

Docker Deployment on AWS Cloud

Add new user

Go to AWS → IAM → Users

Add new user

Select Access key — Programmatic access checkbox

Set user details

You can add multiple users at once with the same access type and permissions. [Learn more](#)

User name*

[+ Add another user](#)

Select AWS access type

Select how these users will primarily access AWS. If you choose only programmatic access, it does NOT prevent users from accessing the console using an assumed role. Access keys and autogenerated passwords are provided in the last step. [Learn more](#)

Select AWS credential type*

- ☒ **Access key - Programmatic access**
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.
- ☐ **Password - AWS Management Console access**
Enables a **password** that allows users to sign-in to the AWS Management Console.

Attach policy

Give this user, administrator permissions for AWS Elastic Beanstalk resources. Attach the policy 'AdministratorAccess-AWSElasticBeanstalk' to the list of permissions.


Add permissions to aws_cli_user


1


2

Grant permissions

Use IAM policies to grant permissions. You can assign an existing policy or create a new one.

 Add user to group

 Copy permissions from existing user

 Attach existing policies directly

Create policy

↺

Filter policies ▾		Q awselasticbeanstalk	Showing 14 results	
	Policy name ▾	Type	Used as	
<input checked="" type="checkbox"/>	AdministratorAccess-AWSElasticBeanstalk	AWS managed	None	
<input type="checkbox"/>	AWSElasticBeanstalkCustomPlatformforEC2Role	AWS managed	None	
<input type="checkbox"/>	AWSElasticBeanstalkEnhancedHealth	AWS managed	Permissions policy (1)	
<input type="checkbox"/>	AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy	AWS managed	Permissions policy (1)	

Access credentials

Skip the tags for this user. Note down access key and access secret.

Review

Review your choices. After you create the user, you can view and download the autogenerated password and access key.

User details

User name	aws_cli_user
AWS access type	Programmatic access - with an access key
Permissions boundary	Permissions boundary is not set

Permissions summary

The following policies will be attached to the user shown above.

Type	Name
Managed policy	AdministratorAccess

Tags

No tags were added.

AWS Elastic Beanstalk CLI

What is AWS Elastic Bean?

Introduction, refer the slides

Installation of CLI

<https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/eb-cli3-install.html>

<https://github.com/aws/aws-elastic-beanstalk-cli-setup>

Elastic Bean Initiate

Go to the project directory containing the ML code with Dockerfile.

Run this command

```
eb init
```

*Follow the prompts and input your **access key id** and **secret access key** if prompted. These are the settings I used for my specific project. The region will differ depending on your location and latency.*

- *Default region: 16*
- *Select an application to use: 3 [Create new Application]*
- *Application name: flask-api*
- *Docker: Y*
- *Select platform branch: 1 (Docker running on 64bit Amazon Linux 2)*
- *CodeCommit: n*
- *SSH: n*

`eb init` should do the following.

You will notice that the AWS Elastic Beanstalk CLI will have created an `.elasticbeanstalk` folder containing a `config.yml` detailing deployment configurations.

Elastic Bean Application Create

Now in the same project folder, run the following command

```
eb create
```

Elastic Beanstalk will automatically handle all the infrastructure deployment required to help you deploy your project so you can concentrate on developing the codebase. It will go through the process of automatically creating all the necessary resources such as launching an EC2 instance, setting load balancer, auto scaling groups and routes to serve your application.

Upon successful deployment, Elastic Beanstalk will return an address where the application will be available.