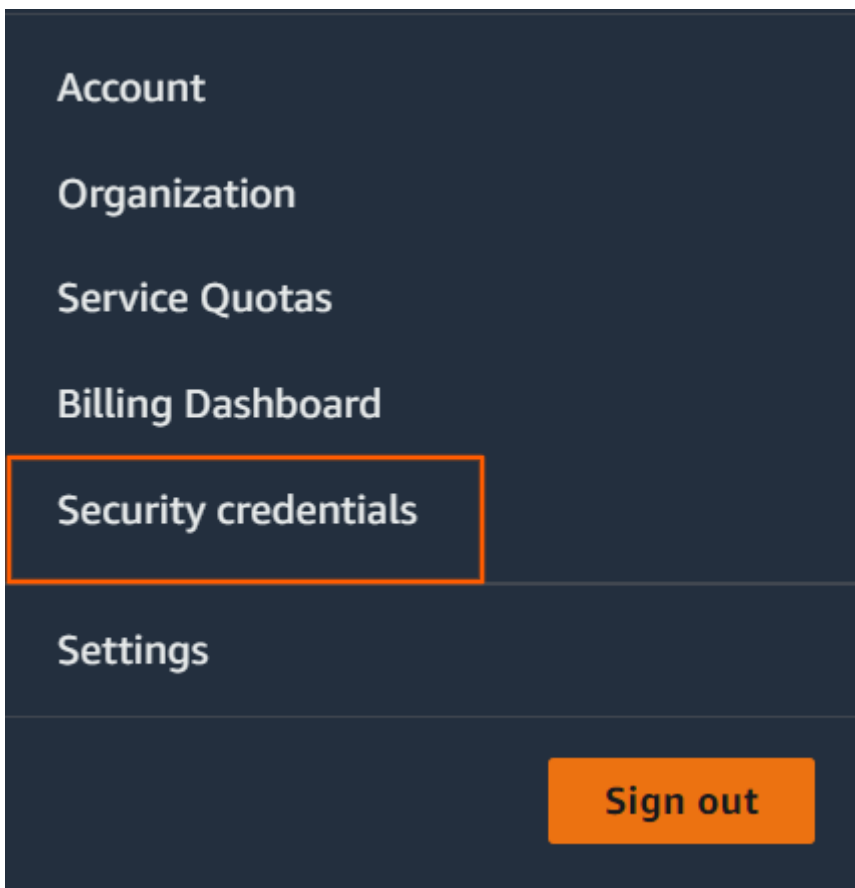


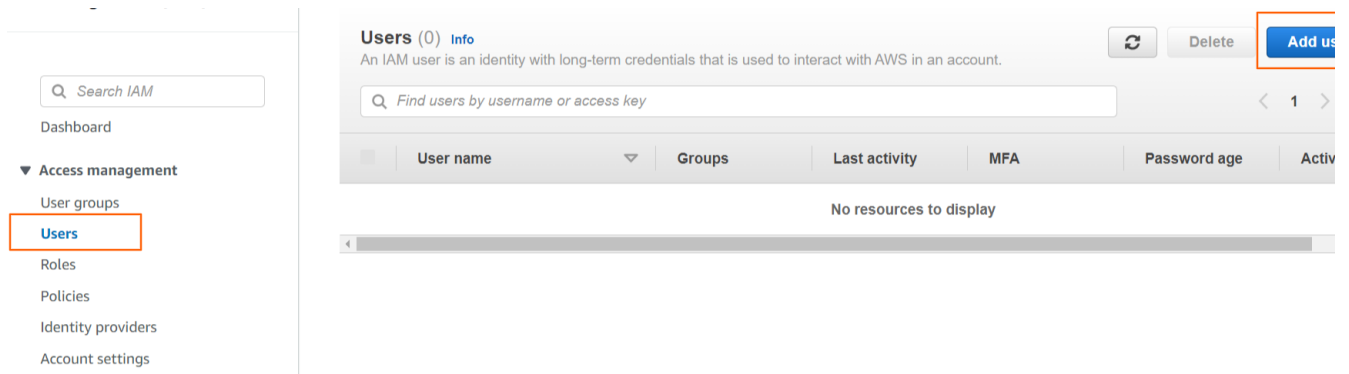
# How to Create AWS Access and Secret Key for BYOAWS

To create your AWS Access and Secret Key, follow these steps.

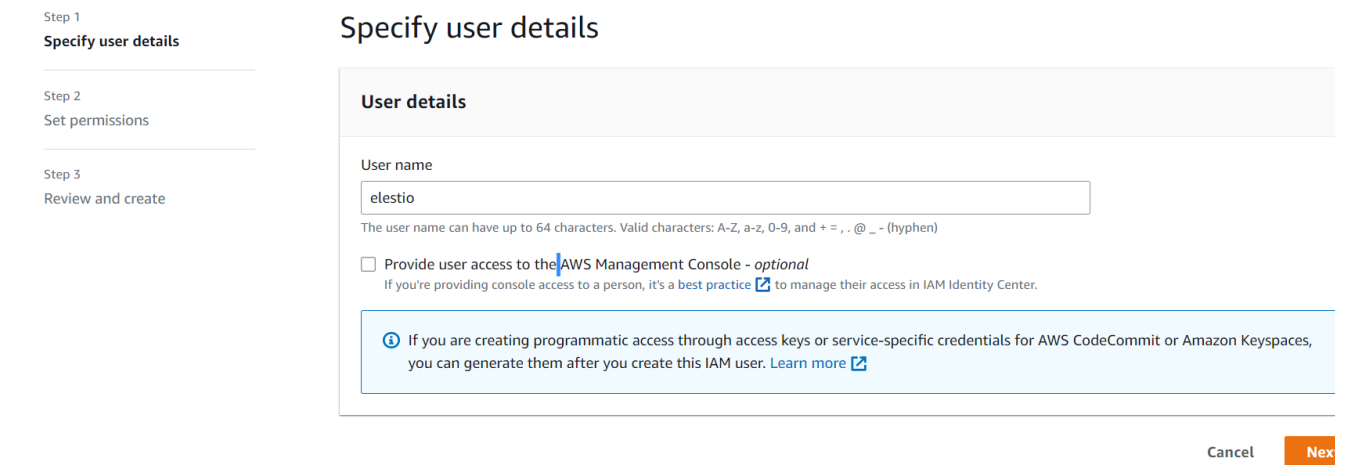
- **Step 1**:- Use your AWS account ID or account alias, your IAM user name, and your password to sign in to the [IAM console](#)
- **Step 2**:- In the navigation bar on the upper right, choose your user name, and then choose **Security credentials**.



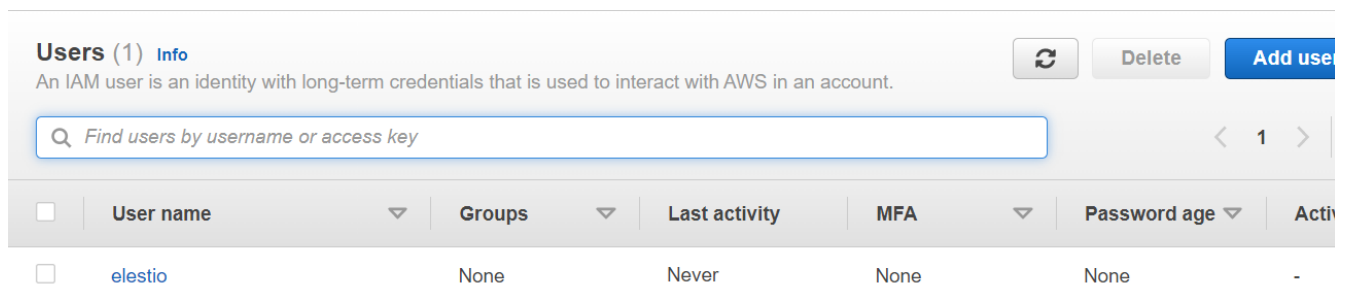
- **Step 3**:- If you want to create the AWS Access and Secret credentials with the root user, skip to **step 4**; otherwise, click **Users** on the left side of access management and then **Add users**.



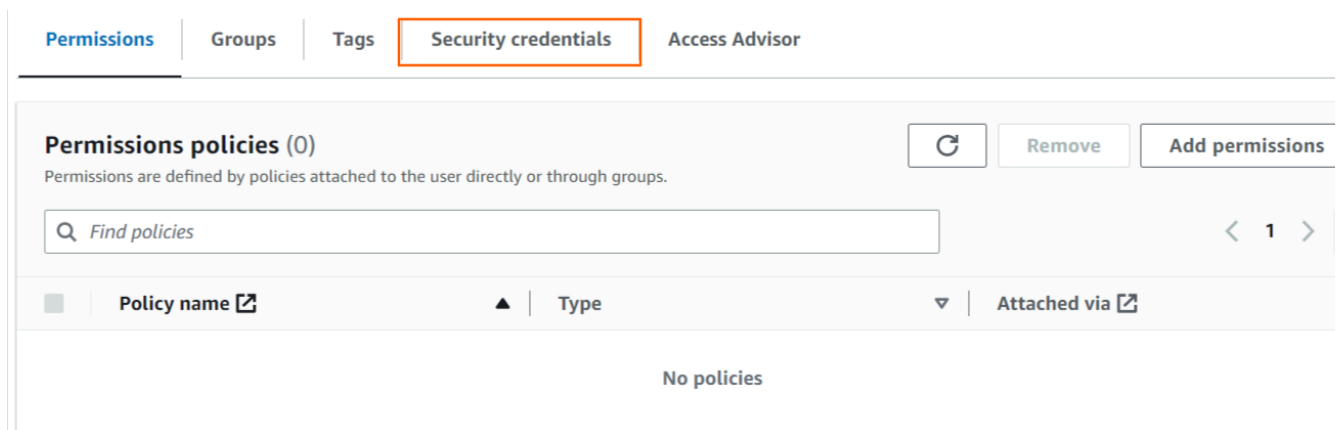
Enter your AWS Account access user name and click Next, Next, Next and Create User to proceed.



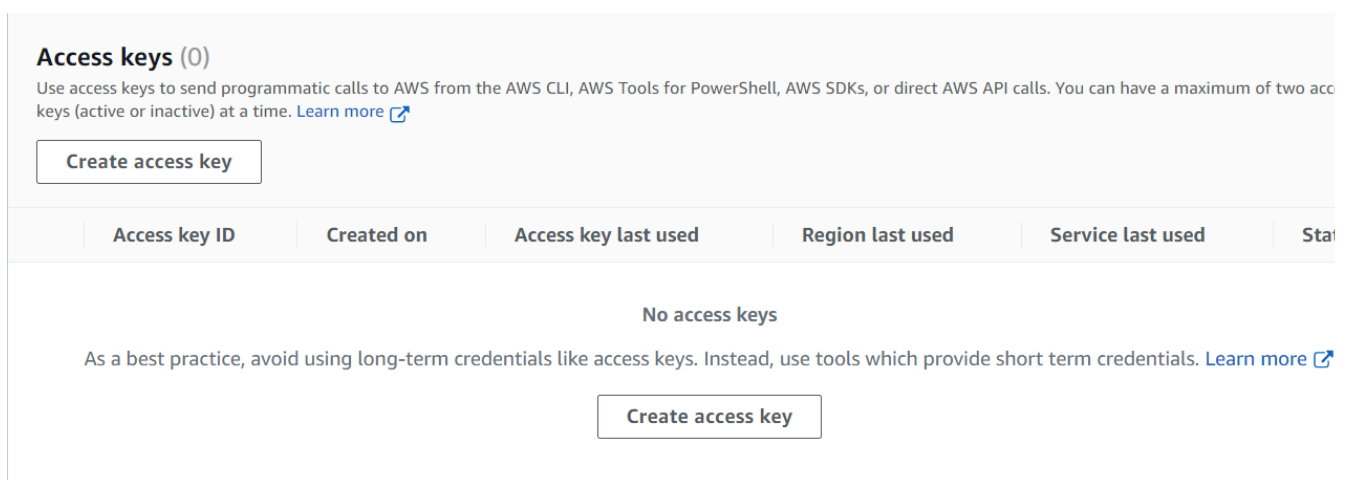
After clicking the Create User button, you will be taken to the user list. Choose your created user name from the user list and click on it.



Navigate to **Security Credentials** under User Details.



- **Step 4:-** Select **Create access key** from the **Access keys section**. If you already have two access keys, this button is disabled, and you must delete one before creating a new one.



- On the **Access key best practices & alternatives** page, choose your use case to learn about additional options which can help you avoid creating a long-term access key. If you determine that your use case still requires an access key, choose **Other** and then choose **Next**.
- (Optional) Set a description tag value for the access key. This adds a tag key-value pair to your IAM user. This can help you identify and rotate access keys later. The tag key is set to the access key id. The tag value is set to the access key description that you specify. When you are finished, choose to **Create access key**.

**For Non-Root Users:-** Choose **Command Line Interface** as an alternative and check the box, then click Next and **Create Access Key**.

Step 1  
Access key best practices & alternatives

Step 2 - optional  
Set description tag

Step 3  
Retrieve access keys

## Access key best practices & alternatives

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

**Command Line Interface (CLI)**  
You plan to use this access key to enable the AWS CLI to access your AWS account.

**Local code**  
You plan to use this access key to enable application code in a local development environment to access your AWS account.

**Application running on an AWS compute service**  
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.

**Third-party service**  
You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.

**Application running outside AWS**  
You plan to use this access key to enable an application running on an on-premises host, or to use a local AWS client or third-party AWS plugin.

**Other**  
Your use case is not listed here.



### Alternatives recommended

- Use [AWS CloudShell](#), a browser-based CLI, to run commands. [Learn more](#)
- Use the [AWS CLI V2](#) and enable authentication through a user in IAM Identity Center. [Learn more](#)

I understand the above recommendation and want to proceed to create an access key.

**For Root User:-** Check the box and click **Create Access Key**.

IAM > Security credentials > Create access key

Step 1  
**Alternatives to root user access keys**

Step 2  
Retrieve access key

## Alternatives to root user access keys

**Root user access keys are not recommended**

We don't recommend that you create root user access keys. Because you can't specify the root user in a permissions policy, you can't limit its permissions, which is a best practice.

Instead, use alternatives such as an IAM role or a user in IAM Identity Center, which provide temporary rather than long-term credentials. [Learn More](#)

If your use case requires an access key, create an IAM user with an access key and apply least privilege permissions for that user. [Learn More](#)

Continue to create access key?

I understand creating a root access key is not a best practice, but I still want to create one.

Cancel **Create access key**



- **Step 5:-** On the **Retrieve access keys** page, choose either **Show** to reveal the value of your user's secret access key, or **Download .csv file**. This is your only opportunity to save your secret access key. After you've saved your secret access key in a secure location, choose **Done**.

Step 1  
Alternatives to root user access  
keys

Step 2  
**Retrieve access key**

## Retrieve access key

**Access key**  
If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key	Secret access key
 [Redacted]	 [Redacted] <a href="#">Hide</a>

**Access key best practices**

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the [Best practices for managing AWS access keys](#).

[Download .csv file](#) [Download](#)

- **Step 6:-** Now grant **Administration** access to the newly created keys.

“ Elestio requires a *AmazonEC2FullAccess* permission to manage your Ec2 services

Navigate to permission in the user details to add the permission and select **Add Permission**

[Permissions](#) | [Groups](#) | [Tags](#) | [Security credentials](#) | [Access Advisor](#)

**Permissions policies (0)** [Refresh](#) [Remove](#) [Add permissions](#)

Permissions are defined by policies attached to the user directly or through groups.

[<](#) **1** [>](#)

<input type="checkbox"/>	Policy name <a href="#">↗</a>	▲	Type	▼	Attached via <a href="#">↗</a>
No policies					

Select the **Attach policies directly** Tab to attach permission.

## Add permissions

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

### Permissions options

Add user to group

Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.

Copy permissions

Copy all group memberships, attached managed policies, inline policies, and any existing permissions boundaries from an existing user.


Attach policies directly

Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

Now Enter *AmazonEC2FullAccess* into the search box and then select the *AmazonEC2FullAccess* permission from the results.

Permissions policies (2/1043) ↻ Create policy

🔍 AmazonEC2FullAccess ✕ 1 match < 1 >

<input checked="" type="checkbox"/>	Policy name <a href="#">↗</a>	Type	Attached entities
<input checked="" type="checkbox"/>	 AmazonEC2FullAccess	AWS managed	1

After Selecting both Click **Next** and then click **Add permission** to proceed.

## Review

The following policies will be attached to this user. [Learn more](#) [↗](#)

### User details

User name  
elstio

---

### Permissions summary (1)

< 1 >

Name <a href="#">↗</a>	Type	Used as
AmazonEC2FullAccess	AWS managed	Permissions policy

Cancel Previous Add permission

Follow these steps if you already have an access key and want to activate it instead of creating a new one.

- In the **Access keys** section, find the key to activate.

**Access keys (1)**  
Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. [Learn more](#)

Actions ▼ Create access key

Access key ID	Created on	Access key last used	Region last used	Service last used	Status
● [redacted]	11 minutes ago	None	N/A	N/A	⊖ Inactive

- To activate the key, go to **Actions** and select **Activate**.

**Access keys (1)**  
Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. [Learn more](#)

Actions ▲ Create access key

- Deactivate
- Activate
- Delete

ID	Created on	Access key last used	Region last used	Service last used	Status
[redacted]	11 minutes ago	None	N/A	N/A	⊖ Inactive

Before adding keys to elestio, you must grant Administration access to them.