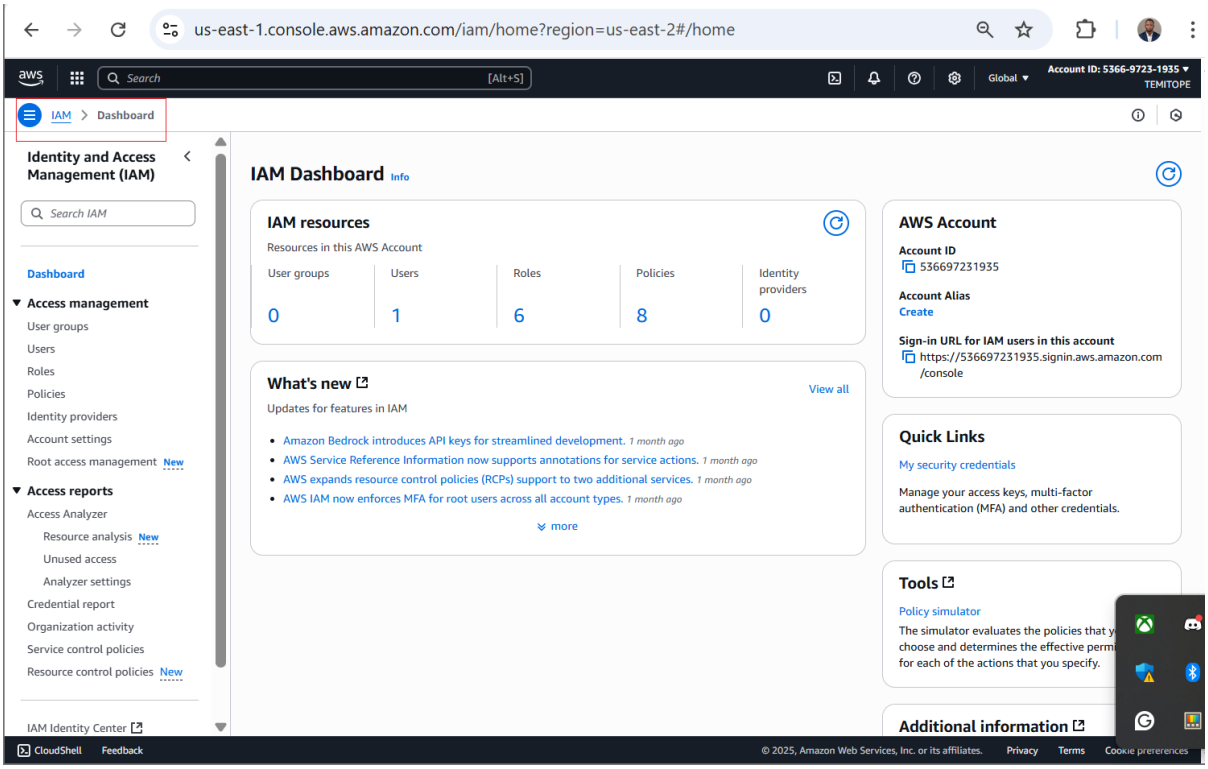
- Use the service search.



- Search for IAM and click on it.



- Now on IAM dashboard.



---

4. Creating IAM Policies

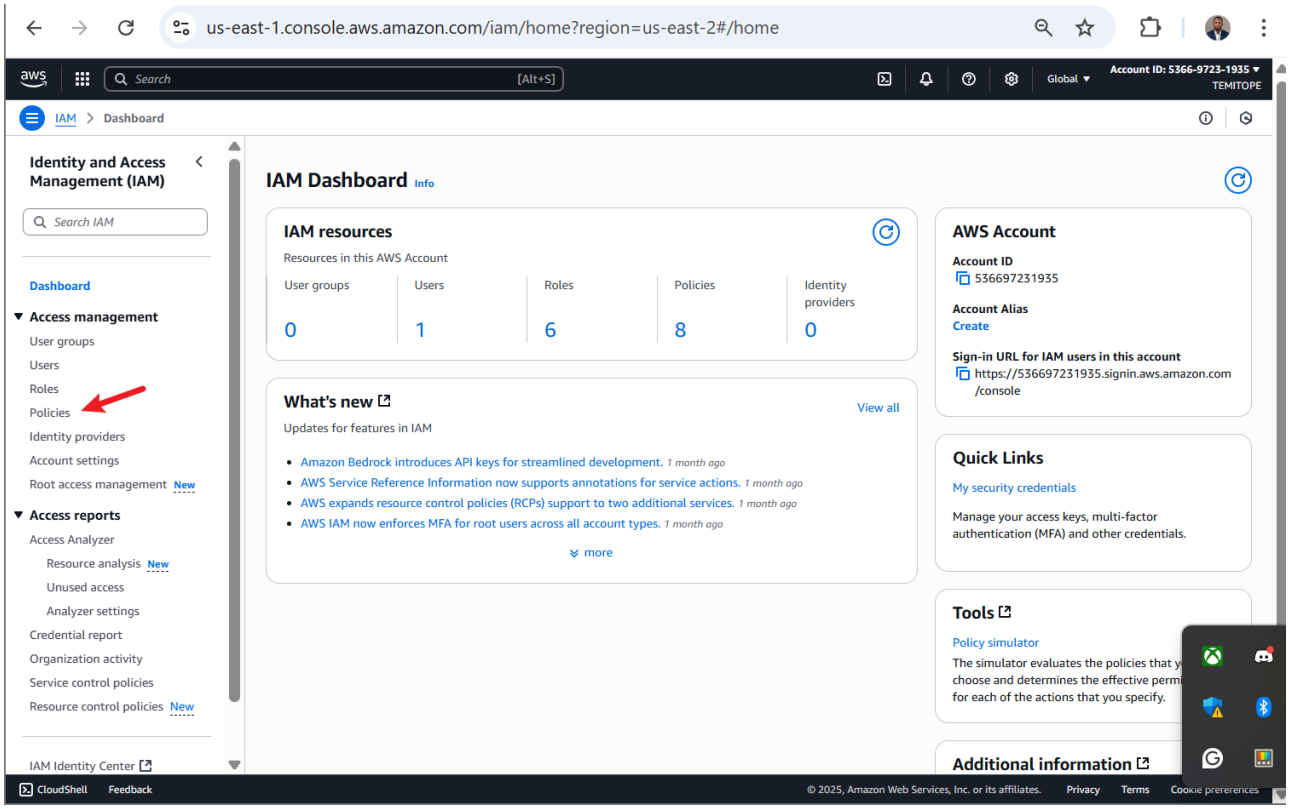
Developer Policy (EC2 Access)

- Created a **customer-managed policy** for backend developers.
  - Service: **EC2**.
  - Actions: **All EC2 actions**.
  - Resources: **All**.

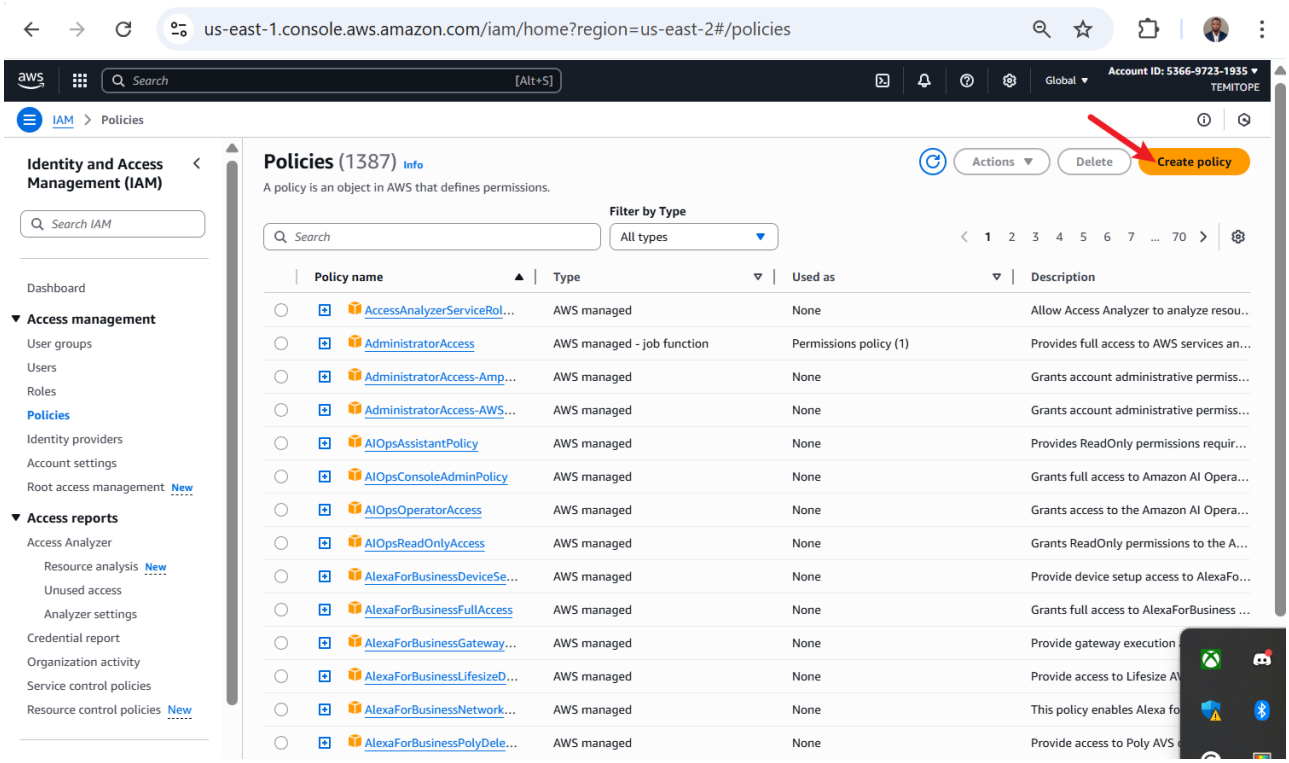
- Policy Name: `developer-policy`.

Steps:

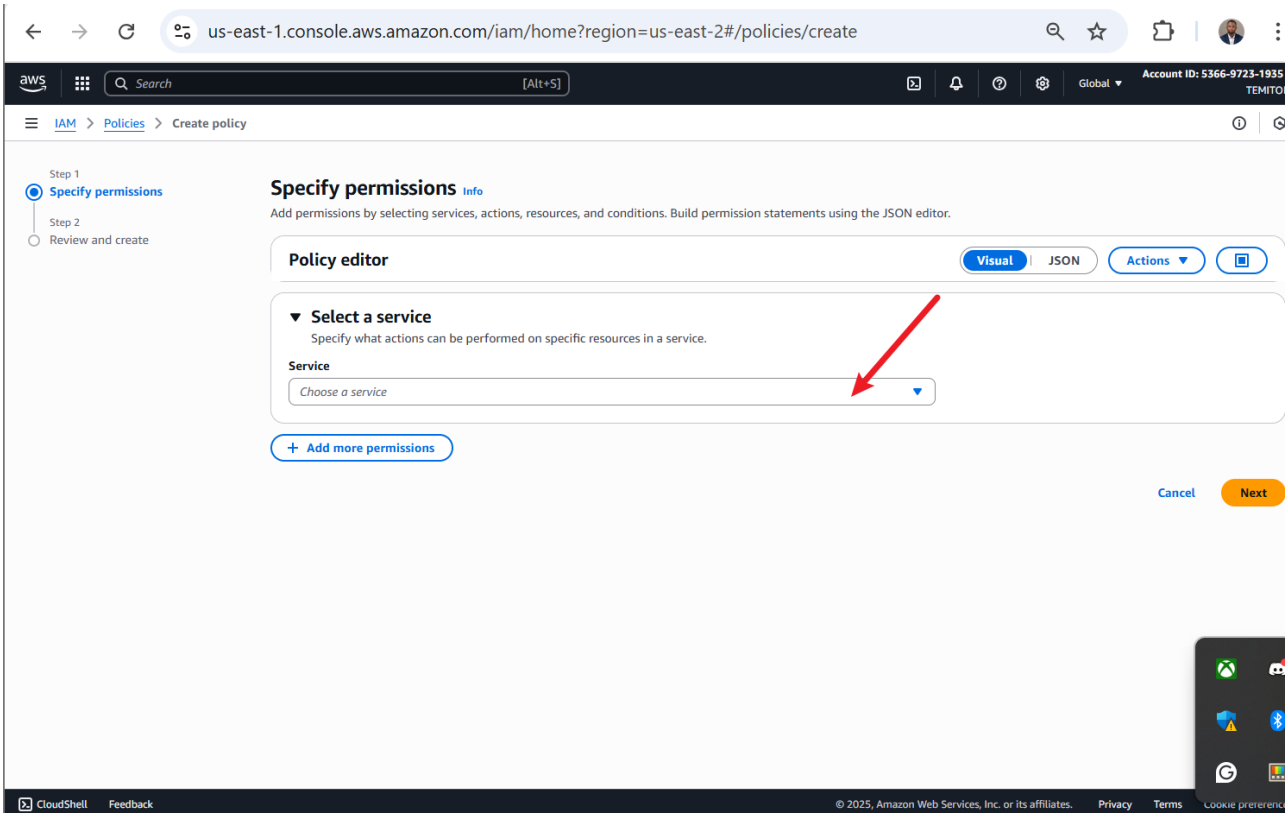
- Click on policies.



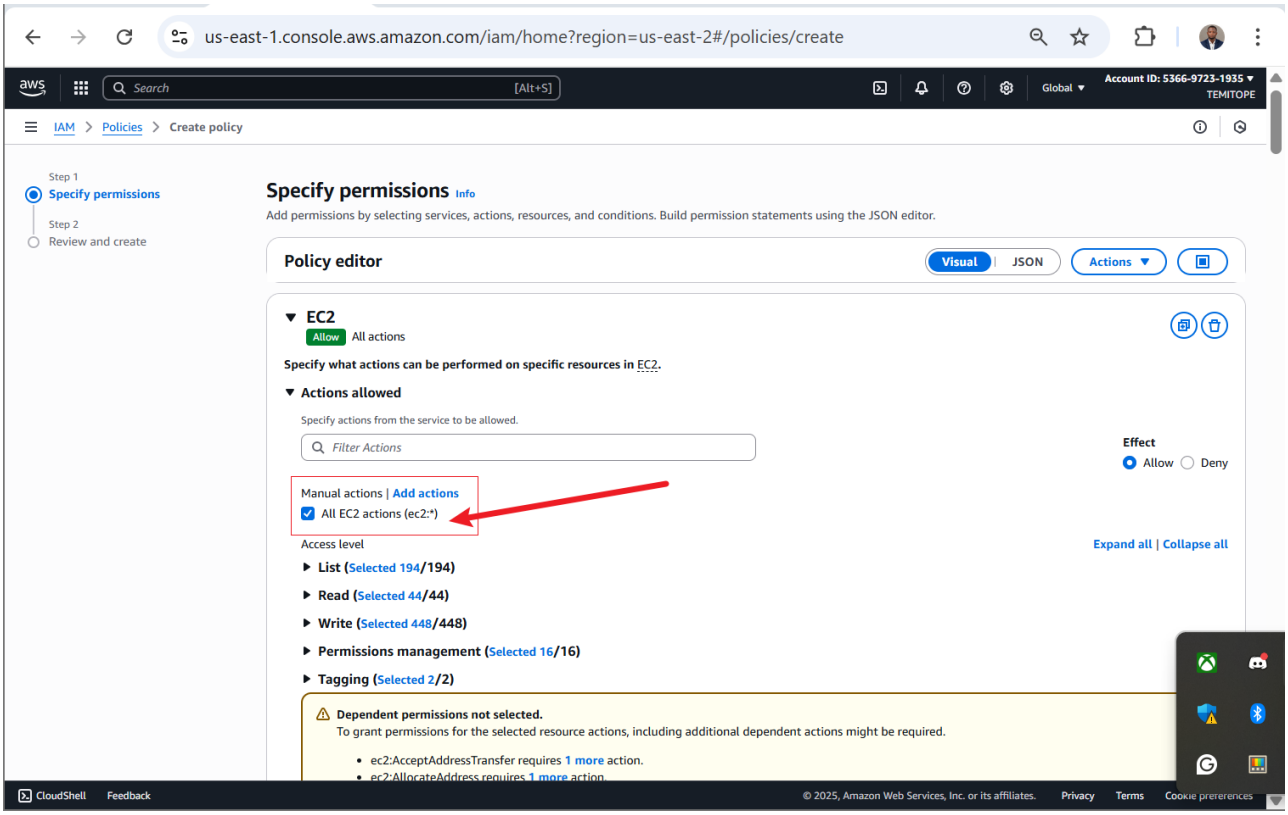
- Click on create policy.



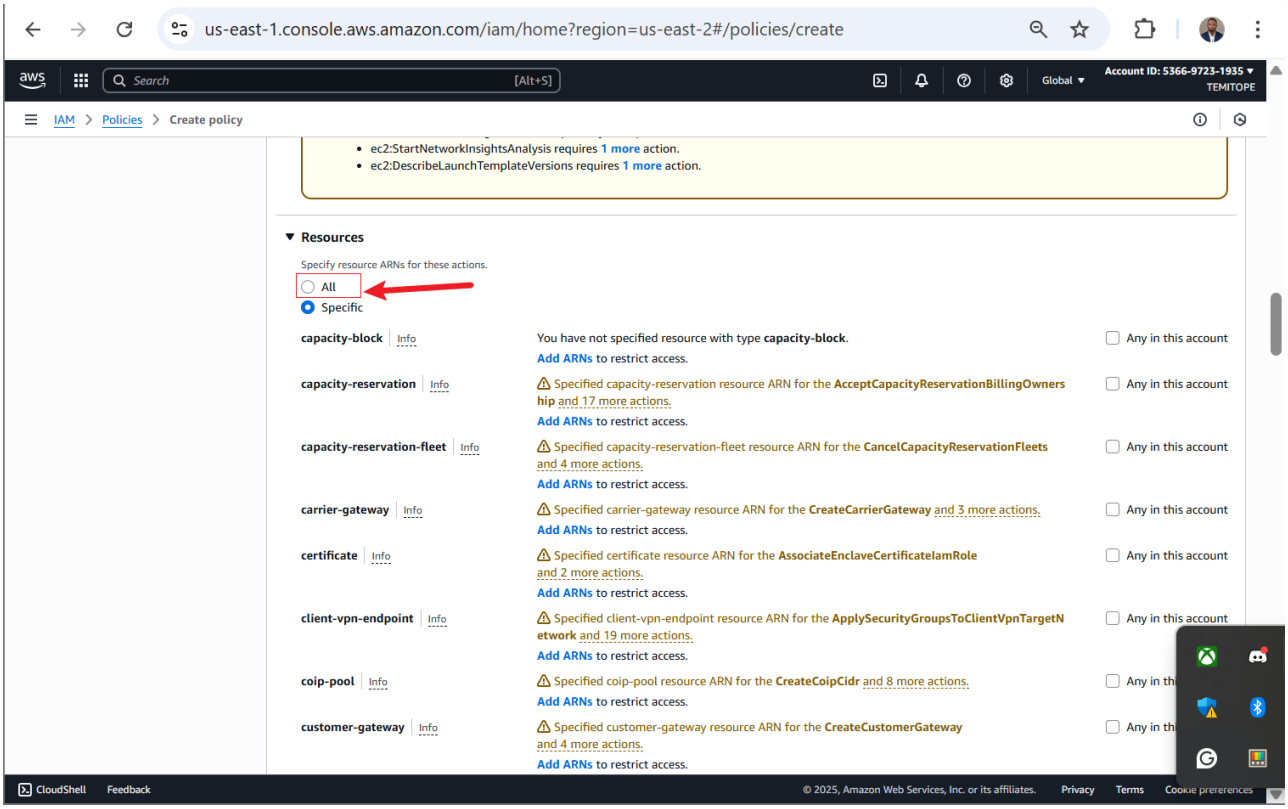
- Under select service section, choose EC2 service.



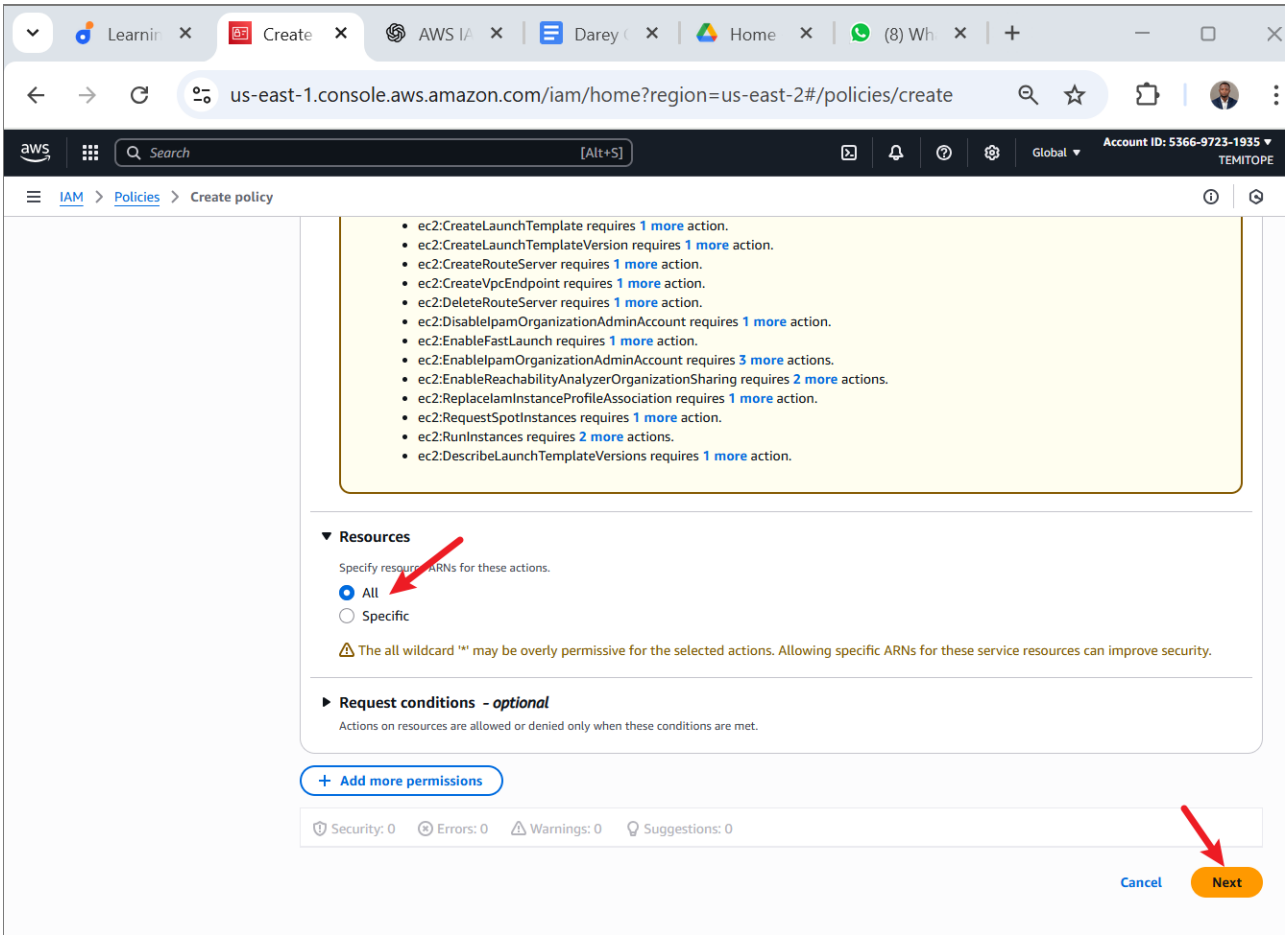
- Under action, check all EC2.



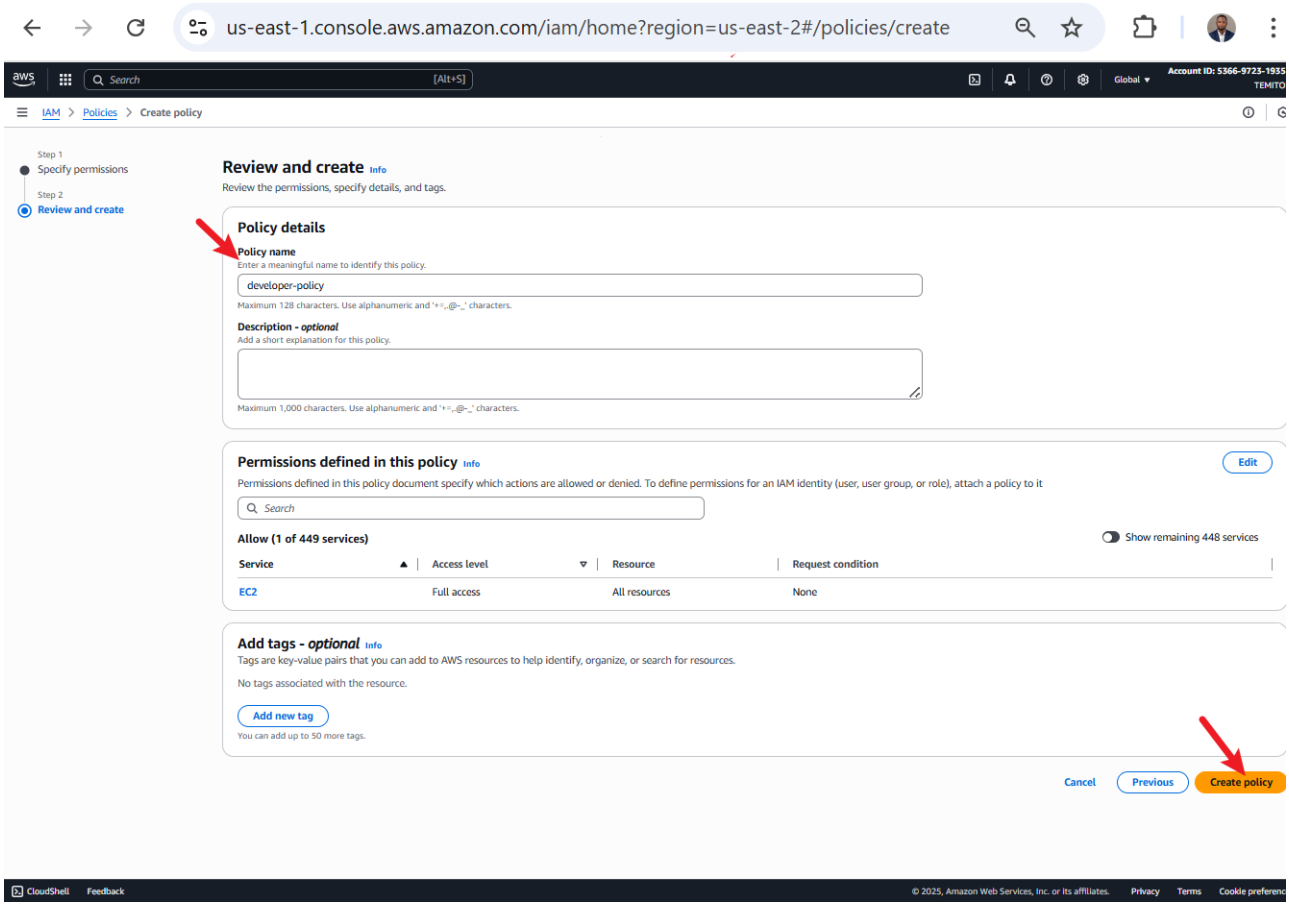
- Under resources, check all.



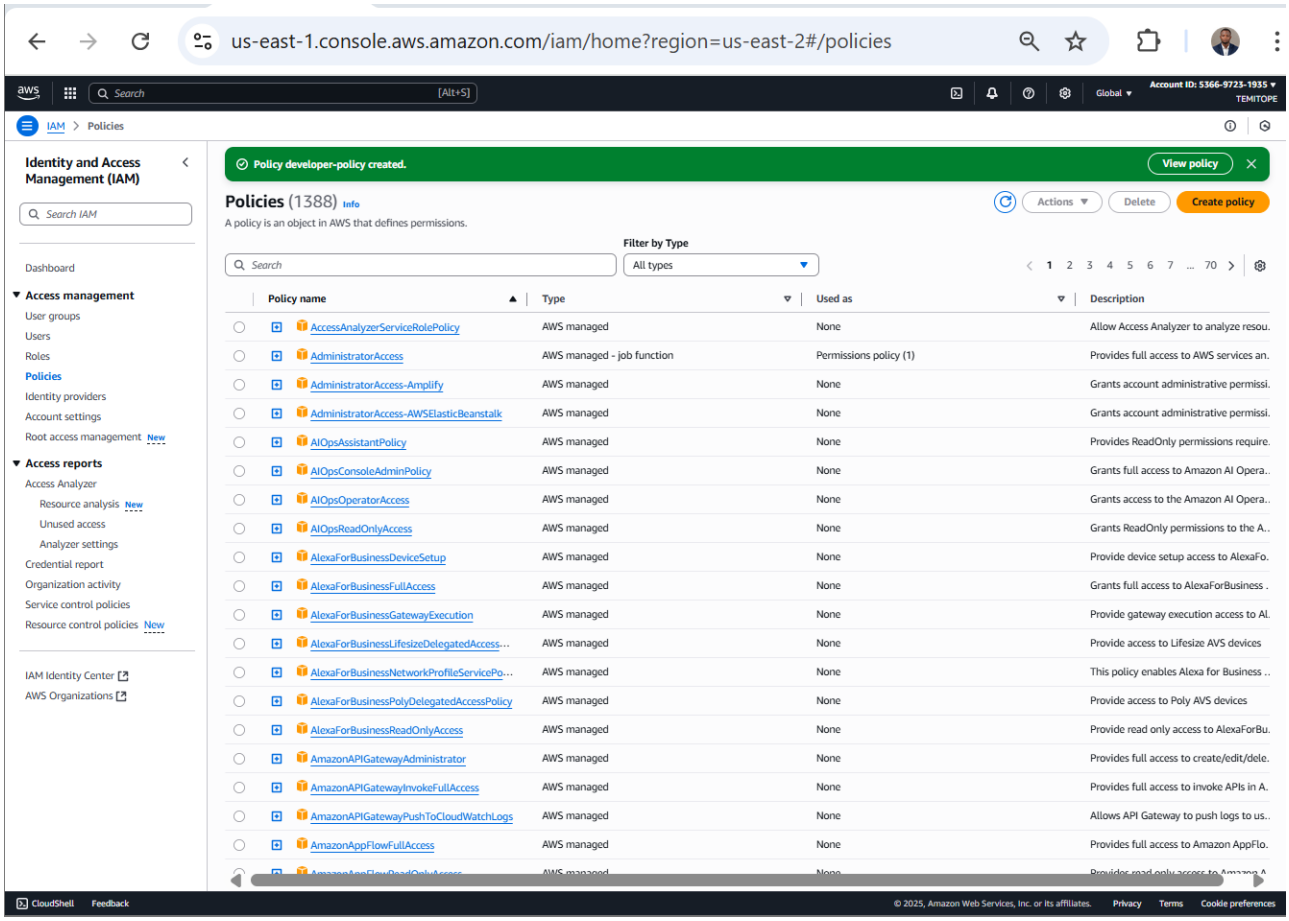
- Having checked all, click next.



- Name policy and click create policy.



- Developer policy successfully created.

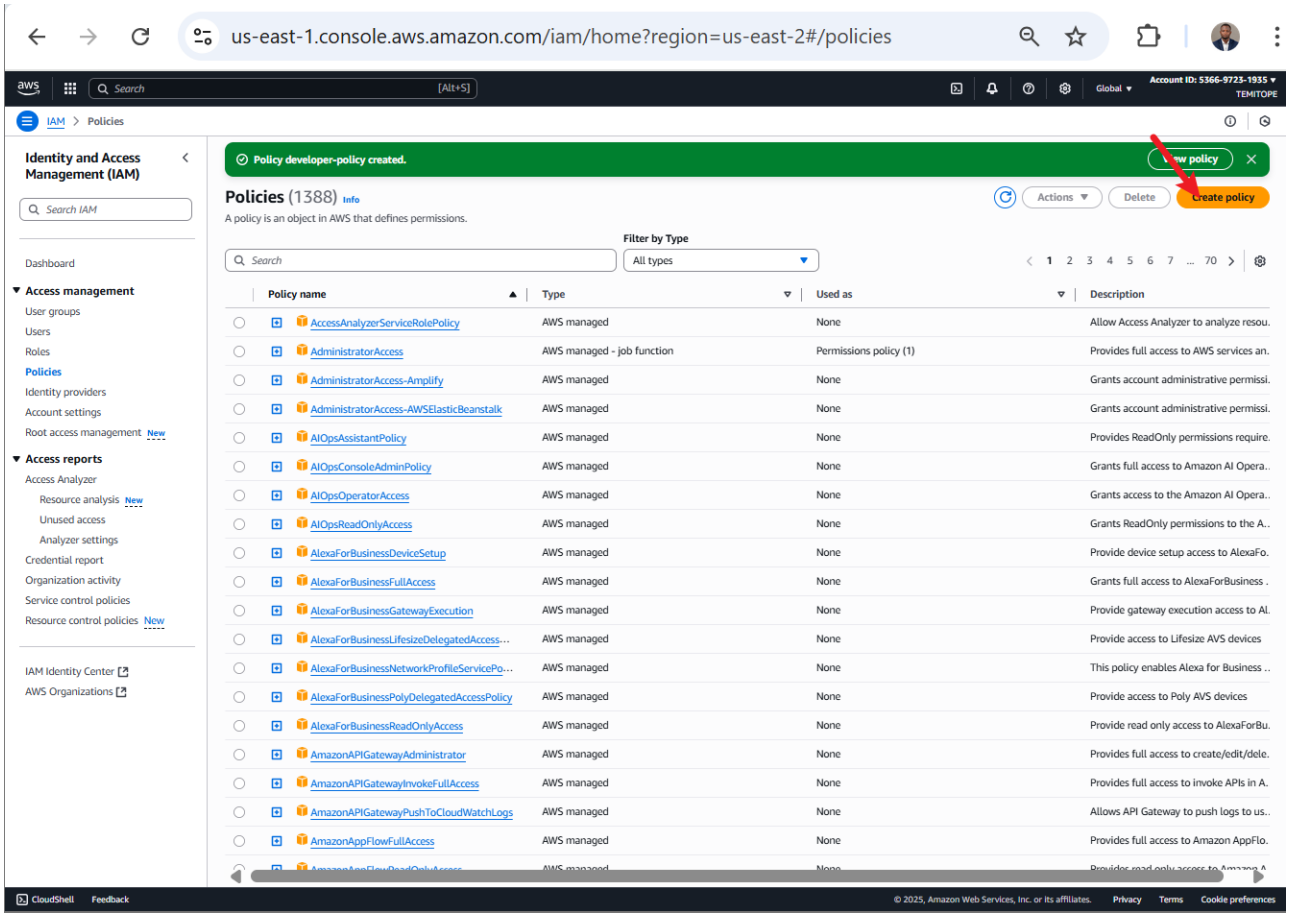


Analyst Policy (S3 Access)

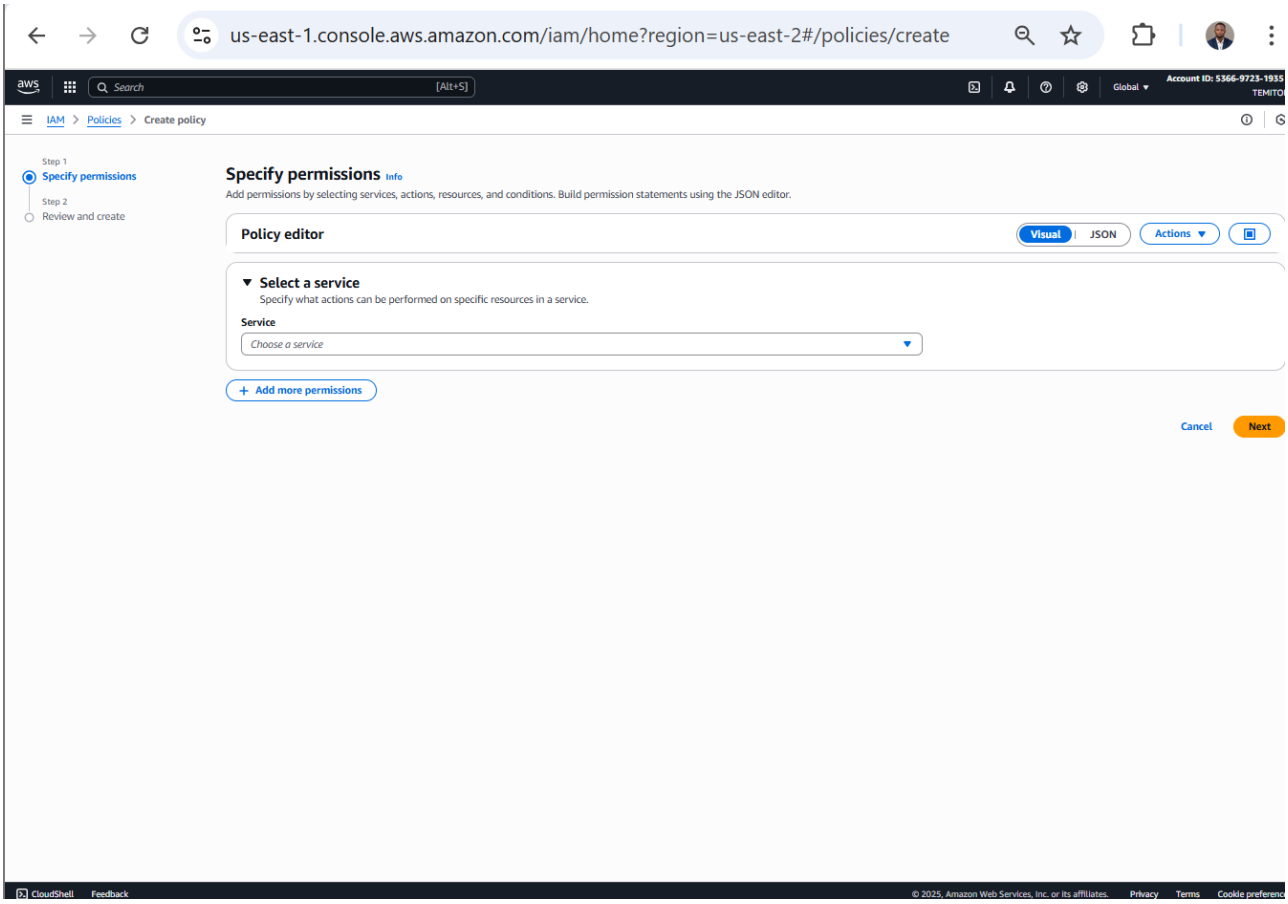
- Created a **customer-managed policy** for data analysts.
  - Service: **S3**.
  - Actions: **All S3 actions**.
  - Resources: **All**.
  - Policy Name: `analyst-policy`.

Steps:

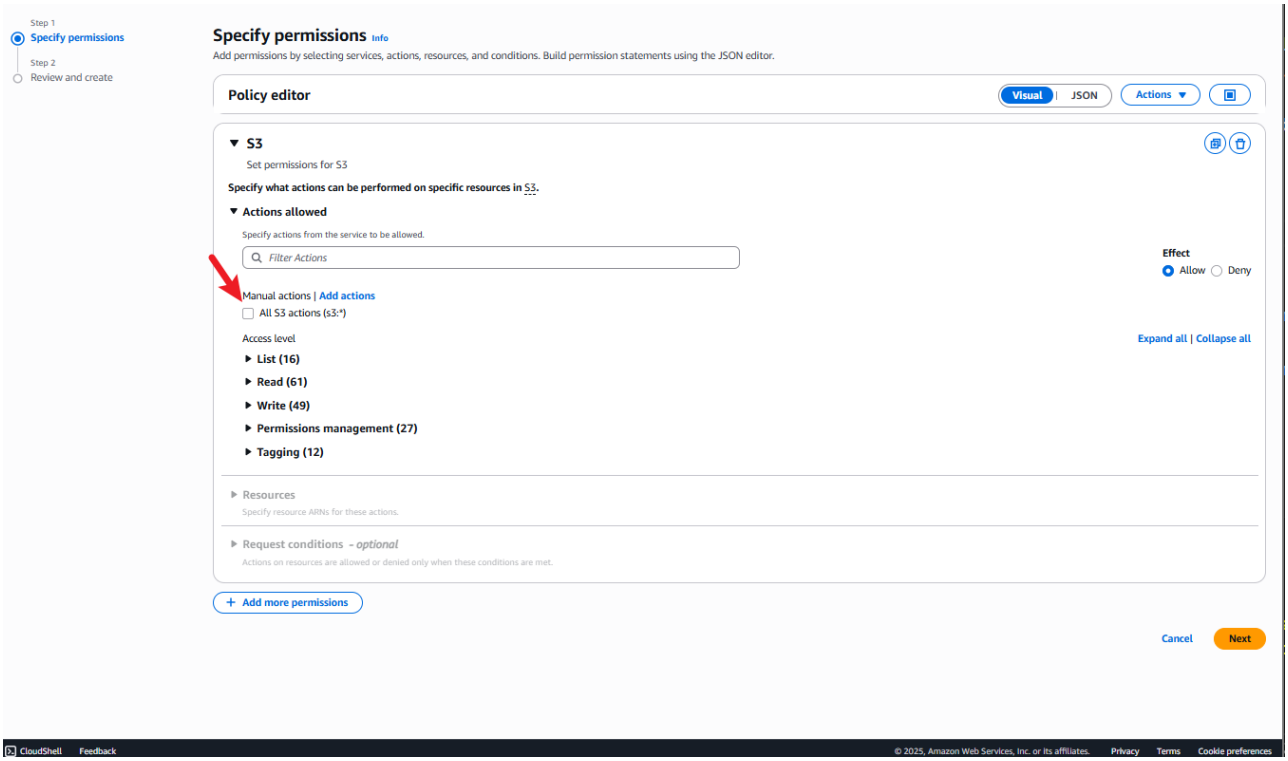
- Click create policy to create another one.



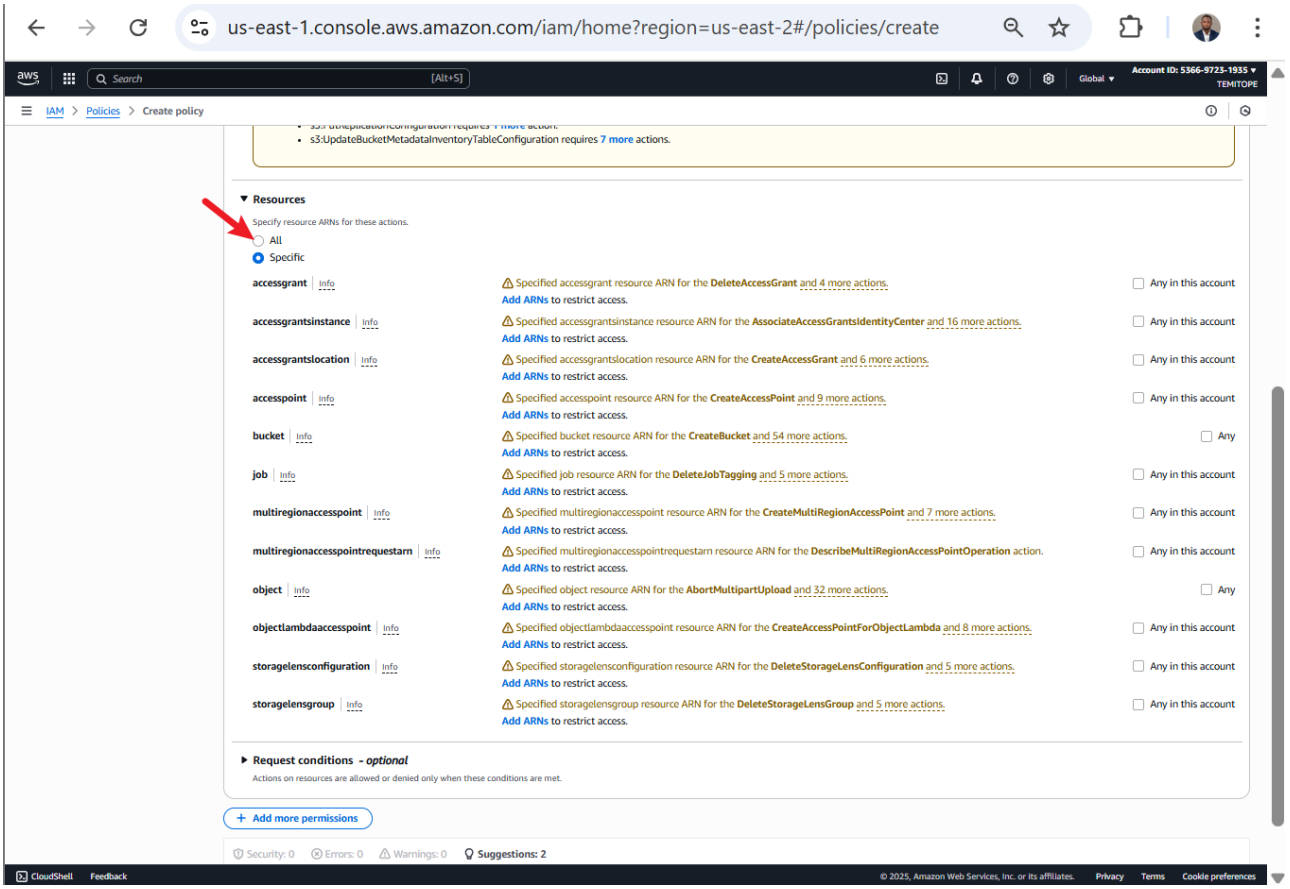
- Under select a service, choose S3.



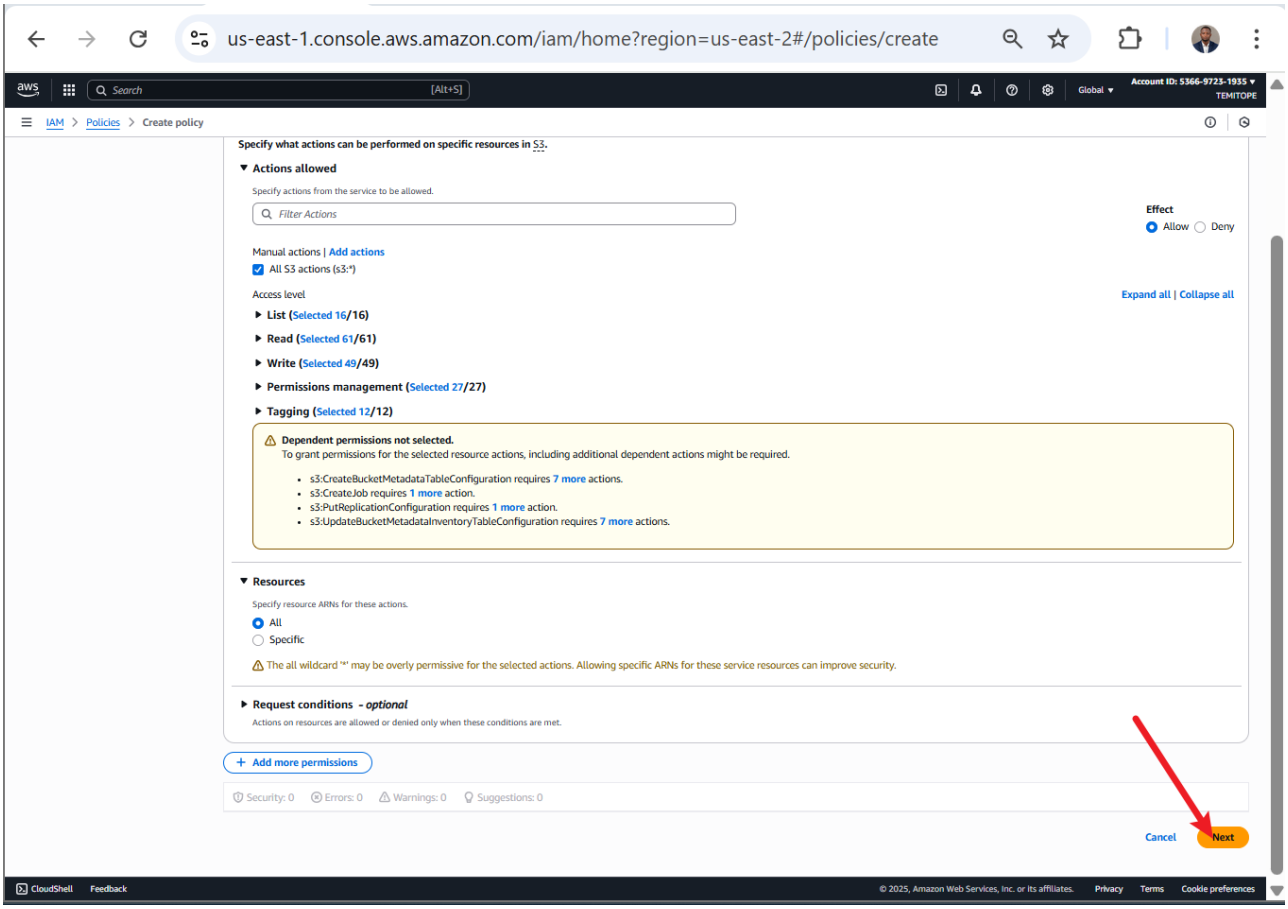
- Under action, check all S3.



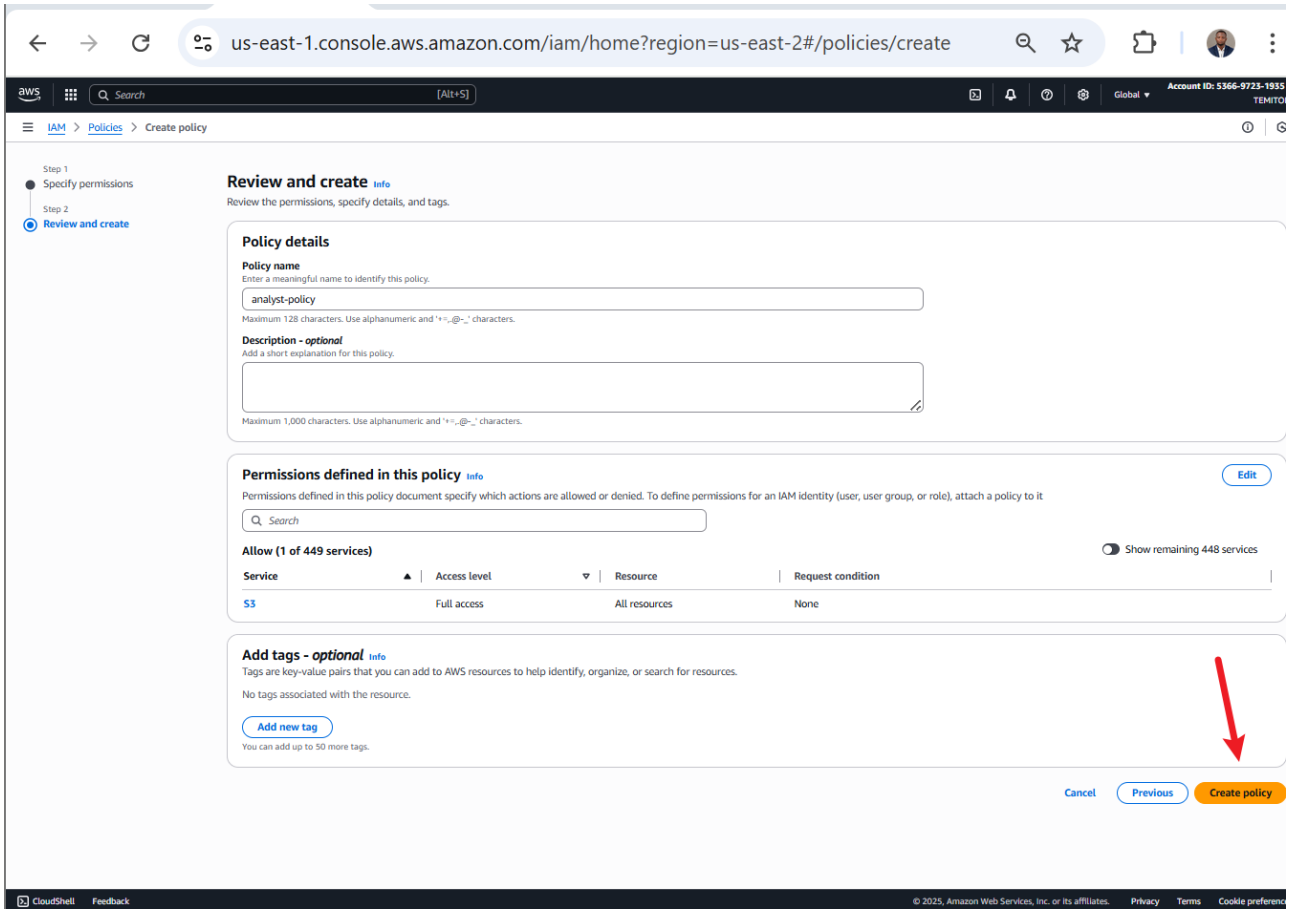
- Under resources, check all.



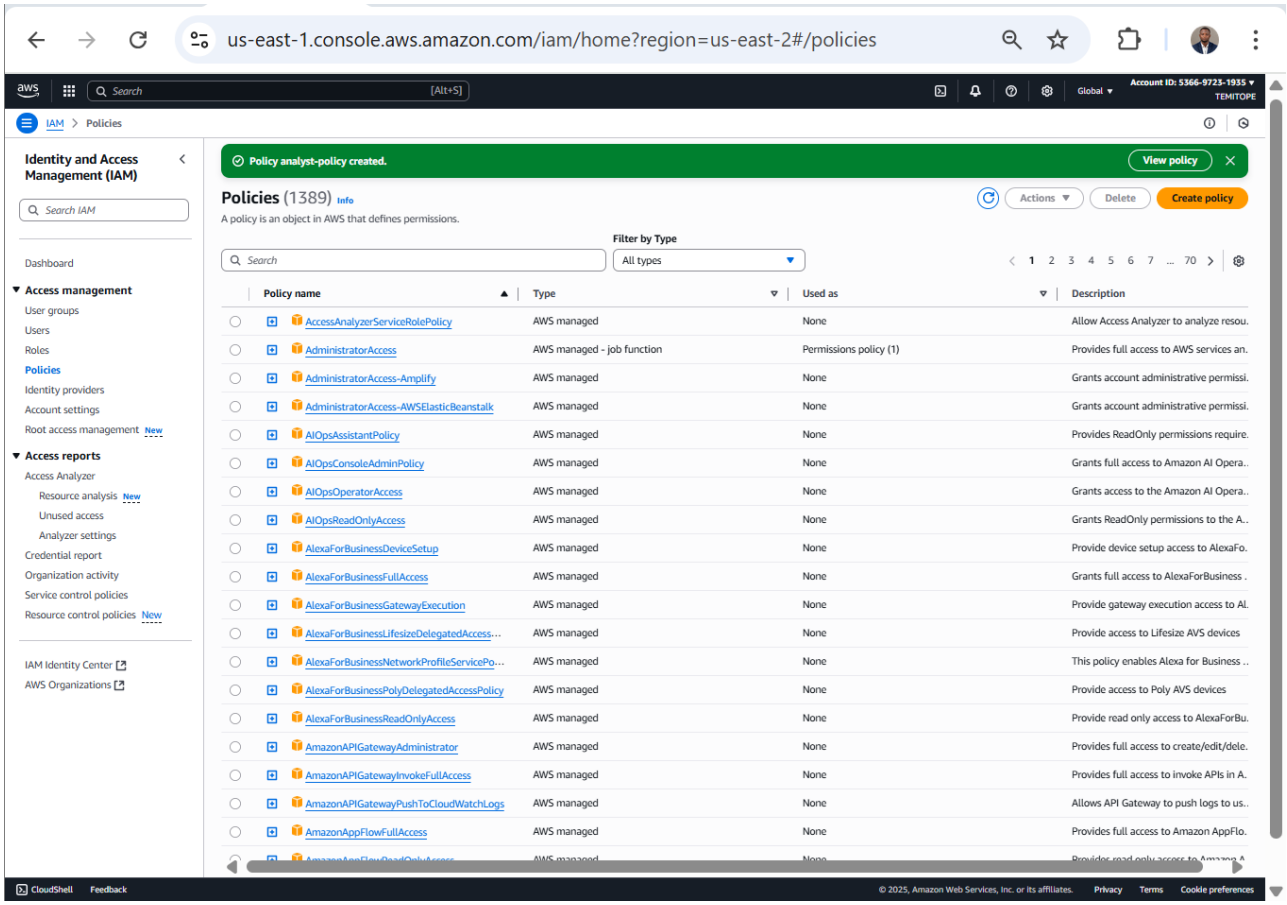
- Having checked all, click next.



- Name policy and click create policy.



- Analyst policy successfully created.

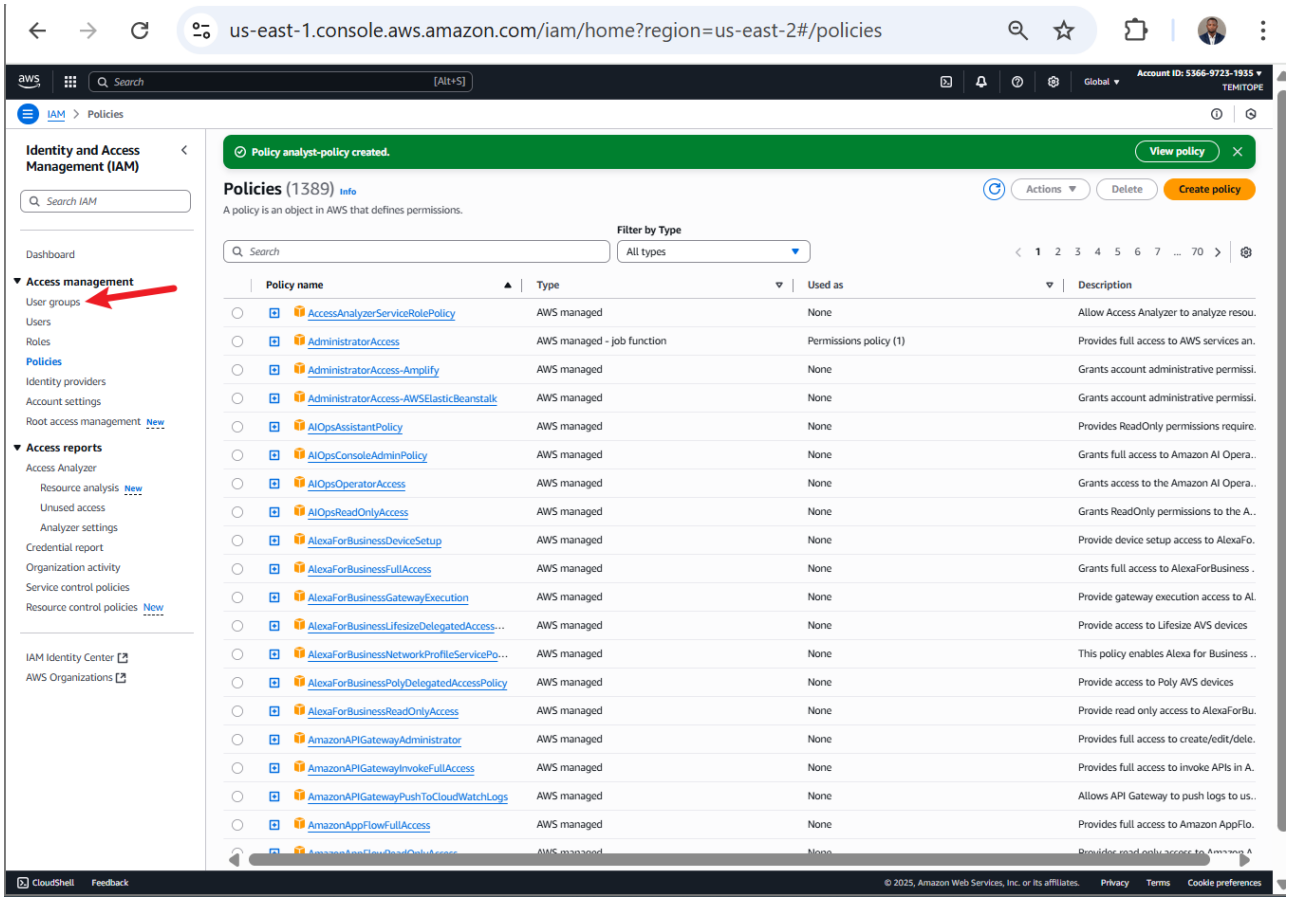


5. Creating IAM Groups

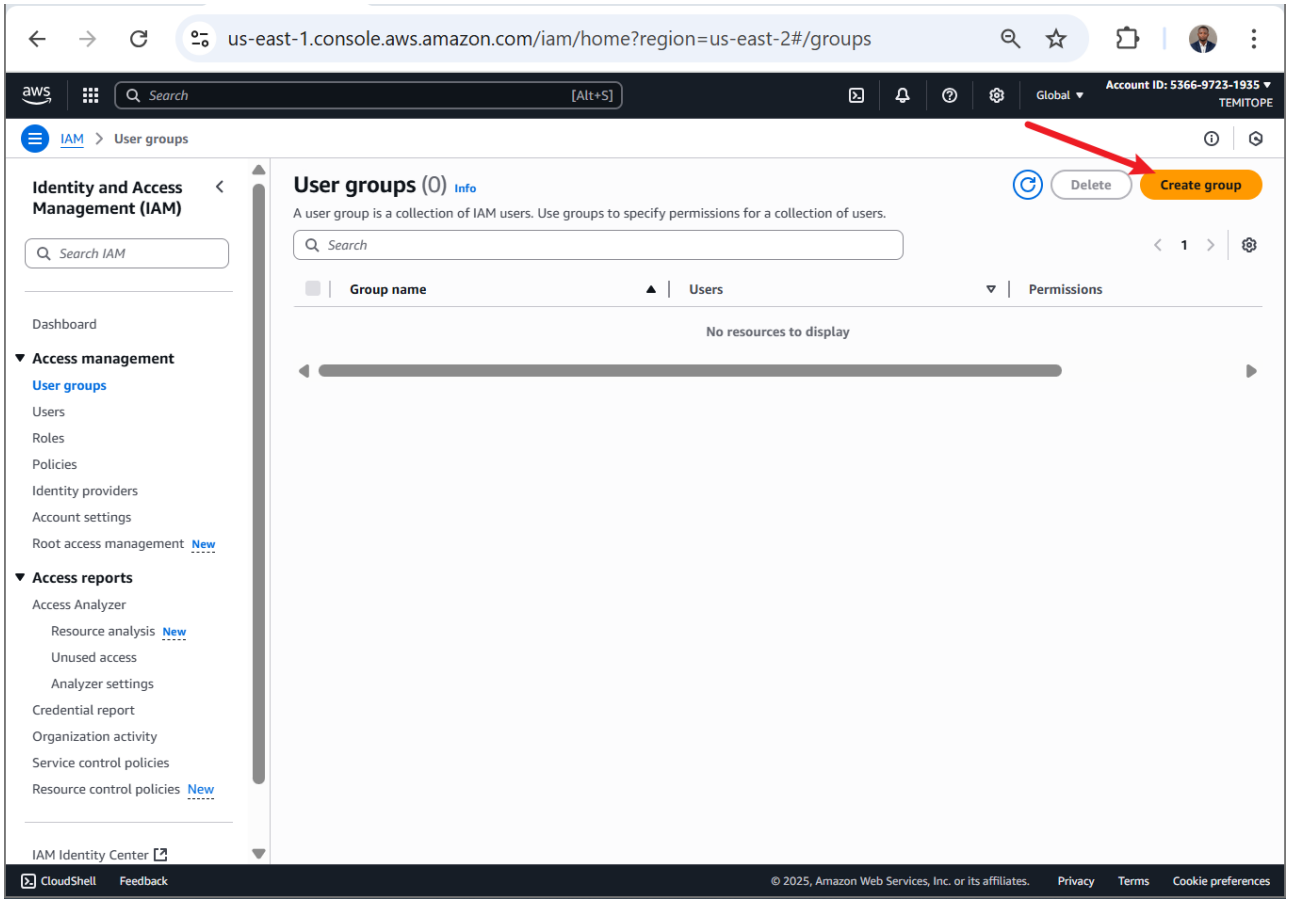
- **Development-Team** → Attached `developer-policy`.
- **Analyst-Team** → Attached `analyst-policy`.

Steps for Development-Team:

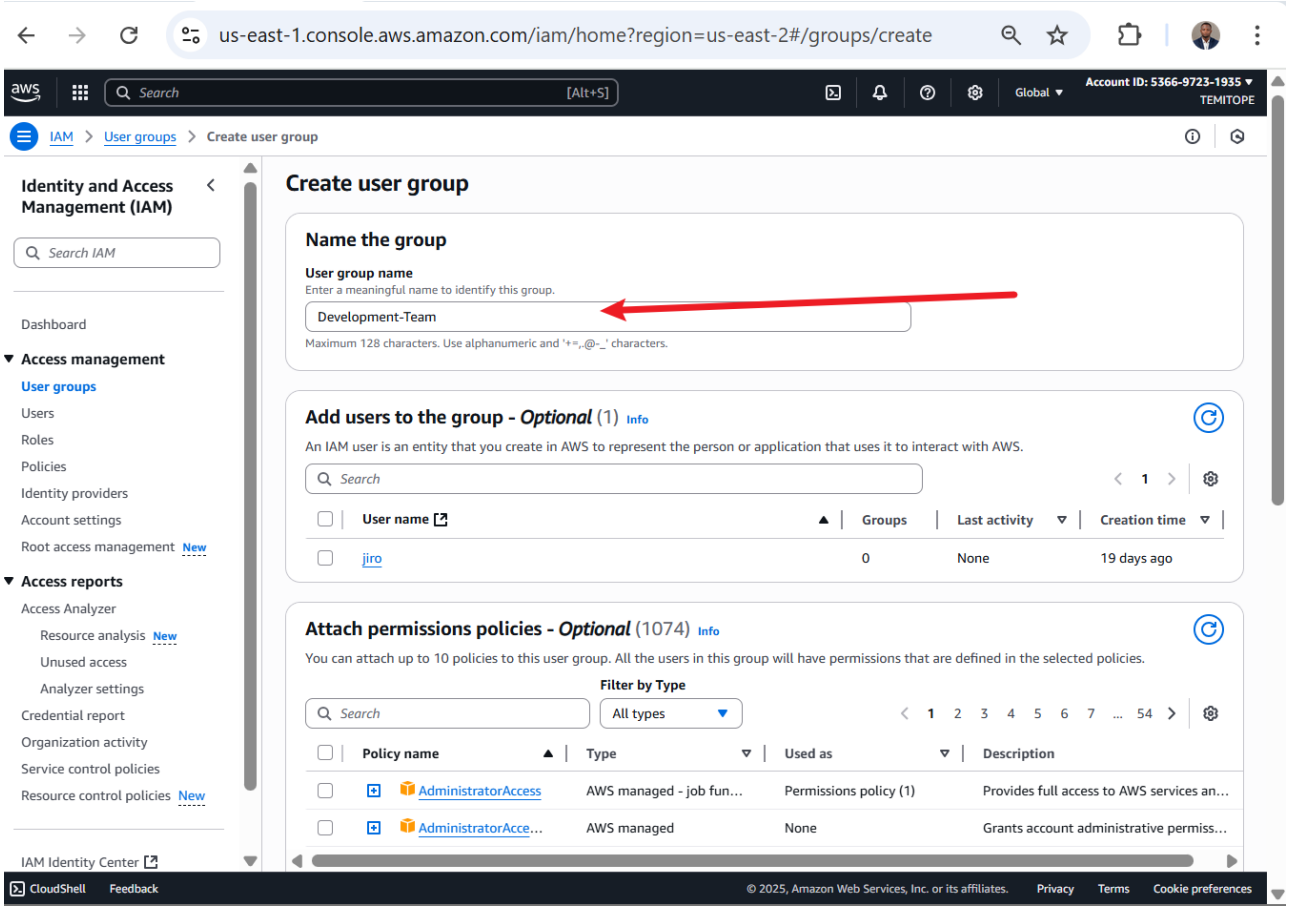
- Click on user group.



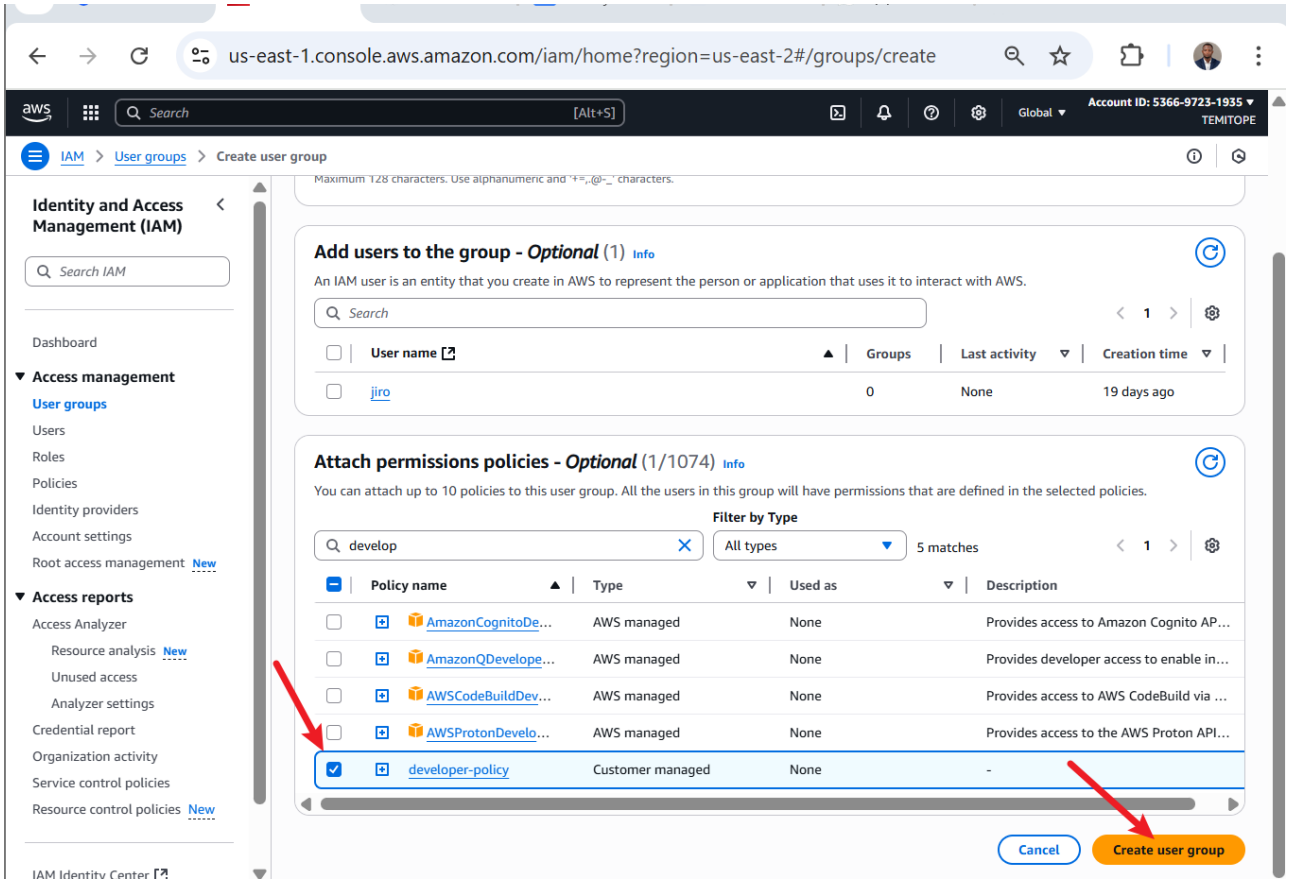
- Click on create group.



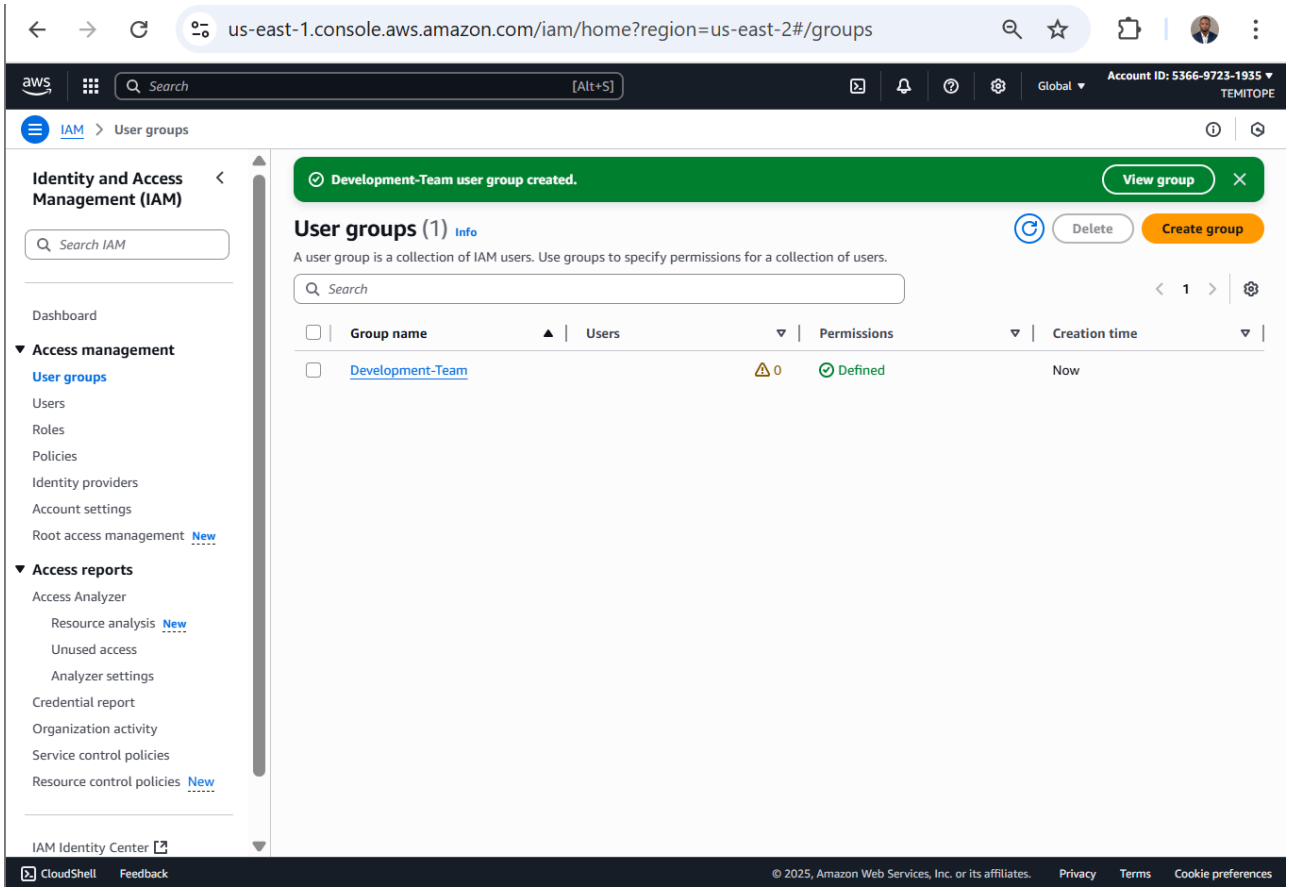
- Name the group based on their role.



- Attach developer policy and click create group.

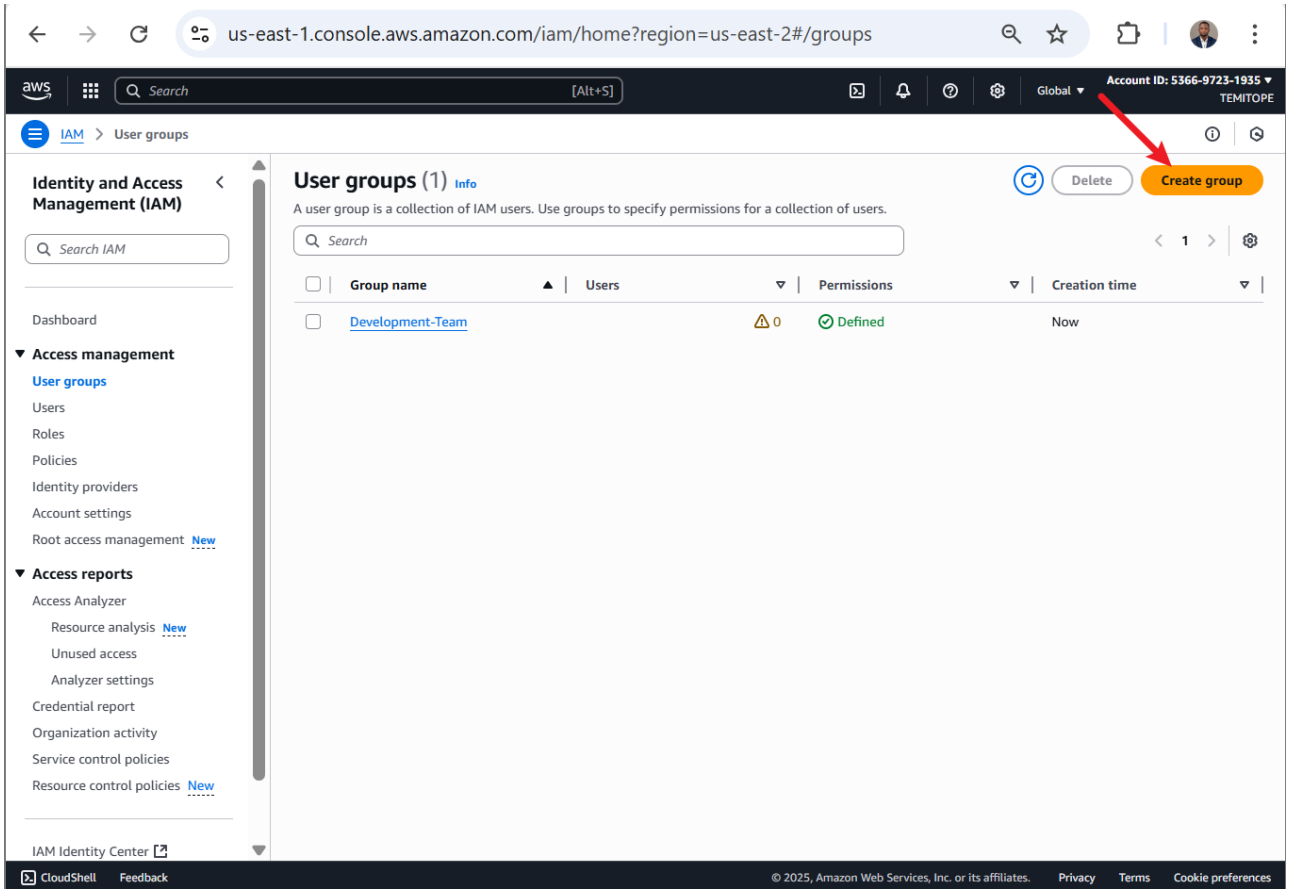


- Development user group successfully created.

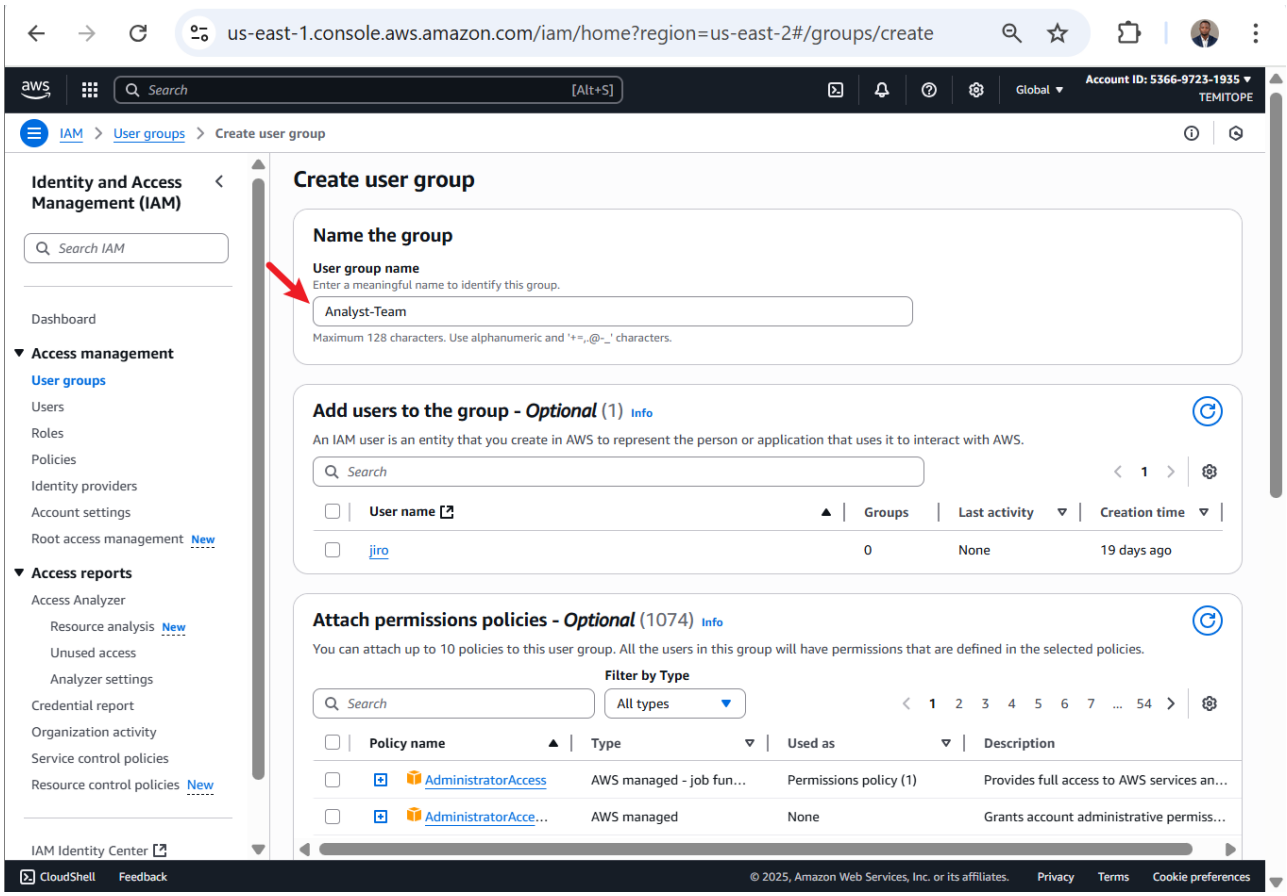


Steps for Analyst-Team:

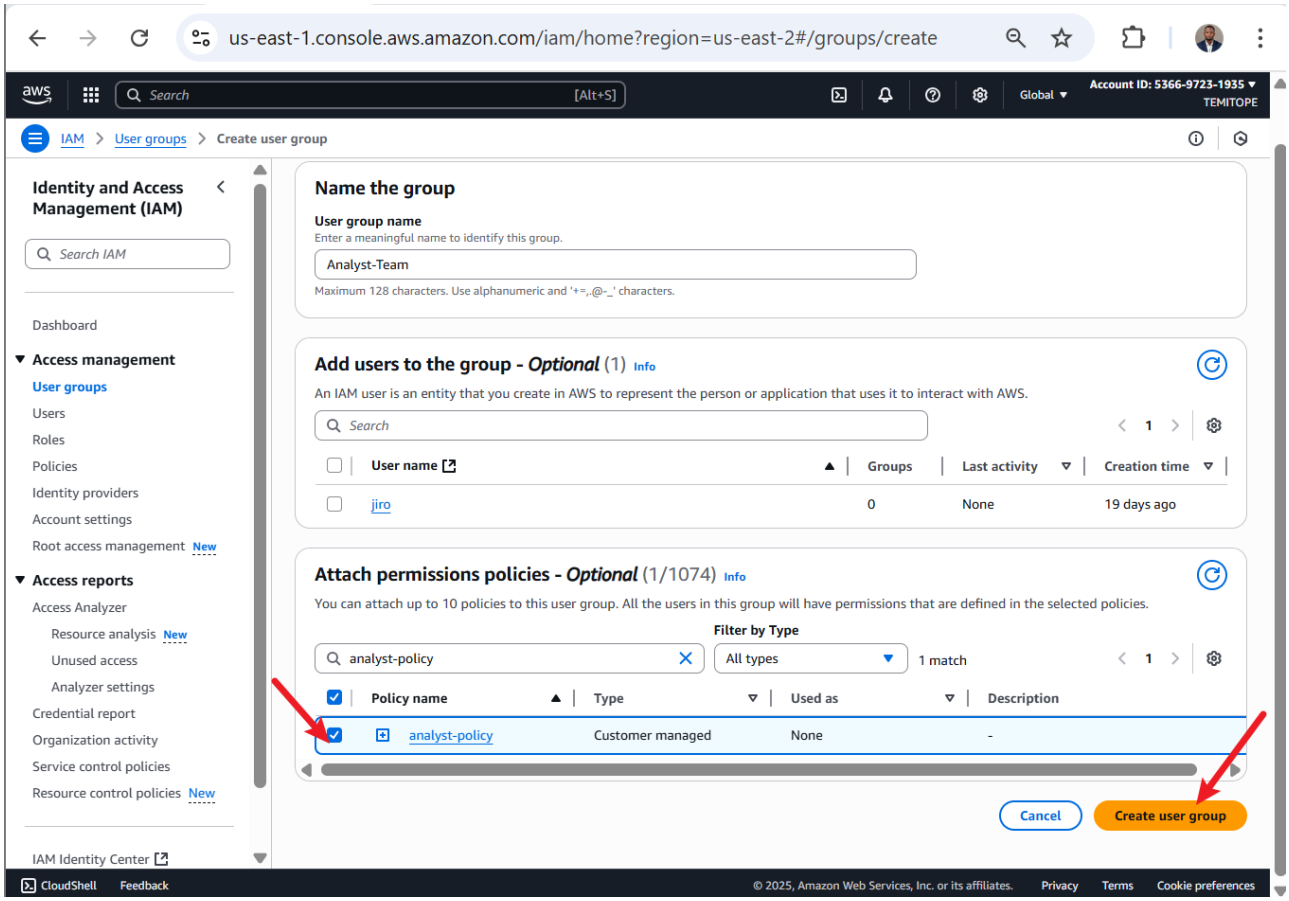
- Click on create group to create one more.



- Name the group based on their role.

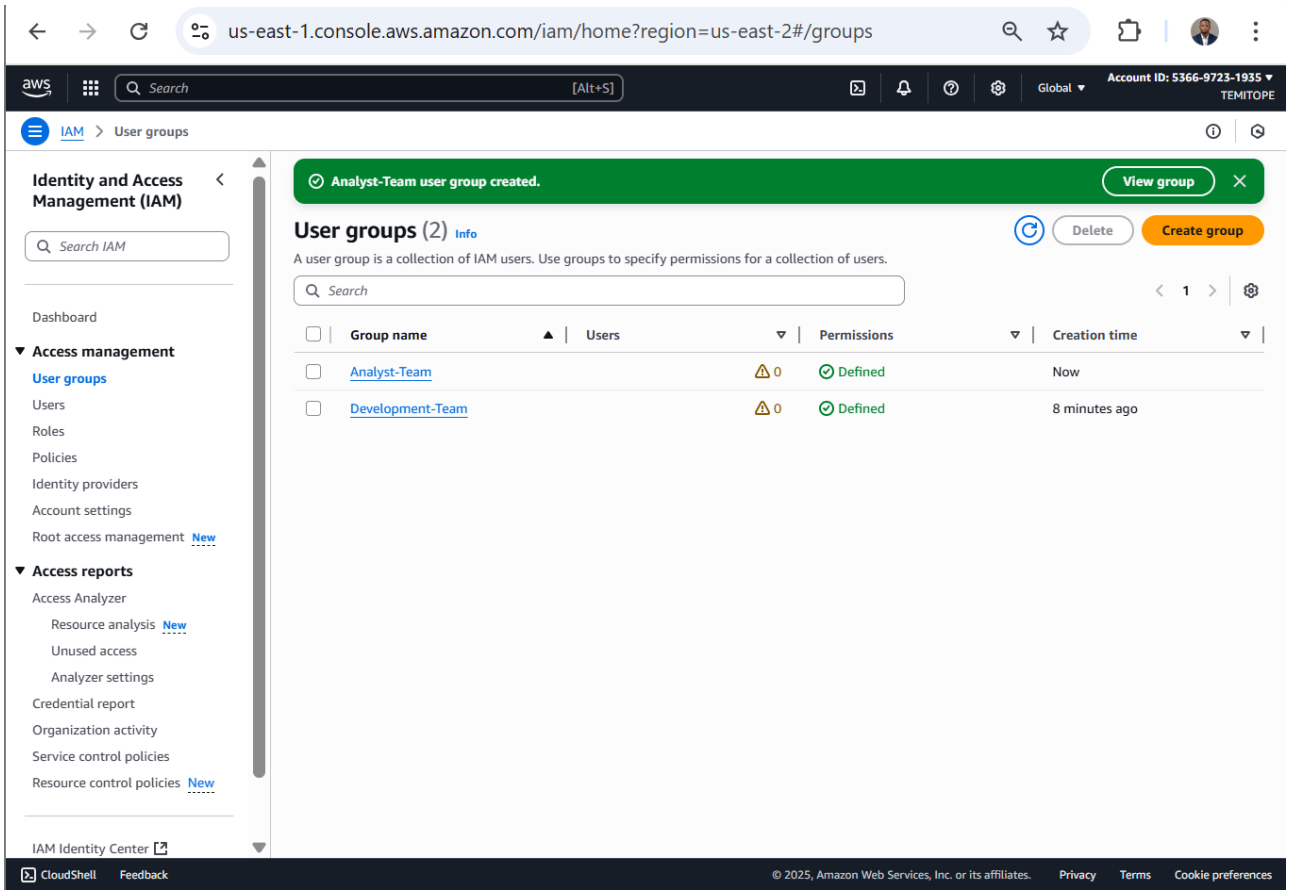


- Attach analyst policy and click create group.



(Note: Filename references developer policy, but context indicates analyst policy attachment.)

- Analyst user group successfully created.



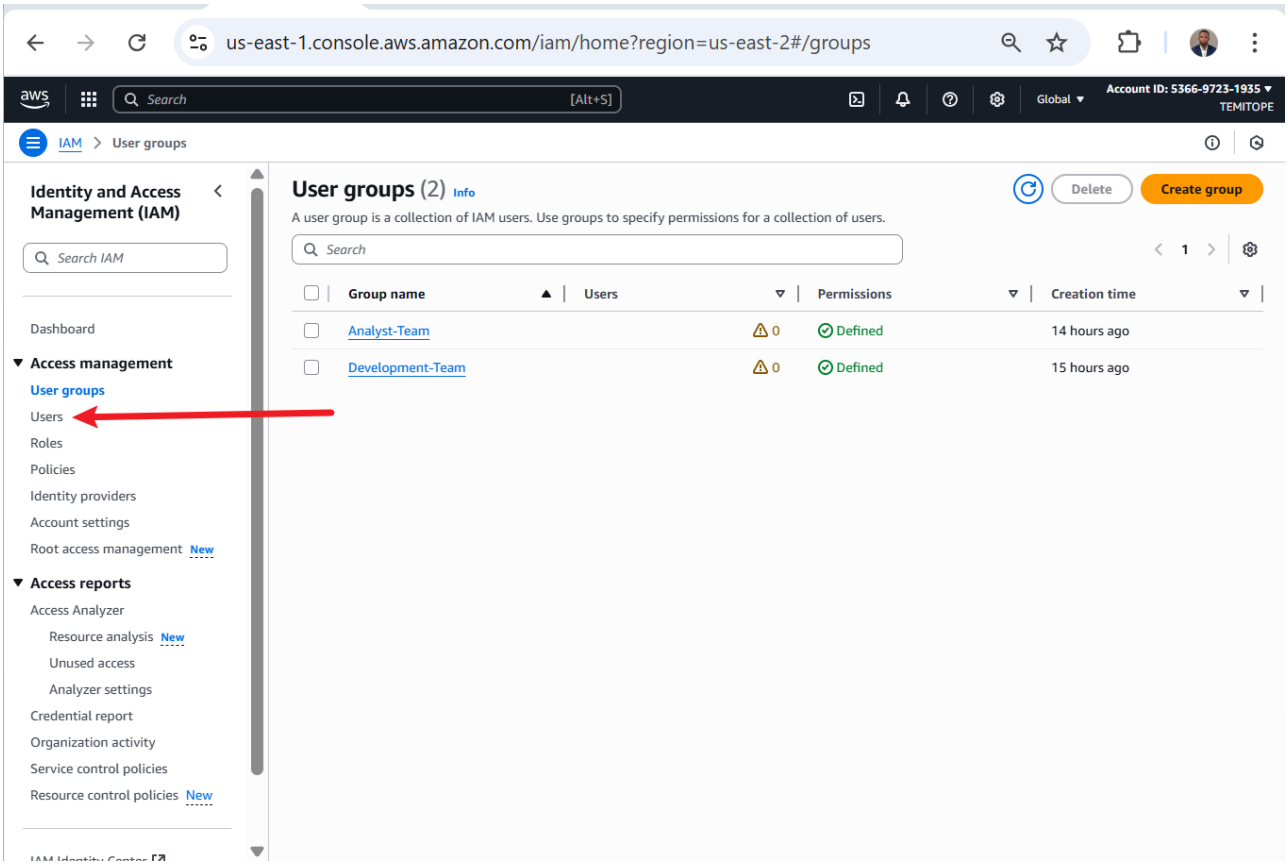
6. Creating IAM Users

John (Backend Developer)

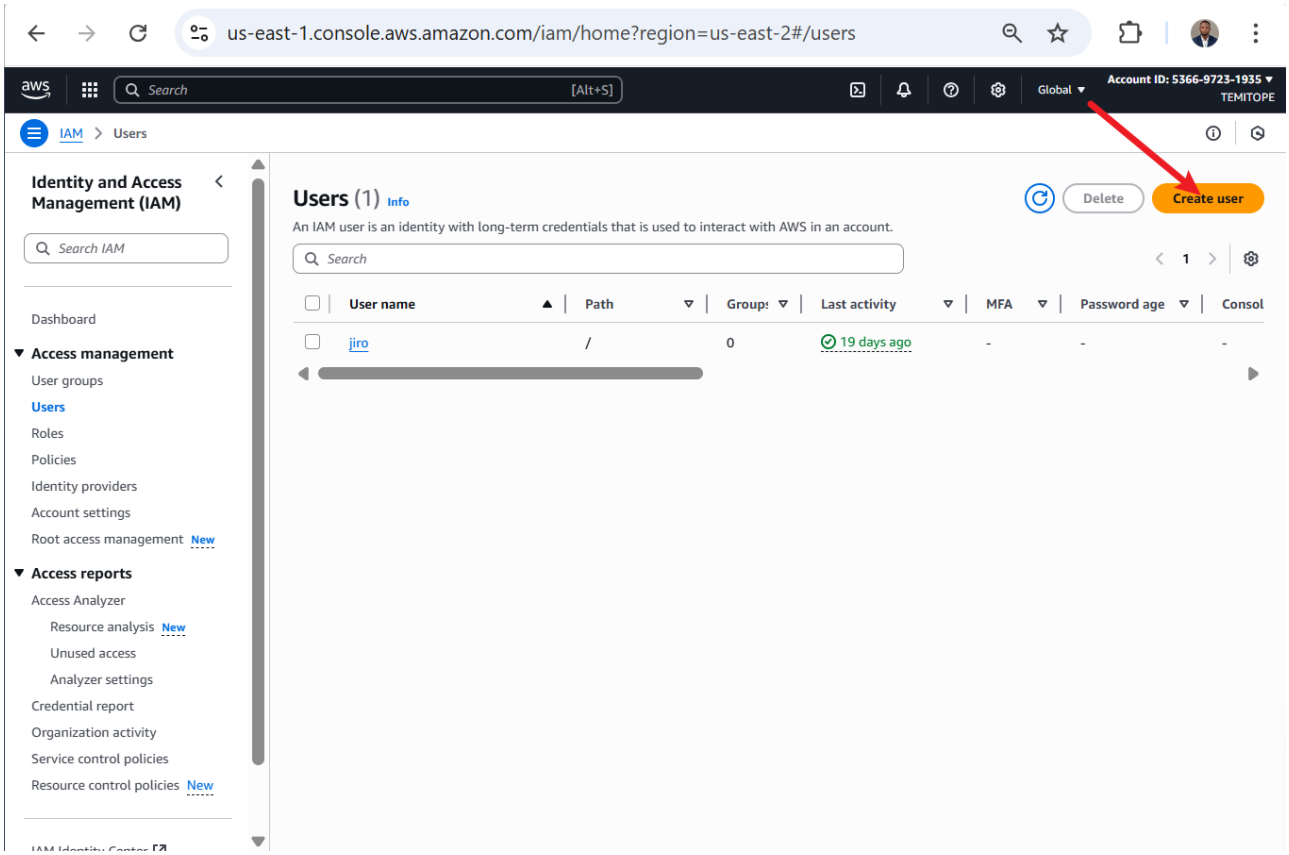
- User created: **John**.
- Added to **Development-Team** group.
- Console login enabled.
- Credentials downloaded.

Steps:

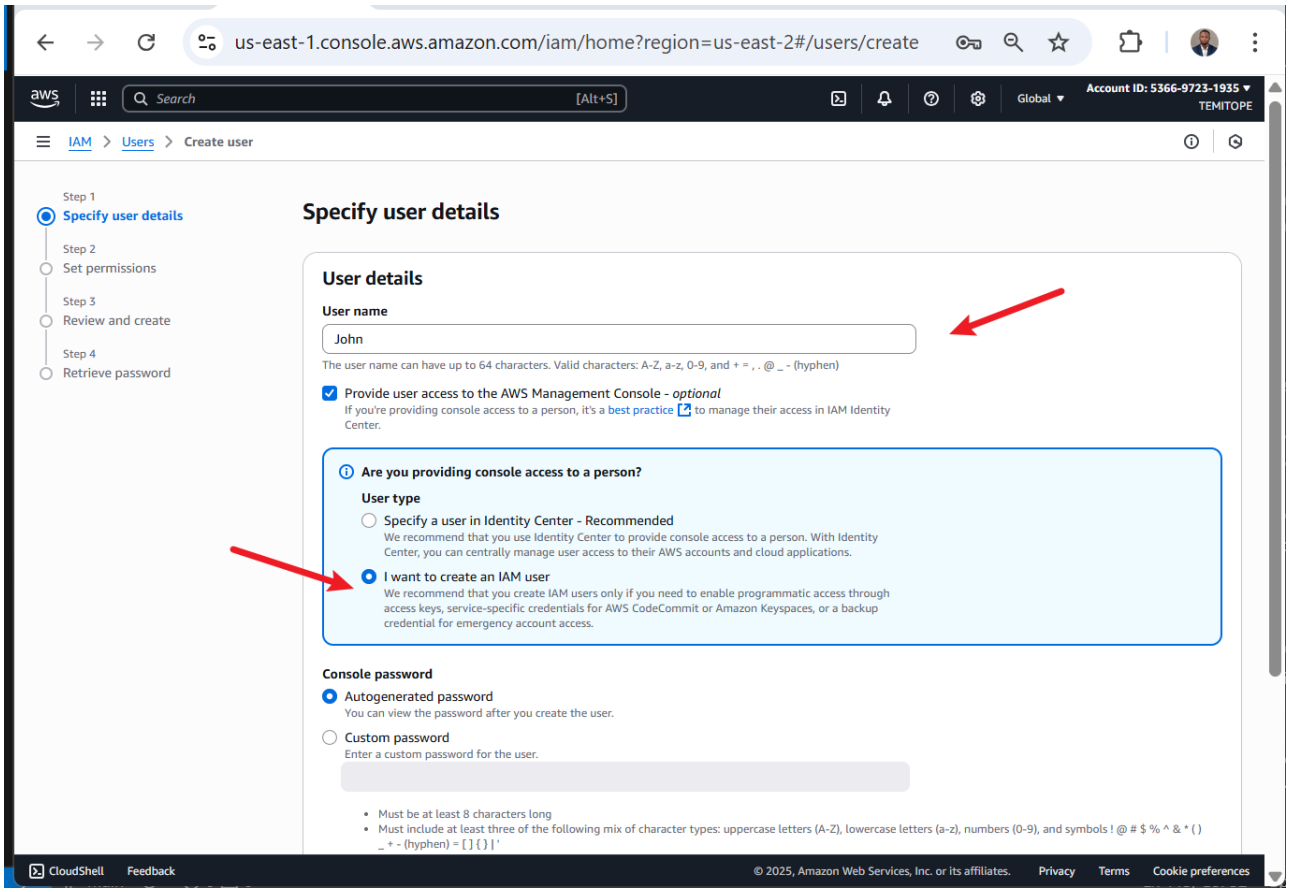
- Click users to create users.



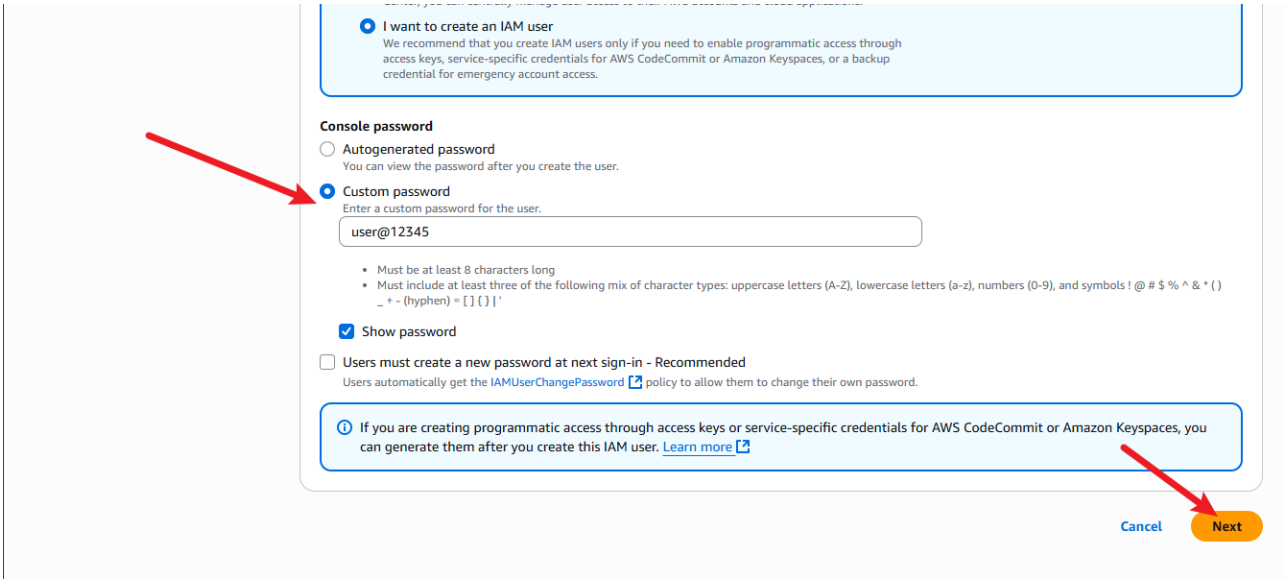
- Click create user.



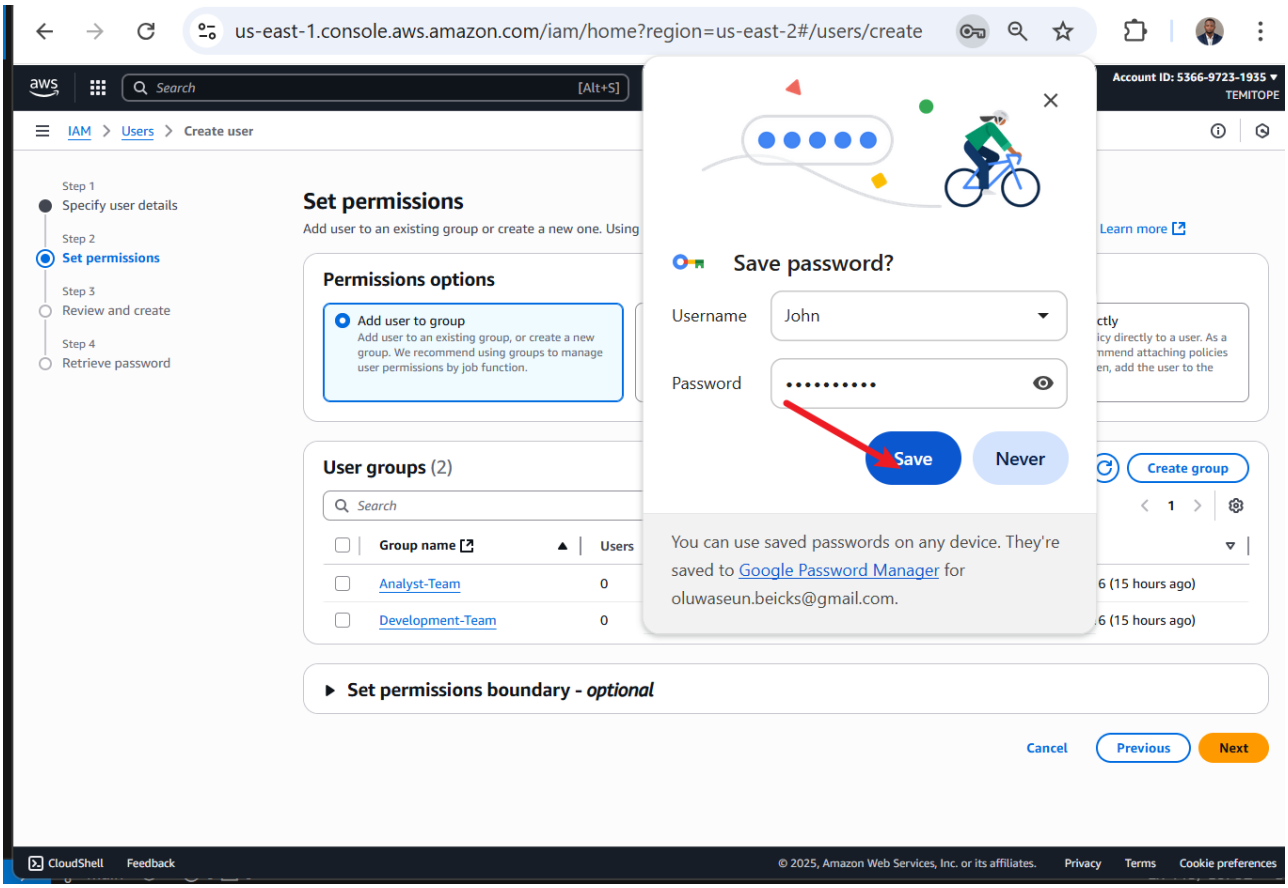
- Enter name, provide access to console, and check IAM user.



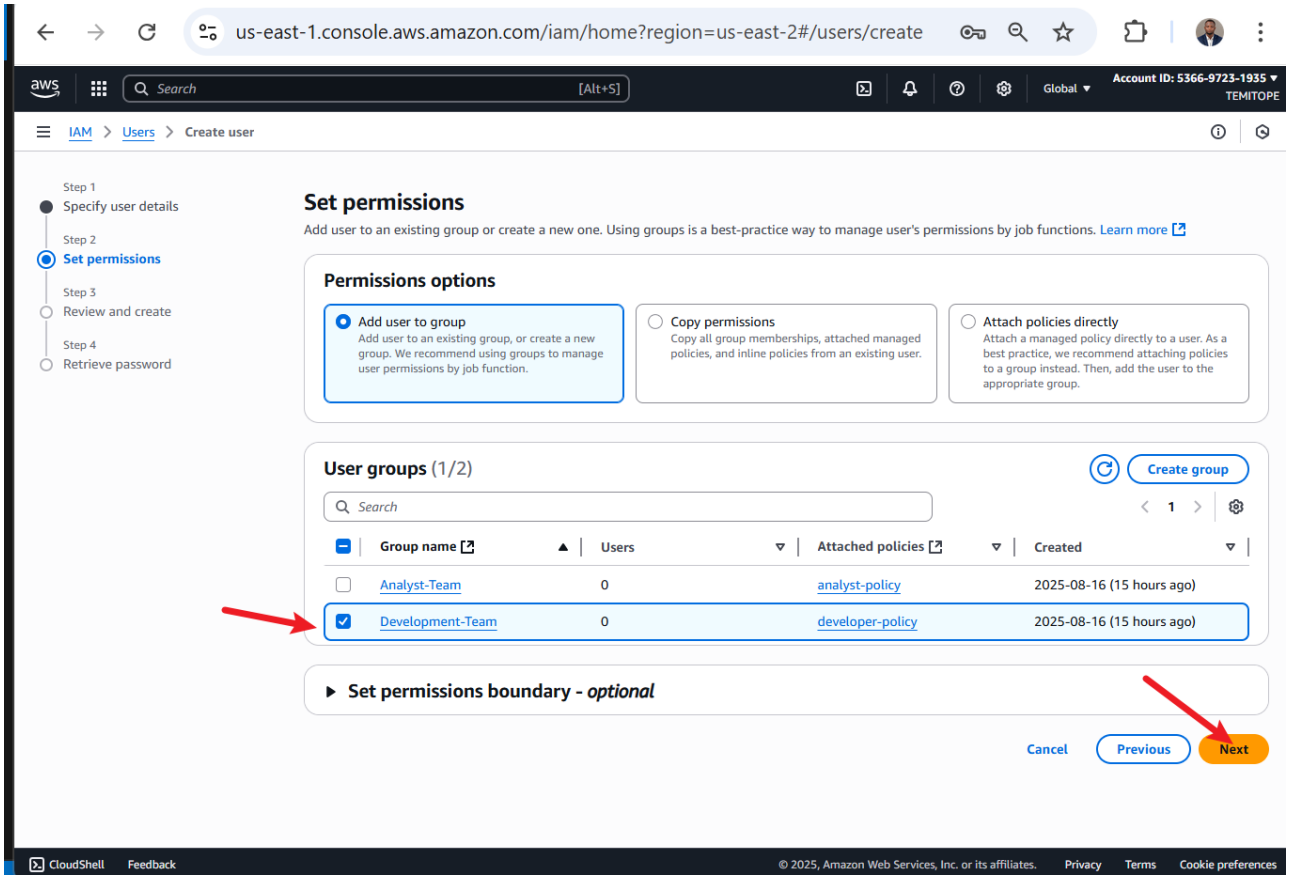
- Choose password, uncheck password change on next login.



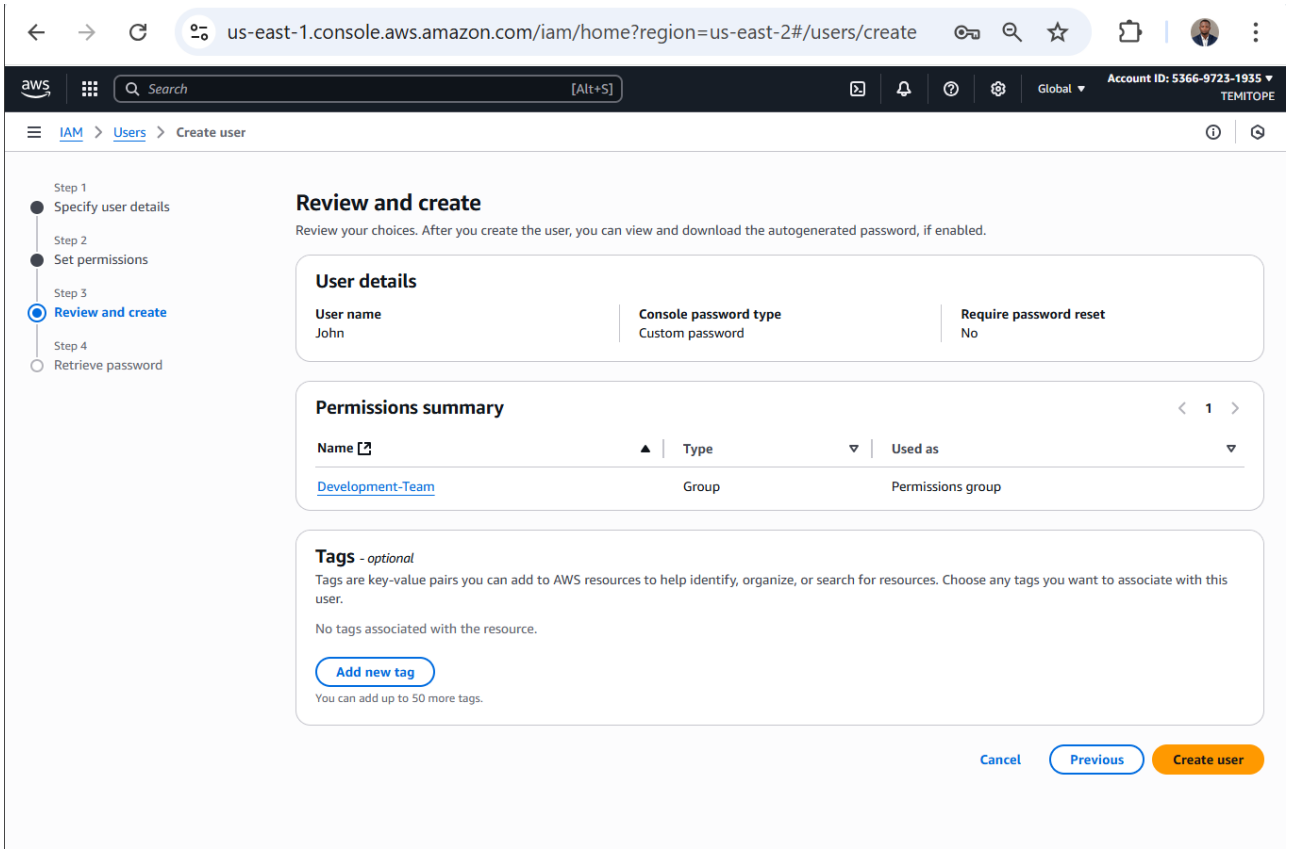
- Save password for next login.



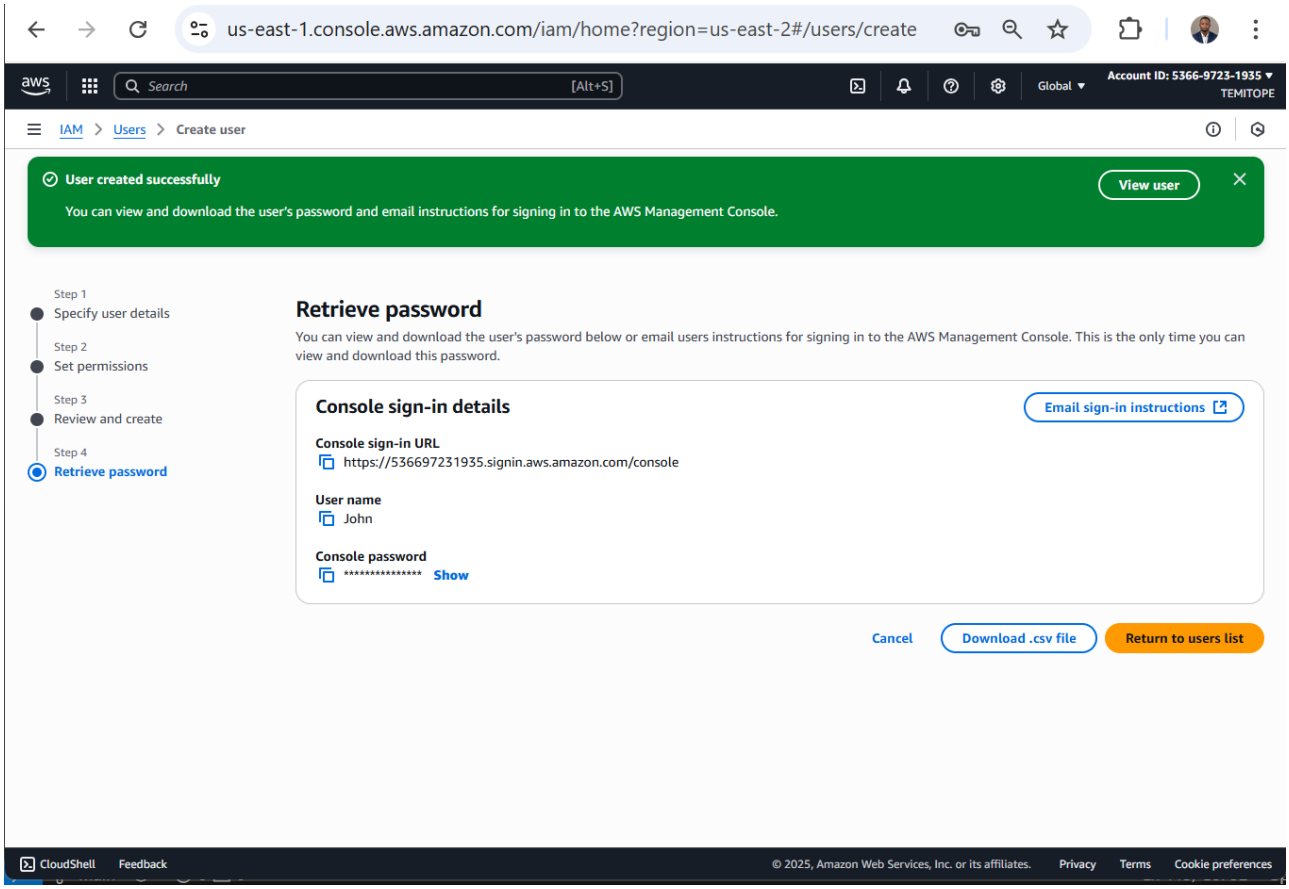
- Check to add to development group and click next.



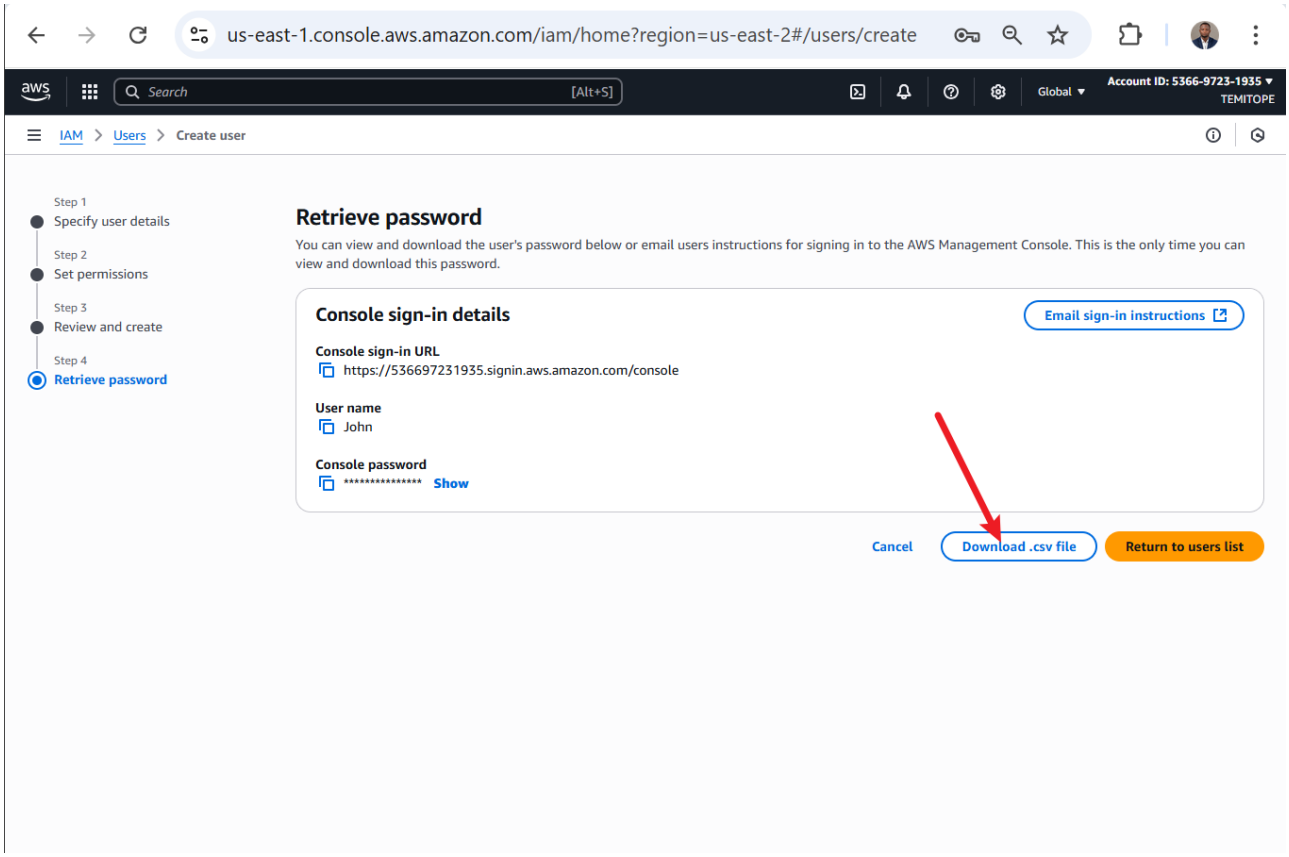
- Review and create user.



- Successfully create user.



- Download user login CSV.



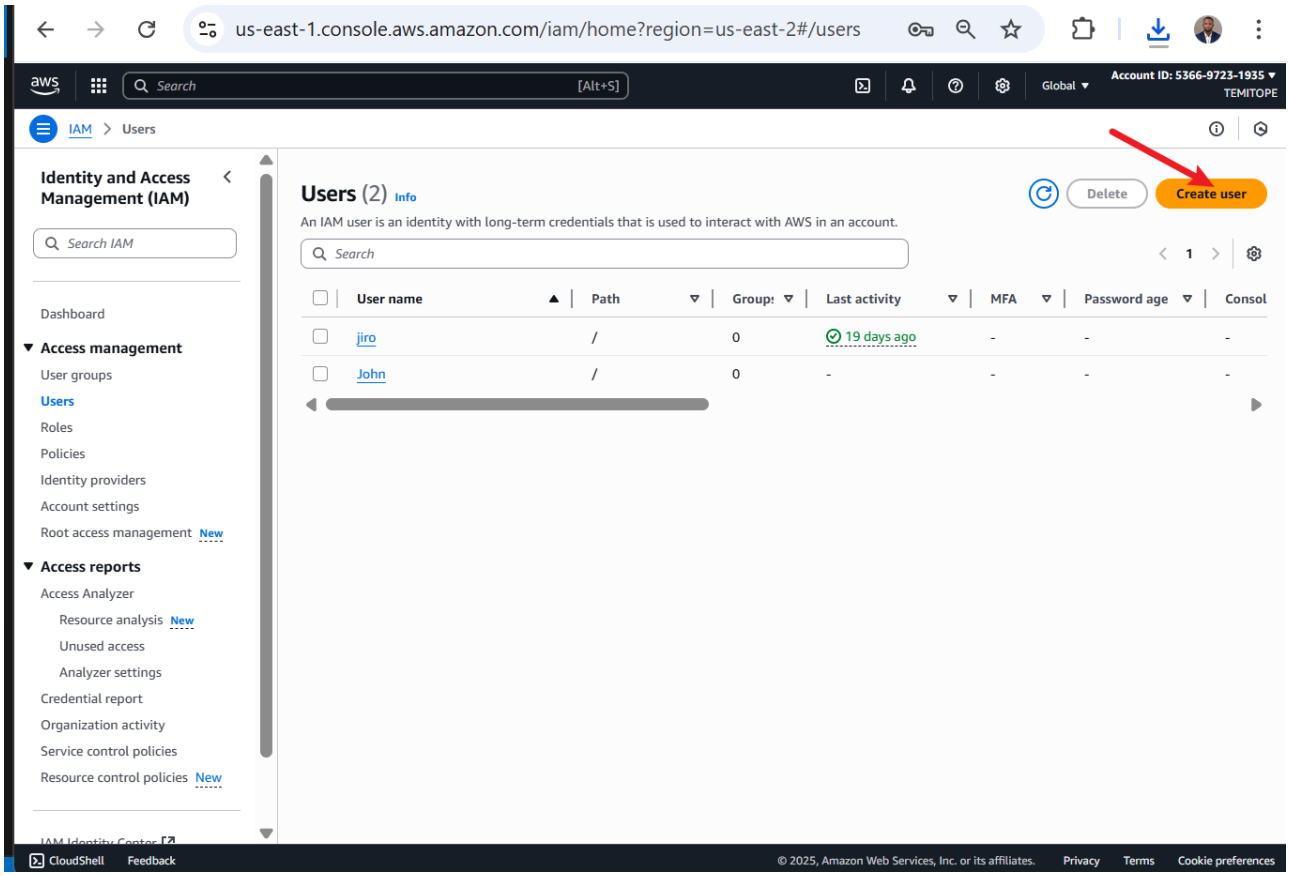
Mary (Data Analyst)

- User created: **Mary**.
- Added to **Analyst-Team** group.

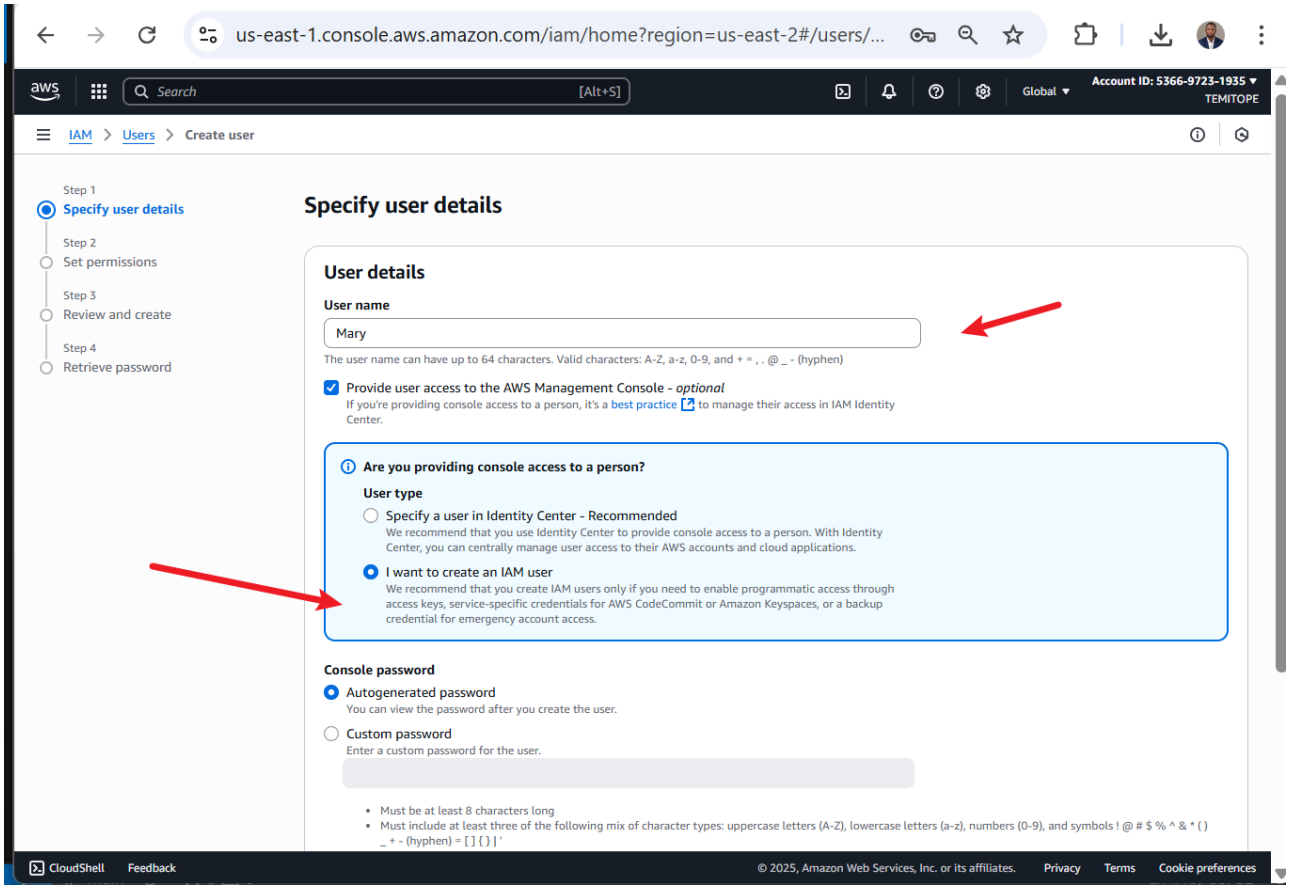
- Console login enabled.
- Credentials downloaded.

Steps:

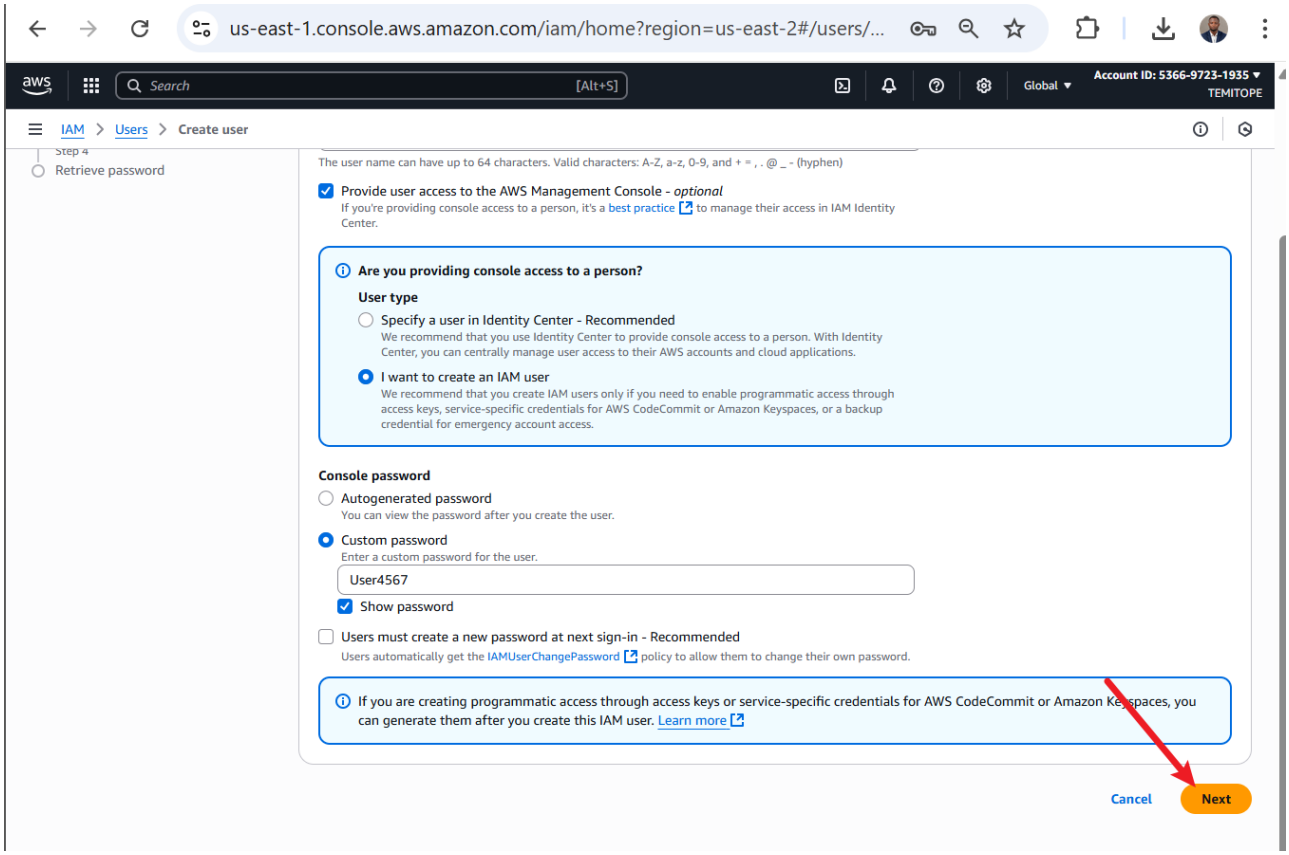
- Click create new user to create new user Mary.



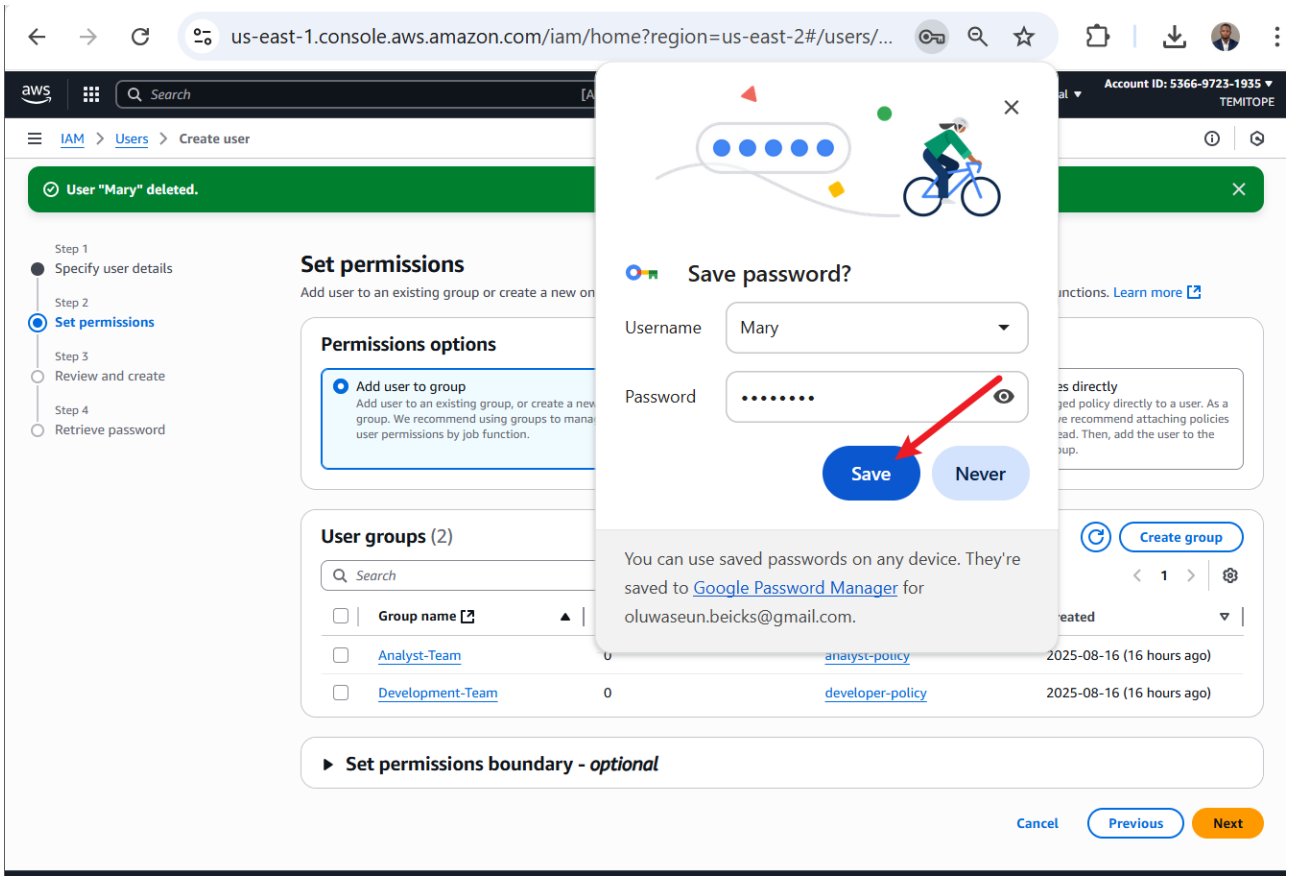
- Enter name, provide access to console, and check IAM user.



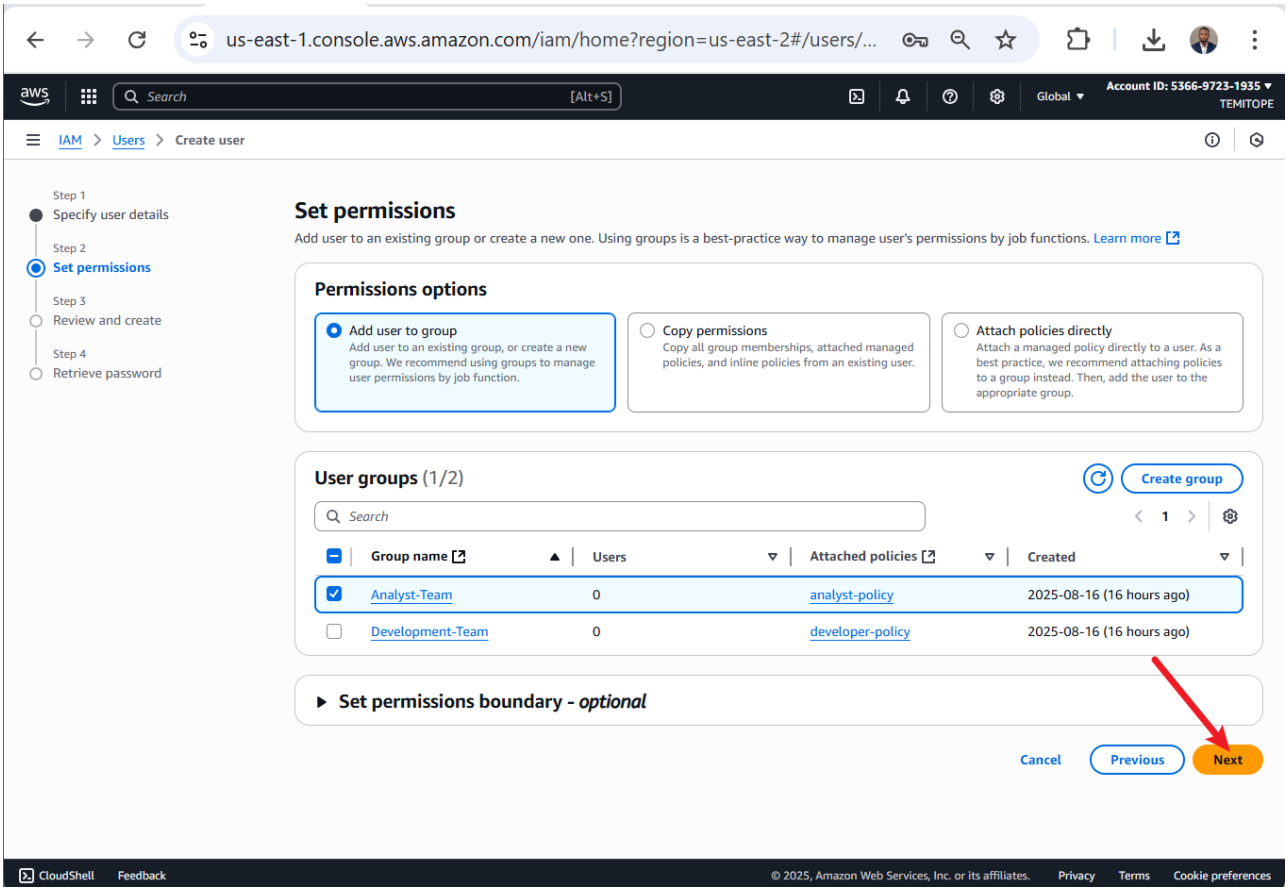
- Choose password, uncheck password change on next login.



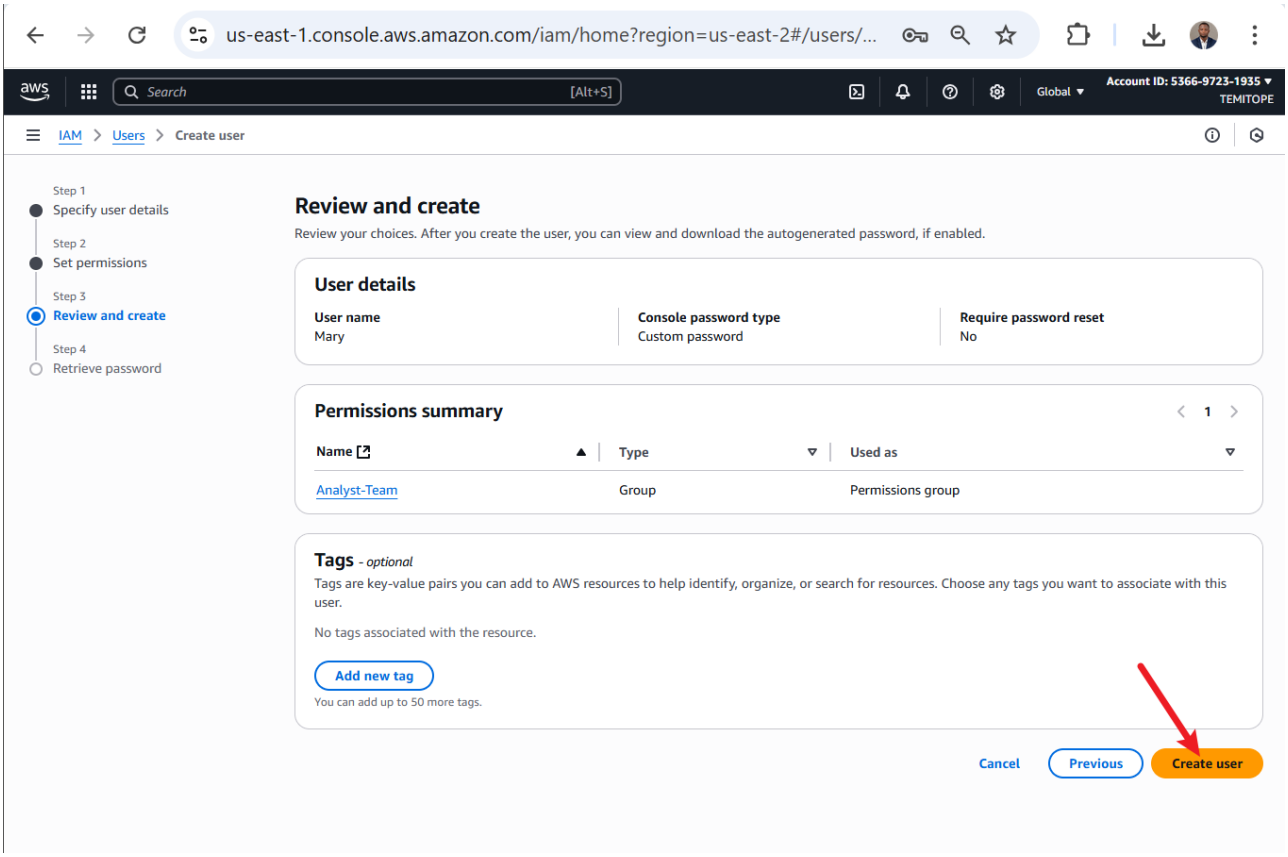
- Save password for next login.



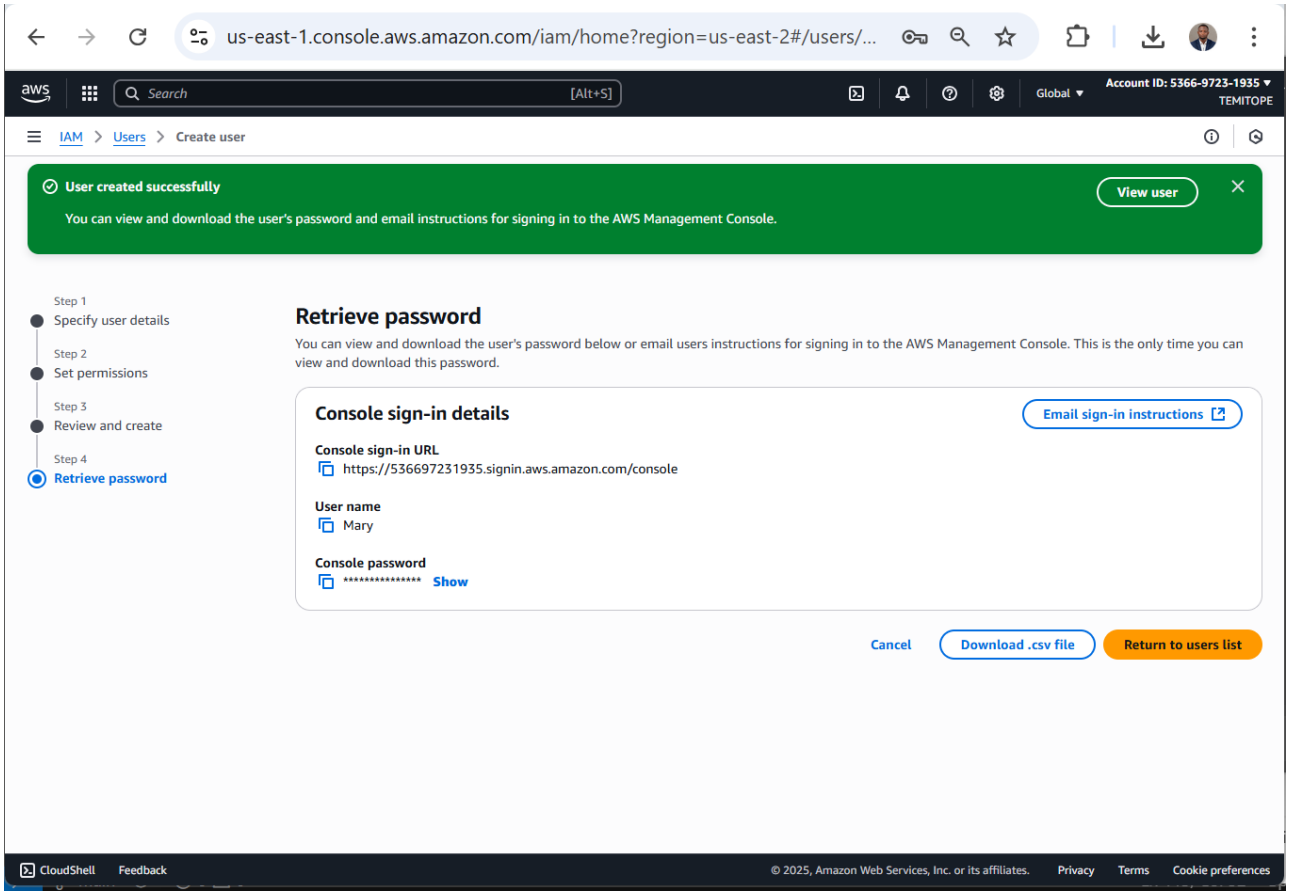
- Check to add to analyst group and click next.



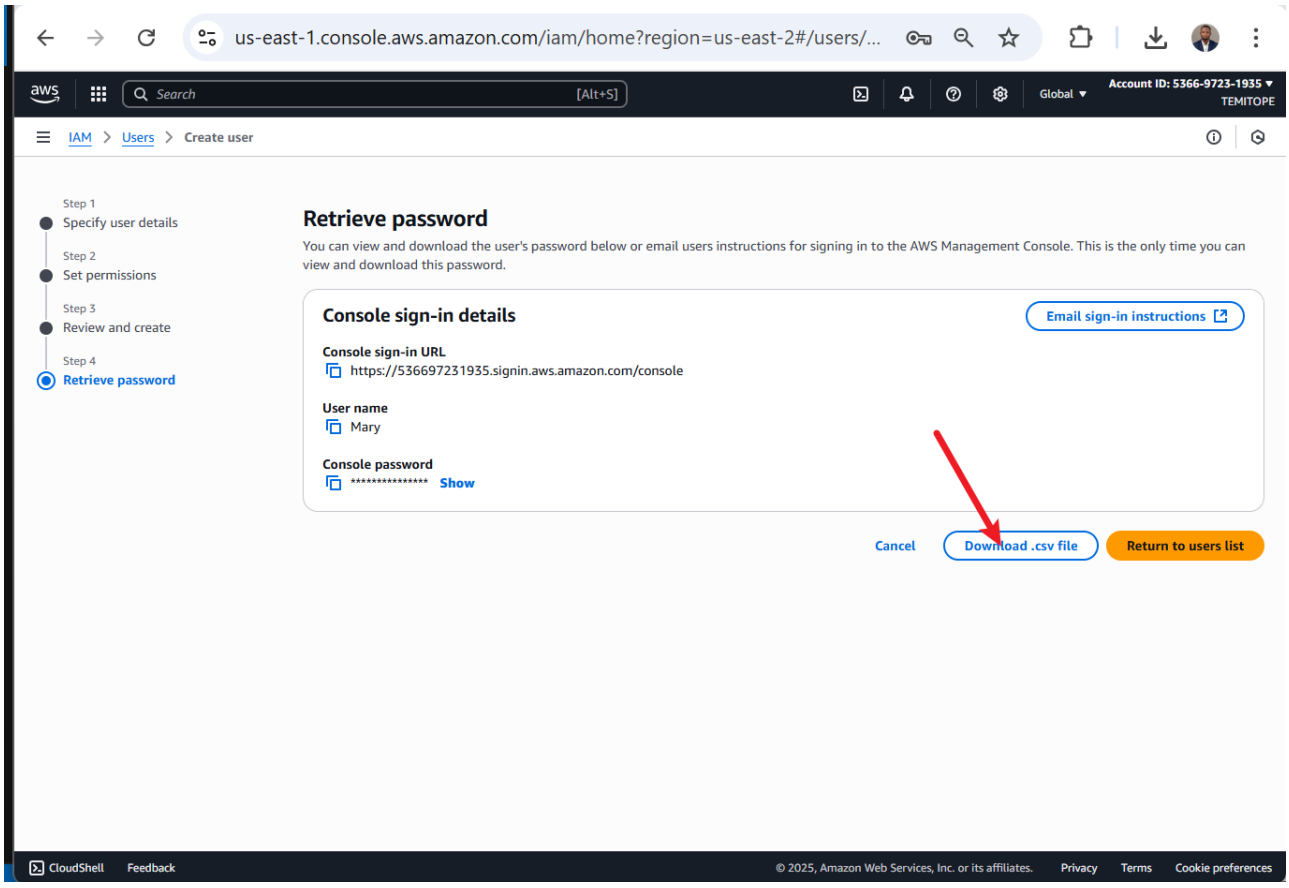
- Review and create user.



- Successfully create user.

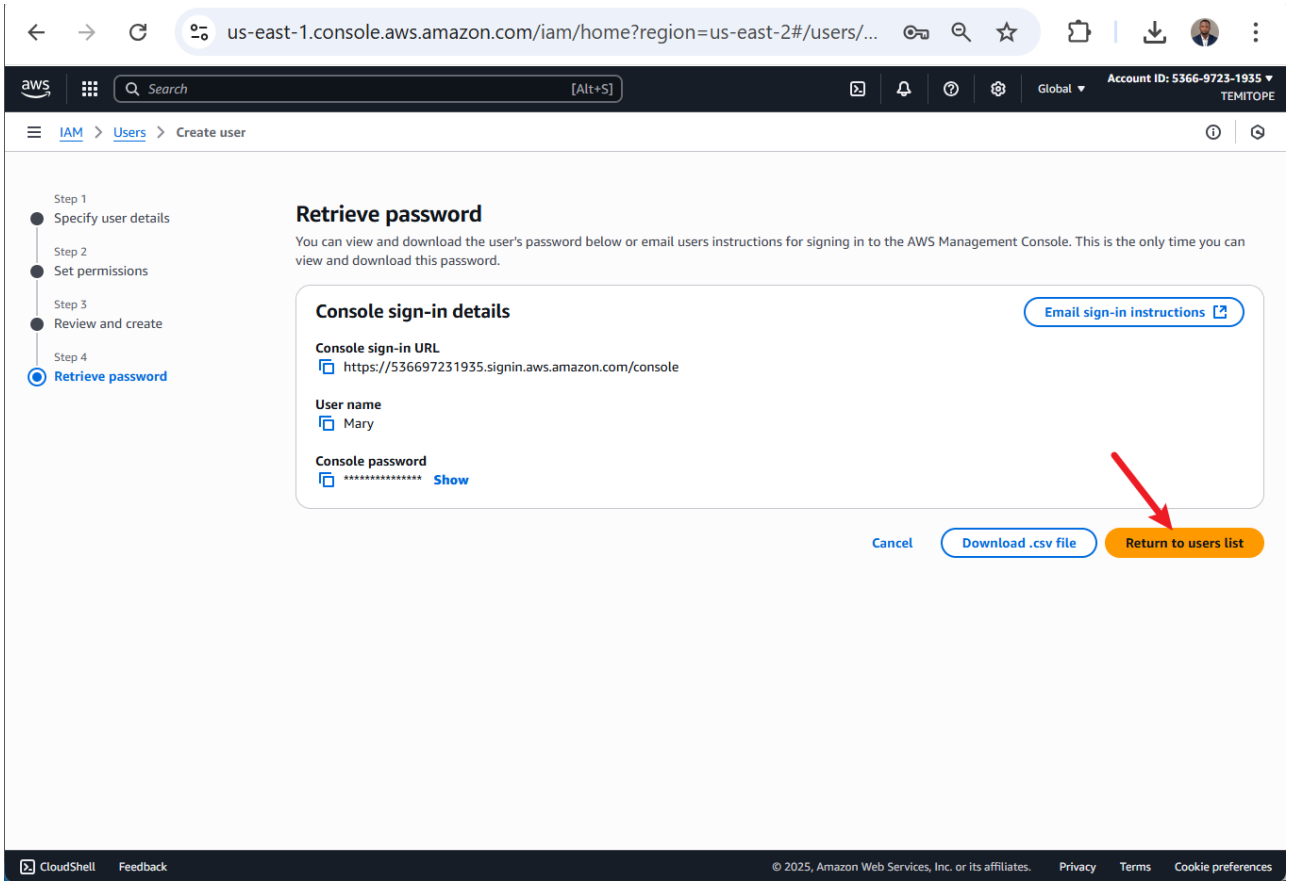


- Download user login CSV.

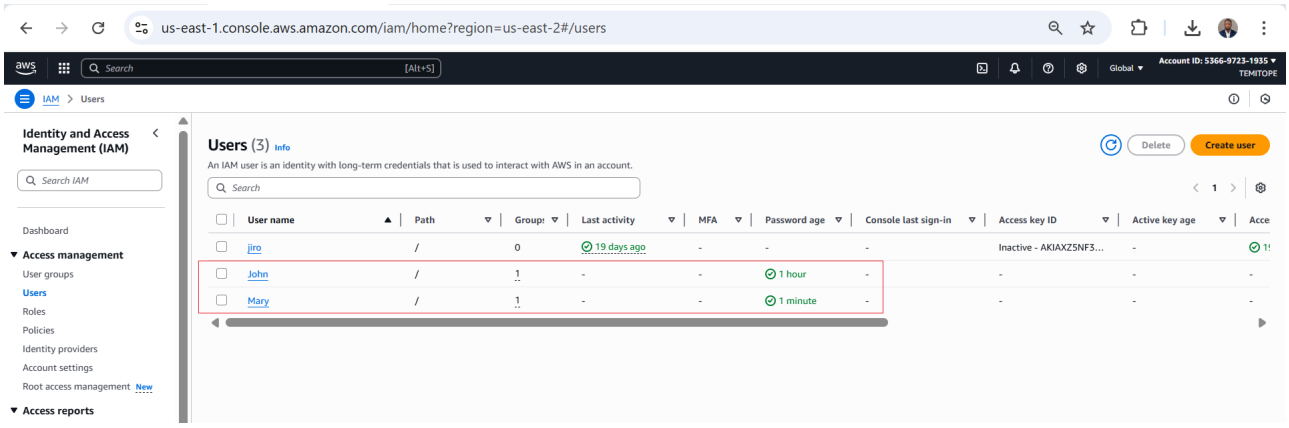


7. Testing and Validation

- Return user list.



- Landed on user list page.

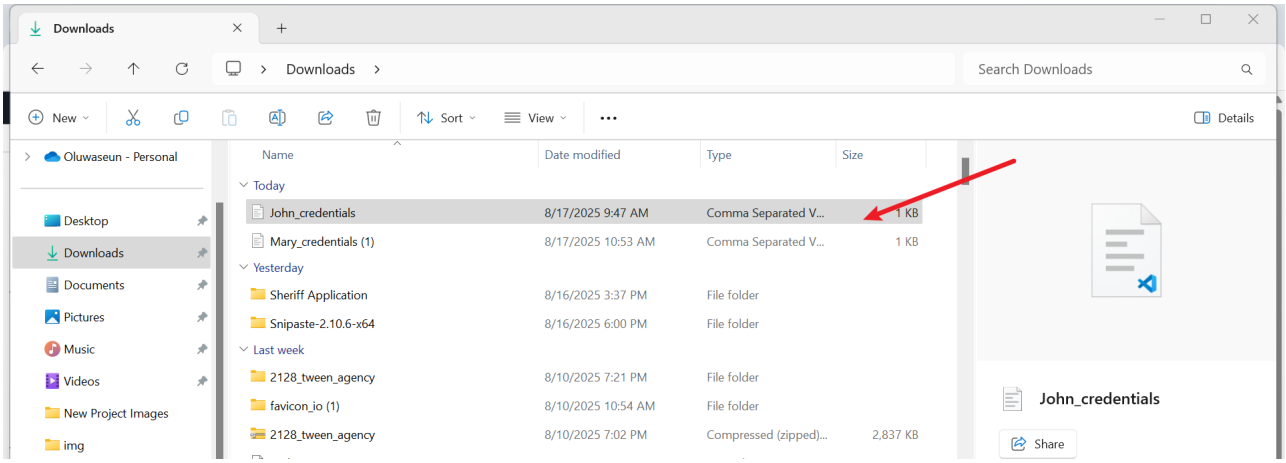


John's Access

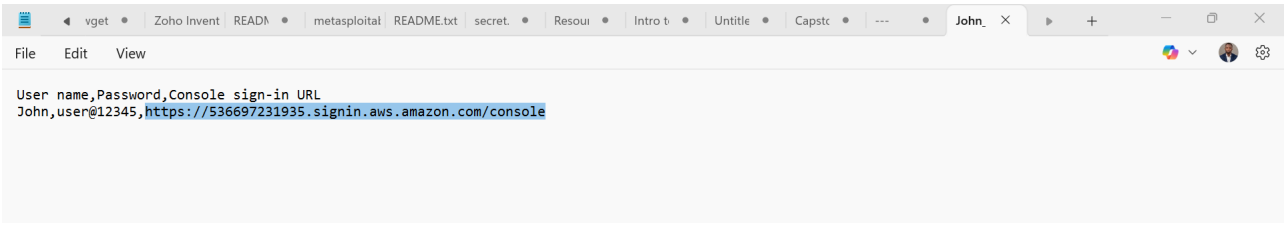
- Logged in as John.
- Accessed **EC2 Dashboard** ☑.
- Attempted to access **S3 Dashboard** ❌ (denied).

Steps:

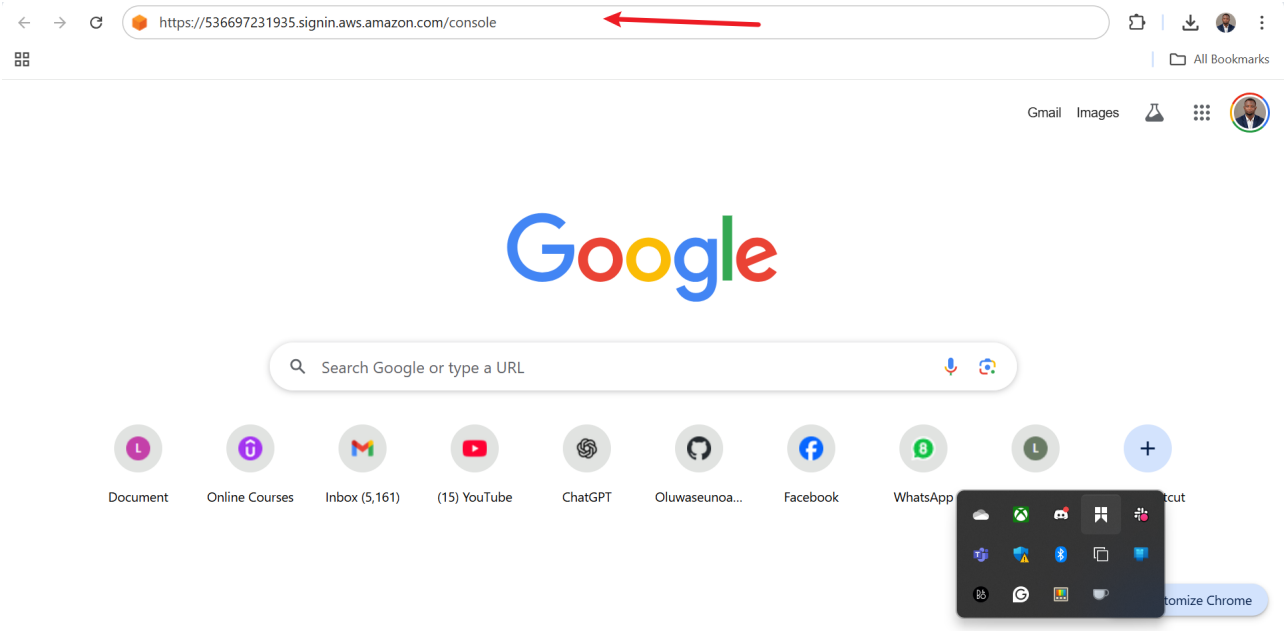
- Locate and open John downloaded CSV file.



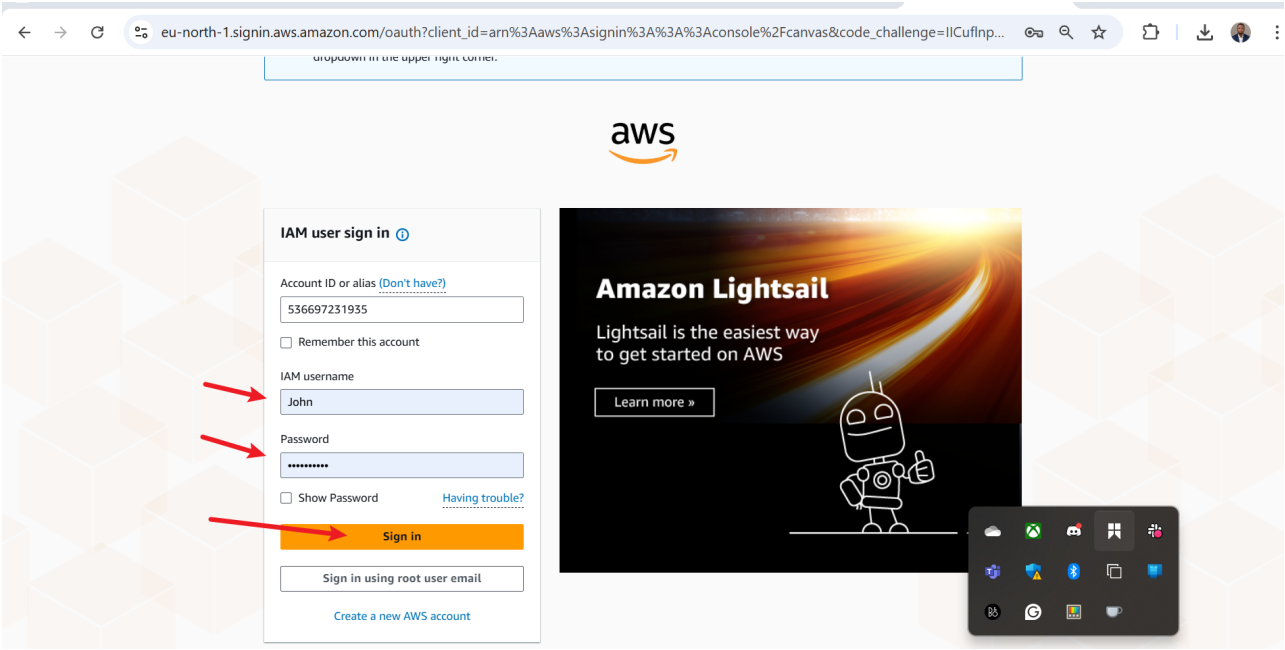
- Copy the HTTPS link.



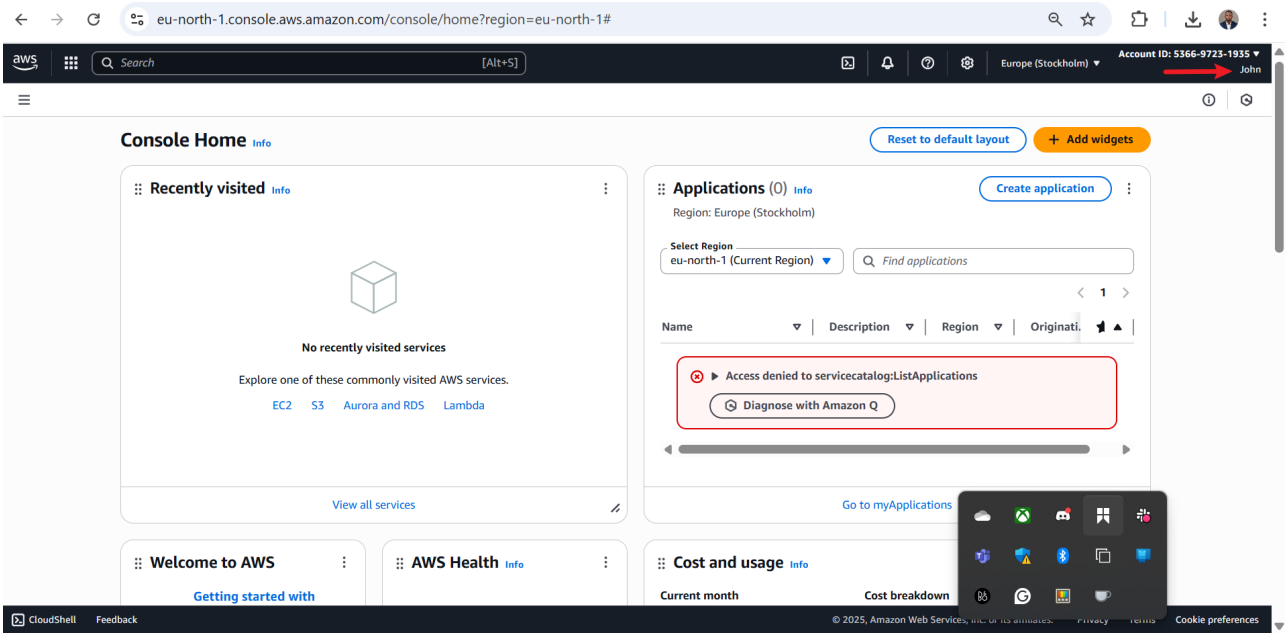
- Paste it on the browser.



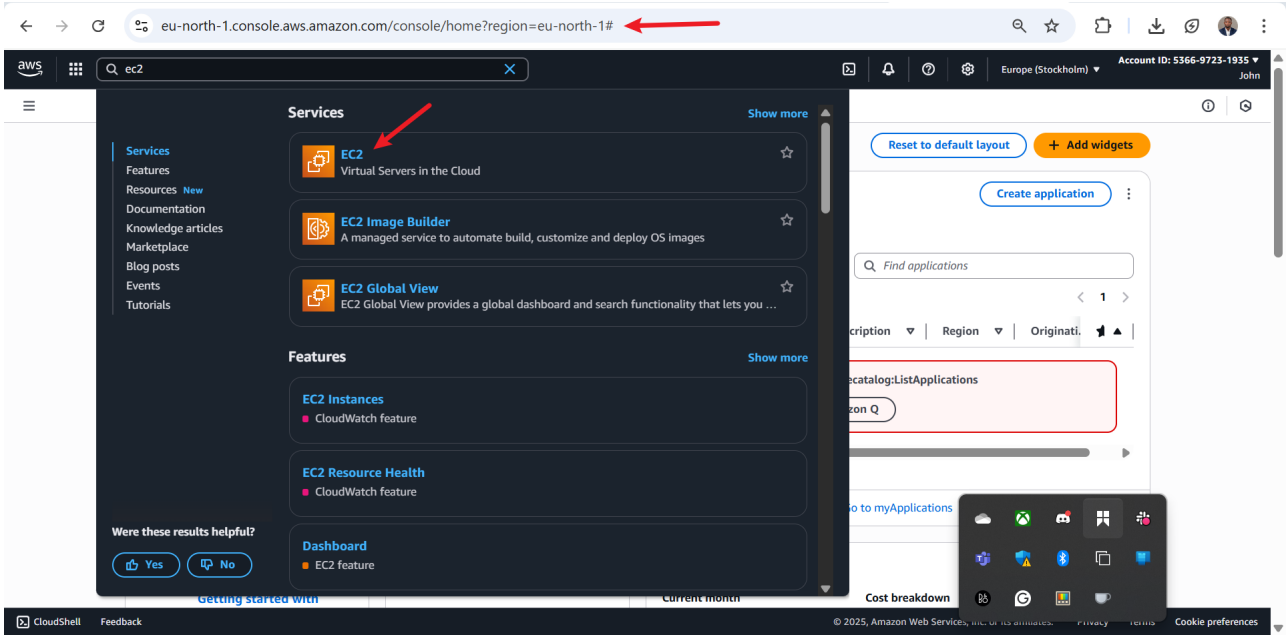
- Sign in as IAM user using your login and account ID which is fetched by the link.



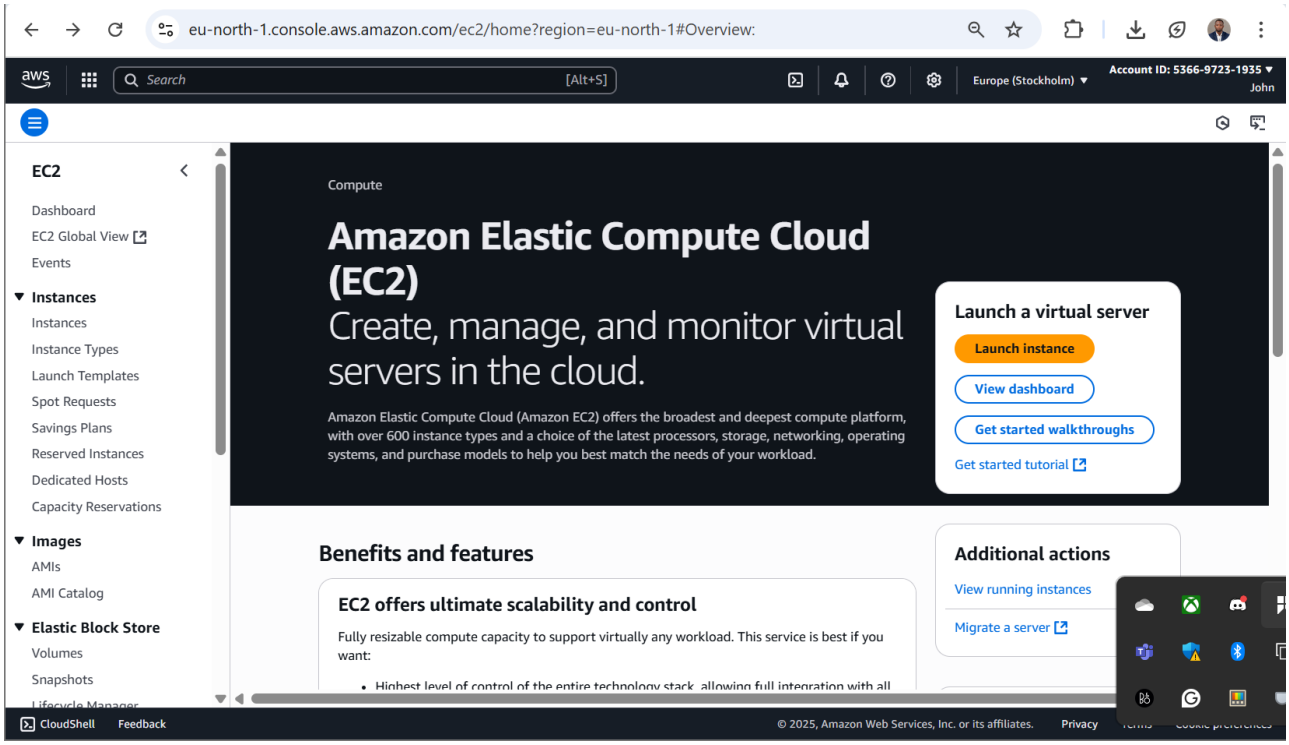
- Logged in as John.



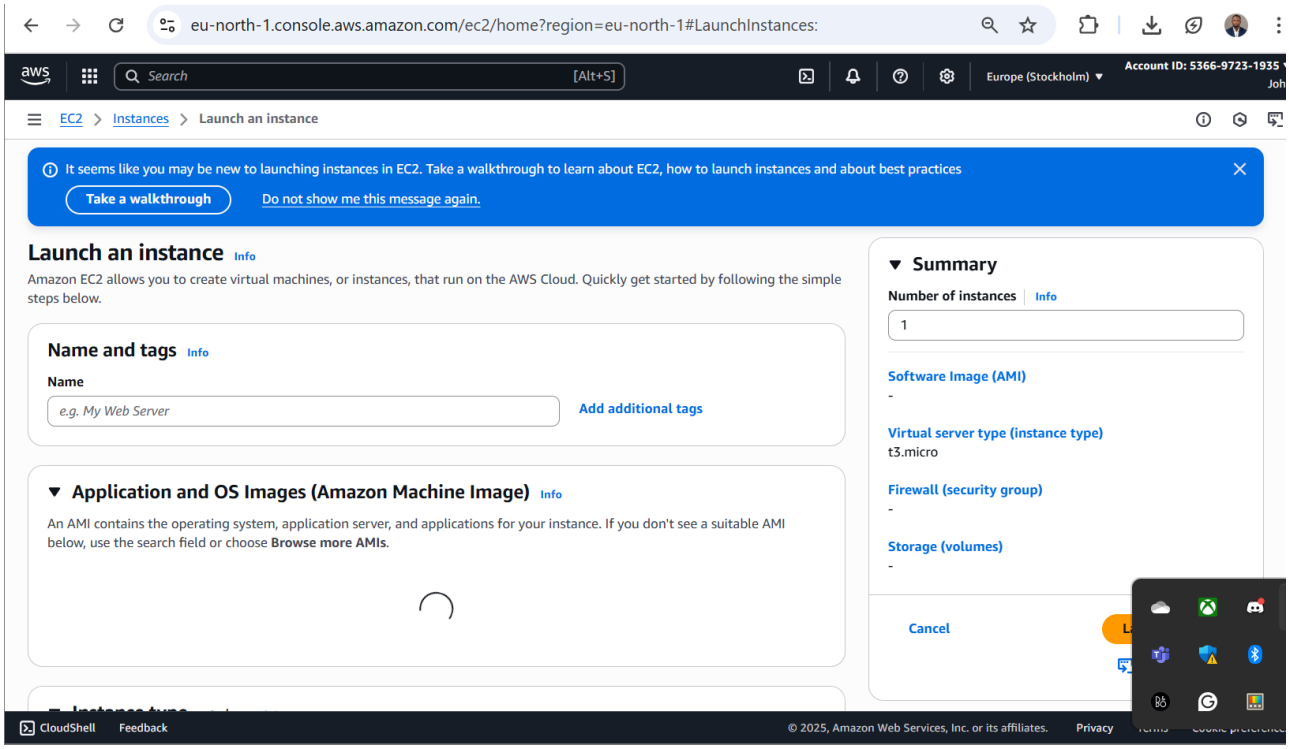
- Search for EC2 and click on it.



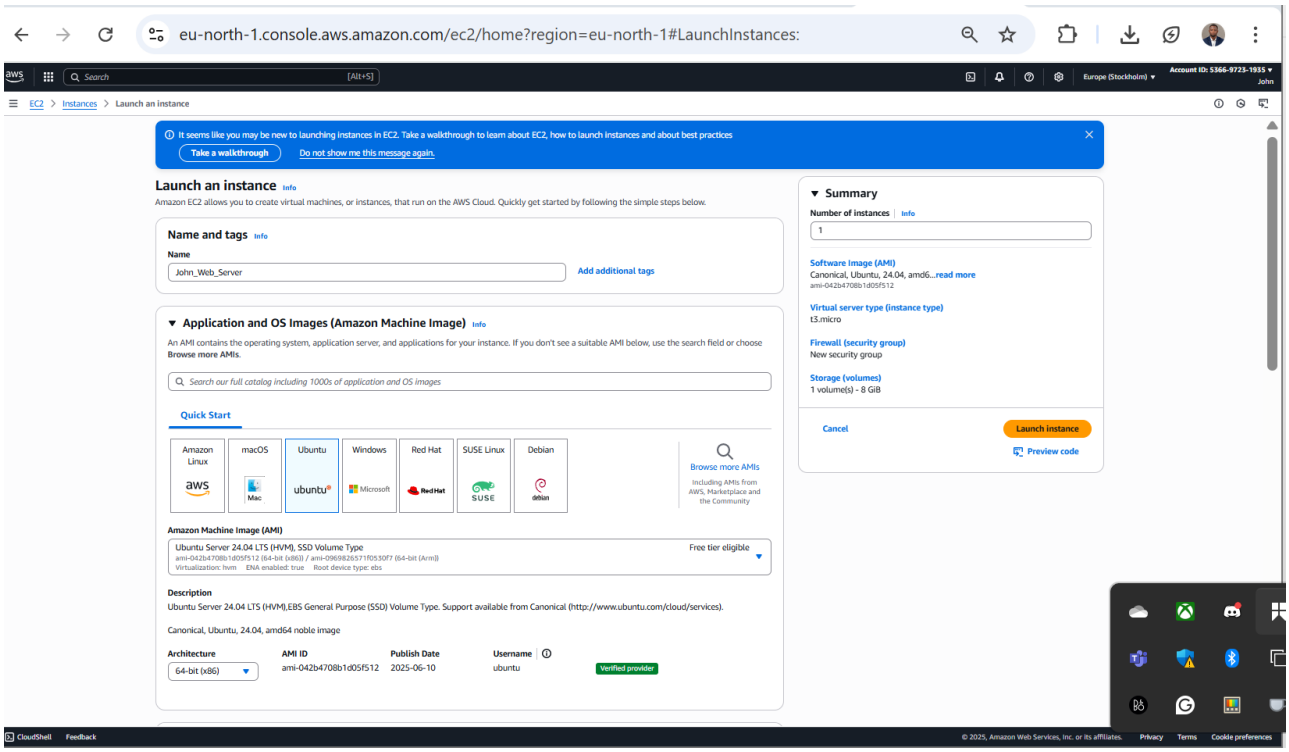
- Successfully access EC2 dashboard, click launch instance.



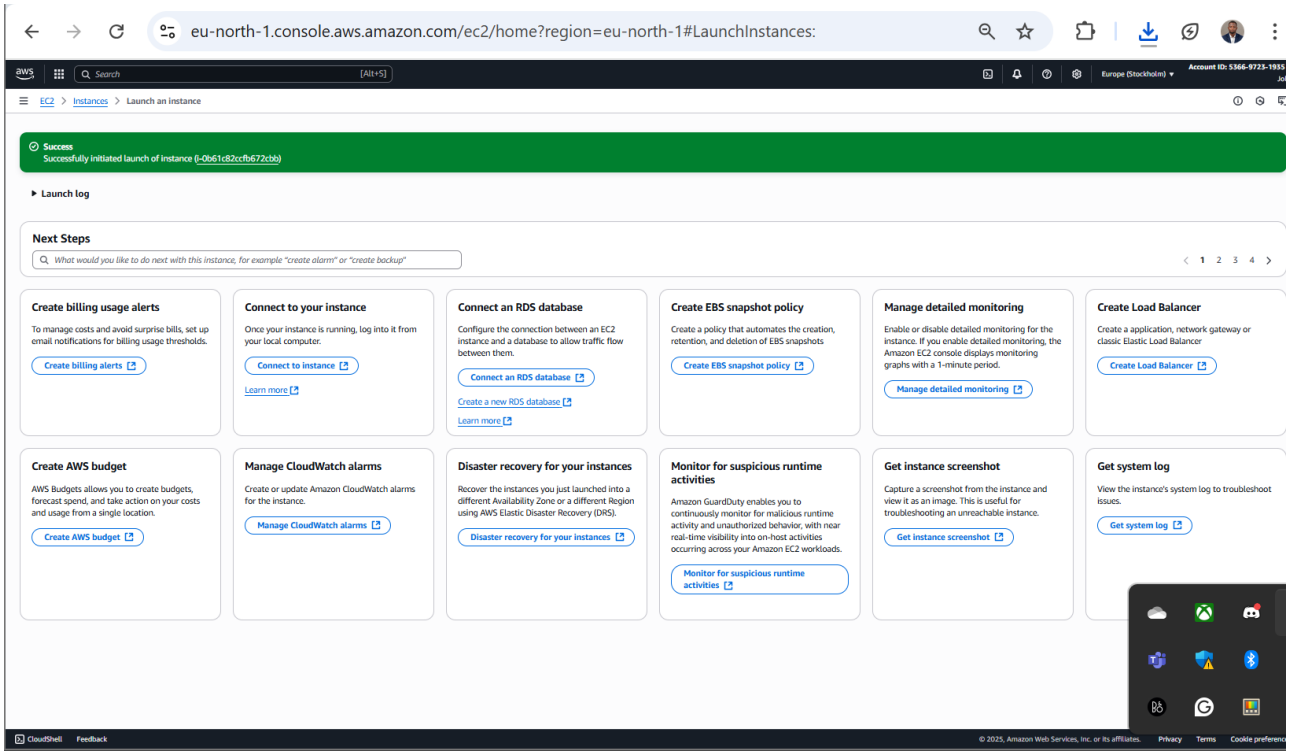
- Launch an instance page appear.



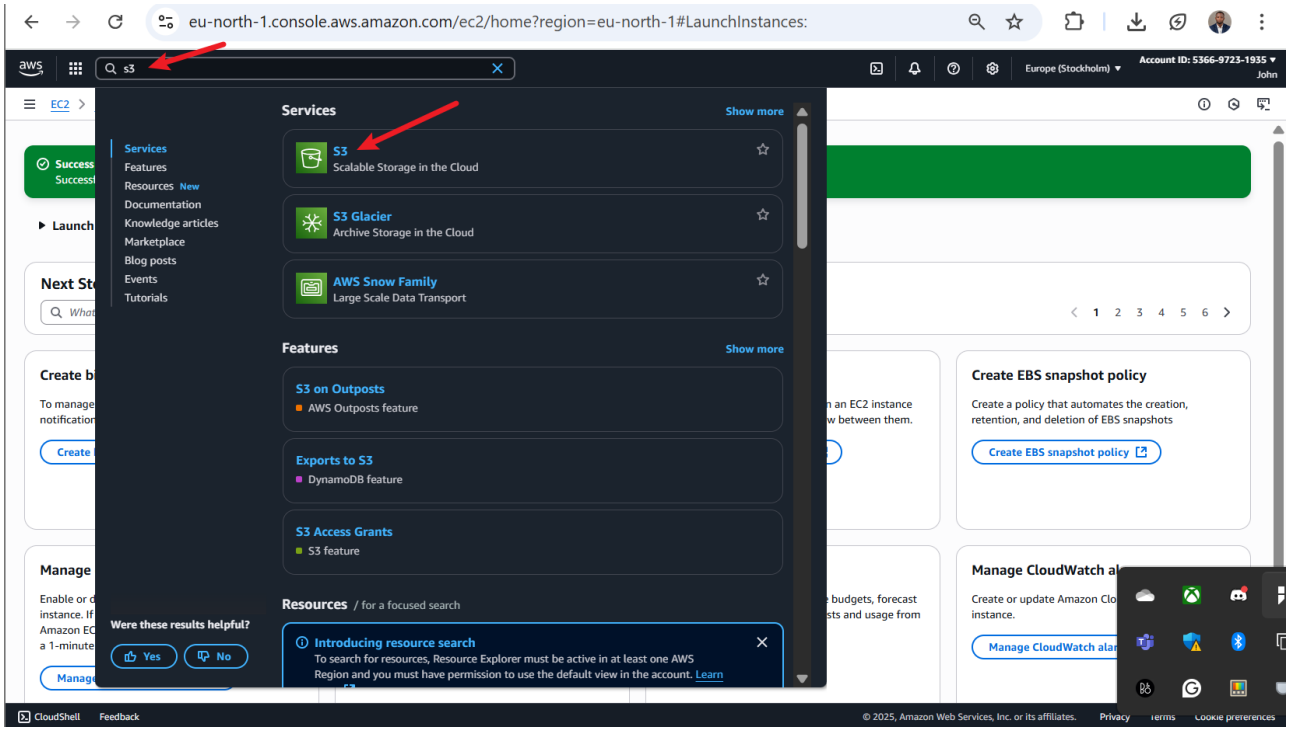
- Name server and set it up, also set key pairs and launch instance.



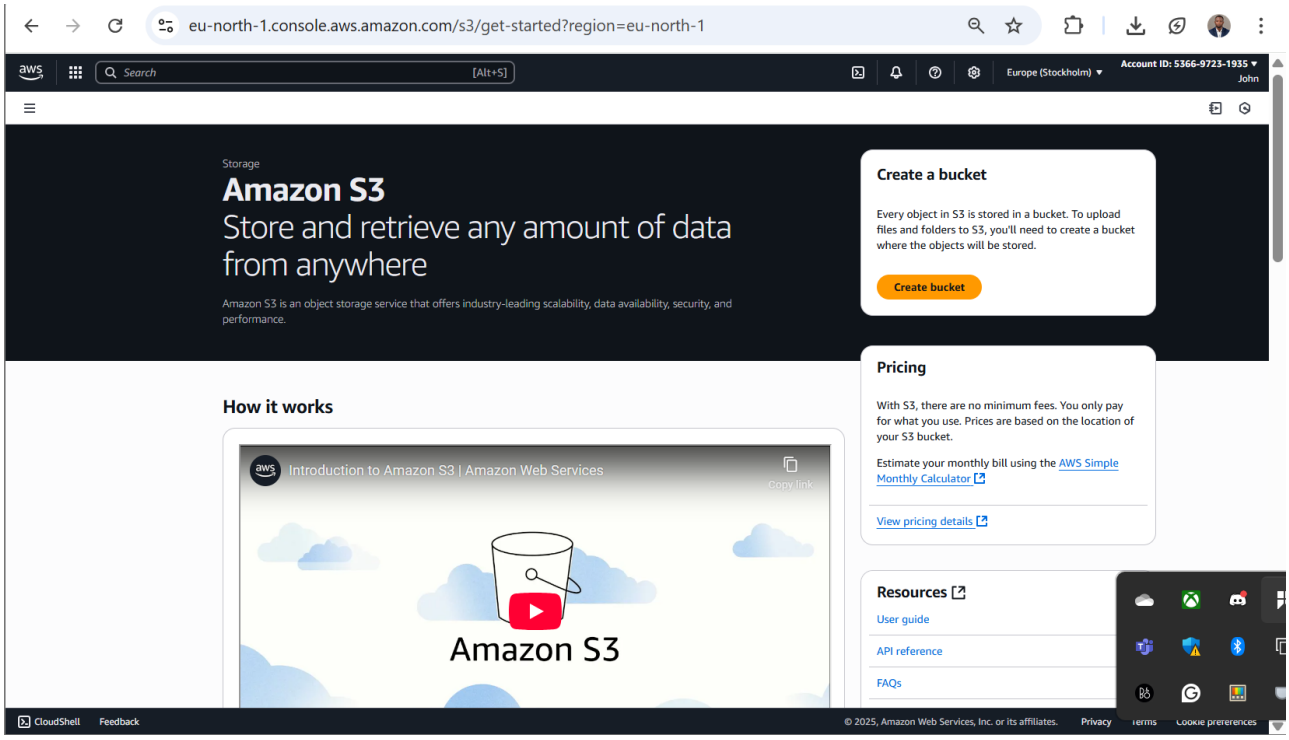
- Instance successfully launched.



- Search for S3 and click on it.

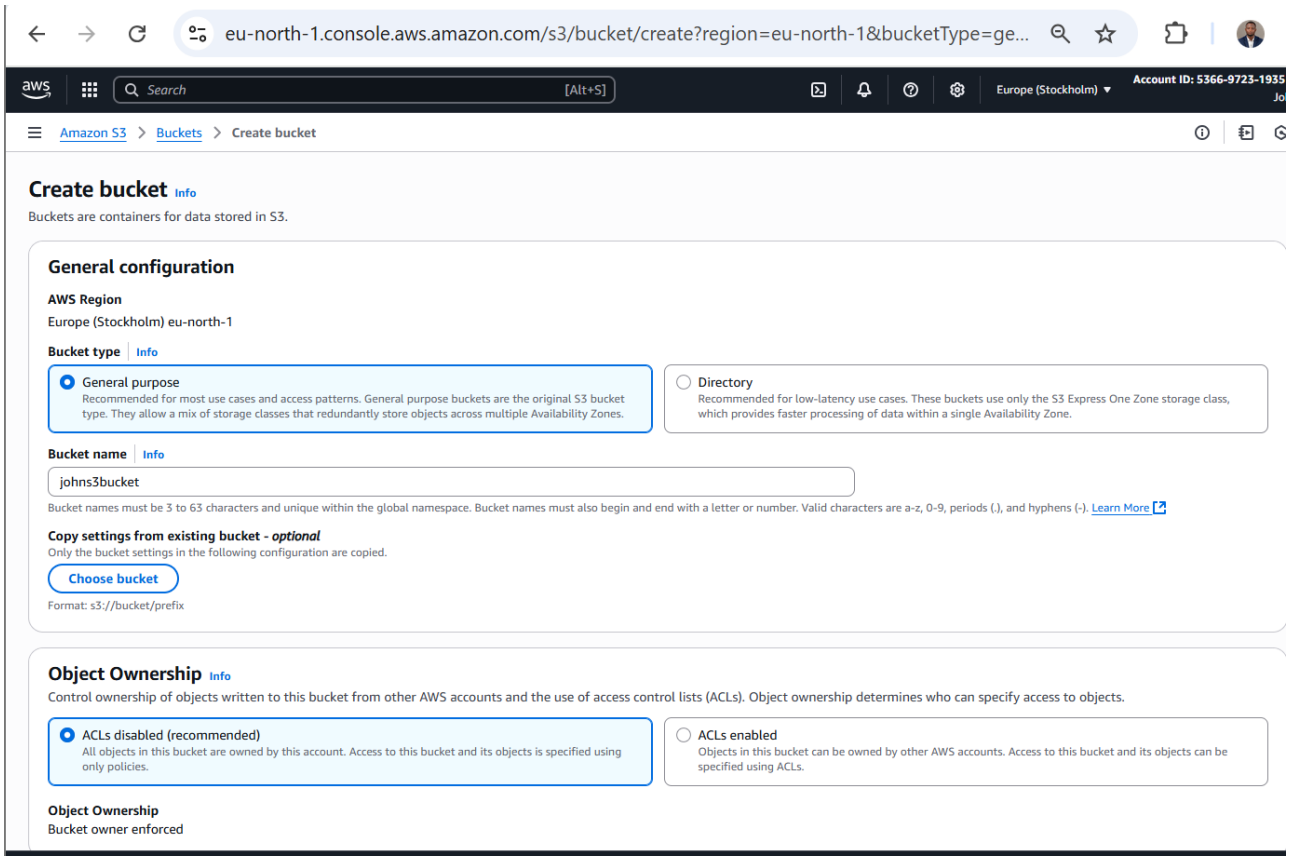


- Successfully access S3 dashboard, click create bucket.

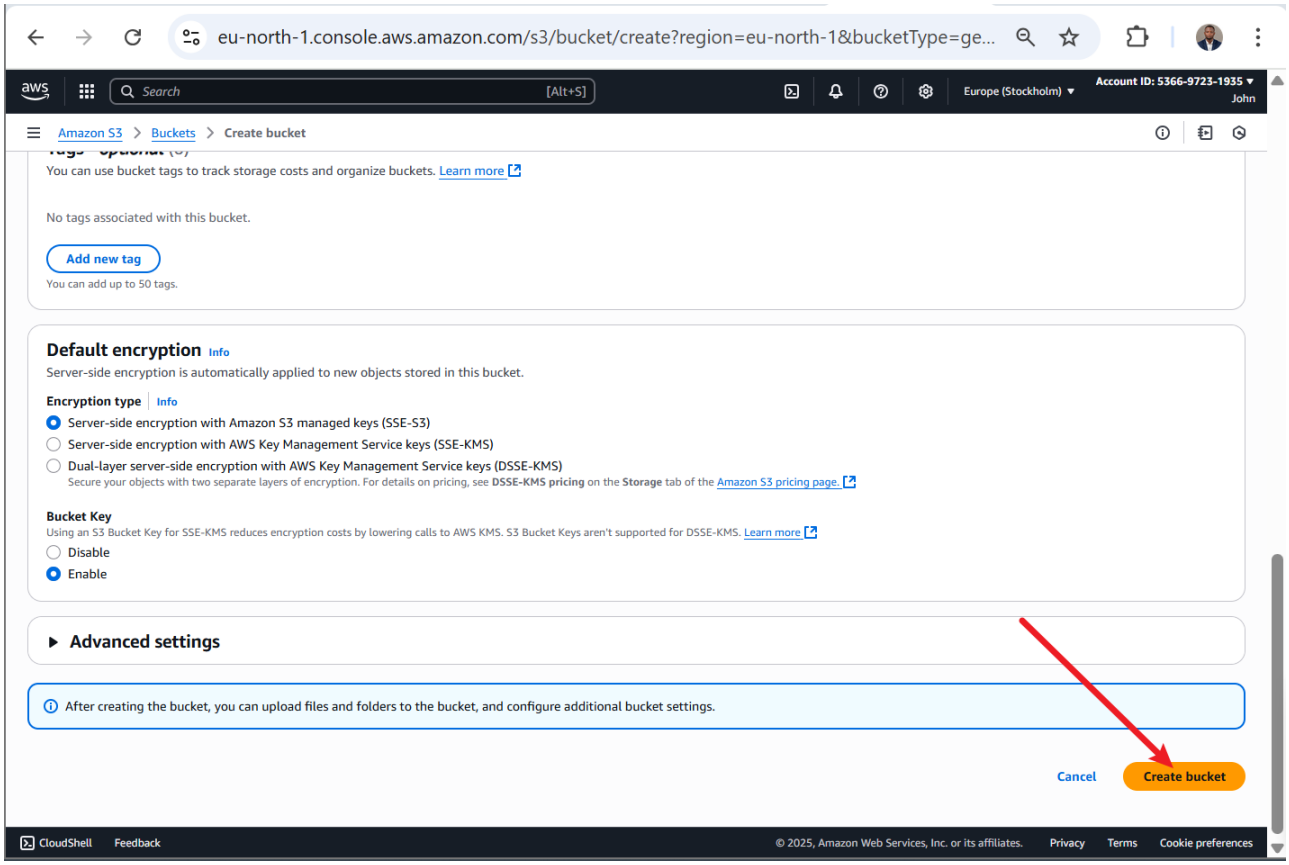


(Note: Context indicates access attempt leading to denial.)

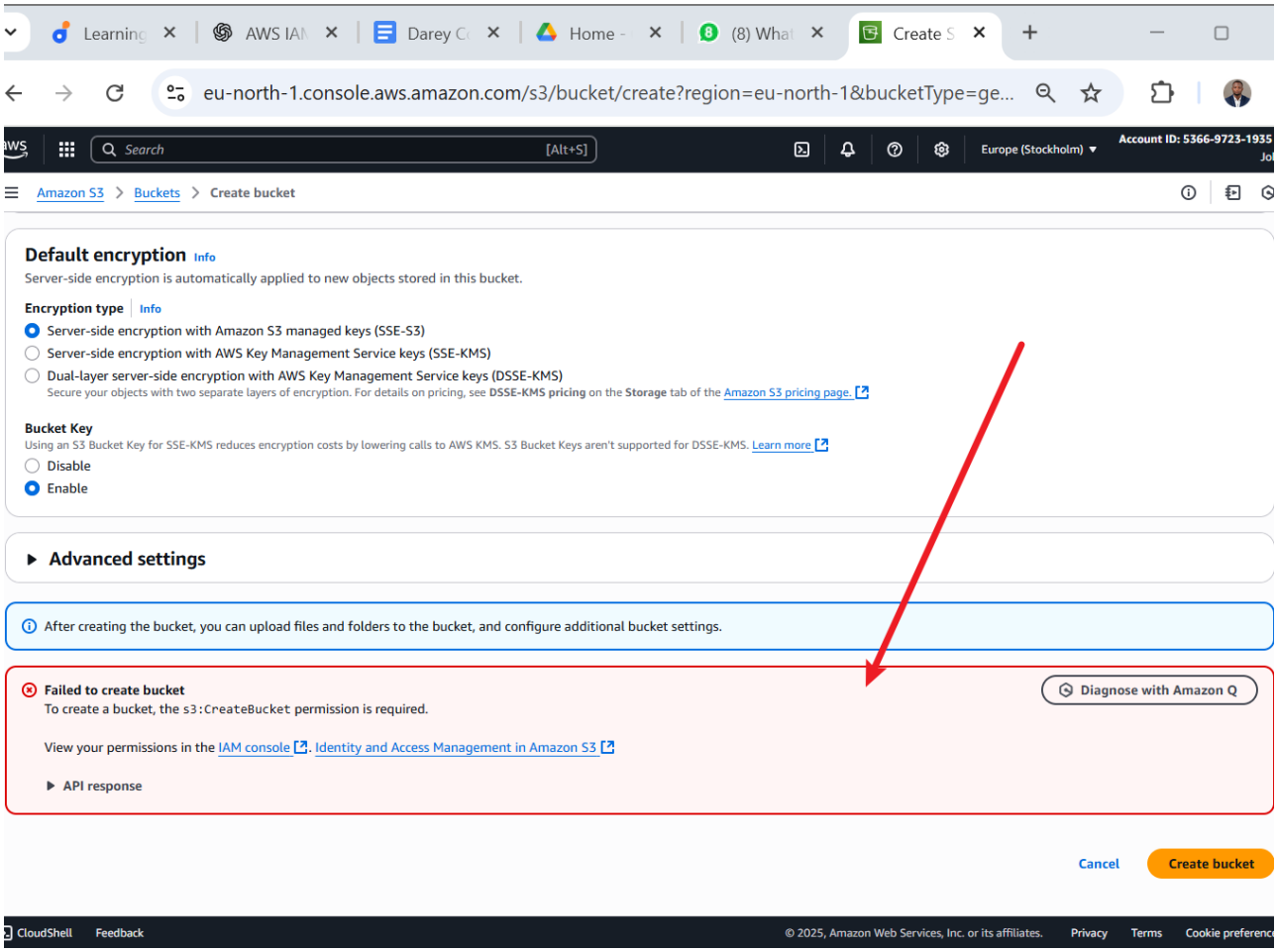
- Name bucket and set it up.



- Scroll down and click create bucket.



- Failed to create bucket.

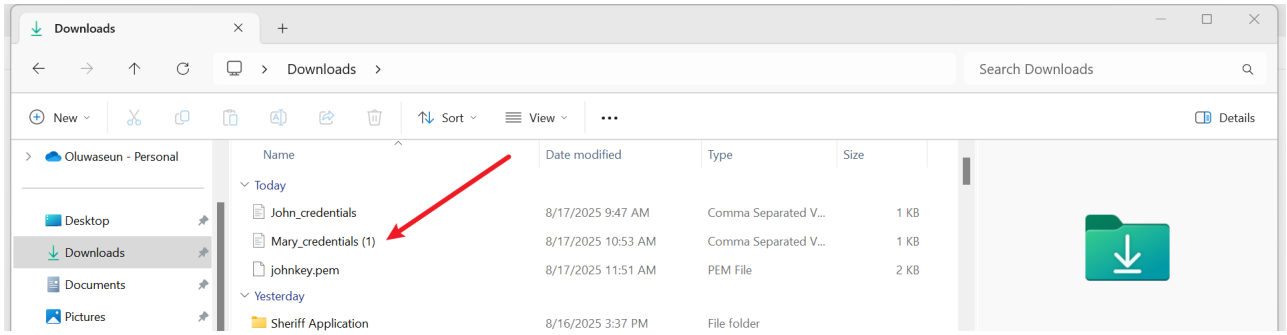


Mary's Access

- Logged in as Mary.
- Accessed **S3 Dashboard** ☑.
- Attempted to access **EC2 Dashboard** ❌ (denied).

Steps:

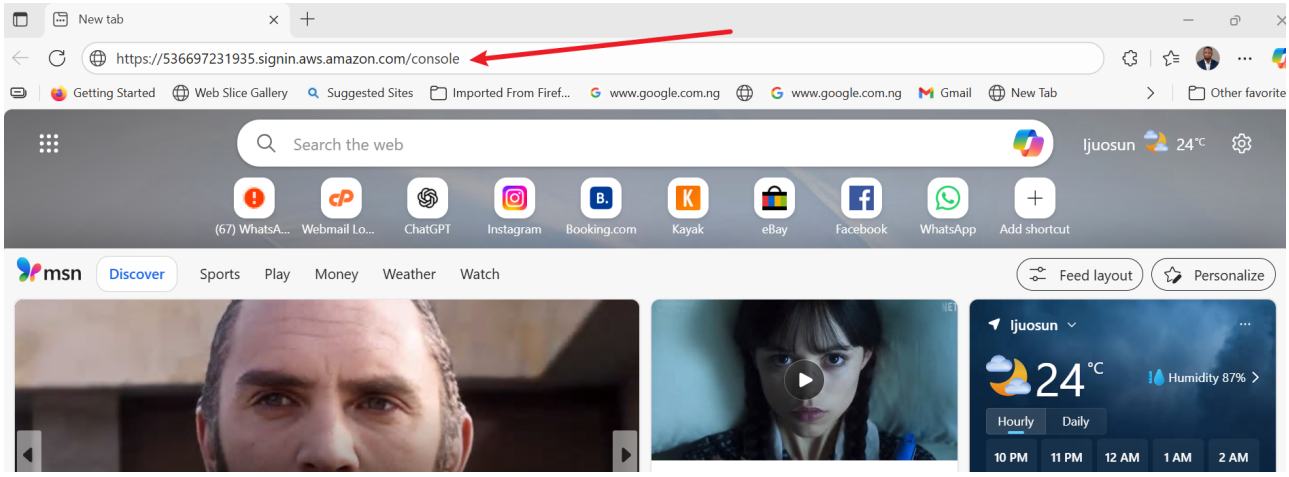
- Locate and open Mary downloaded CSV file.



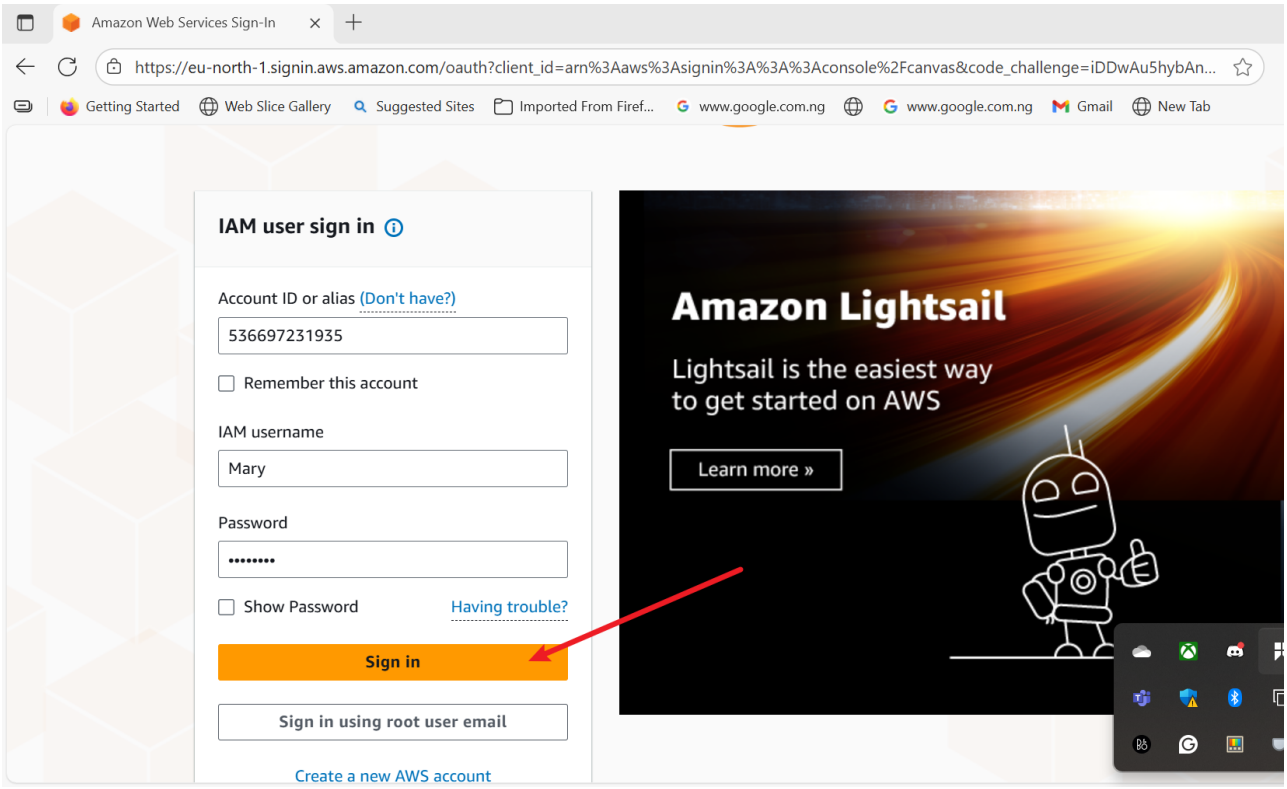
(Note: Filename references John, but context indicates Mary's file.)

- Copy the HTTPS link.

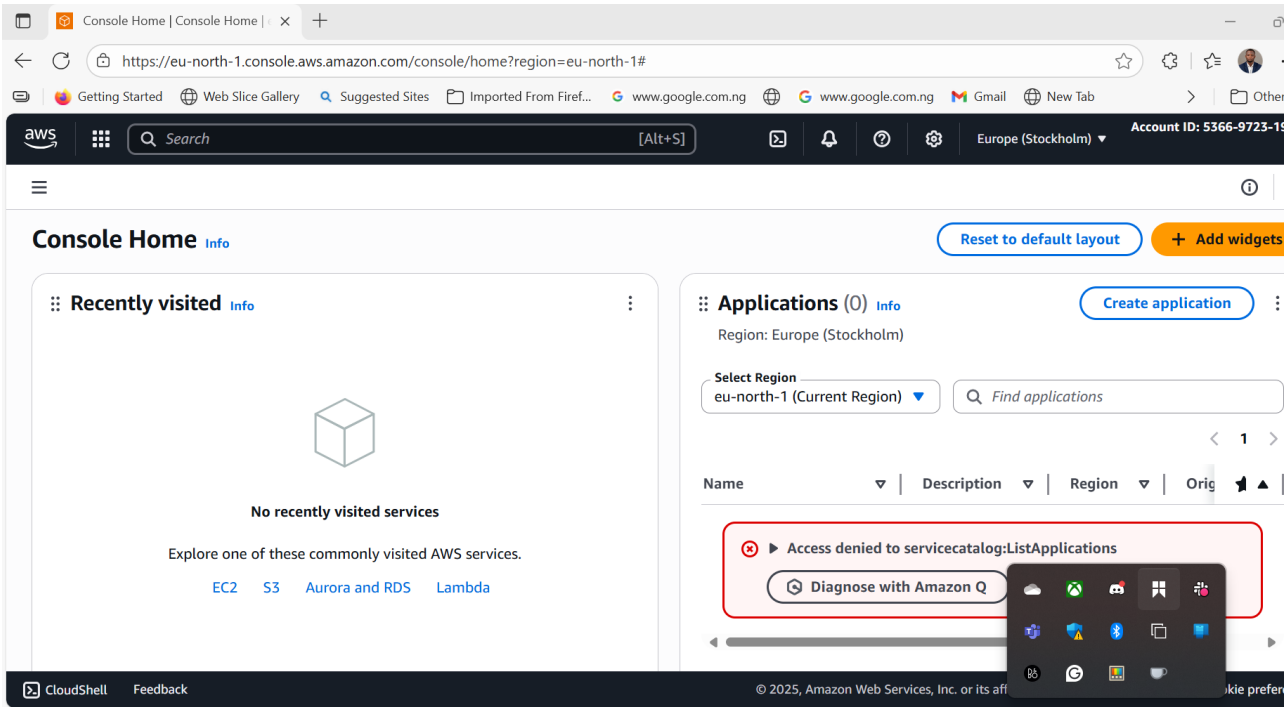
- Paste it on the browser.



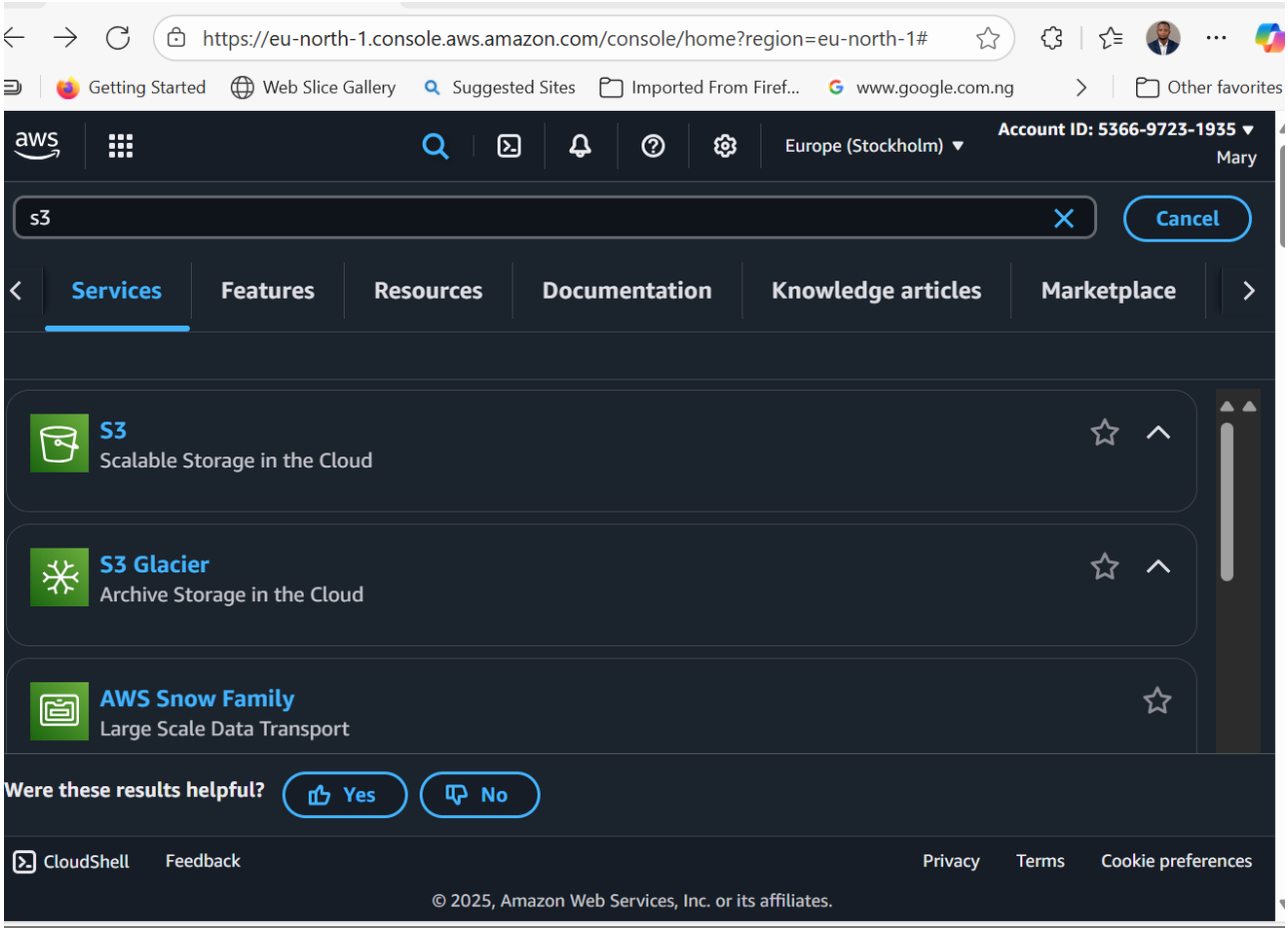
- Sign in as IAM user using your login and account ID which is fetched by the link.



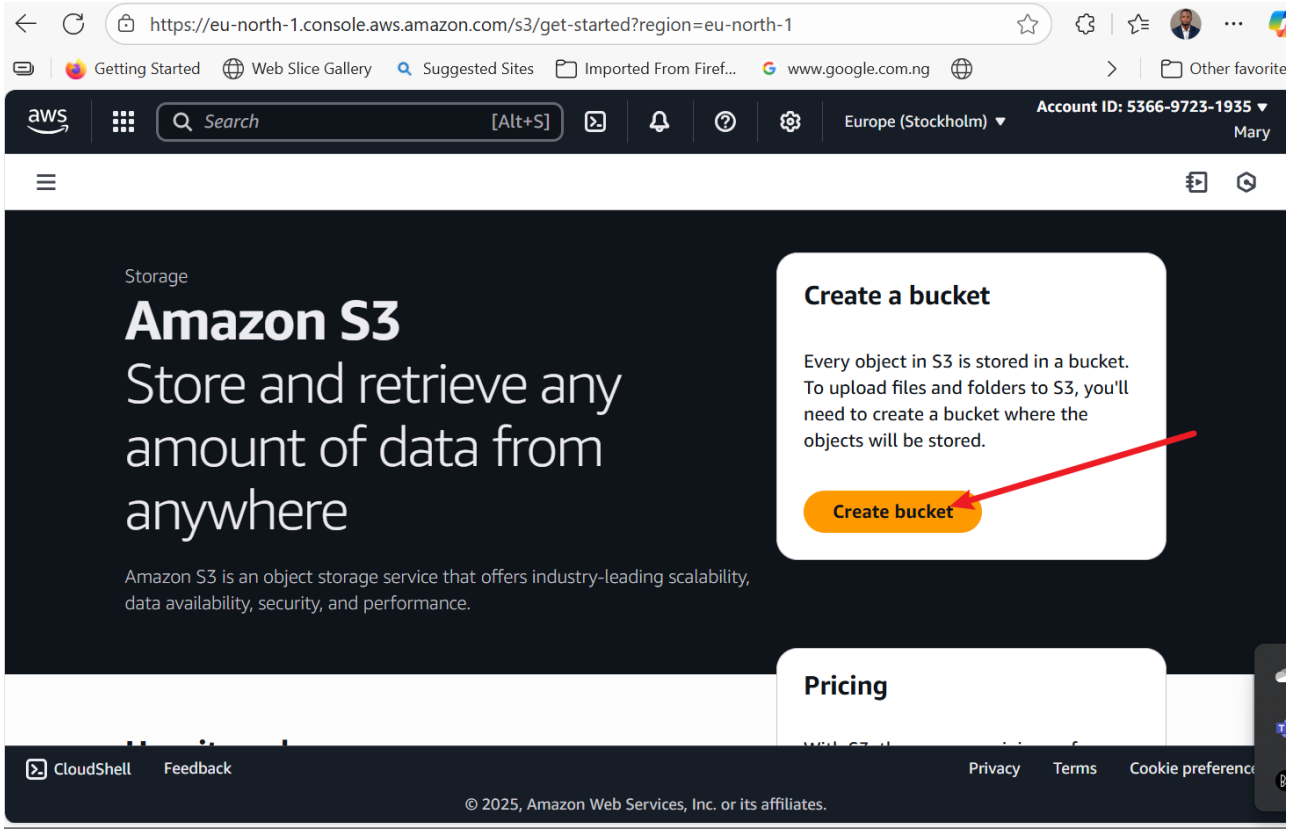
- Logged in as Mary.



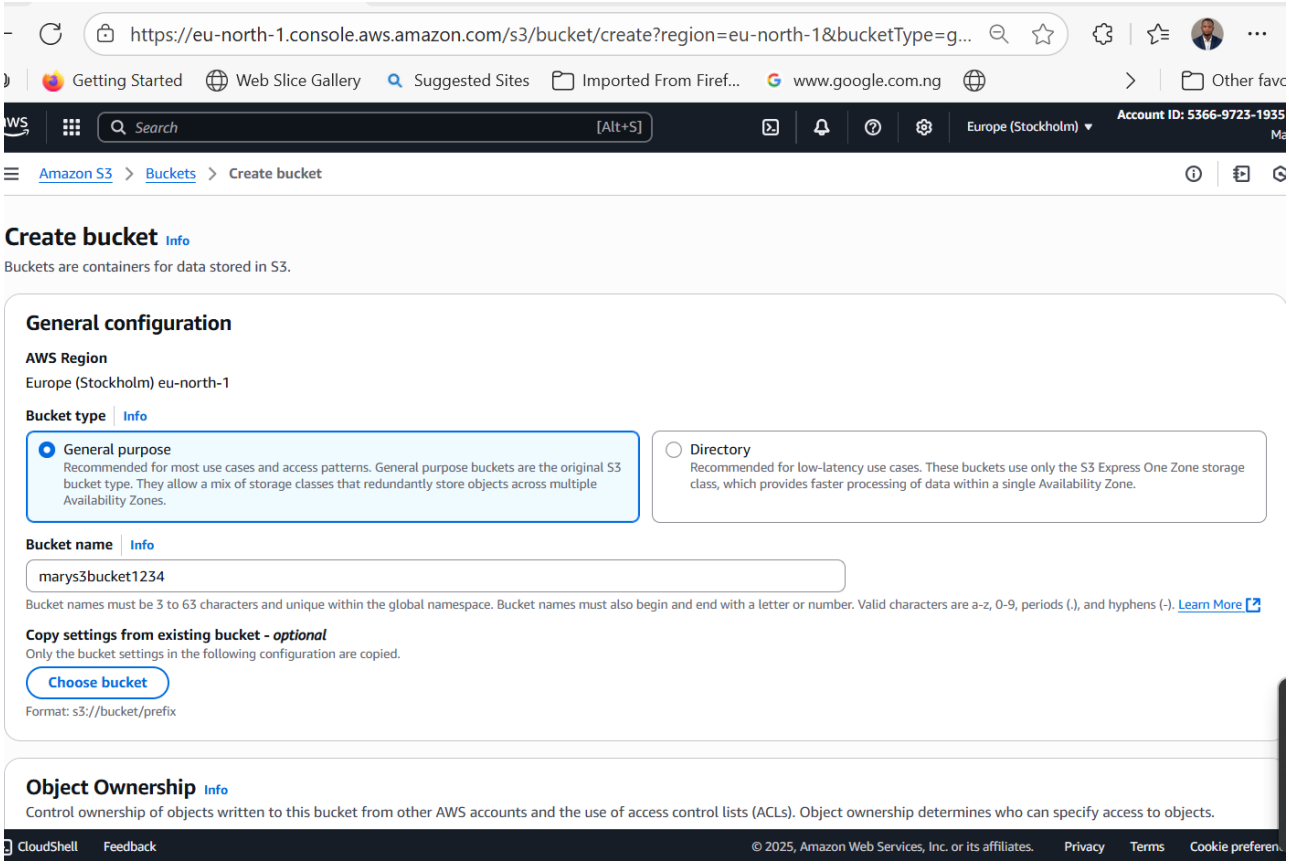
- Search for S3 and click on it.



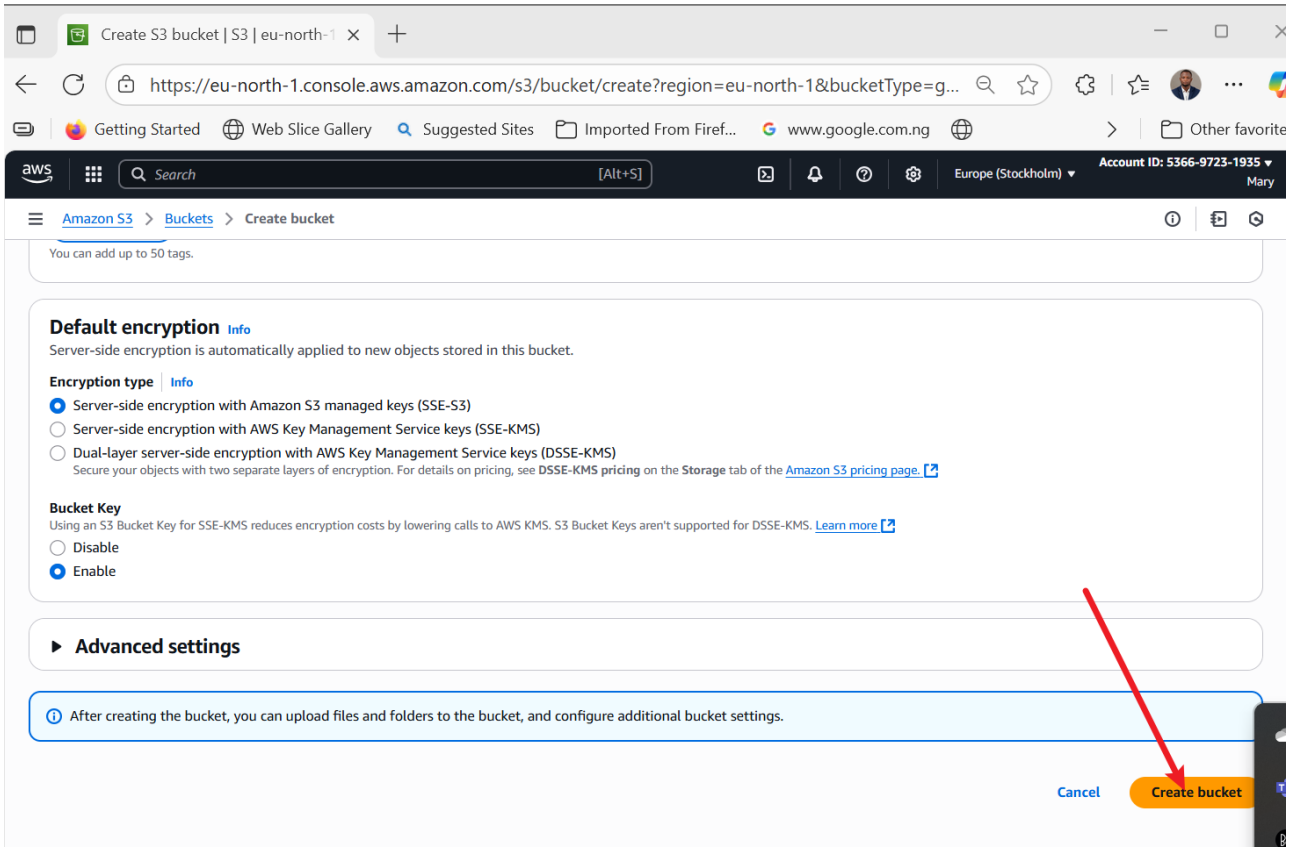
- Successfully access S3 dashboard, click create bucket.



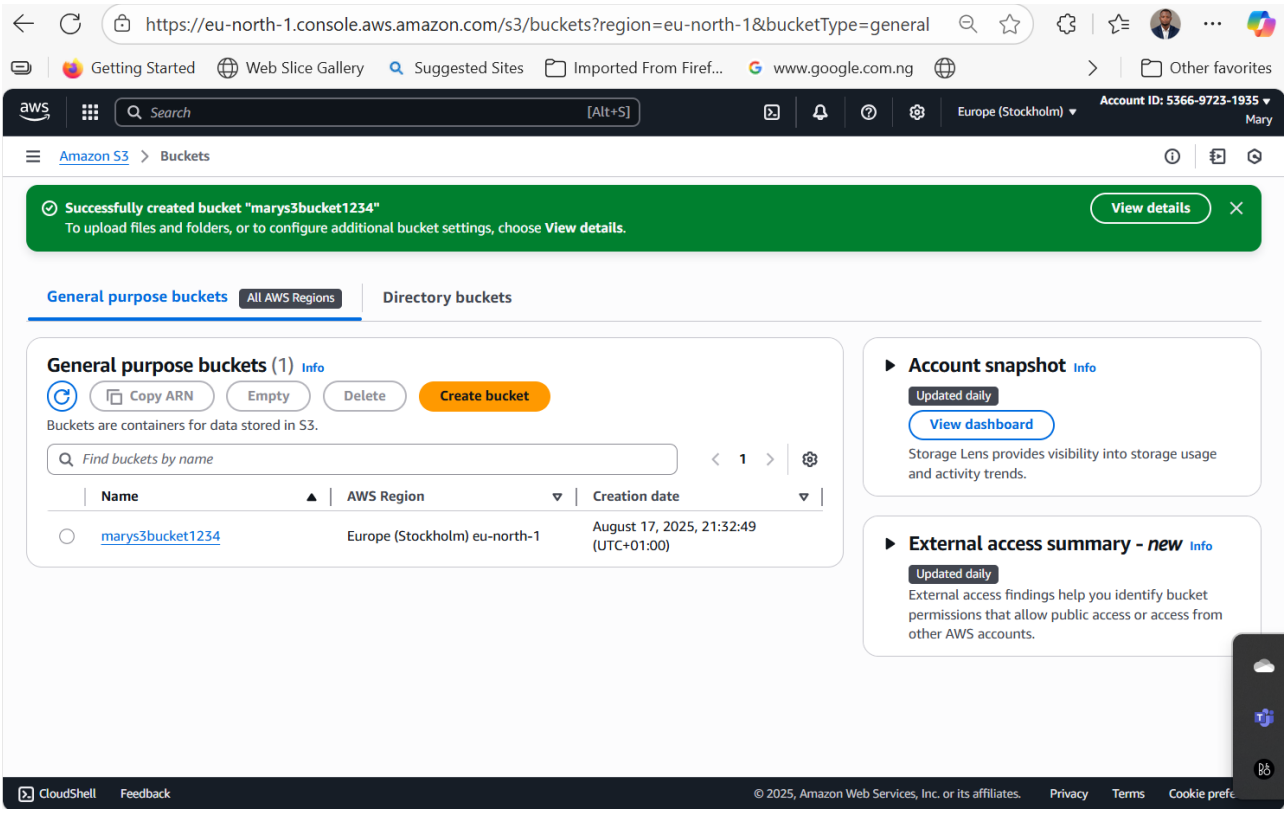
- Name bucket and set it up.



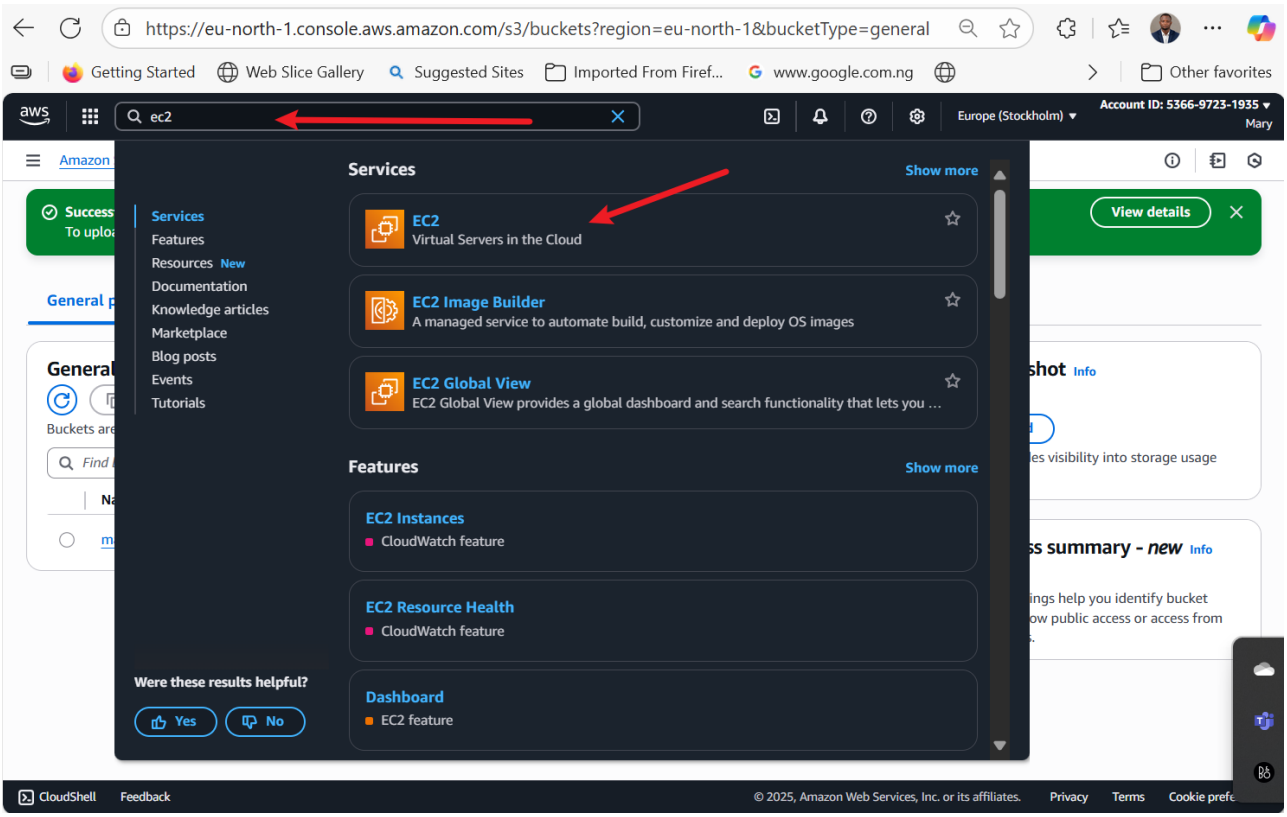
- Scroll down and click create bucket.



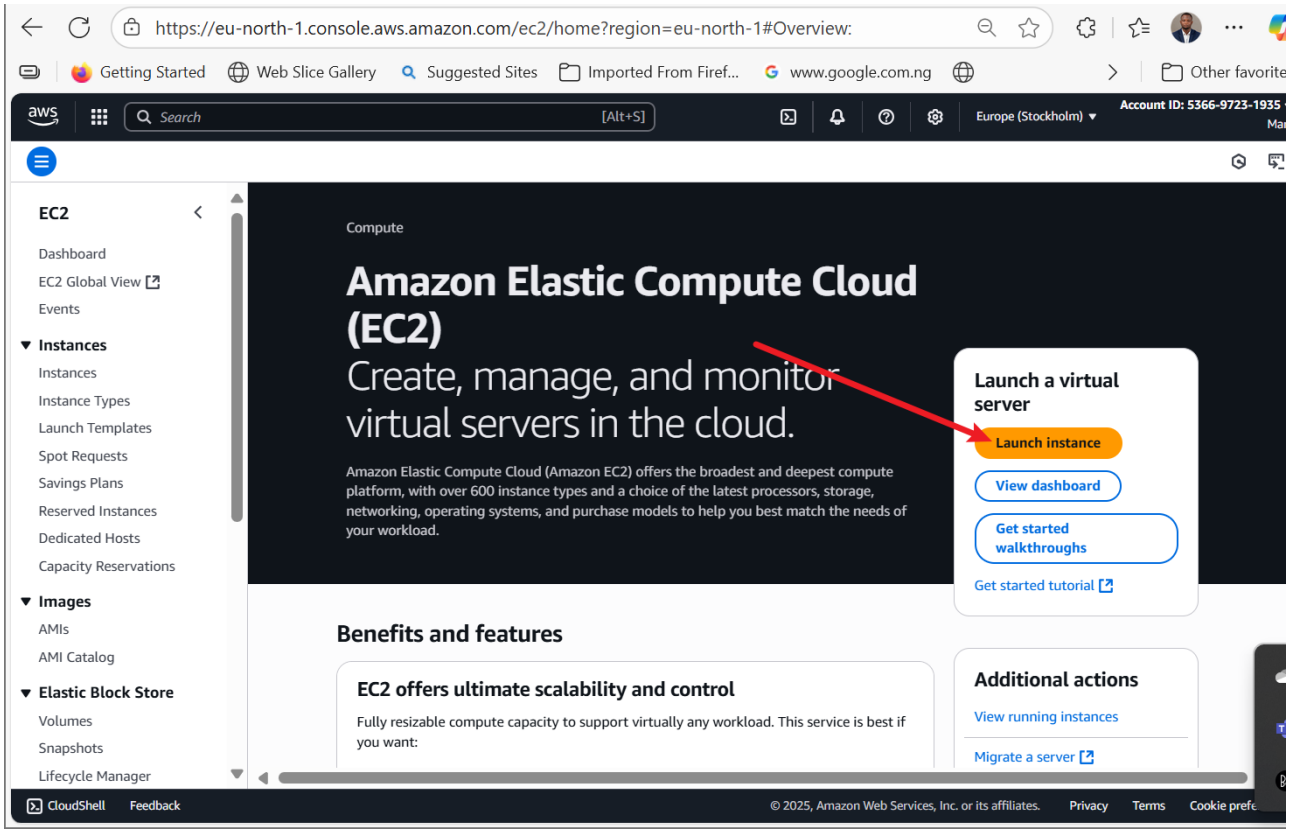
- Bucket successfully created.



- Search for EC2 and click on it.

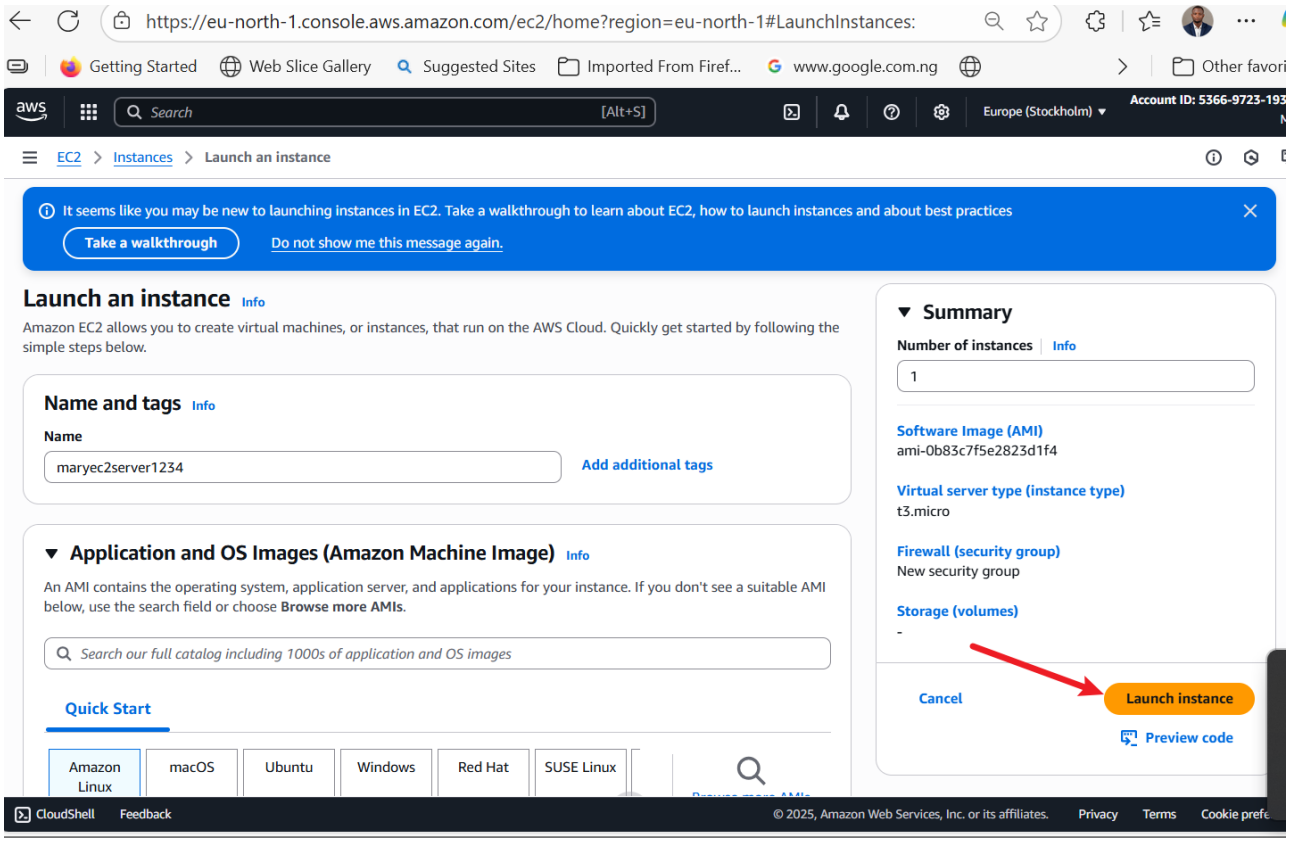


- Successfully access EC2 dashboard, click launch instance.

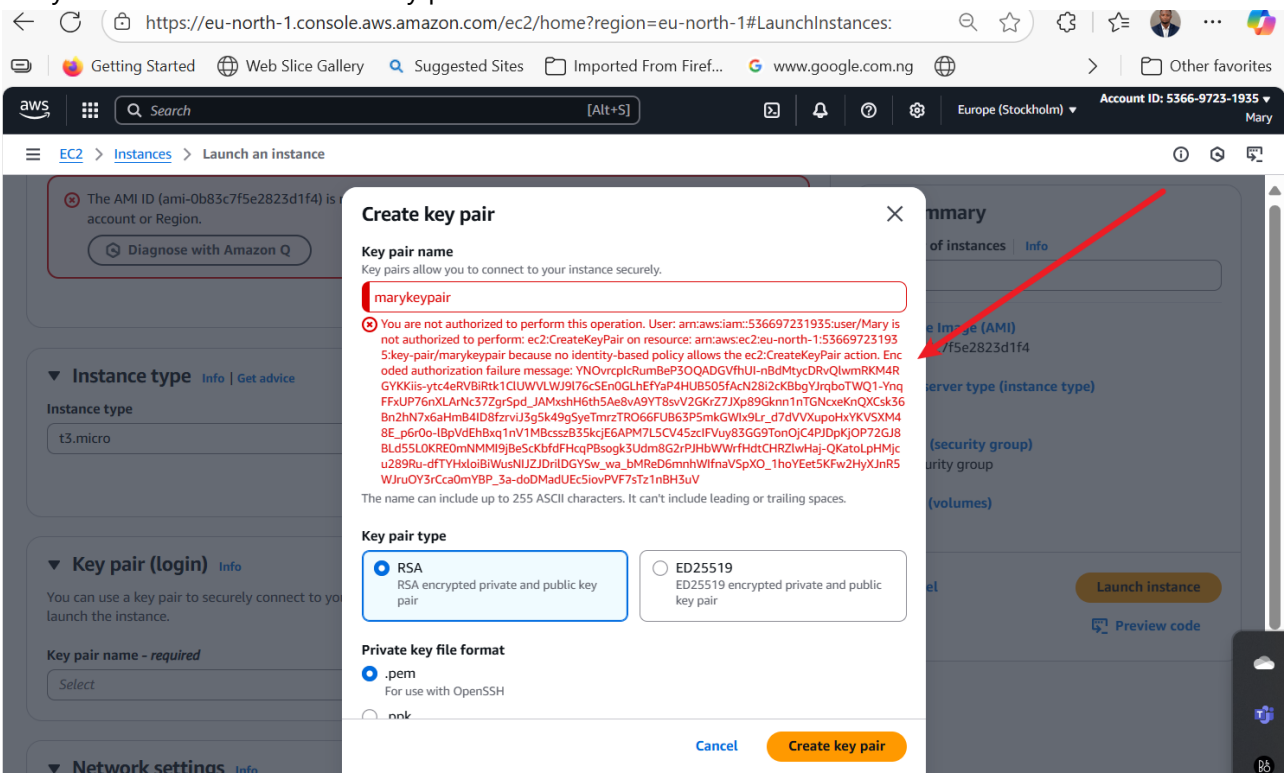


(Note: Context indicates access attempt leading to denial.)

- Name server and set it up, also set key pairs and launch instance.



- Mary not authorized to create key pair.



This confirmed that users had **role-specific permissions** aligned with the principle of least privilege.

---

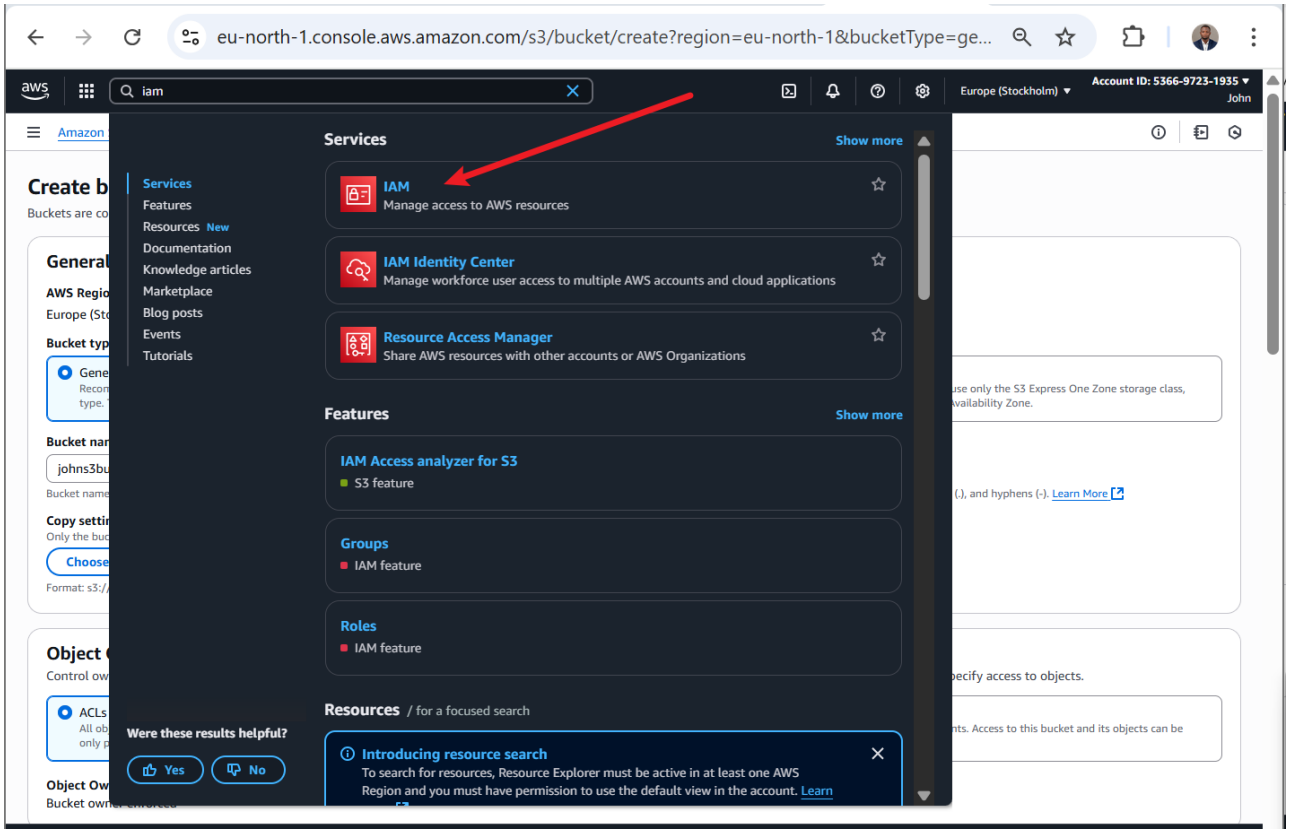
8. Enabling Multi-Factor Authentication (MFA)

John (Backend Developer)

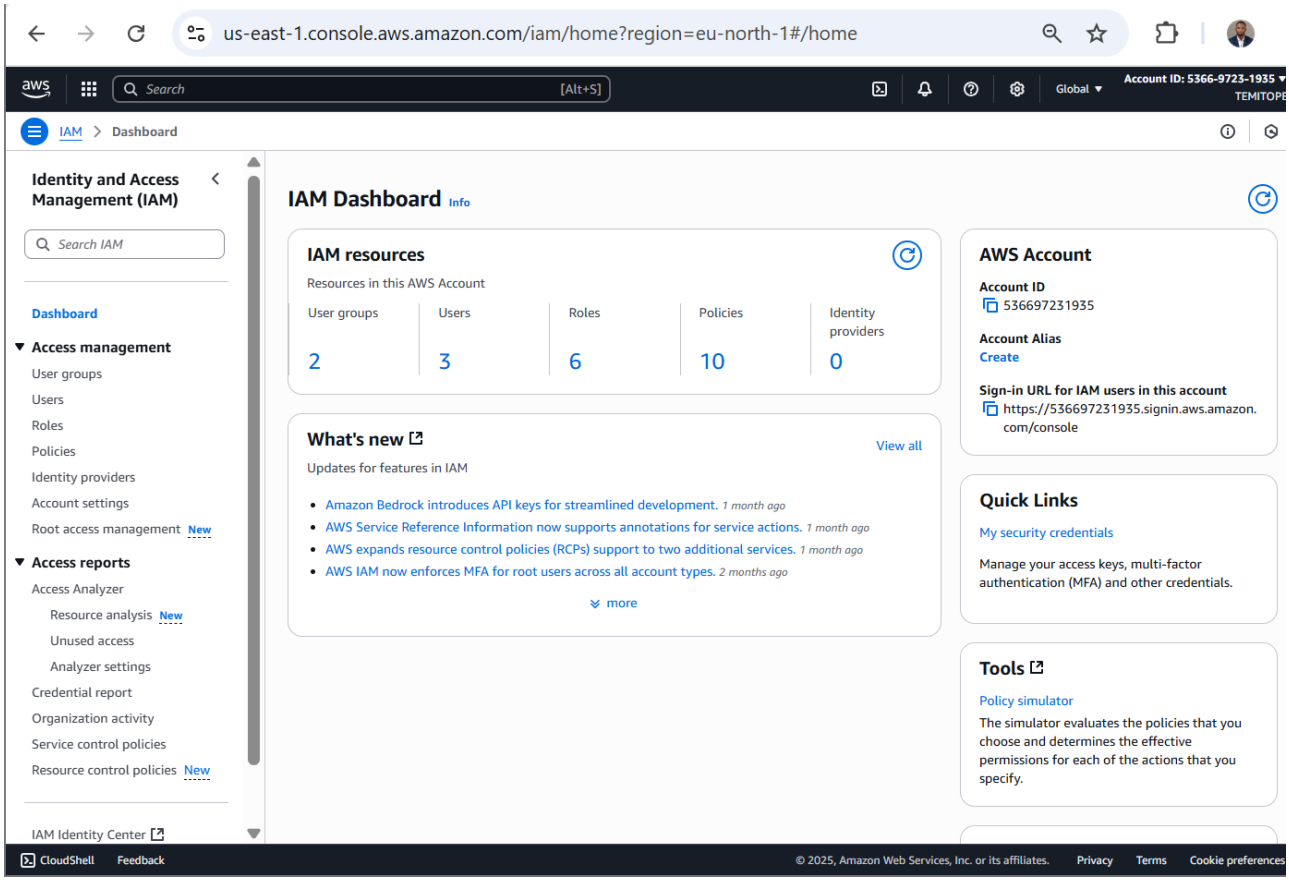
- Enabled MFA.
- Linked with **Google Authenticator**.
- Tested successful login with MFA.

Steps:

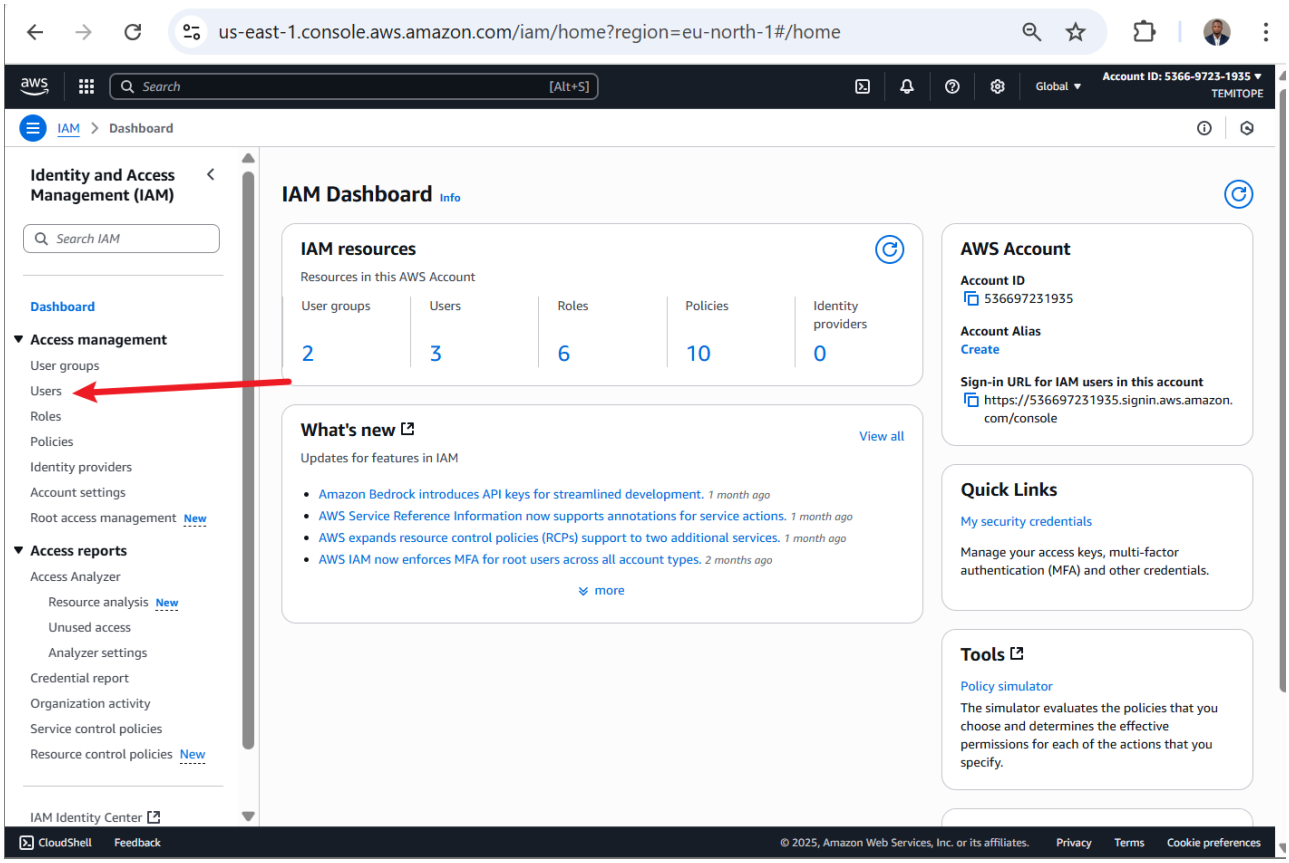
- Search IAM to set MFA for John.



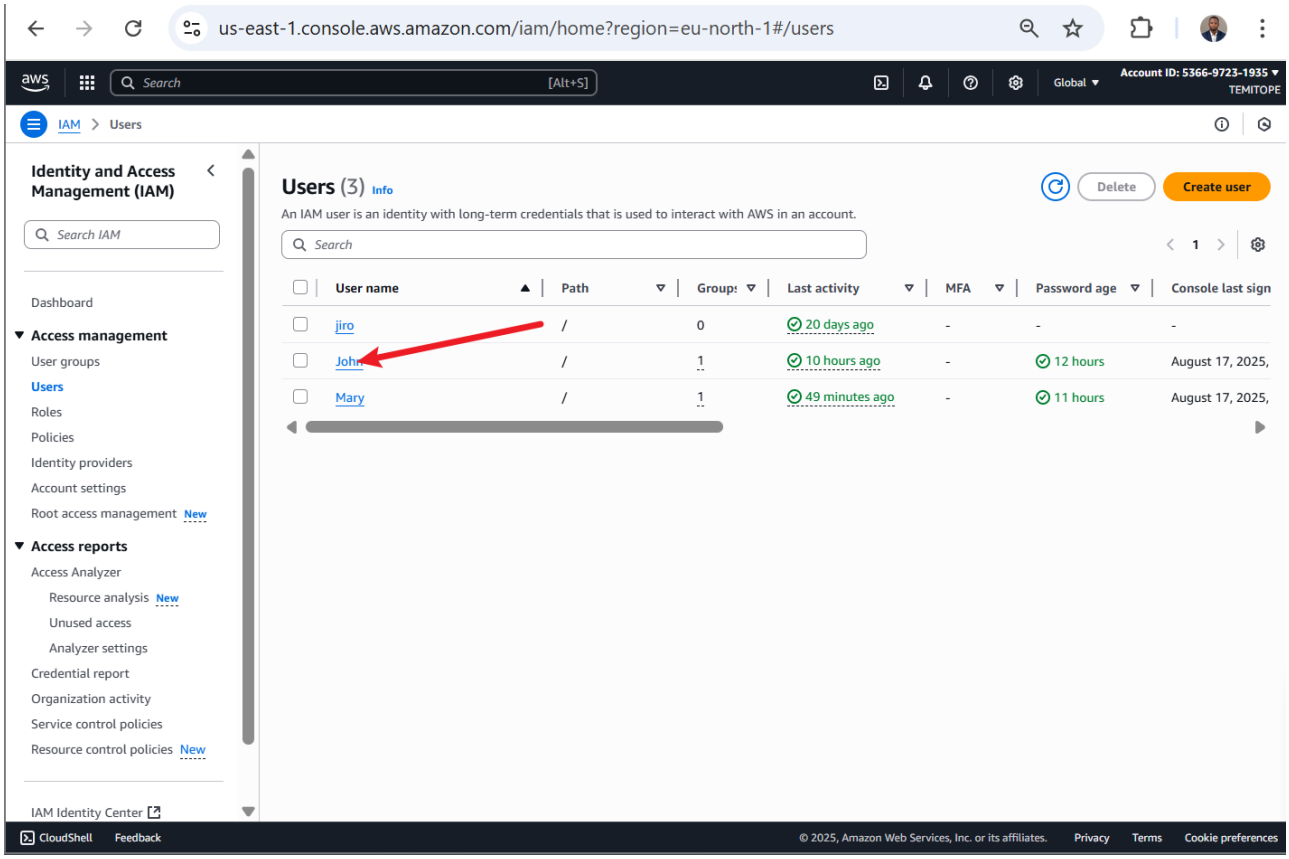
- Login as IAM admin and navigate to IAM dashboard.



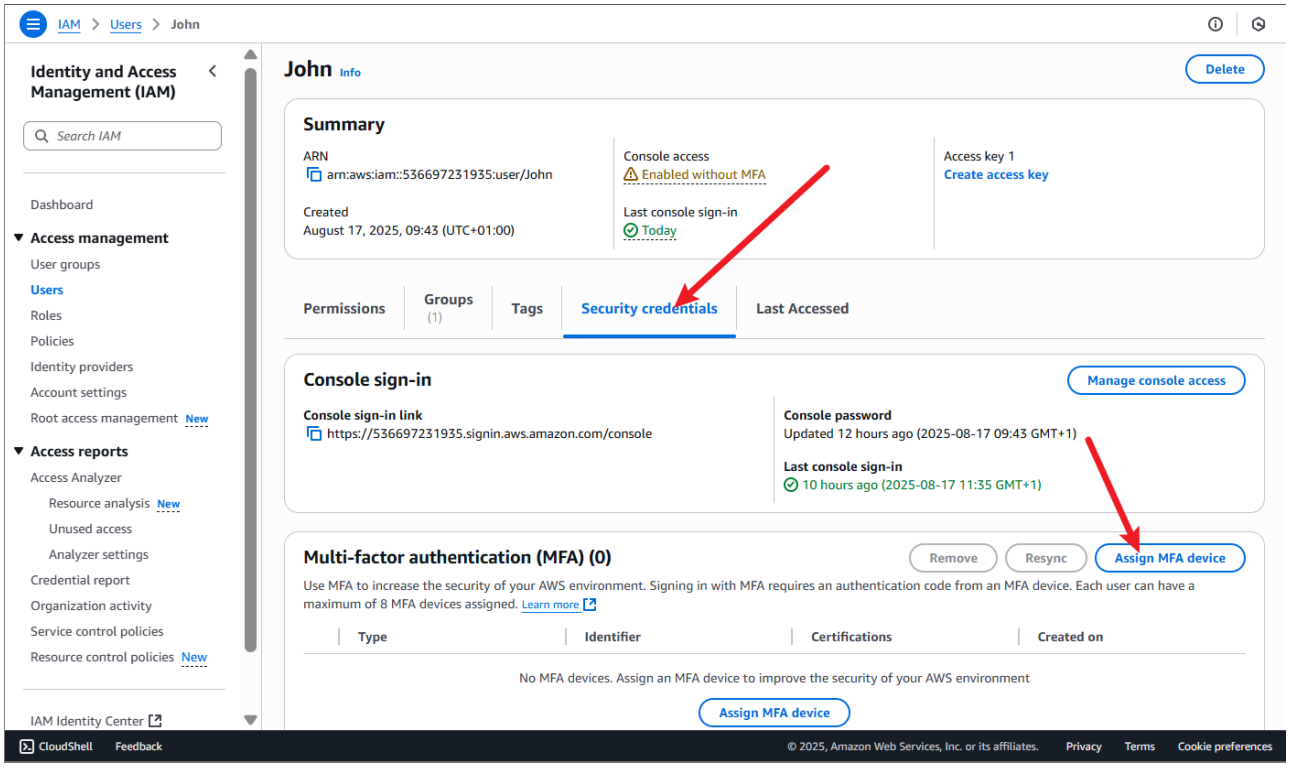
- Click on users.



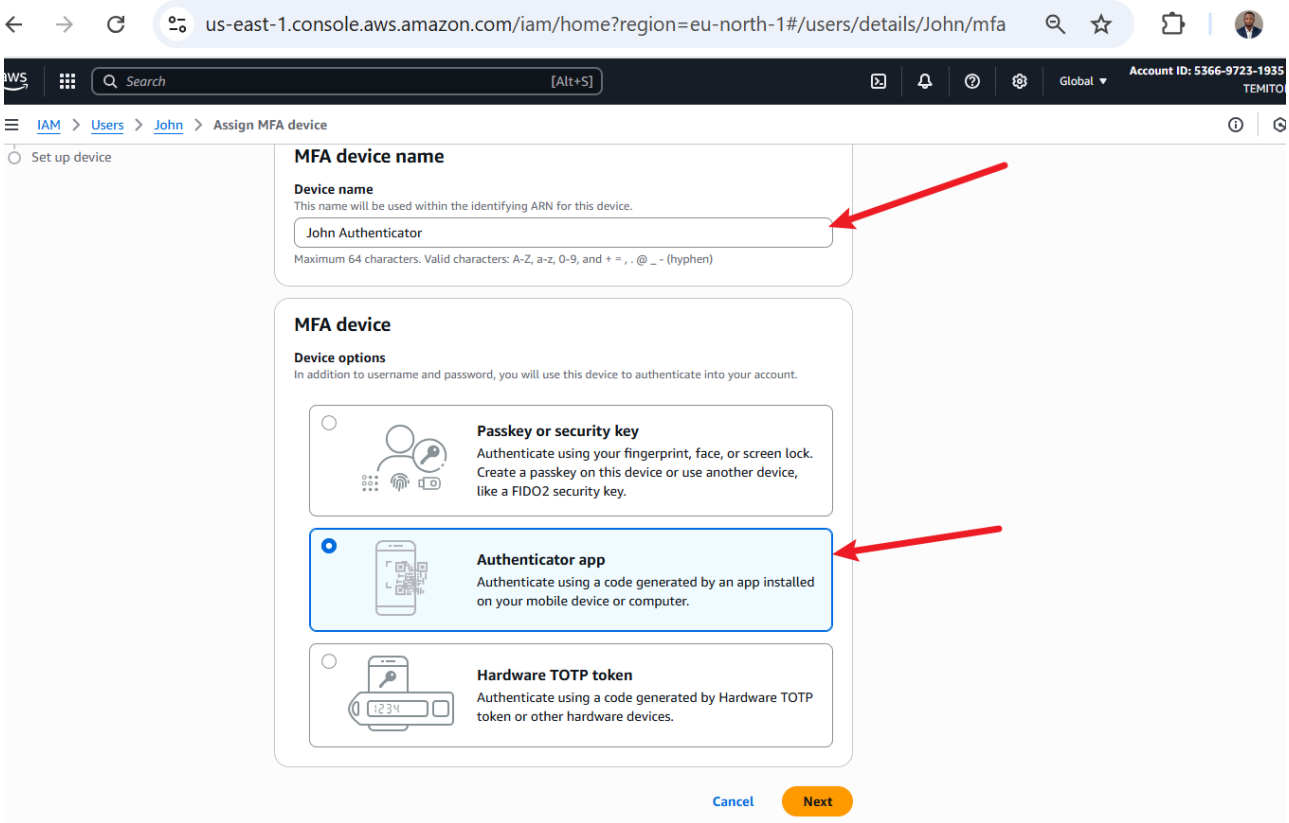
- Click on John.



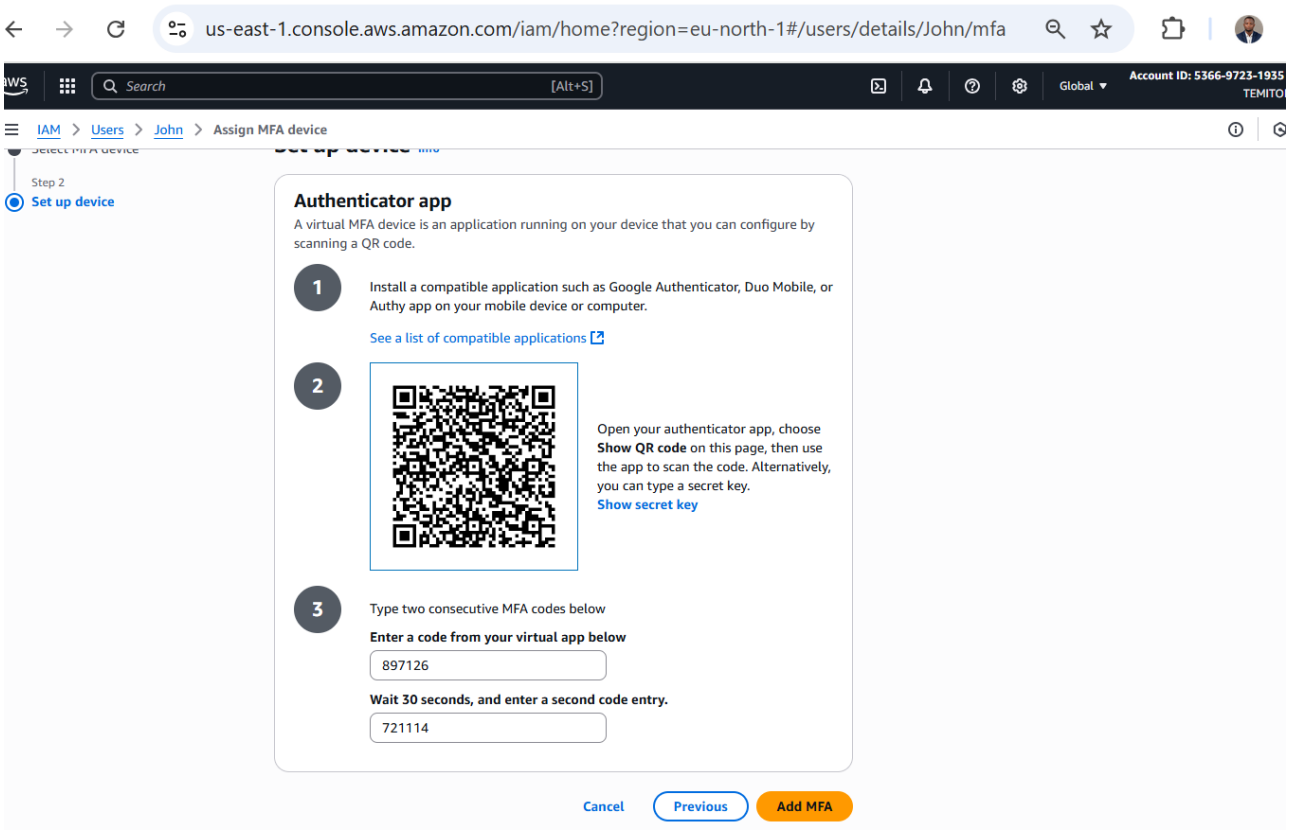
- Click security credential and assign MFA.



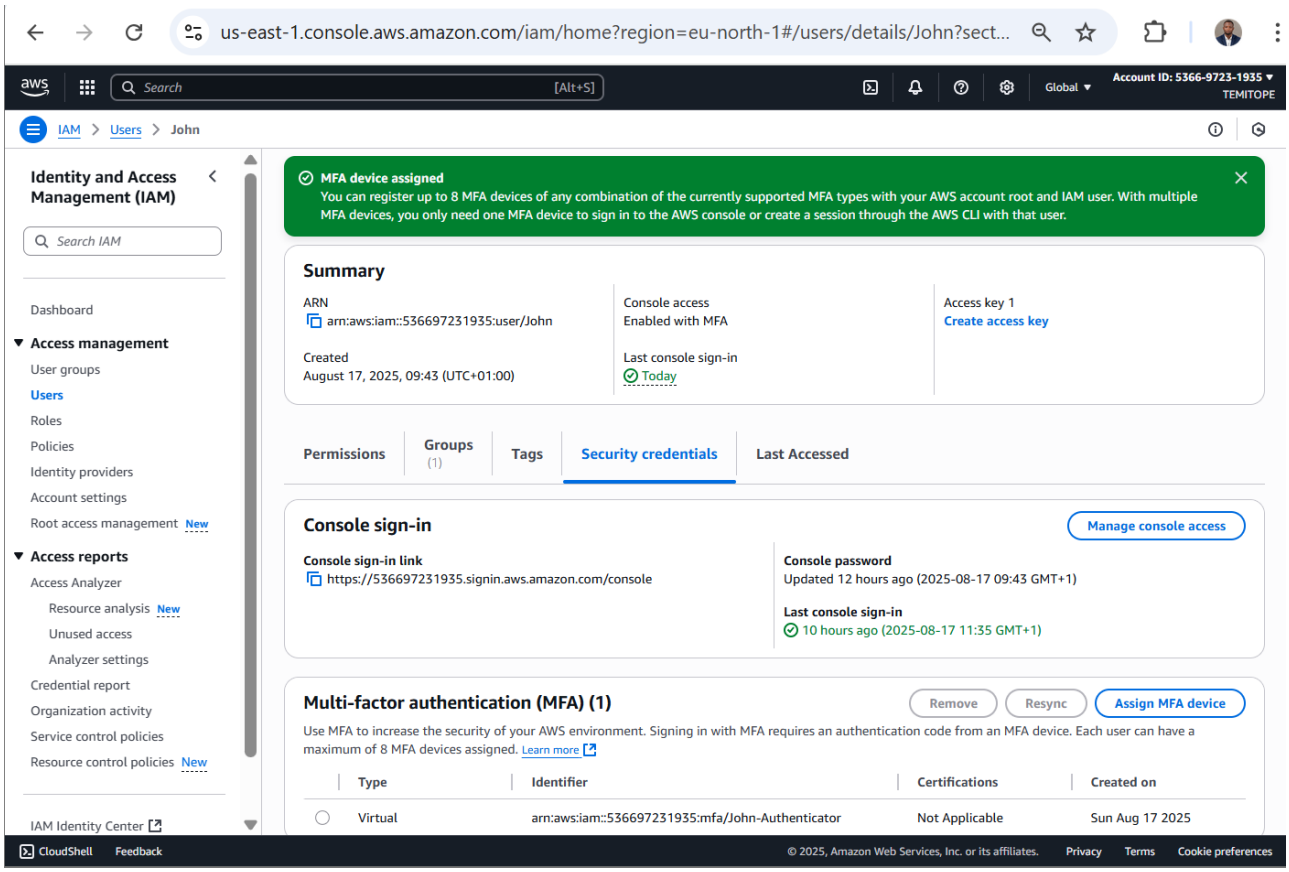
- Set name and choose authenticator app and click next.



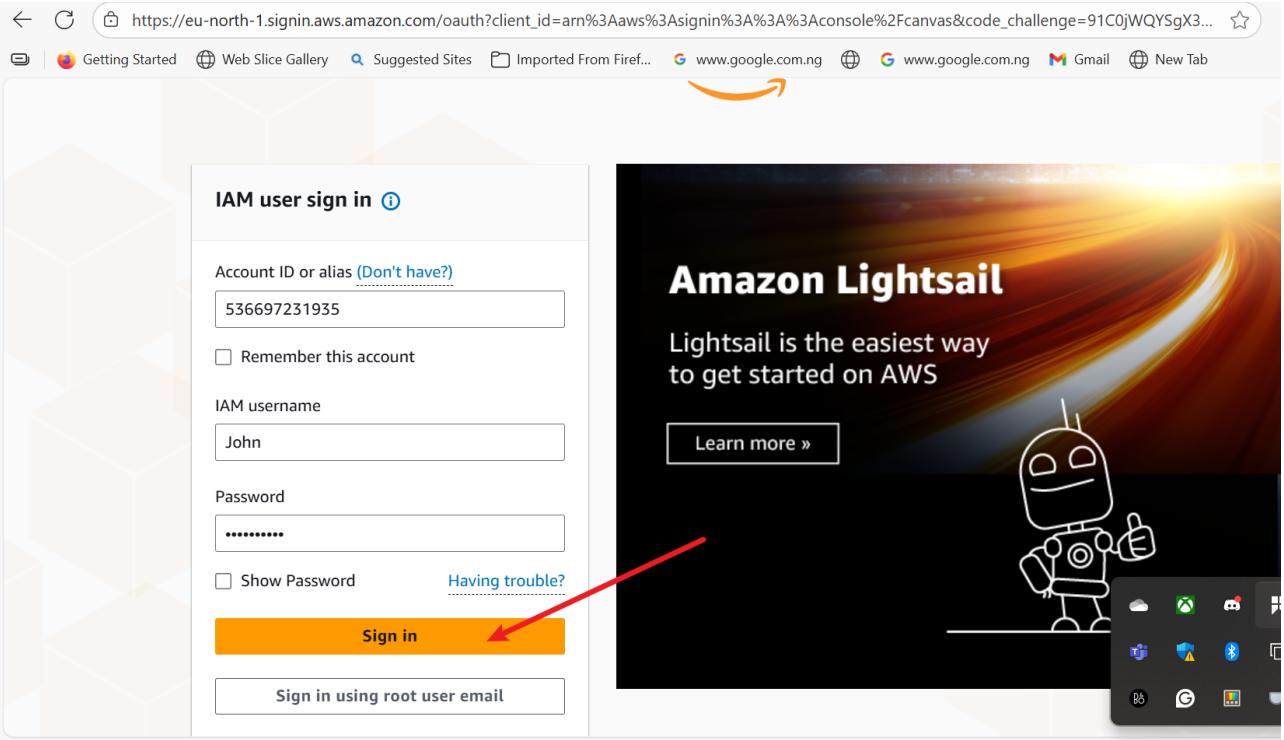
- Show QR, scan it and supply code as instructed.



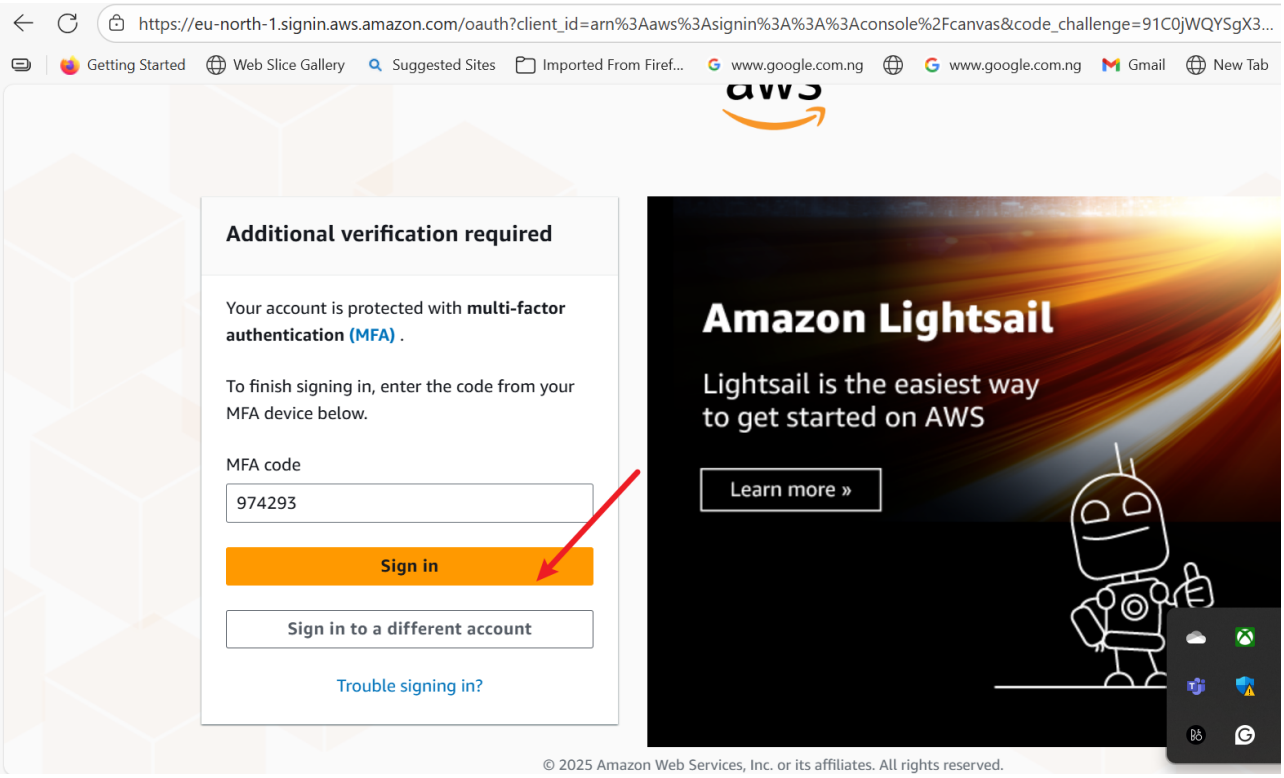
- MFA successfully added.



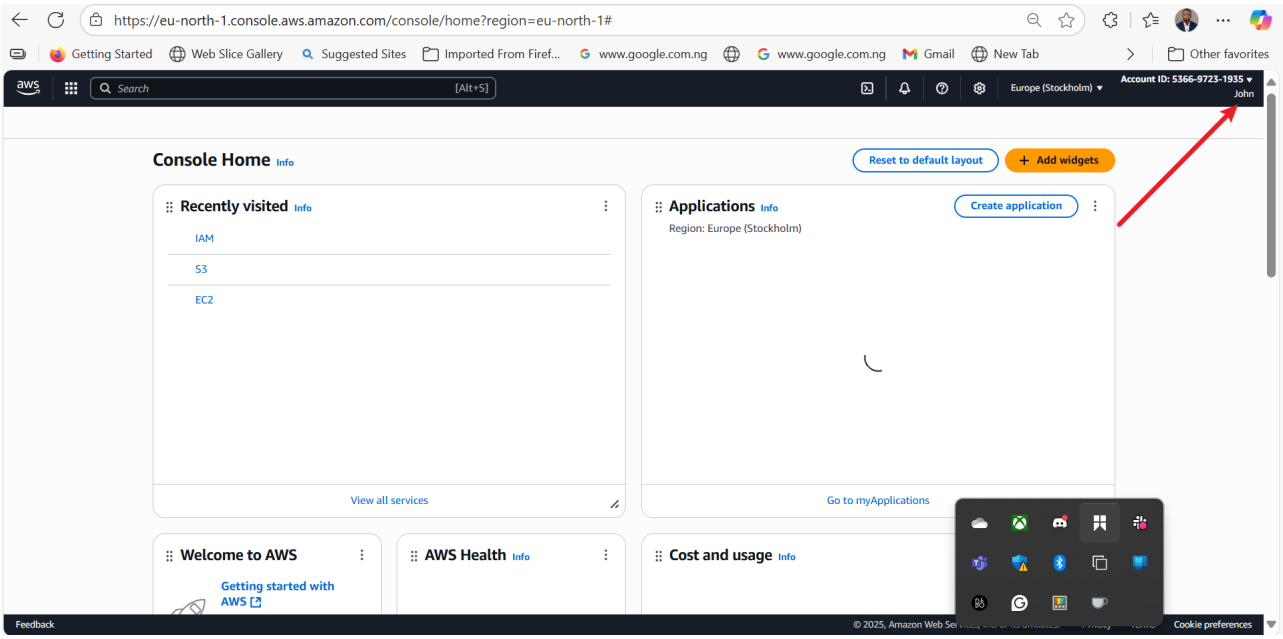
- Sign-in as John.



- Supply required MFA and login.



- Successfully logged in.

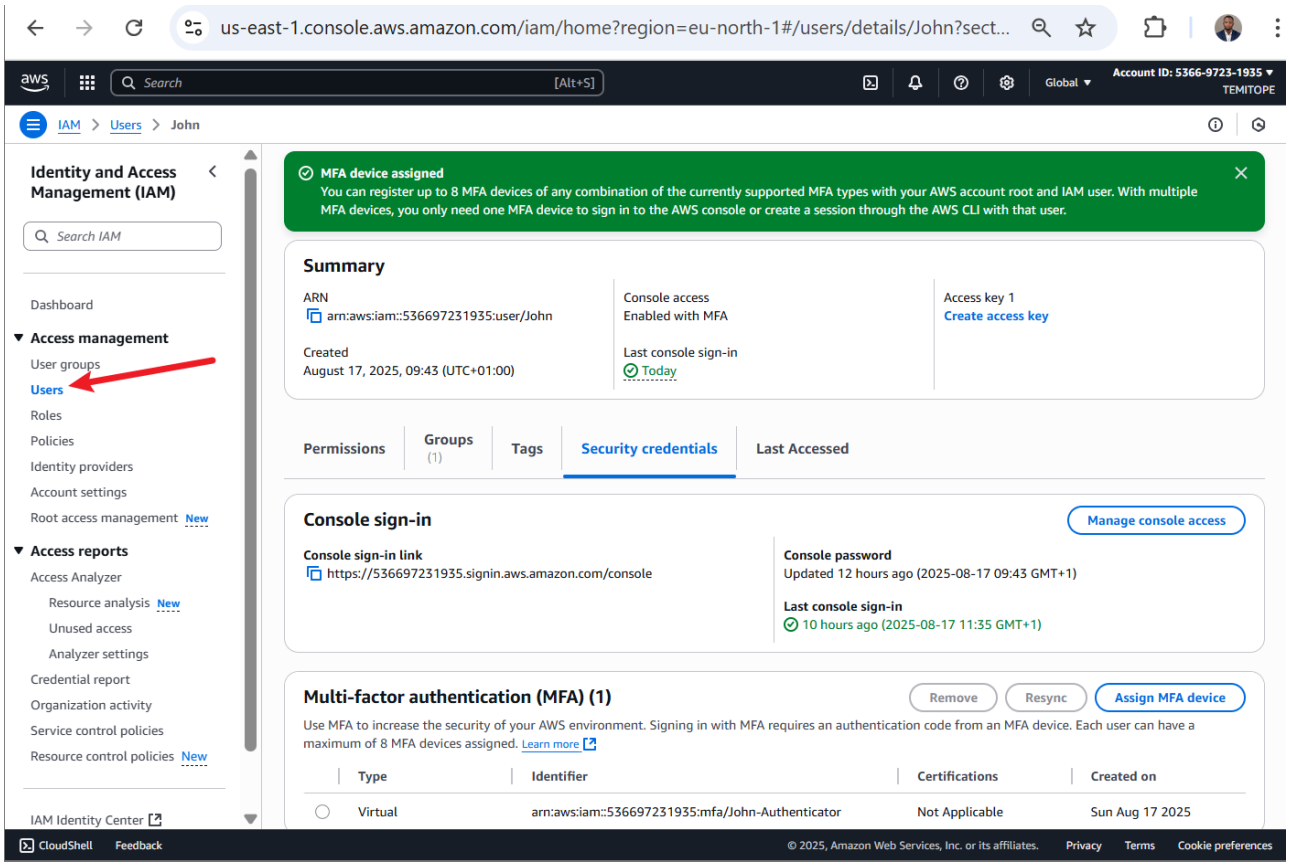


Mary (Data Analyst)

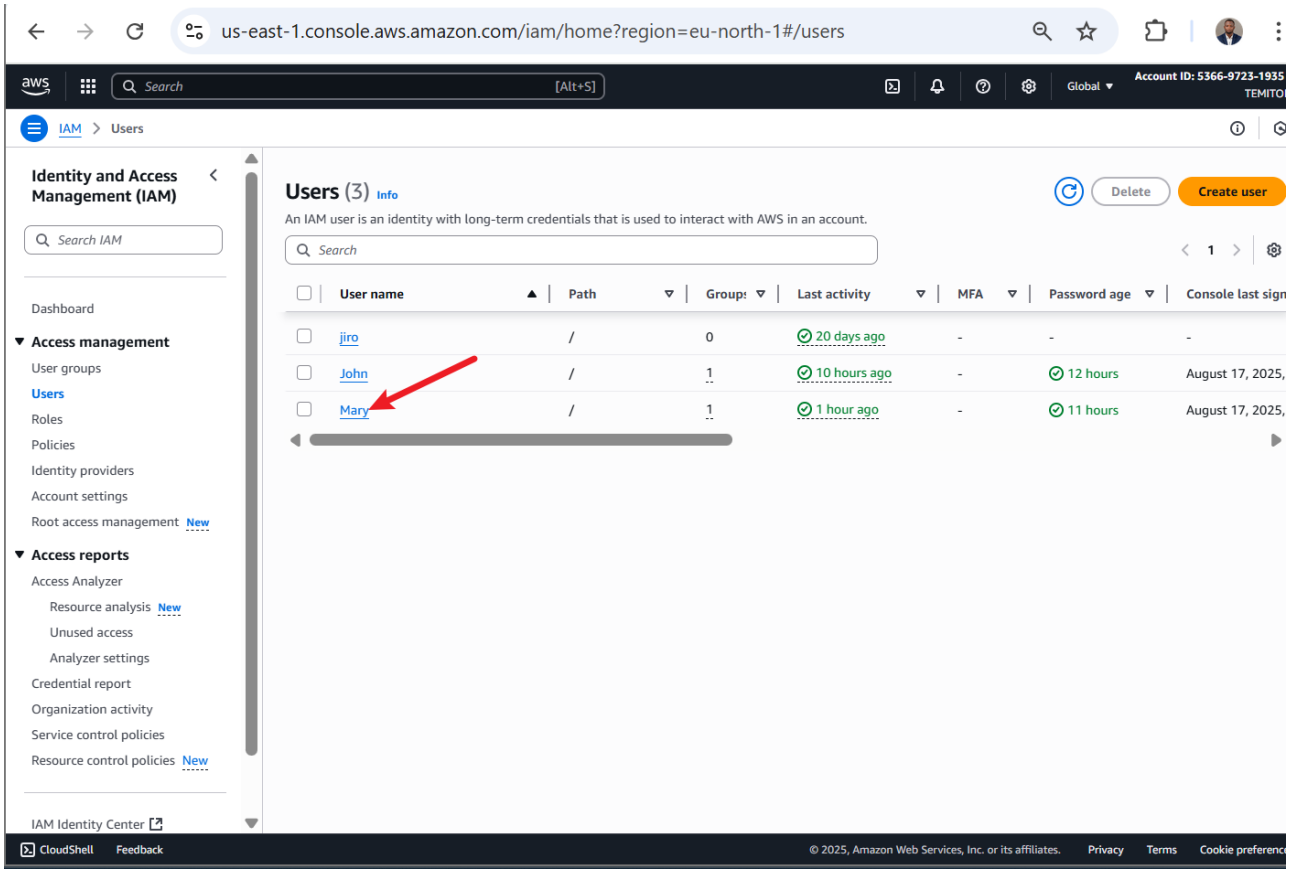
- Enabled MFA.
- Linked with **Microsoft Authenticator**.
- Tested successful login with MFA.

Steps:

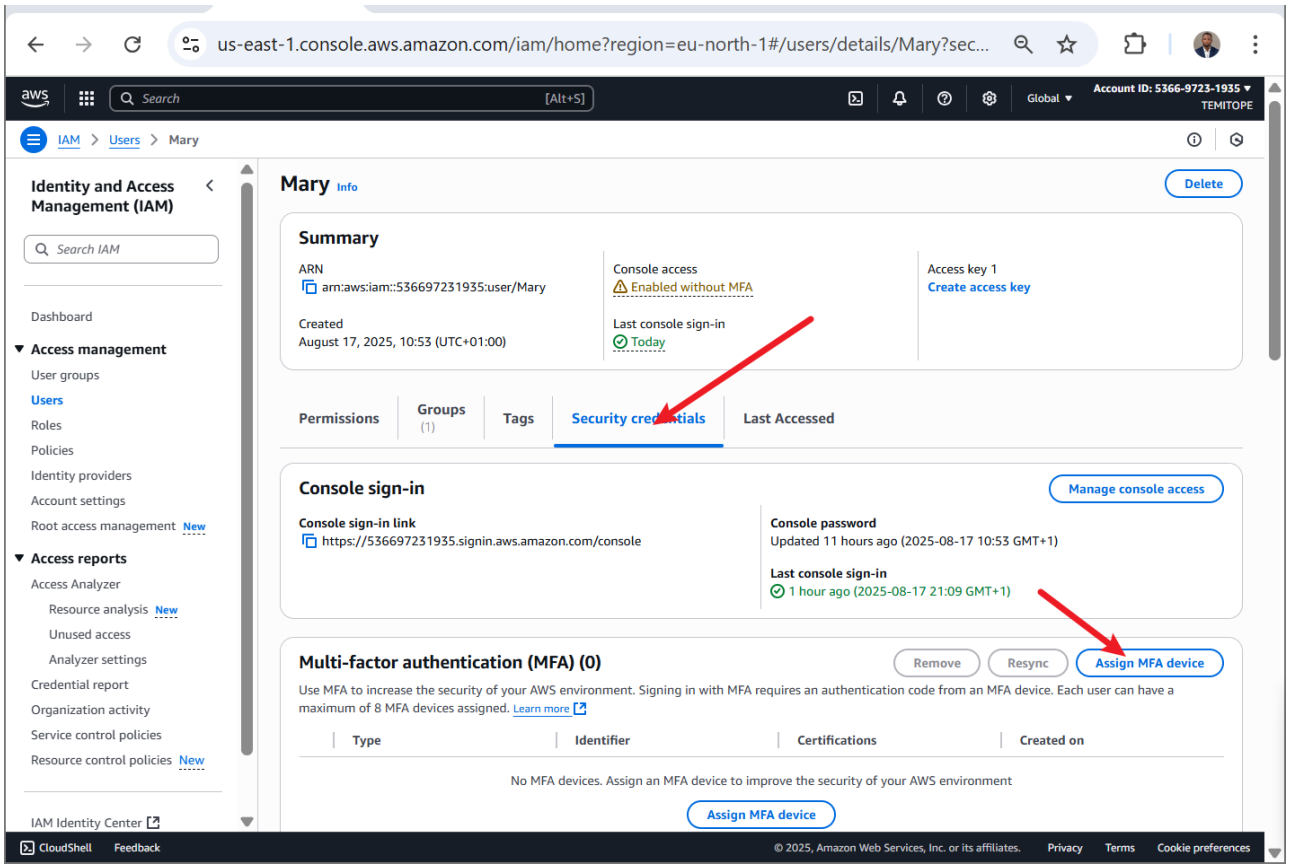
- Click users to repeat the same step for Mary.



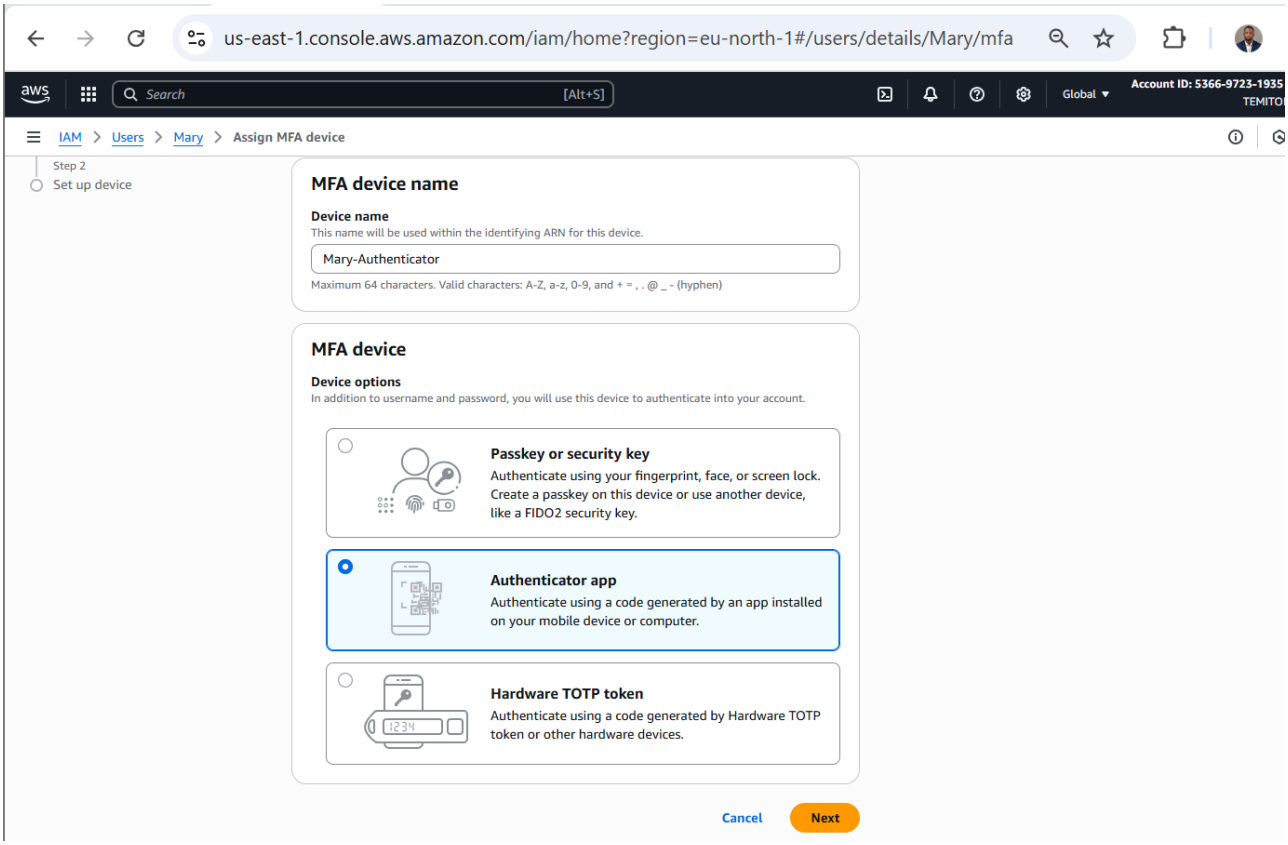
- Click on Mary.



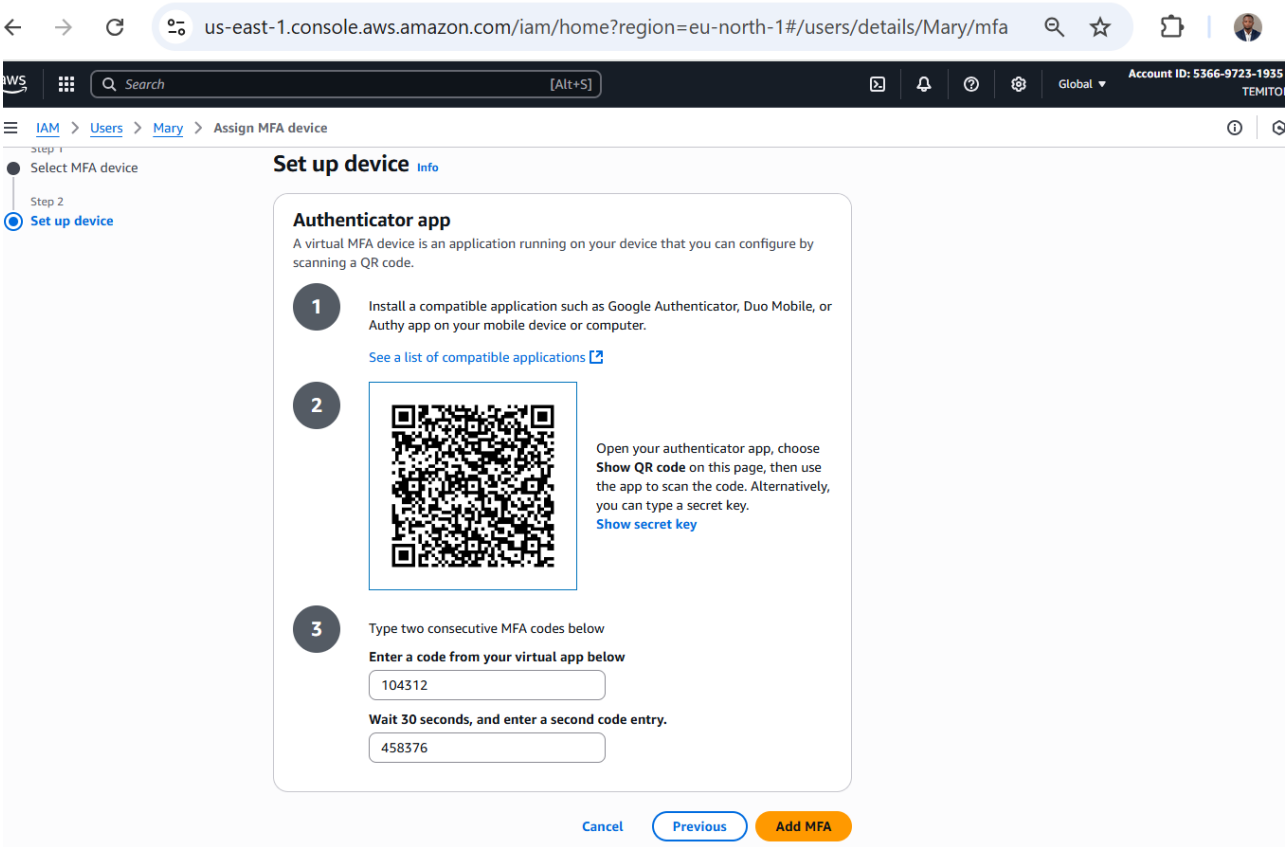
- Click security credential and assign MFA.



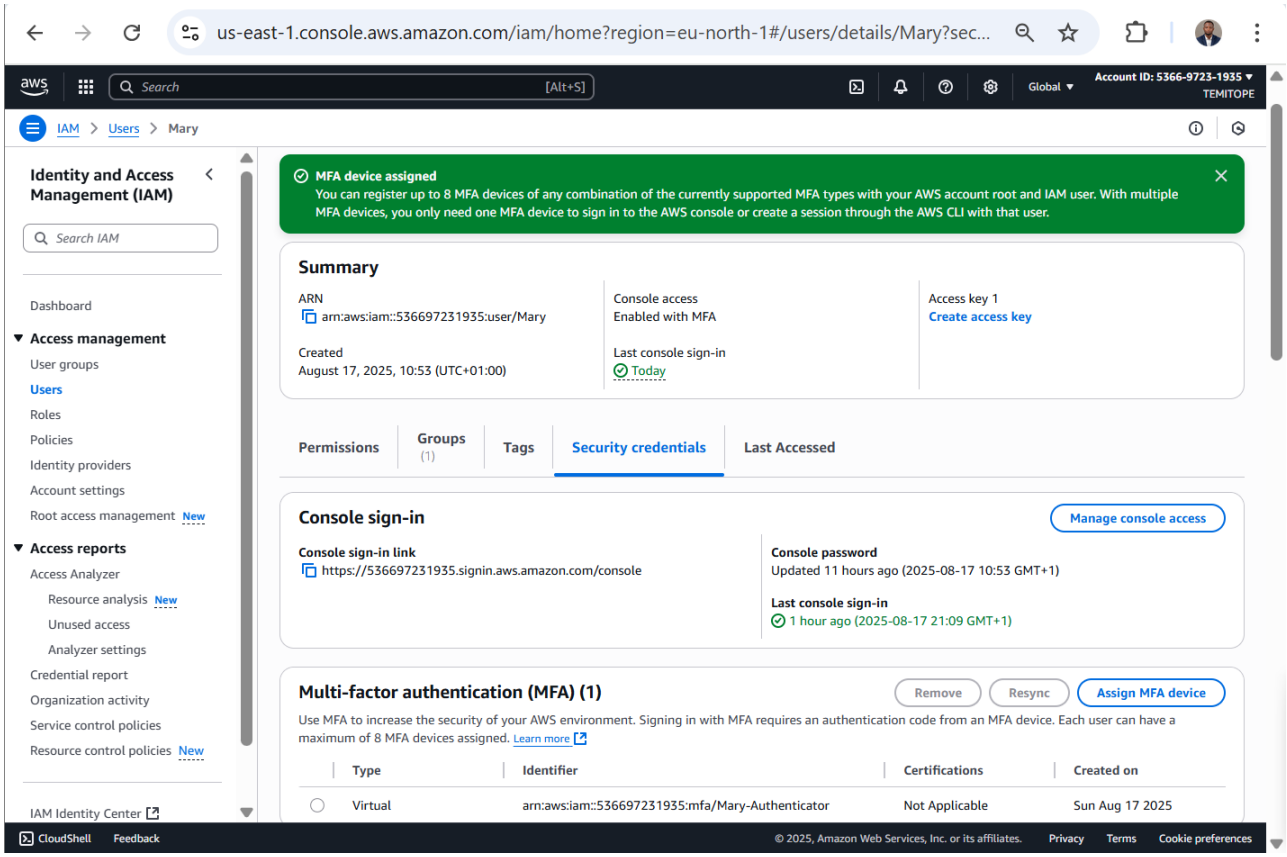
- Set name and choose authenticator app and click next.



- Show QR, scan it and supply code as instructed.



- MFA successfully added.



---

9. Project Reflection

9.1 Role of IAM in AWS

IAM defines **who** can access AWS and **what actions** they can perform. It enhances security, compliance, and efficient resource management.

9.2 Difference Between IAM Users and Groups

- **IAM Users** → Individual identities (e.g., John, Mary).
- **IAM Groups** → Collections of users with shared permissions (e.g., Development-Team, Analyst-Team). Groups simplify permission management at scale.

9.3 Process of Creating IAM Policies

- Identify required service.
- Select allowed actions.
- Define resources.
- Save as **custom policy**.
- Attach policy to a user or group.

9.4 Significance of Principle of Least Privilege

Granting users only the access they need minimizes risks and prevents accidental or malicious misuse of AWS resources.

9.5 Scenario with John and Mary

- **John** → User in Development-Team → EC2 access only.
- **Mary** → User in Analyst-Team → S3 access only.
- Both secured with **MFA**.
- This setup mirrors real-world fintech security practices.

---

10. Conclusion

This project demonstrated how AWS IAM helps enforce **secure, role-based access control** for Zappy e-Bank. By applying IAM concepts such as **users, groups, policies, and MFA**, the project successfully built a secure environment that aligns with fintech compliance requirements and the **principle of least privilege**.